A Bipartite Graph Matching Algorithm in Human-Computer Collaboration

Junfeng Man\textsuperscript{a}, Longqian Zhao\textsuperscript{a}, Ming Liu\textsuperscript{a}, Cheng Peng\textsuperscript{a}, and Qianqian Li\textsuperscript{b,\ast}

\textsuperscript{a}School of Computer Science, Hunan University of Technology, Zhuzhou, 412007, China
\textsuperscript{b}School of Business, Hunan University of Technology, Zhuzhou, 412007, China

Abstract

The emergence of human-machine collaboration has adapted to the requirements of big data for high performance computing and complex artificial reasoning, which uses the huge Internet user group and cluster to deal with the increasingly complicated data altogether. In this paper, a bipartite graph matching strategy is proposed to solve the problem of how the crowd and the cluster can collaborate effectively to complete the large data task. The Hopcroft-Karp algorithm of bipartite graph matching not only enhances and extends the Hungarian algorithm, but also considers the field of adaptive segmentation tasks, the degree of association, and the evaluation of the background and ability of the crowd to maximize the matching between the crowd and the segmented task group. The algorithm calculates each influence factor after each match and optimizes the next match, making the best match between the crowd and the task. Through the experiment, the accuracy of the task completion is verified to be the highest.

Keywords: big data; human-machine collaboration; bipartite graph matching algorithm; Hopcroft-Karp algorithm

(Submitted on July 3, 2018; Revised on August 10, 2018; Accepted on September 15, 2018)

© 2018 Totem Publisher, Inc. All rights reserved.

1. Introduction

Due to the big data era, individual computers cannot effectively deal with the rapid growth of data. The current solutions of big data tasks mainly depend on machine learning, distributed computing, etc. However, the large-scale data, low value density, and diversity of features impact current computer technology. In order to better solve the challenges of big data, more and more researchers have invested in the research of big data. The associate professor Li Guoliang and his team at Tsinghua University put forward the concept of collaboration between the crowd and the cluster \[1-2\], analyzed the characteristics of clusters and groups, and studied the computing power of the machine and the cognitive reasoning capabilities of the human brain, enabling the human-machine to complete complex tasks through collaboration under the big data environment \[3\].

The task of distribution of tasks must be involved in the collaboration between the crowd and the cluster, and tasks involving calculations should be handled by the computer. However, the task of how cognitive tasks can be effectively assigned to the crowd is a problem that needs to be solved.

There are many algorithms for matching, but not all matching algorithms are suitable for task matching in human-machine collaboration. For example, the random matching algorithm is too random, although there may be excellent matching, and there may be extremely poor conditions. The instability is not suitable for the match between people and tasks in human-machine collaboration \[4\]. With the ladder matching algorithm \[5\], the number of tasks that can be matched at the same time is not more than \(1w\) in general; otherwise, its performance will plummet. It is suitable for character matching in games, but it is not suitable for matching characters and tasks in human-machine collaboration. The violence matching algorithm \[6\], also known as the naive string matching algorithm, is suitable for matching the same things such as strings, but not for relevance matching between characters and tasks. The bipartite graph matching algorithm is used in this paper and is well adapted to the matching of characters and tasks in human-machine collaboration.

\ast Corresponding author.

E-mail address: 25716332@qq.com
The task assignment algorithm in human-machine collaboration is mainly composed of several modules in Figure 1. The task modeling, quality assessment module, and task release/collection module are complementary to each other. The task release/collection module is used to publish tasks and collect feedback results of tasks. The mechanism adaptively cuts the task into multiple sub-tasks and matches the task to the people using a matching mechanism. For the matching algorithm in the matching mechanism, we proposed the bipartite matching algorithm to solve the blindness and randomness caused by random matching of tasks [7]. The bipartite graph is a formal theoretical model based on graphs proposed by computer scientist Robin Milner. The purpose of this model is to provide a platform for pervasive computing [8]. The bipartite application is very extensive, such as element search, tree and graph traversal, etc. It is also well suited to the matching of tasks and people in human-machine collaboration [9].

As shown in Figure 2, the better credibility evaluation and self-adaptive task sets are regarded as two sets of elements U and V, and matching points in U and V are searched by the even-image matching algorithm for maximum matching to assign the task to the right people and ultimately achieve higher quality feedback [10-11]. This matching algorithm has more complexity than the random matching algorithm, but the matching effect is improved.

