A Reinforced Hungarian Algorithm for Task Allocation in Global Software Development

Xiao Yu
State Key Lab. of Software Engineering,
Computer School, Wuhan University,
Wuhan, China
xiaoyu_whu@yahoo.com

Man Wu
School of Computer Science and Information Engineering,
HuBei University,
Wuhan, China
wuman_1@qq.com

Xiangyang Jia*
State Key Lab. of Software Engineering,
Computer School, Wuhan University,
Wuhan, China
Corresponding author email: jxy@whu.edu.cn

Ye Liu
School of Computer Science and Information Engineering,
HuBei University,
Wuhan, China
liuye061@qq.com

***Abstract*—The allocation of software development tasks is a critical management activity in distributed development projects. One of the most important problem is to find the lowest-cost way to assign tasks in global software development, which can be solved by Hungarian algorithm. However, the original Hungarian algorithm only assume that a task can only be solved by one development site. The assumption is not agreed with the actual case where a software development task is usually be solved through a collaboration among several sites. To address such an issue, this paper proposes a reinforced Hungarian algorithm (RHA) for task assignment in global software development. RHA consists of three major stages. First, RHA transforms a *n*×*m* cost matrix into two *n*×*n* cost matrix by adding (2*n*-*m*) virtual development sites. Second, RHA performs the original Hungarian algorithm on the two *n*×*n* cost matrix to get the optimal assignment results. Finally, RHA removes the (2*n*-*m*) virtual development sites and gets the final optimal assignment result for *m* tasks. Simulation results indicate that RHA is a viable approach for the task assignment problem in global software development.**

Keywords—task allocation; global software development; Hungarian algorithm

I. INTRODUCTION

As global and distributed software development is becoming a norm in the software industry [1-2], a tight budget and a shortage of resources and time have motivated many software enterprises to start looking for outside partners. With the advent of big data era [3-4] and the development of network technology [5-6], known as global software development (GSD), the allocation of software development tasks over several sites, which may be spread across different countries, has become a common practice in industrial software engineering [7-8].

DOI reference number: 10.18293/SEKE2017-042

Software development task assignment is one of the core steps to reasonably allocate limited development resources and effectively exploit the capabilities of the development sites in global software development. However, task allocation is still one of the major challenges in global software development.

A large number of possible combinations for task assignment makes the process more complicated. For example, if a software development project consists of five different tasks which need to be assigned among four development sites, theoretically there are 4^5=1024 different combinations available for task assignment [9]. It is evident that evaluating all possibilities is impossible and it needs an approach to assign the task to the available development sites.

There are several possible strategies for allocating tasks that use different criteria for allocation.

Bass et.al [10] state that in practice, allocating task to low-cost countries has become a cost-saving strategy for many organizations. Therefore, one of the most important problem is to find the lowest-cost way to assign tasks in global software development, which usually be solved by Hungarian algorithm [11]. However, the original Hungarian algorithm solves the problem that the project manager allocates *n* tasks to *n* sites with the least total cost by performing all operations on a *n*×*n* cost matrix. It only assumes that the number of tasks is equal to the number of sites and a task can only be solved by one development site. The assumption is not agreed with the actual case where the number of tasks and the number of site are not equal in general, and a software development task is usually be solved through a collaboration among several sites.

To address such an issue, this paper proposes a reinforced Hungarian algorithm (RHA) for task assignment in global software development. The algorithm solves the problem that the project manager allocates *n* tasks to *m* sites at the least total cost by performing all operations on a *n*×*m* matrix. RHA

consists of three major stages. First, RHA transforms a $n \times m$ matrix into two $n \times n$ matrix by adding (m-n) virtual tasks. Second, RHA performs the original Hungarian algorithm on the two $n \times n$ cost matrix to get the optimal assignment for $2n$ tasks. Finally, RHA removes the (m-n) virtual tasks and gets the final optimal assignment for m tasks.

The effectiveness of RHA is demonstrated by comparing it with the expansion matrix method [12]. Simulation results indicate that RHA can get the optimal assignment with the lowest cost in global software development.

The remainder of this paper is organized as follows. Section II reviews the Hungarian algorithm. Section III describes the proposed reinforced Hungarian algorithm (RHA) for task assignment in global software development and Section IV shows the operational process of the algorithm by an example. Section V demonstrates the experimental results. Section VI presents the related work. Finally, Section VII addresses the conclusion and points out the future work.

