

# ACS-based resource assignment and task scheduling in grid

Qi Chao<sup>1,2</sup> Zhang Jing<sup>1</sup> Li Junhuai<sup>1</sup>

(<sup>1</sup> School of Computer Science and Engineering, Xi'an University of Technology, Xi'an 710048, China)

(<sup>2</sup> School of Computer Science, Shaanxi Normal University, Xi'an 710062, China)

**Abstract:** To solve the deadlock problem of tasks that the interdependence between tasks fails to consider during the course of resource assignment and task scheduling based on the heuristics algorithm, an improved ant colony system (ACS) based algorithm is proposed. First, how to map the resource assignment and task scheduling (RATS) problem into the optimization selection problem of task resource assignment graph (TRAG) and to add the semaphore mechanism in the optimal TRAG to solve deadlocks are explained. Secondly, how to utilize the grid pheromone system model to realize the algorithm based on ACS is explicated. This refers to the construction of TRAG by the random selection of appropriate resources for each task by the user agent and the optimization of TRAG through the positive feedback and distributed parallel computing mechanism of the ACS. Simulation results show that the proposed algorithm is effective and efficient in solving the deadlock problem.

**Key words:** grid; resource assignment; task scheduling; ant colony system (ACS); task resource assignment graph (TRAG); semaphore

The essential aspect of resource assignment and task scheduling (RATS) in grid computing<sup>[1]</sup> is to assign all tasks to corresponding heterogeneous resources on the bases of applications requirements and resources information in order to gain minimum makespan and maximum resource utilization. As a result, RATS is NP-hard in general<sup>[2]</sup>. So far, there has been no effective algorithm found for the NP-complete problem. It becomes necessary to employ heuristics to arrive at a near-optimal solution.

The ant colony algorithm has a better positive feedback mechanism and adopts a distributed parallel computing mechanism, which means that it is easy to integrate it with other methods and it possesses relative strong robustness<sup>[3]</sup>. The ant colony system (ACS)<sup>[4]</sup> presented by Dorigo and Gambardella is one of the improved ant colony optimization (ACO) algorithms<sup>[5]</sup>. It shows obvious advantages regarding aspects of solving many complex combination-optimization problems.

Based on the ant colony algorithm, existing approaches adopted to solve the problem of RATS in grid can find the optimal assignment scheduling scheme quickly. But they fail to consider the interdependence between tasks. There is also the relationship of precedence constraints among the tasks in general. This paper studies the problem based on ACS and proposes an im-

proved assignment scheduling algorithm.

## 1 Improvement of RATS Problem

Usually a huge application system will be decomposed into many interrelated tasks. With each task as a node, communication and precedence constraints among tasks as a directed edge, the grid application can be expressed as a directed acyclic graph (DAG)<sup>[6-8]</sup>.

We define  $T$  and  $M$  to express the collection of tasks and resources. To select one resource from  $M$  for each task in  $T$  randomly, there will be a possible mapping from  $T$  to  $M$ . Then, a task resource assignment graph (TRAG) can be composed<sup>[9]</sup>. If there are  $N$  available resources and  $K$  tasks, there will be  $K^N$  TRAG. The execution time of the application includes task execution time  $t_s$  and communication time  $t_c$ . For the purpose of minimizing the completion time of an application, we must select the TRAG with the shortest key path  $P$  which can be expressed as  $T_p$ , namely the sum of  $t_s$  and  $t_c$  of the tasks on  $P$ . Thus, we transform the RATS problem to the TRAG optimization selection problem.

The quantity of tasks is generally more than that of the resources in grid. So, the precedence constraints between tasks must be considered when task scheduling maps tasks to resources. It is possible that several tasks are mapped to the same resource. Given that task  $t_i$  and  $t_j$  are mapped to resource  $m_k$ , and  $t_i$  is prior to  $t_j$ , the deadlock will occur if  $t_j$  obtains  $m_k$  first. To avoid deadlock, the tasks with precedence constraints rela-

Received 2007-05-18.