2. The Bipartite Graph Structure

**Definition 1** (The bipartite graph) The bipartite graphs is also known as “bipartite graphs”, that is to say, the element set V in an undirected graph G <V, E> is divided into two groups U1 and U2, which satisfies U1 ∪ U2 = V, and U1 ∩ U2 = Ø. E is the edge set of G, where two endpoints of either side of E belong to U1 and the other belongs to U2. The graph G can also be expressed as G<U1, U2, E>. In human-machine collaboration, U1 is viewed as a cluster of people for credibility evaluation, U2 as a set of tasks after adaptive cutting, and E as the matching edge of human clusters and task sets.

**Definition 2** (The maximum matching and the perfect matching) In the even graph, the matching is a set of edges, and no two edges have common vertices. This means that after a large task is cut into multiple small tasks, the evaluated person in the crowd can only match one of them to improve the efficiency of task completion. As shown in Figure 3, initially, each user can be matched with multiple small tasks as in part (a), and the user with professional background and high credibility can be matched with related tasks through bipartite graph algorithm. The matched users and tasks are no longer matches with other users or tasks. For example, after U1 and V1 in part (b) are matched, all other edges associated with U1 and V1 are useless edges. All matchings of all small tasks divided into one large task after adaptive cutting contain the matching
with the largest number of matching edges and are called the maximum matching of even images. For example, each element in the U set in part (c) can find one in the V set. The element is associated with it, the maximum number of matching edges is four, and the two ends of the matching edge are U set and V set respectively. If the users participating in the matching after the credibility evaluation are able to correspond to the adaptively cut small task sets, that is, each element in the U set and V set is a matching point, the matching is called perfect matching. Obviously part (c) is also a perfect match.

Figure 3. A bipartite graph matching process

What is the relationship between the user and the task in the even image matching process? This requires us to model the task. In the bipartite graph matching process, it ensures that users and task users can obtain the user’s professional background, knowledge reserve ability, etc. after a trusted evaluation. A task is divided into many different small tasks after adaptive segmentation. They may belong to different fields, but they may also belong to similar fields. Matching users and tasks with the same or similar fields may even map matching. The quality of the tasks completed by the user can also be used as a reference for the next task matching. Such a virtuous cycle will make matching. Is the effect improving? This requires us to model the task. Therefore, it is necessary to establish the degree of correlation between the small tasks after the same task is cut, and the degree of correlation between the tasks depends on the field of the task and describes the degree of similarity. The above two factors can be combined to form Equation (1).

\[ TS = \theta \times AS + \varphi \times SS \]  

(1)

Where \( \theta \) and \( \varphi \) are based on the dataset, \( TS \) is the degree of association between divided mini-tasks, \( AS \) is the task domain similarity, and \( SS \) is the task description similarity. The \( AS \) takes the values 0, 0.5, and 1 according to the overlapping degree of the domain and abstracts the description of the task into a set. The description can be used to determine the size of the \( SS \) through the intersection of the small tasks to describe the set according to keywords, topics, and other abstractions. As shown in Figure 4, Task1 and Task4 belong to two different domains, so the domain similarity between them is 0, and the domain similarity between Task3 and other tasks is 0.5. The above constitutes the correlation degree of the edges on the graph, thus establishing the relationship between tasks, so that the task can be more effectively matched to users with high task relevance in bipartite graph matching.

3. The Bipartite Graph Matching Principle

There are two factors that need to be considered in the bipartite graph matching process. One is the cost of the user to complete the task, \( M \), and the other is the increment value of the user after the task is completed, \( H \). The biograph graph matches the user of the cost \( M \), and the incremental value \( H \) is used to match the next user with the task, given by Equation
(2) and Equation (3).

\[ H = \varphi \cdot \sum_{x=1}^{X}EI(T_x) + \omega \cdot \sum_{y=1}^{Y}EI(RT_y) \]  

(2)

\[ M = \sum_{x=1}^{X}a_x + \sum_{y=1}^{Y}b_y \]  

(3)

