Computer Engineering and Intelligent Systems
ISSN 2222-1719 (Paper) ISSN 2222-2863 (Online)
Vol.8, No.8, 2017

A Task Allocation Algorithm with Weighted Average Velocity Based on Online Active Period

Chuang-Wei Wang* Wen Wang Rui Gao Ke-Ming Tang
College of Information Engineering, Yancheng Teachers University, Jiangsu, 224002, China

Abstract
In some complex scientific calculation, the resources of the calculation are very large. To a certain extent, the improvement of the computer level has met the needs of many computing, but a lot of more complex calculation cannot still be effectively resolved. Volunteer computing is a computational method that divides the complexity of computing tasks into simple subtasks, and collects the results of volunteer computing resources to solve the subtasks. In this calculation process, the task assignment module is an extremely important part of the whole computing platform. Many of the existing task allocation algorithms (TAA) are used to group by the similar conditions of the volunteer computer. TAA used in this work grouped by the computers with similar online active period, and the computation efficiency is improved by using the weighted average velocity as a parameter. The experimental results showed that TAA with the weighted average velocity based on online active period can effectively improve the performance of the volunteer computing platform.
Keywords: Volunteer computing; Task allocation algorithm; Weighted average velocity; Online active period

1. Introduction

In some areas, especially in the field of scientific computing, we cannot often find the right performers for a large computing task. The main reason may be the large computational cost, or the long calculation period. To solve above problem, volunteer computing come into the world. With comparison to computational cost of supercomputers, volunteer computing is a low-cost, high-efficiency computing model by collecting idle resources of volunteer computers. Compared to common computers, volunteer computing combines their computing resources and computing power together to play a huge role. How to organize and improve cooperation mechanisms between computers? In order to manage these computers, we need more effective division management module and allocation module of tasks [1].

Most of the existing task allocation algorithm (TAA) is based on a computer's performance initial detected, except for some simple TAA [2], such as first come first served (FCFS) algorithm, random allocation algorithm [3].

In order to accurately evaluating the overall performance of a computer, some computing platform carry out hardware detection and resource assessment for volunteer computers [4], followed by weight depending on the proportion of results, finally get a parameter. The above allocation algorithm has higher reliability in once task allocation, but the frequent change of the size of idle computing resources will reduce the efficiency of the algorithm. Existing TAAs mostly group in term of the performance computers, and the large amounts of tasks is preferentially assigned to a high-performance groups. Moreover, some computers with in good hardware may be moved to low-performance groups by sever when it perform very complex tasks, resulting in few resources. In addition, some TAA based on credibility make similar hardware performance computers into a group. And the same set of tasks is assigned to every computer in this group. Then sever finally get an optimal value by comparing the results [5]. The above method may be result in low utilization of computing resources, although it can effectively avoid erroneous results, and the confidence is high.

Taking into account the above practical problems, this paper aims to proposing dynamic TAA with the weighted average velocity based on online active period [6], which grouped the volunteer computers according to their online active period, and can effectively improve the performance of the volunteer computing platform [7].

2. TAA with weighted average velocity based on online active period

2.1 The basic idea of the algorithm

To facilitate the management of volunteer computer, we made the online active period of each computer. The volunteer computer with similar online active period is divided into a group [8, 9]. Due to the difference of computer performance in a group, dynamic TAA with weighted average velocity as an important parameter was designed to improve the overall efficiency of computing platform, which can match the size of task to each computer [10]. When a low-performance computer is detected, less subtasks is assigned to it, and vice versa. The weighted average velocity as the basis for judging the ability of a computer to perform tasks is also a parameter of operating velocity of a computer, and calculating the size of the next. The general steps of the algorithm are substantially as follows.

Step 1: Check the state of all logged volunteer computer on volunteer computing platform, which include

major hardware and network.

Step 2: The volunteer computers grouped by online active period. Servers collect on-line time period for each computer and analyze, and the management modules of servers manage all the computers in the list.

Step 3: Servers divide big tasks into subtasks and assign them to the volunteer computer in all active group. Servers record the calculation time, the task size and maximum velocity during execution period. Then average velocity and maximum velocity are weighted for the calculation of the next task size.

Step 4: Volunteer computers return the calculation results, and servers deduce whether get the returned result. If successful, servers end the run of the project, otherwise, servers need to search the computers and assign tasks again, namely, repeat the above steps.