**Biographies:** Qi Chao (1975—), male, graduate; Zhang Jing (corresponding author), male, doctor, professor, zhangjing@xaut.edu.cn.

tionships must access the same resource mutually exclusively. We can add a semaphore mechanism in the

assignment scheduling strategy to realize it. The algorithm is shown in Fig. 1.

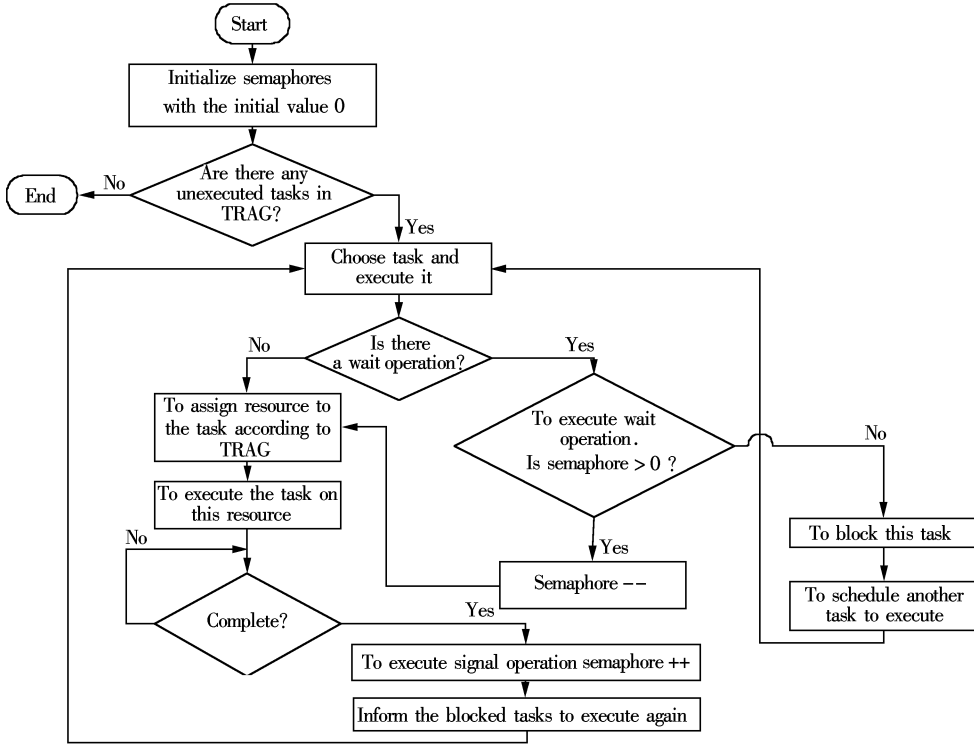


Fig. 1 The algorithm of avoiding deadlock

Both wait operation and signal operation are atomicity operations. For example, task  $t_1$  is prior to  $t_2$  and  $t_3$ . Task  $t_2$  is prior to  $t_4$  and  $t_5$ . Task  $t_3$ ,  $t_4$  and  $t_5$  are prior to  $t_6$ . Resource  $m_1$  is assigned to task  $t_1$  and  $t_3$ , and resource  $m_4$  is assigned to task  $t_2$  and  $t_4$ . Task  $t_5$  and  $t_6$  gain resources  $m_3$  and  $m_7$ , respectively. A set of semaphores  $(a, b, c, d, e, f, g)$  with the initial value 0 is defined. Then we can realize the parallelism between tasks and access resources mutually exclusively.

$t_1 \rightarrow \text{Begin assign } m_1; \text{signal}(a, b); \text{end};$   
 $t_2 \rightarrow \text{Begin wait}(a); \text{assign } m_4; \text{signal}(c, d); \text{end};$   
 $t_3 \rightarrow \text{Begin wait}(b); \text{assign } m_1; \text{signal}(e); \text{end};$   
 $t_4 \rightarrow \text{Begin wait}(c); \text{assign } m_4; \text{signal}(f); \text{end};$   
 $t_5 \rightarrow \text{Begin wait}(d); \text{assign } m_3; \text{signal}(g); \text{end};$   
 $t_6 \rightarrow \text{Begin wait}(e, f, g); \text{assign } m_7; \text{end}.$

Only after task  $t_1$  is completed, will the value of semaphores  $a$  and  $b$  be changed from 0 to 1. And then, task  $t_2$  and  $t_3$  will start. Although resource  $m_1$  is allocated to task  $t_1$  and  $t_3$ , the deadlock will not occur, and  $t_2$  and  $t_3$  can continue parallel executing.