The increment value \( H \) is composed of two parts: one is the expected amount of revenue \( EI(T_x) \) for the user to complete the task, and the other is the user to complete the same field and associated expected amount of revenue \( EI(RT_y) \) with the task, where \( T \) represents the matched task and \( RT \) represents a task that is associated with task \( T \) but has not been matched or completed. The expectations that the user obtains vary from task to task, and the parameters \( \varphi \) and \( \omega \) are determined by the user amount, the task amount, and the dataset. The task cost \( M \) is also composed of two parts: one is the task difficulty factor \( a_x \), and each task has a different degree of difficulty. The degree of difficulty is determined by the field in which the task is located, the user evaluation, and the amount of completion, and the other is the completion of the user. The task domain factor is \( b_y \). The task may only involve a field or may also involve multiple fields. From the point of view of task separation, the smaller the field involved in the small task, the more thorough the division, so the value of \( b_y \) is related to the involved field. The user’s incremental value \( H \) is changed from task to task, and we can get the expected mission revenue for the user to complete a specific task, given by Equation (4).

\[ EI(T) = E_\varphi \left[ \log \int \frac{p_r(x|\theta)}{p_r(x)} p_r(x|\theta)dx \right] \]  

(4)

\( EI(T) \) indicates the expected amount of revenue obtained by the user for completing a specific task \( T \), where \( x \) indicates the task variable, \( \theta \) indicates the domain area, and \( p_r(x|\theta) \) indicates the qualified quality of the task completed by the user. It is a criterion for the completion of the task. The higher the quality of the tasks performed by the user, the greater the value of \( p_r(x|\theta) \). \( p_r(x) \) represents the probability of the standard given by the task itself, what is expected by the task publisher.

The bipartite graph matching algorithm firstly considers the task situation and then searches for matching users. After the task is adaptively cut, we can obtain the task domain and associated task conditions so that the user group’s field and professional background can be determined. The evaluated population can select tasks in related fields to perform even map simulation and matching to calculate the cost required to complete the task. The bipartite graph matches the expected incremental value obtained by the user in the past to accomplish similar tasks in the same field with the appropriate task, thereby improving the quality and efficiency of the task completion.

4. The Bipartite Graph Matching Algorithm (Hopcroft-Karp Algorithm)

The bipartite graph matching in human–machine collaboration mainly uses the Hopcroft-Karp algorithm, which is the inheritance and extension of the Hungarian algorithm. Therefore, it is necessary to understand the principles and characteristics of the Hungarian algorithm.

**Definition 3** (Alternate matching path) Starting from an unmatched user, by means of multiple virtual matches, the path formed by the unmatched edge and the matched edge in turn is called an alternate matching path.

**Definition 4** (Augmented matching path) Similar to the alternate matching path, it starts from an unmatched user and takes the alternate path. If there is another unmatched point on the way except for the starting point, then this special alternate matching path is called the augmented matching path. As shown in Figure 5(a), it is an alternate matching path taken during matching of a user and a task, and (b) is an augmented matching path extracted from (a); starting from the unmatched user of V4, the non-matching edge and the matching edge alternately cycle through the path, and the points encountered halfway through are matching points until an unmatched point U2 is encountered to form an augmented matching path. It is not difficult to see that the non-matching edge ratio is matched in the augmented matching path. The edge is one more hop, and this is also an important feature of the augmented matching path. If the matching edge and the non-matching exchange are used, then the alternate matching path will have one more matching edge, so that there will be one more matching edge without destroying the even image matching. We study augmenting matching to improve
In the matching process, even maps are converted from tree matches to tree matches, that is, Hungarian trees. The Hungarian tree starts from an unmatched user, alternately matches the road, performs augmented matching, and continuously searches for an augmented matching road until no unmatched point can be found. As shown in Figure 6, Figure 6(b) is a tree converted from Figure 6(a). There is a non-matching leaf node 7 in the converted tree, and the Hungarian tree requires that all leaf nodes must be matching nodes. The tree is not a Hungarian tree. Because node 7 is a non-matching node, node 7 can be removed and then matched. Figure 6(a) is converted to Figure 6(c). All nodes except node 2 are matching nodes, and node 2 is the root node and converted to a Hungarian tree. Similar to the breadth-first search (BFS) tree, all leaf nodes of the transformed tree are matching nodes, and matching at this moment reaches the maximum match.