II. HUNGARIAN ALGORITHM

In this section, we first give the problem definition. Then, we briefly review the Hungarian algorithm.

A. Problem definition

The usual assignment problem is defined as follows: assign n tasks to n development sites with the least total cost, if task i is assigned to site j with a non-negative integer cost c_{ij}. This problem is a special case of the linear programming problem, defined as follows:

$$\text{Min } S=\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \quad (1)$$

$$\text{Subject to } \sum_{i=1}^{n} x_{ij} = 1 \text{ (j=1, 2, ..., n)}, \quad (2)$$

$$\sum_{j=1}^{n} x_{ij} = 1 \text{ (i=1, 2, ..., n)}, \quad (3)$$

$$x_{ij}=1 \text{ or } 0. \quad (4)$$

where x_{ij}= 1 means task i is assigned to site j, otherwise x_{ij}=0, task i is not assigned to site j. The costs c_{ij} form a cost matrix, denoted C.

Theorem 1: If a number is added to or subtracted from all of the entries of any one row or column of a cost matrix C, then the optimal assignment for the resulting cost matrix C' is also an optimal assignment for the original cost matrix C.

Proof: Assume that u_i is subtracted from all of the entries in each row and v_j is subtracted from all of the entries in each column from the original cost matrix C. For the resulting cost matrix C', the problem is defined as follows:

$$\begin{aligned}\text{Min } S' &= \sum_{i=1}^{n}\sum_{j=1}^{n}(c_{ij}-u_i-v_j)\times x_{ij} \quad (5)\\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} - \sum_{i=1}^{n}\sum_{j=1}^{n} u_{ij}x_{ij} - \sum_{i=1}^{n}\sum_{j=1}^{n} v_{ij}x_{ij}\end{aligned}$$

Because $\sum_{i=1}^{n} x_{ij} = 1$ (j=1,2,3…,n) and $\sum_{j=1}^{n} x_{ij} = 1$ (i=1,2,3…,n),

then

$$\begin{aligned}\text{Min } S' &= \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}\times x_{ij} - \sum_{i=1}^{n} u_i - \sum_{j=1}^{n} v_j\\ &= \text{Min } S - \sum_{i=1}^{n} u_i - \sum_{j=1}^{n} v_j\end{aligned}$$

Since $\sum_{i=1}^{n} u_i + \sum_{j=1}^{n} v_j$ is a constant number, the optimal assignment for the resulting cost matrix C' is also an optimal assignment for the original cost matrix C

B. The Hungarian algorithm

The Hungarian algorithm applies the above mathematical model and theorem 1 to a given cost matrix to find an optimal assignment.

Step 1. Subtract the smallest entry in each row from all the entries of its row.

Step 2. Subtract the smallest entry in each column from all the entries of its column.

Step 3. Draw lines through appropriate rows and columns so that all the zero entries of the cost matrix are covered and the minimum number of such lines is used.

Step 4. Test for Optimality: (1) If the minimum number of covering lines is, an optimal assignment of zeros is possible and we are finished. (2)If the minimum number of covering lines is less than, an optimal assignment of zeros is not yet possible. In that case, proceed to Step 5.

Step 5. Determine the smallest entry not covered by any line. Subtract this entry from each uncovered row, and then add it to each covered column. Return to Step 3.

C. Example

A software development project consists of four different tasks: Task 1, Task 2, Task 3 and Task 4. There are four development sites: S1, S2, S3, and S4. They each demand different pay for various tasks. The problem is to find the lowest-cost way to assign the tasks. The 4×4 cost matrix of this problem is shown as follows.

$$\begin{bmatrix} 9.4 & 4.8 & 5 & 8.2 \\ 9.6 & 10 & 6.4 & 8.8 \\ 6.6 & 11 & 8 & 7.6 \\ 8.2 & 8.4 & 5 & 8.2 \end{bmatrix}$$

Step 1. Subtract 4.8 from Row 1, 6.4 from Row 2, 6.6 from Row 3, and 5 from Row 4. The resulting matrix is as follows.

$$\begin{bmatrix} 4.6 & 0 & 0.2 & 3.4 \\ 3.2 & 3.6 & 0 & 2.4 \\ 0 & 4.4 & 1.4 & 1 \\ 3.2 & 3.4 & 0 & 3.2 \end{bmatrix}$$