## 2 Description of Improved Algorithm Based on ACS

In the grid pheromone system model<sup>[10]</sup>, the union between the task manager and the scheduling manager acts as a user agent(UA), namely the interface between

the consumer and the grid system.

### 2.1 Construction of TRAG

The UA creates a new ant for each task  $k$ . First, the ant should obtain the basic information about task  $k$  and the basic requirement information about grid resources needed. Secondly, the UA can obtain a set of available resources  $R$  for each task based on the information. Then a certain resource  $r$  can be selected randomly from  $R$  for task  $k$ . Finally, a certain TRAG is constructed. And the optimized TRAG can be found based on ACS.

### 2.2 Procedure of algorithm

1) Obtain the information about available grid resources before preallocation.

2) Select the serial number of the resource according to the pseudo-random proportional rule:  $s_k = \arg \max_{u \in J_k} \{[\tau_j(t)]^\alpha [\eta_j(t)]^\beta\}$ ,  $q \leq q_0$ ; otherwise,  $s_k = S_k$ . Where  $s_k$  denotes that task  $k$  chooses resource  $s_k$ ;  $q$  is a random number;  $S_k$  submits to the random proportional rule.

3) Compute  $p_{kj}$ :  $p_{kj} = \frac{[\tau_j(t)]^\alpha [\eta_j(t)]^\beta}{\sum_{j \in J_k} [\tau_j(t)]^\alpha [\eta_j(t)]^\beta}$ ,  $j, u \in J_k$ .

Where  $p_{kj}$  denotes the probability of resource  $j$  being chosen by task  $k$ ;  $J_k$  is the collection of grid resources on which task  $k$  can execute.

4) Update local pheromone:  $\tau_j(t) = (1 - \rho)\tau_j(t)$ ,  $\tau_{s_k}(t) = \tau_{s_k}(t) + \rho\Delta\tau_{s_k}(t)$ , where  $\rho$  is the local volatilization coefficient,  $\Delta\tau_{s_k}(t) = -K_k$ .

5) The UA constructs the TRAG and calculates the length of the key path:  $\min T = \min(T_p, \min T)$ .

6) Record the whole sequence  $(k, s_k)_{\text{step}}$  which  $\min T$  corresponds to. All the tasks have been preallocated once in terms of the current optimum solution.

7) Recover the pheromone to the value before the current loop.

8) Repeat until stopping criterion is satisfied, otherwise return to 2).

9) The UA sets up a semaphore in accordance with optimum TRAG.

10) Update global pheromone after the tasks are completed on the resources:

$$\tau_{s_k}(t) = (1 - \theta)\tau_{s_k}(t) + \theta\Delta\tau_{s_k}(t), \Delta\tau_{s_k}(t) = K_k$$

### 2.3 Simulation experiment

We create and dispatch 5, 10, 20, 40 and 80 tasks to 15 resources, respectively. Based on the existing algorithm and the improved algorithm presented in this paper, we performed the simulation experiments separately. We set  $\rho_0 = 0.8$ ,  $\theta = 0.9$ ,  $\alpha = 0.5$ ,  $\beta = 0.5$ . The results are shown in Fig. 2.

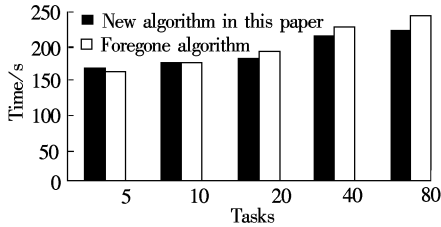


Fig. 2 Experimental results

The results shows that the two algorithms have the approximate completion time when the quantity of tasks is small. The reason is that both of them apply the global pheromone update rule only to the optimum ant path, which swells the positive feedback mechanism and accelerates the searching efficiency. Once the quantity of tasks becomes large, the completion time of the improved algorithm will be less than that of the foregone algorithm. Because the tasks are interdependent, the probability of deadlock on the shared resources increases with the increase in the number of tasks.