2.2 Algorithm Description

Make $agent_j$ represents the j-th Agent, n is the current total number of Agents involved in the calculation, and $j \in [1, n]$. In the $agent_j$, task (i, j) represents the size of the i-th task split. w_1 is the weigh value of the average velocity, $V_{average}$ (i, j), w_2 is the weight for maximum velocity, V_{max} (i, j), and $w_1 + w_2 = 1$.

Table 1 Symbols used in algorithm description

Symbols	Description
n	Total numbers of agents
K	Small group numbers in a large group
t_0	Test-run time of the first task
T	Execution cycle of a task
Time	Time online of agent
Task(i,j)	The i-th task spit, the task size of the j-th agent
V(i,j)	The i-th task spit, the weighted average velocity
t(i,j)	The i-th task spit, and running time of the j-th Agent
V_{max}(i,j)	The i-th task spit, and maximum velocity of the j-th Agent
$V_{average}$(i,j)	The i-th task spit, average velocity of the j-th Agent

When a project is to run, sever first search online Agents with the current time t0 as a starting point, and group as online period: t0 + T min, t0 + 2T minutes, t0 + 3T minutes, and so on. If online period doesn't achieve t0 + T, Agents do not need to be assigned into the group, whereas online period is longer than t0 + KT min, Agents are divided into the K-th group. Then server assigned tasks to all agents in each large group. For the first test-run, sever receive an average velocity and a maximum velocity from each agents. And then a weighted average rate may be obtained, according to which servers will determine the size of the next task to be executed by agents, and make execution time of each task is approximately equal to T min. When the first round task is completed, if the results are not desirable, each agent will continue to be assigned tasks again. For the second distribution, the task will only be assigned to the rest group of agents except for the first group, and to be run like previous run until only the K-th group agents left. If the expected value is finally not found, and the project is not implemented, the K-th agents are all disbanded. Continually, severs assign the tasks to other agents in other large group unless find the desired value of the project. The diagram of grouped agents is shown in Fig. 1. Symbols used in the algorithm description are described in Table 1.

Considering the differences of computing performance of volunteer computers, and optimal utilization and unified management of resources, the algorithm will firstly perform a test-run of small tasks o obtain the performance parameters of individual agents. This method will return total time and maximum velocity to obtain average velocity, $V_{average}$. Additionally, the instability of the computer's performance leads to that mere one average velocity does not accurately reflect the computing power of a computer, so the weighted average velocity, V, was introduced, and it is calculated as follows:

$V = w_1 * V_{average} + w_2 * V_{max}$, $(w_1 + w_2 = 1)$.