Figure 6(c) illustrates the Hungarian matching process: (1) Initial inspection, virtual matching. Taking node 1 as an example, during the initialization, the user (node 1) acquires professional backgrounds and areas of expertise and finds that the adaptively-cut small tasks (nodes 7 and 9) satisfy the matching conditions and records information, as do other nodes. (2) Initial match. Match node 1 and node 7 according to the weights. (3) Add node 3 and find that node 3 can match nodes 7 and 8. According to conditions, it is found that node 7 is more suitable, but node 7 is already a matching node and is matched with node 1, and find the alternate path of node 3, 3-7-1-9, Remove the edges 1-7 that were already on the match on the alternate path, and add the remaining edges 3-7, 1-9 to the matching edges. (4) Repeatedly process other nodes to complete all the matches.

The Hopcroft-Karp algorithm is based on this. The Hungarian algorithm is dominated by alternating matching paths, and only one augmented matching path can be found at a time. The Hopcroft-Karp algorithm uses this feature of the Hungarian algorithm and proposes to find more times at a time. It increases the number of matching routes to increase multiple matches, continuously improves matching, and improves matching efficiency. From the point of view of time complexity, the time complexity of the Hungarian algorithm is $O(V \cdot E)$, and the time complexity of the Hopcroft-Karp algorithm is $O(\sqrt{V \cdot E})$. As the amount of matching data increases, the advantages become more obvious.

5. Experimental Results and Analysis
In order to understand the matching accuracy and matching effect of Hopcroft-Karp algorithm, the random matching algorithm and Hungarian algorithm were respectively used as experimental objects reference. Considering that the crowdsourced platform does not support even-image matching algorithms, we can only use simulated iterative experiments under specific experiments offline. The following simulation experiments and real experiments were conducted to test the effect of the Hopcroft-Karp algorithm.

5.1. Simulation Experiment

In the simulation experiment, we used three themes: drug theme, music theme, and computer equipment theme. The number of tasks involved in the three themes reached \( n = 1200 \), 400 tasks per theme. Suppose there are 90 users participating in the trial, and each user is good at one of the domain topics, then the accuracy of the user's response to the topic of good subject to Gaussian distribution, and the accuracy rate of data sampling is from \( \text{gauss}(0.7, 0.3) \).

At the beginning of the experiment, the task with high correlation is adaptively divided into small tasks with low coupling. Because the selected topic is clear, there is no domain relevance of 0.5, the relevance of different background tasks is 0, and the same background tasks are associated degree is 1. In the experiment, the task was divided into 20 batches, with 60 tasks in each batch. During the task distribution process, the difficulty of the task gradually decreased. The task uses a random allocation algorithm, a Hungarian algorithm, and a Hopcroft-Karp algorithm to perform 1000 matches, collects the completion of each task, and finally takes the average accuracy.

As shown in Figure 7, the mission difficulty is according to beta distribution \( \beta(a, b) \), where \( (a, b) \) takes values \( (2, 1), (4, 1), (7, 1), (10, 1) \) respectively. The value of \( a \) from 2 to 10 indicates that the mission is difficult to achieve. The ordinate of the graph is the average accuracy obtained by repeating 1000 matches in the experiment. From the figure, we can see that the tasks are randomly assigned as the difficulty of the task is reduced, and the average accuracy rate is improved, but it is not obvious. Random allocation does not consider the field and background of the task, nor does it take into account the user’s professionalism and ability. The difficulty of the task has little effect on random allocation. The Hungarian algorithm and Hopcroft-Karp algorithm are greatly affected by the beta distribution, because the two algorithms need to consider the task area and associated conditions and the user's trusted evaluation situation in order to calculate the expected amount of revenue \( E(T) \). As the difficulty of the task is reduced, it is easier for the user to complete the task of the benefits, the cost is smaller, and thus the average accuracy of the completion of the task is higher. Since the Hopcroft-Karp algorithm is complementary and extended to Hungary, it does not have much impact on the quality of matching, and the number of experiments is higher, so the average accuracy of the two is basically the same.