Step 2. Subtract 1 from Column 4. The resulting matrix is as follows.

$$\begin{bmatrix} 4.6 & 0 & 0.2 & 2.4 \\ 3.2 & 3.6 & 0 & 1.4 \\ 0 & 4.4 & 1.4 & 0 \\ 3.2 & 3.4 & 0 & 2.2 \end{bmatrix}$$

Step 3. Cover all the zeros of the matrix with the minimum number of horizontal or vertical lines.

$$\begin{bmatrix} \sout{4.6} & \sout{0} & \sout{0.2} & \sout{2.4} \\ 3.2 & 3.6 & 0 & 1.4 \\ \sout{0} & \sout{4.4} & \sout{1.4} & \sout{0} \\ 3.2 & 3.4 & 0 & 2.2 \end{bmatrix}$$

Step 4. Since the minimal number of lines is less than 4, we have to proceed to Step 5.

Step 5. Note that 1.4 is the smallest entry not covered by any line. Subtract 1.4 from each uncovered row. The resulting matrix is as follows.

$$\begin{bmatrix} 4.6 & 0 & 0.2 & 2.4 \\ 1.8 & 2.2 & -1.4 & 0 \\ 0 & 4.4 & 1.4 & 0 \\ 1.8 & 2 & -1.4 & 0.6 \end{bmatrix}$$

Now add 1.4 to each covered column. The resulting matrix is as follows.

$$\begin{bmatrix} 4.6 & 0 & 1.6 & 2.4 \\ 1.8 & 2.2 & 0 & 0 \\ 0 & 4.4 & 2.8 & 0 \\ 1.8 & 2 & 0 & 0.6 \end{bmatrix}$$

Now return to Step 3.

Step 3. Cover all the zeros of the matrix with the minimum number of horizontal or vertical lines.

$$\begin{bmatrix} \sout{4.6} & \sout{0} & \sout{1.6} & \sout{2.4} \\ \sout{1.8} & \sout{2.2} & \sout{0} & \sout{0} \\ \sout{0} & \sout{4.4} & \sout{2.8} & \sout{0} \\ \sout{1.8} & \sout{2} & \sout{0} & \sout{0.6} \end{bmatrix}$$

Step 4. Since the minimal number of lines is 4, an optimal assignment of zeros is possible and we are finished.

$$\begin{bmatrix} 4.6 & \boxed{0} & 1.6 & 2.4 \\ 1.8 & 2.2 & 0 & \boxed{0} \\ \boxed{0} & 4.4 & 2.8 & 0 \\ 1.8 & 2 & \boxed{0} & 0.6 \end{bmatrix}$$

Since the total cost for this assignment is 0, it must be an optimal assignment. Here is the same assignment made to the original cost matrix

$$\begin{bmatrix} 9.4 & \boxed{4.8} & 5 & 8.2 \\ 9.6 & 10 & 6.4 & \boxed{8.8} \\ \boxed{6.6} & 11 & 8 & 7.6 \\ 8.2 & 8.4 & \boxed{5} & 8.2 \end{bmatrix}$$

So we should assign Task 1 to Site 2, Task 2 to Site 4, Task 3 to Site 1, and Task 4 to Site 3.

III. RHA

In this section, we first give the problem definition. Then, we present our RHA method for task allocation in global software development.

A. Problem definition

The original Hungarian algorithm only assume that the number of tasks is equal to the number of sites and a task can only be solved by one development site and a task can only be solved by one development site. The assumption is not agreed with the actual case in global software development where the number of tasks and the number of site are not equal in general, and a software development task is usually be solved through a collaboration among several sites.

Therefore, the task assignment problem in global software development is defined as follows: assign n tasks to m development sites with the least total cost ($m>n$), if task i is assigned to site j with a non-negative integer cost c_{ij}. This problem is a special case of the linear programming problem, defined as follows:

$$\text{Min } S=\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \tag{6}$$

$$\text{Subject to } \sum_{i=1}^{m} x_{ij} = 1 \text{ (j=1, 2, ..., n)}, \tag{7}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \text{ (i=1, 2, ..., m)}, \tag{8}$$

$$x_{ij}=1 \text{ or } 0. \tag{9}$$

For this case, we need to find an optimal assignment given a $n\times m$ cost matrix.

