Efficient Service Task Assignment in Grid Computing Environments

Dr Angelos Michalas

Technological Educational Institute of Western Macedonia

Dr Malamati D. Louta

Harokopio University of Athens

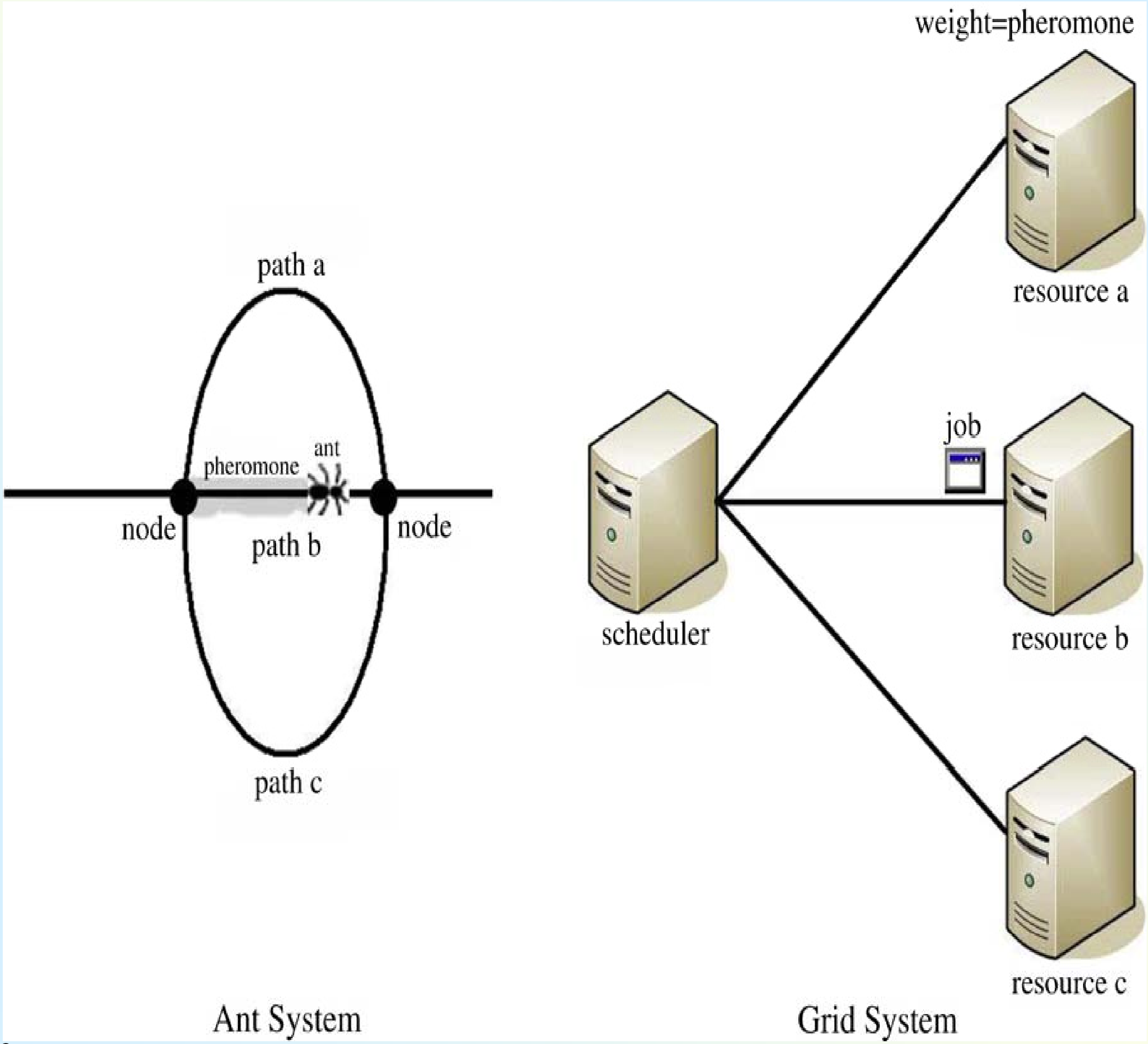
Task Assignment Framework (1)

- **Assuming that a user wishes to perform a specific service task, which can be served by various candidate Grid nodes (CGNs), a problem that should be addressed is the assignment of the requested service task to the most appropriate Grid node.**
- **The pertinent problem is called Service Task Allocation/Assignment (STA).**

Service Task Assignment

- GIVEN
  - the set of candidate Grid nodes and their layout,
  - the set of service tasks constituting the required services
  - the resource requirement of each service task in terms of CPU utilization
  - the current load conditions of each grid node and of the network links
- FIND
  - best assignment pattern of tasks to service nodes
- Our approach uses an Ant Colony Optimization algorithm (ACO) for service task allocation.
- ACO algorithms are based in a behavioral pattern exhibited by ants and more specifically their ability to find shortest paths using pheromone, a chemical substance that ants can deposit and smell across paths.
- ACO is used to solve many NP-hard problems including routing, assignment, and scheduling.