### 3 Conclusion

There has been no integrated theory of RATS of the grid system up to now. So the research in this field has a long way to go to develop and reach perfection. At present, the research of the heuristics scheduling al-

gorithm is still at the phase of analog simulation. The granularity of the research of the scheduling algorithm needs to be finer. A system of resource assignment and task scheduling in grid must be provided with a definite security mechanism in order to offer secure and reliable task scheduling services to grid.

### References

- [1] Foster I, Kesselman C. *The grid: blueprint for a future computing infrastructure* [M]. Morgan Kaufmann Publishers, 1998.
- [2] Zhang Yingfeng, Li Yulin. Grid computing resource management scheduler based on evolution algorithm[J]. *Computer Engineering*, 2003, **29**(15): 110 – 175.
- [3] Dorigo M, Gambardella L M. Ant colony system: a cooperative learning approach to the traveling salesman problem [J]. *IEEE Trans on Evolutionary Computation*, 1997, **1**(1): 53 – 66.
- [4] Gambardella L M, Dorigo M. Solving symmetric and asymmetric TSPs by ant colonies[C]//*Proceedings of the IEEE Conference on Evolutionary Computation*. IEEE Press, 1996: 622 – 627.
- [5] Colomi A, Dorigo M, Maniezzo V, et al. Distributed optimization by ant colonies [C]//*Proceedings of European Conference on Artificial Life*. Paris, 1991: 134 – 142.
- [6] He Liang, Jarvis Stephen A. Mapping DAG-based applications to multicusters with background workload[C]//*IEEE International Symposium on Cluster Computing and the Grid*. Cardiff, Wales, UK, 2005: 855 – 862.
- [7] Aggarwal Mona, Kent Robert D, Ngom Alioune. Genetic algorithm based scheduler for computational grids[C]//*Proceedings of the 19th International Symposium on High Performance Computing Systems and Applications (HPCS'05)*. Guelph, Ontario, Canada, 2005: 756 – 761.
- [8] Malewicz Grzegorz, Rosenberg Arnold L, Yurkewych Matthew. On scheduling complex dags for internet-based computing[C]//*Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05)*. Denver, CA, USA, 2005: 645 – 650.
- [9] Chen Tingwei, Zhang Bin, Hao Xianwen, et al. Task scheduling in grid based on particle swarm optimization[C]//*Proceedings of the Fifth International Symposium on Parallel and Distributed Computing (ISPDC'06)*. Timisoara, Romania, 2006: 455 – 461.
- [10] Qi Xuguang, Liang Zhengyou. Ant colony algorithm based resource allocation and task scheduling of grid[J]. *Journal of Guangxi University for Nationalities: Natural Science Edition*, 2006(2): 123 – 125. (in Chinese)

# 基于蚁群系统的网格资源分配与任务调度

祁 超<sup>1,2</sup> 张 璟<sup>1</sup> 李军怀<sup>1</sup>

(<sup>1</sup> 西安理工大学计算机科学与工程学院, 西安 710048)

(<sup>2</sup> 陕西师范大学计算机科学学院, 西安 710062)

**摘要:**为了解决基于启发式算法的资源分配和任务调度过程中由于没有考虑任务间的相互依赖关系而出现的任务死锁问题,提出了一种基于蚁群系统的改进算法.首先阐述了如何将分配调度问题映射到任务资源分配图的优化选择问题上和如何将信号量机制引入到最优任务资源分配图中来解决死锁问题.其次说明了基于蚁群系统如何利用网格信息素系统模型实现该算法,涉及任务资源分配图的构造,以及通过蚁群的正反馈和分布式并行计算机制优化任务资源分配图.最后模拟试验结果说明所提出的算法可以有效地解决网格中任务死锁问题.

**关键词:**网格;资源分配;任务调度;蚁群系统;任务资源分配图;信号量

**中图分类号:**TP393