Theorem 2: If a $n\times m$ matrix is transformed into two $n\times n$ matrix by adding ($2n$-m) virtual sites with the zero entries, then an optimal assignment for the resulting cost matrixes is also an optimal assignment for the original cost matrix.

Proof: For the resulting cost matrix *C'*, the problem is defined as follows:

$$\text{Min } S'' = \sum_{i=1}^{n}\sum_{j=1}^{2n} c'_{ij} \times x'_{ij} \tag{10}$$

$$\text{Subject to } \sum_{i=1}^{n} x'_{ij} = 1 \ (j = 1,2,\dots,2n) \tag{11}$$

$$\sum_{j=1}^{n} x'_{ij} = 1 \ (i = 1,2,\dots,n) \tag{12}$$

$$\sum_{j=n+1}^{2n} x'_{ij} = 1 \ (i = 1,2,\dots,n) \tag{13}$$

$$x'_{ij} = 1 \ or \ 0 \ \ (\text{i} = 1,2,\dots,\text{n};\text{j} = 1,2,\dots,2\text{n}) \tag{14}$$

$$\begin{aligned} \text{Min } S'' &= \sum_{i=1}^{n}\sum_{j=1}^{2n} c'_{ij} \times x'_{ij} \\ &= \sum_{i=1}^{n}\sum_{j=1}^{m} c'_{ij}\, x'_{ij} + \sum_{i=1}^{n}\sum_{j=m+1}^{2n} c'_{ij}\, x'_{ij} \end{aligned}$$

Because $c'_{ij} = 0 \ \ (i = 1,2,\dots,n; j = m+1, m+2, \dots, 2n)$

$$\begin{aligned} \text{Then Min } S'' &= \sum_{i=1}^{n}\sum_{j=1}^{2n} c'_{ij} \times x'_{ij} + 0 \\ &= \sum_{i=1}^{n}\sum_{j=1}^{2n} c'_{ij} \times x'_{ij} \end{aligned}$$

$$= \text{Min } S'$$

B. The RHA method

The following algorithm applies the above mathematical model and theorem 2 to a given $n \times m$ cost matrix to find an optimal assignment. Algorithm 1 presents the pseudo-code of our proposed RHA method.

Algorithm 1. RHA approach

Input: $n \times m$ cost matrix

Output: optimal assignment result

1. Subtract the smallest entry in each row from all the entries of its row;
2. Subtract the smallest entry in each column from all the entries of its column;
3. Transform $n \times m$ matrix into two $n \times n$ matrix by adding (2n-m) virtual sites with the zero entries;
4. Performs the original Hungarian algorithm on the two $n \times n$ cost matrix**;**
5. Removes the (2n-m) virtual development sites ;
6. **return** optimal assignment result;

IV. EXAMPLE

In this section, we illustrate the execution of RHA by an example.

Example 1: A software development project consists of four different tasks: Task 1, Task 2, Task 3 and Task 4. There are six development sites: S1, S2, S3, S4, S5, and S6. They each demand different pay for various tasks. The problem is to find the lowest-cost way to assign the tasks. The 6×8 cost matrix of this problem is shown as follows.

$$E_{ij}=\begin{bmatrix} 9.4 & 4.8 & 5 & 8.2 & 12 & 7.4 \\ 9.6 & 10 & 6.4 & 8.8 & 8.8 & 9 \\ 6.6 & 11 & 8 & 7.6 & 7.6 & 6.6 \\ 8.2 & 8.4 & 5 & 8.2 & 6.4 & 8.6 \end{bmatrix}$$

Step 1. Subtract 4.8 from Row 1, 6.4 from Row 2, 6.6 from Row 3, and 5 from Row 4. The resulting matrix is as follows.

$$\begin{bmatrix} 4.6 & 0 & 0.2 & 3.4 & 7.2 & 2.6 \\ 3.2 & 3.6 & 0 & 2.4 & 2.4 & 2.6 \\ 0 & 4.4 & 1.4 & 1 & 3.4 & 0 \\ 3.2 & 3.4 & 0 & 3.2 & 1.4 & 3.6 \end{bmatrix}$$

Step 2. Subtract 1 from Column 4, and 1.4 from Column 5. The resulting matrix is as follows.