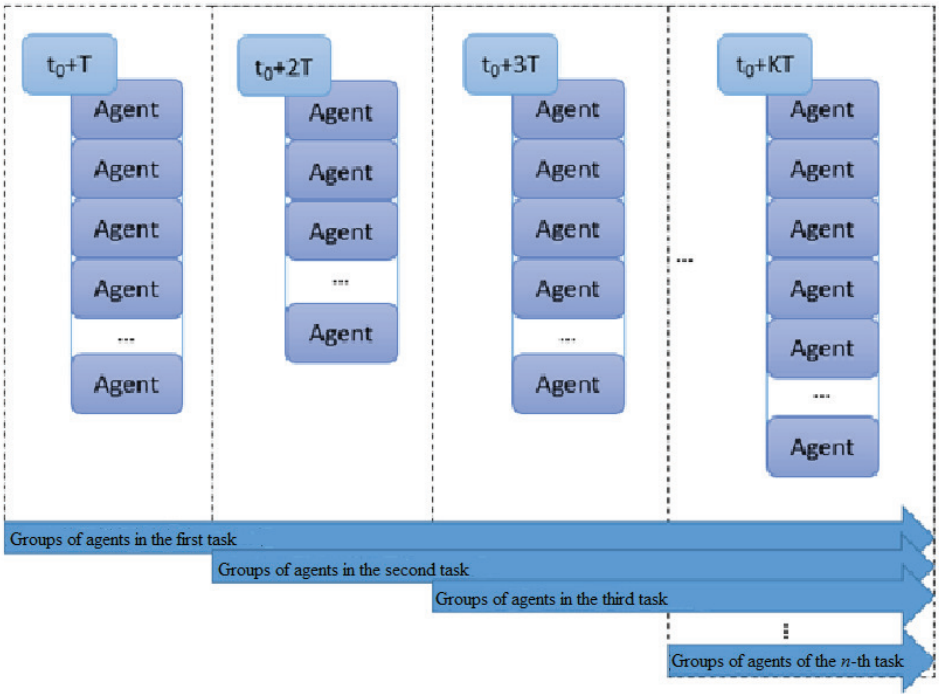

Fig. 1 Diagram of grouped agent

The first large group of agent is distributed. And serve may operate the first task test-run. As shown in Fig. 2, with the first task split, a portion of the Project is divided into the same size subtasks to be assigned to each Agent. After the completion of task, the maximum speed V_{max} (1, j), the task time t (1, j) for each Agent, and results of this task are all back to the server with data packets. Then average velocity, $V_{average}$ (1, j) and the weighted average velocity, V (1, j) are obtained by the server, and which are the basis of the second task distribution. When all the Agents will complete their task runs, server receives the results packet and deduce whether to find the desired value. if the answer is yes, the project is finished, or server continues to assign tasks to individual Agents. And so on. The n-th task allocation is displayed in Fig. 3.