Mapping between the ant system and the grid system.



BACKGROUND (1)

- (Yan, 2005) uses the basic idea of MMAS ACO . The pheromone deposited on a trail includes:
  - ✓ an encouragement coefficient when a task is completed successfully and the resource is released,
  - ✓ a punishment coefficient when a job failed and returned from the resource
  - ✓ a load balancing factor related to the job finishing rate on a specific resource.
- (Chang, 2009) uses a balanced ACO which performs job scheduling according to resources status in grid environment and the size of a given job.
  - ✓ Local pheromone update function updates the status of a selected path after job assignment.
  - ✓ Global pheromone update function updates the status of all existing paths after the completion of a job.

BACKGROUND (2)

- (Dornermann, 2007) presents a metascheduler which decides where to execute a job in a Grid environment consisting of several administration domains controlled by different local schedulers.
  - ✓ AntNests offer services to users based on the work of autonomous agents called Ants.
  - ✓ A grid node hosts one running AntNest which receives, schedules and processes Ants as well as sends Ants to neighboring AntNests.
  - ✓ State information carried by Ants is used to update pheromones on paths along AntNests.

Service Task Assignment Architecture

- **Grid Resource Provider Agent (GRPA):** selecting on behalf of the Grid resource provider the best service task assignment pattern.
- **User Agent (UA):** promoting the service request to the appropriate GRPA.
- **The Grid Node Agent (GNA):** promoting the current load conditions of a Candidate Grid Node (CGN).
- **The Network Provider Agent (NPA):** providing current network load conditions (i.e., bandwidth availability) to the appropriate GRPA.
- **The GRPA**
  - ✓ Interacts with the UA in order to acquire the user preferences, requirements and constraints, analyzes the user request in order to identify the respective requirements in terms of CPU.
  - ✓ Interacts with the GNAs so as to obtain the CGN’s load conditions and with the NPAs so as to acquire the network load conditions.
  - ✓ Ultimately selects the most appropriate service task assignment pattern.

The ACO Algorithm

- The initial pheromone value of each CGN is given by the formula:

$$\tau_j(0) = CPU\_Speed_j \cdot (1 - CPU\_Load_j) \quad (1)$$

- Pheromone trails are updated upon assignment of a task on a CGN and upon termination of a task according to the formula:

$$\tau_j^{post} = \rho \cdot \tau_j^{pre} + \Delta\tau_{ij} \quad (2)$$

  - ✓ When task i is assigned to CGN j, $\Delta\tau_{ij} = -M$, while when task i is completed and CGN j is released $\Delta\tau_{ij} = M$.
  - ✓ M is a positive value relevant to the computation workload of the task.

- The desirability of assigning task i to CGN j is defined by the following formula:

$$des_{i,j} = \tau_j - Com\_Cost_{i,j} \quad (4)$$

  - ✓ The factor $Com\_Cost_{i,j}$ is the cost of migration to CGN j

Service Provisioning Modules

- Simulated grid environment composed of six service nodes reside on a 100Mbit/sec Ethernet LAN, running the Linux Redhat OS.
- The overall Grid Resource Provisioning System (GRPS) has been implemented in Java.
- Voyager mobile agent platform used for the realisation of the software components as well as for the inter-component communication.
- GRPA and monitoring modules GNAs, NPAs implemented as fixed agents
- The service task is implemented as intelligent mobile agent, which can migrate and execute to remote service nodes.

Experimental Results(1)

- To evaluate the efficiency of our service task allocation method the following experimental procedure has been followed which is similar to (Chang, 2009).
- We consider 1500 simple tasks each performing matrix multiplication of real numbers. The matrix sizes are varying from 400x400 up to 1000x1000.
- The task size depends on its matrix size and is about n x n x 4 bytes (each real number is represented by 4 bytes).
- The number of instructions that the task contains, can be drawn from task's complexity.
- Since matrix multiplication has $O(n^3)$ complexity, $2n^3$ instructions are estimated for a $n \times n$ matrix multiplication.
- Communication cost is similar for all hosts only the computation workload of tasks is considered.

Experimental Results(2)

- The same experiments have been conducted with and without using the ACO service task allocation scheme.
- In the latter case, service tasks are assigned in a round robin fashion to service nodes.
- In order to measure the efficiency of both methods we use the standard deviation of CPU load of CGNs.
- The load of each CGN is sampled after each task assignment and the standard deviation of each method is computed per 100 samples from 100 to 1500 tasks.
- The standard deviation is computed as:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})}$$

- Where σ is the standard deviation x_i is the CPU load of resource i and $\bar{x}$ is the average load of all resources.

Experimental Results(3)

- From the obtained results, we observe a decrease in the standard deviation when the ACO service task assignment scheme is used
- Verifies that the load of CGNs is better balanced compared to the Round Robin assignment scheme.