$$\begin{bmatrix} 4.6 & 0 & 0.2 & 2.4 & 5.8 & 2.6 \\ 3.2 & 3.6 & 0 & 1.4 & 1 & 2.6 \\ 0 & 4.4 & 1.4 & 0 & 2 & 0 \\ 3.2 & 3.4 & 0 & 2.2 & 0 & 3.6 \end{bmatrix}$$

Step 3. Transform 6×4 matrix into two 4×4 matrix by adding 2 virtual sites with the zero entries. The resulting matrixes are as follows.

$$\text{The first matrix:} \begin{bmatrix} 4.6 & 0 & 0.2 & 2.4 \\ 3.2 & 3.6 & 0 & 1.4 \\ 0 & 4.4 & 1.4 & 0 \\ 3.2 & 3.4 & 0 & 2.2 \end{bmatrix}$$

$$\text{The second matrix:} \begin{bmatrix} 5.8 & 2.6 & 0 & 0 \\ 1 & 2.6 & 0 & 0 \\ 2 & 0 & 0 & 0 \\ 0 & 3.6 & 0 & 0 \end{bmatrix}$$

Step 4. Performs the original Hungarian algorithm on the two 4×4 cost matrix. The assignment result of the first cost matrix is as follows:

Task1--->S2

Task2--->S4

Task3--->S1

Task4--->S3

The assignment result of the second cost matrix is as follows:

Task1--->S7

Task2--->S8

Task3--->S6

Task4--->S5

Step 5.Removes the 2 virtual development sites and output the final assignment result:

Task1--->S2

Task2--->S4

Task3--->S1,S6

Task4--->S3,S5

V. EXPERIMENTS

In this section, we evaluate our proposed reinforced Hungarian algorithm (RHA) for task assignment empirically. We first introduce the performance measures. Then, in order to investigate the performance of RHA, we perform some empirical experiments.

A. *Performance measures*

In the experiment, we employ one commonly used performance measures, i.e, successful allocation rate. It is defined and summarized as follows.

- Successful allocation rate (SAR) is the measure of tasks that are successfully allocated to development sites. The SAR=M_s/M, where M_s is the number of cost matrixes that are successfully allocated and M is the number of all cost matrixes. The higher the value of successful allocation rate, the more effective the algorithm is.

B. *Experimental results*

In this experiment, we analyze the effectiveness of RHA with different number of development sites. Here we fix the number of tasks to 4 and evaluate the performance of the evaluated algorithms by increasing the number of development sites from 6 to 12 with an increment of 2. Therefore, we generated four sets of matrix data, which contains 100 *4×6* matrixes, 100 *4×8* matrixes, 100 *4×10* matrixes and 100 *4×12* matrixes, respectively.

Result Analysis: As we can see in Fig.1, the successful allocation rate of RHA is always 1. The successful allocation rate of the expansion matrix method goes up when the number of development sites increases. It shows that RHA can effectively allocate tasks to development sites with the lowest cost when the number of development sites changes.

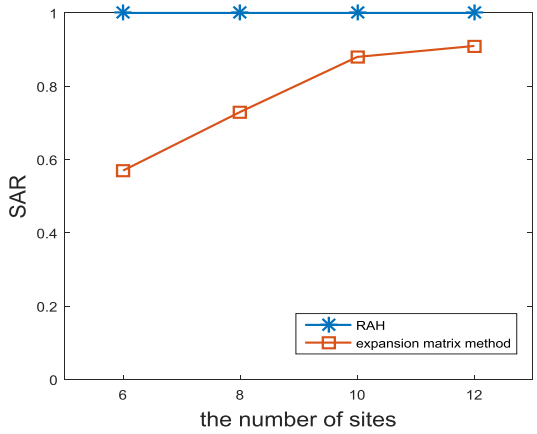

Fig. 1. SAR performances with different number of development sites

Experiment 2: In this experiment, we analyze the effectiveness of RHA with different number of tasks. Here we fix the number of development sites to 4 and evaluate the performance of the evaluated algorithms by increasing the number of tasks from 4 to 10 with an increment of 2. Therefore, we generated four sets of matrix data, which contains 100 *4×4* matrixes, 100 *4×6* matrixes, 100 *4×8* matrixes and 100 *4×10* matrixes, respectively.

Result Analysis: As we can see in Fig.2, the successful allocation rate of RHA is always 1. The successful allocation rate of the expansion matrix method decreases when the number of tasks increases It shows that RHA can effectively allocate tasks to development sites with the lowest cost when the number of tasks changes.