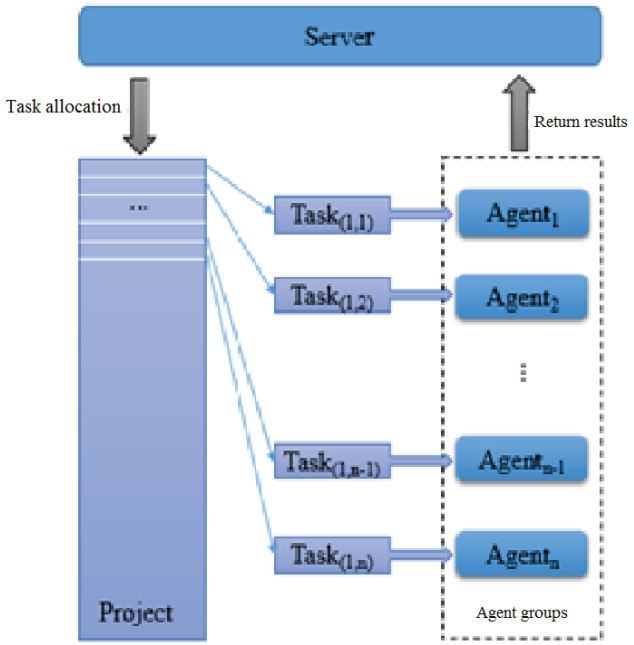

Fig. 2 The diagram of the first task allocation

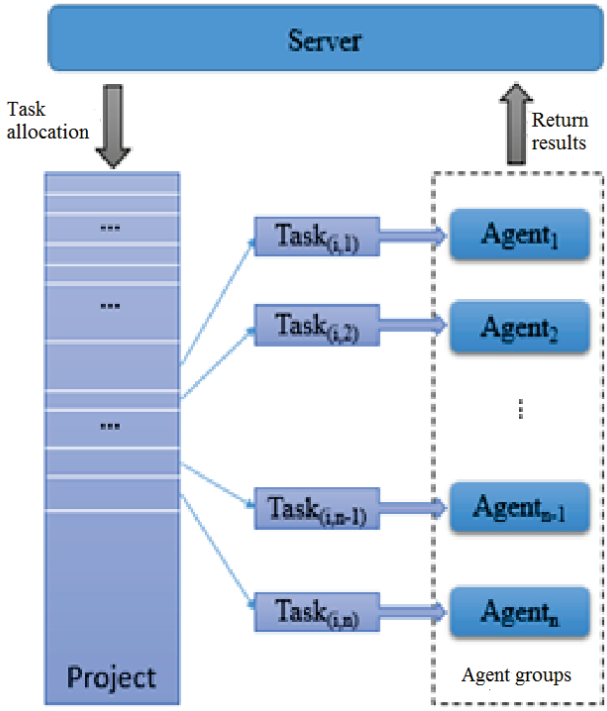

Fig. 3 The diagram of the *n*-th task allocation

The pseudo-code is as follows:

```
Begin:
Initial agents management list, Agent total numbers n=0, Agent group number K, task allocation times i=1;
While(Response == true)
    n ++;;
For each agent_j ∈ UseList
    Switch(agent_j.Time)
    case t0 + T:
        Group1.add(agent_j);
        break;
    case t0 + 2T:
        Group2.add(agent_j);
        break;
        ...
    case t0 + KT:
        Groupk.add(agent_j);
        break;
  While(!Result && !Project)
   IF(i = 1)
        For each agent_j ∈ UseList
            Run task(1, j)
            Return t(1, j), Vmax(1, j),Vaverage(1, j),Result;
        For server
            V(1, j) = w1 * Vaverage(1, j) + w2 * Vmax(1, j)
            task(2, j) = T * V(1, j)
    Else IF(i!=1)
        For each agent_j ∈ UseList
            Run task(i, j);
            Return t(I, j), Vmax(i, j),Vaverage(i,j), Result;
        For server
            V(i, j) = w1 * Vaverage(i, j) + w2 * Vmax(i, j)
            task(i+1, j) = T * V(i, j)
```


Break;
End

3. Experimental

HCEP (HashCatEnigma Project, password recovery project) is run on the algorithm volunteer computing platform. The volunteer computing model will be exploited to achieve password recovery. HCEP is a project with a large amount of calculation, but simple unit task calculation, which is just suitable to volunteer computing.

3.1 Illustrations of experimental data

HCEP can recovery password by the way of using a dictionary. And dictionary decoding is a violence way. Each Agent is equipped with a dictionary. Assigned to a task, agents only need to complete the tasks in their scope. For individual, the task to complete is of a small amount. Of course, a project will be easily finished as long as there are more volunteers to join into the project. In this experiment, the size of the task refers to the number of password to attempt.

In this paper, the experiment is divided into three parts:
(i) Data collection experiments from agent tasks,
(ii) Comparison to static allocation algorithm,
(iii) Comparison to the existing computing platform.
The above experiments will be carried out on the voluntary computing platform based on online active period.

3.2 Results and analysis

3.2.1 Data collection experiments from agent tasks

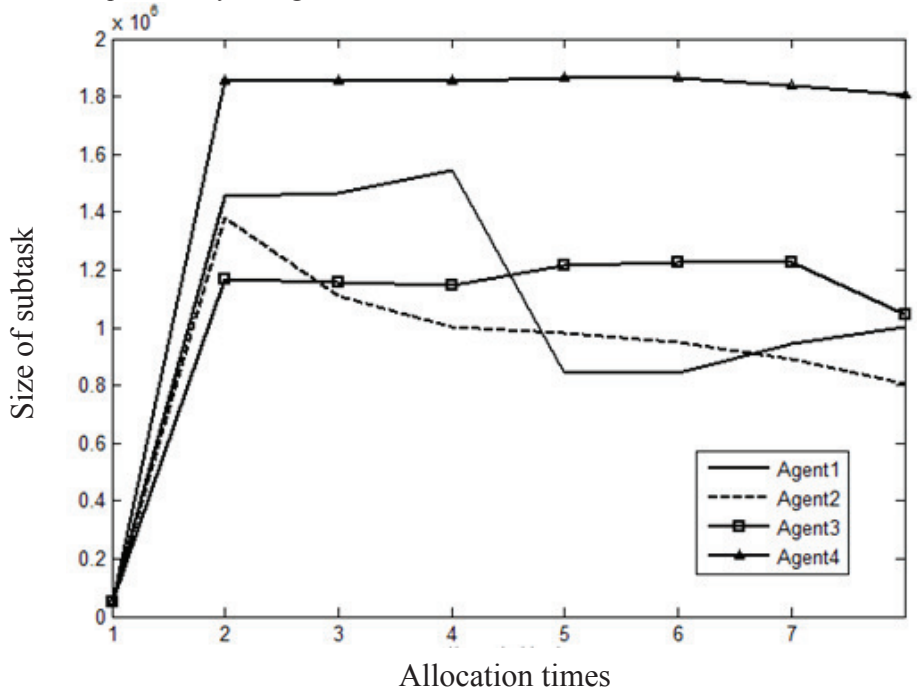

Fig. 4 Diagram of tasks for agents

In this part, the server performs the recovery of passwords, and records the subtasks size of task allocation for eight times from four agents in the decoding process, as shown in Fig. 4. Due to the performance difference between the volunteer computers, the sizes of the task to complete by Agent are also different. From the results in Fig. 4, agent 4 complete more tasks by virtue of excellent performance of GPU.