The accuracy rate of the task completed by the user is observed by a specific \( \beta \) value. Since the average accuracy of the Hungarian algorithm and Hopcroft-Karp algorithm is basically constant, we only compare the average accuracy of the random distribution algorithm and the Hopcroft-Karp algorithm. Figure 8 is a case where 20 rounds of task matching are performed by sampling from \( \beta(4, 1) \). It can be seen from the figure that when using the random allocation algorithm matching, the accuracy of the task completed by the user is 40% on average, and when using Hopcroft-Karp with an increase in the number of rounds, the algorithm has a slight improvement in the accuracy. In the first round, the average accuracy rate is low because the cost of the front-end user's task \( M \) cannot be obtained initially. However, there are many factors to consider, and the matching accuracy is better than the random distribution. In the second matches, there is the \( M \)
value of the front wheel, and the average accuracy rate has a significant increase. Because Hopcroft-Karp has continuously optimized results, the matching effect will be better as the number of rounds increases.

Figure 8. The effect of rounds on matching

5.2. Real Experiment

The real dataset comes from a crowdsourced platform. We take three types of datasets, the medicines dataset, the English dataset, and the law dataset, from a crowdsourcing platform to test the validity of the Hopcroft-Karp algorithm. For the medicines dataset, we collected questions about drug names, drug knowledge questions and answers, etc. For the English dataset, we collected vocabulary questions, grammar questions, and other related English questions. For the law dataset, we collected legal knowledge questions. Each mission experiment was conducted three times. The principle of voting was adopted for tasks with ambiguous answers (the minority was subject to the majority). Figure 9 is the experimental rendering of the Hopcroft-Karp algorithm.

Figure 9. The experimental rendering of the Hopcroft-Karp algorithm

The effect of the three datasets in the three algorithms can be seen in Figure 9. The effect of the random assignment algorithm is related to the quality of the population. For example, the popularity of English makes the random match algorithm match the average accuracy of the task and the task. The average accuracy of the other two datasets is higher, but due to their randomness, the average accuracy obtained is not very high. The Hungarian algorithm and Hopcroft-Karp algorithm have achieved more than 90% accuracy, close to 100%, and the matching effect is very good.

The Hungarian algorithm and Hopcroft-Karp algorithm match quality are similar. Figure 10 is the difference between the time spent by the two algorithms in the case of the amount of data increasing from 1,000 to 10,000. It can be seen from the figure that with an increase in data volume, the Hopcroft-Karp algorithm matching takes less time than Hungarian algorithm matching, which is suitable for task matching under large data volume, and human-computer collaboration in our
group computing is suitable for big data and large crowds.

6. Conclusion

This paper proposes a method for matching people and tasks based on even graphs. The Hopcroft-Karp algorithm fully considers the characteristics of the task area and the crowd, achieves efficient matching between people and tasks, and improves the quality and efficiency of task completion. It has made important contributions to the efficient assignment of man-machine collaboration tasks. In the next study, we will consider the characteristics of the population and the division of tasks from more aspects to improve the matching effect between people and tasks and also continue to improve the efficiency of matching, so that the human-machine collaboration can meet more variable challenges.

Acknowledgement

This paper is supported by Natural Science Foundation of China (No. 61871432, No. 61702178), The Natural Science Foundation of Hunan Province (No. 2018J1063, 2017JJ3065, 2016J5035, 2016J5036), The Major Project of Hunan Province Education Department (No.16A059, 17A052), The Excellent Youth Project of Hunan Province Education Department (No.16B071).

References


Junfeng Man is a professor, Master’s tutor, executive member of the Changsha Computer Center of the China Computer Society, and chairman of the Changsha sub-forum of the China Association of Computer Science and Technology Youth Forum. Main research directions are industrial big data, industrial internet, and networked software.

Longqian Zhao is a master’s student in the School of Computer Science and Technology at Hunan University of Technology. His research interests include industrial big data, industrial internet, and networked software.
Ming Liu is a Master’s student in the School of Computer Science and Technology at Hunan University of Technology. His research interests include industrial big data, industrial internet, and networked software.

Cheng Peng received his M.E. and Ph.D. degrees from the School of Information Science and Engineering at Central South University, Changsha, China. His current research interests include networked software, computer networks, and software engineering.

Qianqian Li received a Master’s degree in Computer Applied Technology. Main research directions are computer education, intelligent information processing, and educational psychology.