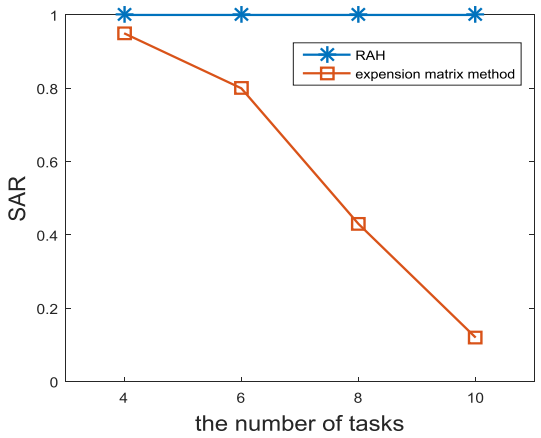

Fig. 2. SAR performances with different number of tasks

VI. Related Work

In this section, we briefly review the existing task assignment methods [13-26] in global software distribution.

Geographically distributed development creates new questions about how to coordinate muhi-site work. Grinter proposes four methods product development organizations to coordinate muhi-site work. He finds that no matter what model is used, it is difficulty to acquire expertise and the distribution of project mass may be unequal [13-14].

Lamersdorf et.al think that distributed development is often driven by some factors which are not distinguished, such as risks, workforce capabilities, the innovation potential of different regions, and cultural factors. They discuss about these factors and find a lack of empirically-based multi-criteria distribution models [15]. A qualitative study aiming to identify and understand the criteria used in practice was conducted by Lamersdorf, et al. The results show that the sourcing strategy and the type of software to be developed have a significant effect on the applied criteria [16]. The criteria and causal relationships were identified in a literature study and refined in a qualitative interview study. Lamersdorf et.al introduce a model which aims at improving management processes in globally distributed projects by giving decision support for task allocation that systematically regards multiple criteria [18]. A planning tool to identify task assignment based on multiple criteria and weighted project goals was developed by Lamersdorf and Munch. The model is called TAMRI (Task Allocation based on Multiple cRIteria) and its implementation is based on Bayesian networks. The application of the tool requires a large amount of knowledge on casual relationships in distributed development. [17].

Though these process models have helped companies to achieve global standards, some social aspects are not considered. Amrit proposes a methodology to test a hypothesis based on how social networks can be used to improve coordination in software industry [21]. Setamanit describes GSD-a hybrid computer simulation model of the software development process in order to identify the practices of work distribution and try to classify criteria [22]. Marques presents a

domain ontology to represent concepts related to task allocation in distributed teams in his paper. This method deals with the lack of standardized vocabulary and achieve knowledge acquisition and sharing [23].

VII. Conclusion and Future Work

In this paper, we address the issue of how to find the lowest-cost way to allocate tasks among development sites. We propose a reinforced Hungarian algorithm (RHA) for task assignment in global software development. RHA consist of two stages. In the first stage, the $n \times m$ matrix are transformed into two $n \times n$ matrix by adding virtual development sites. In the second stage, the two $n \times n$ matrix are solved by the original Hungarian algorithm. Finally, the virtual tasks are removed to get the optimal assignment results. We conduct experiments on the synthetic datasets to evaluate the performance of the proposed algorithm. The experimental results indicate that the proposed algorithm can effectively allocate tasks to development sites with the lowest cost.

In the future, we would like to validate the generalization ability of our method on available real-world datasets. In addition, we plan to investigate the task assignment method based on social data [24-25].

Acknowledgment

This work is partly supported by the grants of National Natural Science Foundation of China (No.61572374, No.U163620068, No.U1135005) and the Academic Team Building Plan from Wuhan University and National Science Foundation (NSF) (No. DGE-1522883).

References

[1] Taylor P S, Greer D, Sage P, et al. Do agile GSD experience reports help the practitioner?[C]//Proceedings of the 2006 international workshop on Global software development for the practitioner. ACM, 2006: 87-93.

[2] DeLone W, Espinosa J A, Lee G, et al. Bridging global boundaries for IS project success[C]//System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on. IEEE, 2005: 48b-48b.

[3] Xu Z, Liu Y, Mei L, et al. Semantic based representing and organizing surveillance big data using video structural description technology[J]. Journal of Systems and Software, 2015, 102: 217-225.