3.2.2 Comparison to static allocation algorithm (SAA)

For SAA, size of subtasks assigned to each agent is the same. Thus agent performance and idle resources will greatly affect the completion time of assigned task in the task execution process. However, each selected project with different size results in different time. As shown in Fig. 5, the performance of SAA in password recovery task for six times is universal inferior to the task allocation algorithm (TAA) with weighted average velocity based on online active period proposed in the work. It can be seen that the time spending on password recovery is shorter when agent has more idle resources. It is concluded that TAA with weighted average velocity based on online active period studied in this paper performs better and more efficiently than SAA.

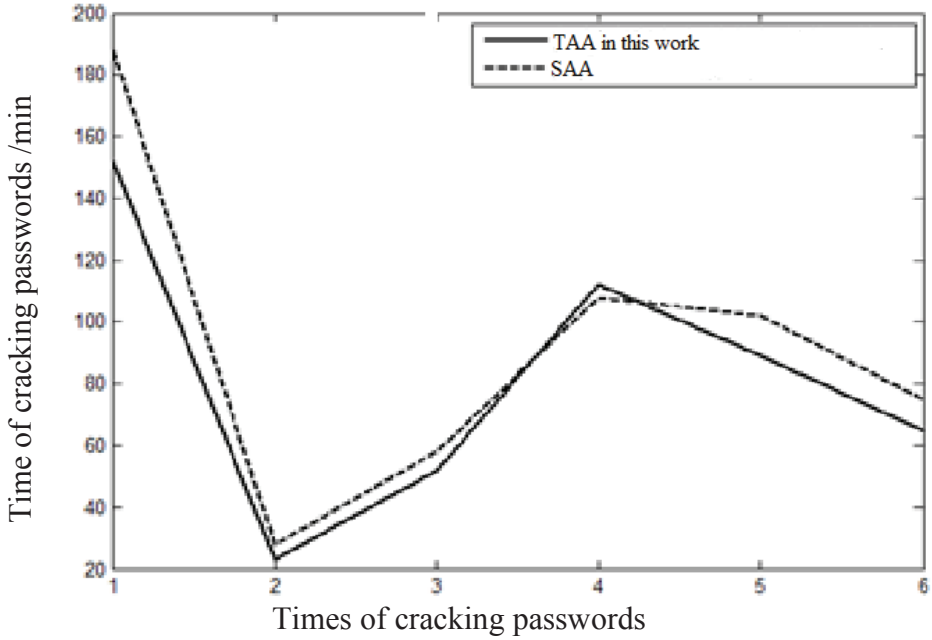

Fig. 5 Comparison of SAA and TAA in this work

3.2.3 Comparison to the existing computing platform on password recovery

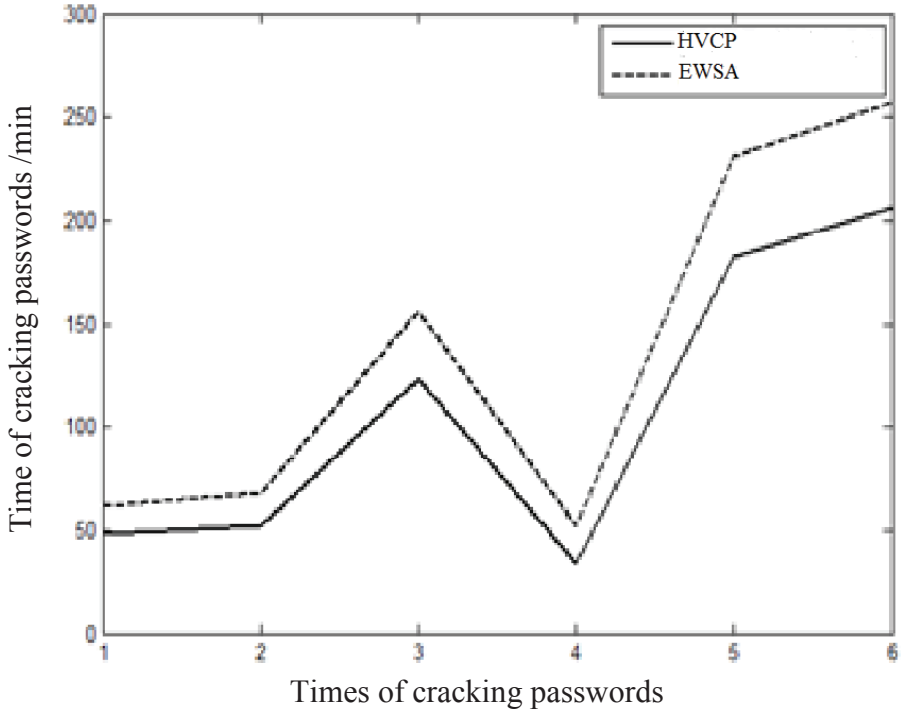

Fig. 6 Comparison of EWSA and HVCP built by TAA

EWSA (Elcomsoft Wireless Security Auditor) is a software on Wifi password recovery produced by Elcomsoft company. The experiment is carried out to compare EWSA and heterogeneous volunteer computing platform (HVCP) [11] built by TAA proposed in the study on to recovery the same password. Fig. 6 presents the data collected in this experiment. It can be found that computational efficiency of TAA with weighted average velocity based on online active period on heterogeneous computing platform is obviously superior to one of EWSA. As can be seen from the Fig. 6, the greater the amount of task project, the longer the time required.

4. Summary

Volunteer computing relying on a large number of personal computers and other resources provides a powerful computing capability for scientific research unable to be finished by using other ways. In order to fully and more effectively use these computing resources to complete more tasks in a limited time [12], task allocation algorithm with a weighted average velocity based online active period is proposed in present work. The task allocation algorithm (TAA) based on weighted average velocity is our previous work, the merits of which are that tasks sizes are assigned according to the performance of volunteer computer and tasks are performed within the unit of time. To extend our research, we consider that grouping of the computer by online time is favor of strengthening management, and avoiding that the task is assigned to those computers coming off the line, which not only saves time and increases the efficiency of volunteer computing.

Acknowledgments

This study was funded by Natural Science Research Project of Yancheng Teachers University (No.10YCKL032), and Jiangsu Province College Students' Innovative Training Project (201410324008Z). All authors declare no financial/commercial conflicts of interest.

References

[1] Fang B, Chen L, Wang H, Dai S, Zhong Q. Research on multirobot pursuit task allocation algorithm based on emotional cooperation factor. Thescientificworldjournal. 2014;2014:864180-.
[2] Sander PV, Peleshchuk D, Grosz BJ. A Scalable, Distributed Algorithm for Efficient Task Allocation. Proceedings of Aamas'. 2010:1191--8.
[3] El-Ghamrawy SM, El-Desouky AI, Saleh AI. Distributed Task Allocation in Multi-Agent System Based on Decision Support Module. 2010.
[4] Zeng W, Aimin YU. Research on improved PSO algorithm based task allocation. Computer Engineering & Applications. 2013.
[5] Fang B, Wang Z, Chen L, Wang H. Research on pursuit task allocation algorithm of emotional robot based on personality. Chinese Automation Congress2015.
[6] Hao YL, Liu ZP. Fusion algorithm of angular velocity weighted averages for GFSINS based on dynamically allocating weights. Journal of Chinese Inertial Technology. 2010.
[7] Sun B, Luo W, Li J. Study on Methods for Extracting Comprehensive Information Based on the Sequential Dynamic Weighted Fusion. Lecture Notes in Electrical Engineering. 2011;102:385-91.
[8] Yang P, Wang T, Liu W, Liu H. Research on multi-task allocation algorithm based on Agent. IEEE International Conference on Software Engineering and Service Science2014. p. 276-8.
[9] Tao XL, Zheng YB. Multi-agent Task Allocation Method Based on Auction: Springer Berlin Heidelberg; 2010.
[10] Wu Q, Zhang R, He Y, Wang A, Ju Y, Wu Q, et al. The Research of Multi-Agent System Task Allocation Based on Auction. Proceedings of the International Conference on Computer Networks & Communication Engineering. 2013;30:214-7.
[11] Christensen H. A Bayesian formulation for auction-based task allocation in heterogeneous multi-agent teams. Proc Spie. 2011;8047:804710--11.
[12] Shao B, Yan Z. Research of Task Allocation Strategy for Moving Image Matching Based on Multi-agent: Springer Berlin Heidelberg; 2015.