[4] Liu J, Yu X, Xu Z, et al. A cloud‐based taxi trace mining framework for smart city[J]. Software: Practice and Experience, 2016.

[5] Liu Z, Wei C, Ma Y, et al. UCOR: an unequally clustering-based hierarchical opportunistic routing protocol for WSNs[C]//International Conference on Wireless Algorithms, Systems, and Applications. Springer Berlin Heidelberg, 2013: 175-185.

[6] Liu Z, Niu X, Lin X, et al. A Task-Centric Cooperative Sensing Scheme for Mobile Crowdsourcing Systems[J]. Sensors, 2016, 16(5): 746.

[7] Huda N, Nahar N, Tepandi J, et al. Key barriers for global software product development organizations[C]//Management of Engineering & Technology, 2009. PICMET 2009. Portland International Conference on. IEEE, 2009: 1081-1087.

[8] Šmite D, Borzovs J. Managing uncertainty in globally distributed software development projects[J]. University of Latvia, Computer Science and Information Technologies, 2008, 733: 9-23.

[9] Wickramaarachchi D, Lai R. A method for work distribution in global software development[C]//Advance Computing Conference (IACC), 2013 IEEE 3rd International. IEEE, 2013: 1443-1448.

[10] Bass M, Paulish D. Global software development process research at Siemens[C]//Third International Workshop on Global Software Development. 2004: 8-11.

[11] Jonker R, Volgenant T. Improving the Hungarian assignment algorithm[J]. Operations Research Letters, 1986, 5(4): 171-175.

[12] Yexin S, Mianyun C, Shuhong Z. An efficient algorithm for solving two multi-object generalized assignment problems and its application[J]. JOURNAL-HUAZHONG UNIVERSITY OF SCIENCE AND TECHNOLOGY CHINESE EDITION, 2001, 29(1): 70-72.

[13] Grinter R E, Herbsleb J D, Perry D E. The geography of coordination: Dealing with distance in R&D work[C]//Proceedings of the international ACM SIGGROUP conference on Supporting group work. ACM, 1999: 306-315.

[14] Edwards H K, Kim J H, Park S, et al. Global software development: Project decomposition and task allocation[C]//International Conference on Business and Information. 2008.

[15] Lamersdorf A, Münch J, Rombach D. Towards a multi-criteria development distribution model: An analysis of existing task distribution approaches[C]//Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference on. IEEE, 2008: 109-118.

[16] Lamersdorf A, Munch J, Rombach D. A survey on the state of the practice in distributed software development: Criteria for task allocation[C]//Global Software Engineering, 2009. ICGSE 2009. Fourth IEEE International Conference on. IEEE, 2009: 41-50.

[17] Lamersdorf A, Munch J. TAMRI: a tool for supporting task distribution in global software development projects[C]//Global Software Engineering, 2009. ICGSE 2009. Fourth IEEE International Conference on. IEEE, 2009: 322-327.

[18] Lamersdorf A, Münch J, Rombach D. A decision model for supporting task allocation processes in global software development[C]//International Conference on Product-Focused Software Process Improvement. Springer Berlin Heidelberg, 2009: 332-346.

[19] Lamersdorf A. Empirically-Based Decision Support for Task Allocation in Global Software Development[C]//ICGSE. 2009: 281-284.

[20] Damian D, Moitra D. Guest editors' introduction: Global software development: How far have we come?[J]. IEEE software, 2006, 23(5): 17-19.

[21] Amrit C. Coordination in software development: the problem of task allocation[C]//ACM SIGSOFT Software Engineering Notes. ACM, 2005, 30(4): 1-7.

[22] Setamanit S, Wakeland W, Raffo D. Using simulation to evaluate global software development task allocation strategies[J]. Software Process: Improvement and Practice, 2007, 12(5): 491-503.

[23] Marques A B, Carvalho J R, Rodrigues R, et al. An ontology for task allocation to teams in distributed software development[C]//Global Software Engineering (ICGSE), 2013 IEEE 8th International Conference on. IEEE, 2013: 21-30.

[24] Xu Z, Mei L, Lu Z, et al. Multi-modal Description of Public Security Events using Surveillance and Social Data[J]. IEEE Transactions on Big Data, 2017.

[25] Xu Z, Zhang H, Hu C, et al. Building knowledge base of urban emergency events based on crowdsourcing of social media[J]. Concurrency and Computation: Practice and Experience, 2016.