Hindawi
Mathematical Problems in Engineering
Volume 2017, Article ID 6738139, 13 pages
https://doi.org/10.1155/2017/6738139

Research Article

Modeling and Simulation Study of Designer's Bidirectional Behavior of Task Selection in Open Source Design Process

Shuo Zhang[1,2] and Yingzi Li[3]

[1]*School of Economics and Management, North China Electric Power University, Beijing 102206, China*
[2]*Beijing Key Laboratory of New Energy and Low-Carbon Development, North China Electric Power University, Beijing 102206, China*
[3]*Donlinks School of Economics and Management, University of Science and Technology Beijing, Beijing 100083, China*

Correspondence should be addressed to Yingzi Li; liyz@ustb.edu.cn

Received 9 January 2017; Revised 25 April 2017; Accepted 2 May 2017; Published 24 May 2017

Academic Editor: M. L. R. Varela

Open source design (OSD) is an emerging mode of product design. In OSD process, how to select right tasks directly influences the efficiency and quality of task completion, hence impacting the whole evolution process of OSD. In this paper, designer's bidirectional behavior of task selection integrating passive selection based on website recommendation and autonomous selection is modeled. First, the model of passive selection behavior by website recommendation is proposed with application of collaborative filtering algorithm, based on a three-dimensional matrix including information of design agents, tasks, and skills; second, the model of autonomous selection behavior is described in consideration of factors such as skill and incentive; third, the model of bidirectional selection behavior is described integrating the aforementioned two selection algorithms. At last, contrast simulation analysis of bidirectional selection, passive selection based on website recommendation, and autonomous selection is proposed with ANOVA, and results show that task selection behavior has significant effect on OSD evolution process and that bidirectional selection behavior is more effective to shorten evolution cycle according to the experiment settings. In addition, the simulation study testifies the model of bidirectional selection by describing the task selection process of OSD in microperspective.

1. Introduction

Open source design (OSD), also called mass collaborative product development (MCPD), is an emerging design mode in recent years. In OSD process, many volunteer designers with different knowledge autonomously contribute to product development, as well as product creation, design, test, and even popularization by means of open network platform [1]. OSD is in rapid development with advantage of high innovation, low cost, and high customer satisfaction. Thus, OSD gradually becomes an important complementary mode of traditional collaborative product development (CPD) [2]. Different from the top-down organization mode of traditional CPD, the organization of OSD is in bottom-up self-organized structure, and OSD including organization and project/product is in continuous evolution [3]. Moreover, OSD can make full use of the emergence of design originality, as well as sharing in technology, resource, and knowledge between designers.

At present, Open Source Software (OSS) is the most successful application of OSD, and some open source communities (OSC) that coexist with OSS are in effective operation like Linux, Apache, Mozilla, and so on [4]. Meanwhile, OSD has been applied in industrial product design [5]. By this mode, the new product originality or CAD model is released to public by volunteer designers or enterprises, and many OSC members will complete it collaboratively by Internet. New ideas and products are constantly emerging in some famous innovation OSC such as Open Source Car, Prosthetics Project, and Lego Mindstorms [6–8]. These cases show that OSD is influencing product design mode deeply and comprehensively.

In OSD process, task selection is an important phase, in which designers endeavor to select appropriate tasks

to promote design process effectively. If this phase is not worked well, it may be a bottleneck that restricts efficiency and quality of task execution. To select matched tasks from numerous candidates, designers perform selection behaviors in consideration of many relevant subjective and objective factors, such as individual preference, technical ability, and motivators. However, there is little research on task selection of OSD process. On this aspect, designers' behaviors are described effectively and expressly from microperspectives in detail by theory of Complex Adaptive System (CAS) and agent modeling [9]. As a result, task selection behavior of designers in OSD process is studied based on CAS in this paper.

The rest of the paper is organized as follows. In Section 2, related work on task selection study of OSD and related area is discussed. In Section 3, bidirectional behavior of task selection integrating passive and autonomous selection methods is proposed. In Section 4, simulation experiments based on an engineering design case of cell phone are designed, and related simulations are carried out to make contrastive analysis of different selection behaviors and analyze designers' bidirectional behaviors of task selection during OSD process from microscopic view. In Section 5, the highlights and future work are represented.

2. Related Work

In OSD process, it is a key point to select tasks matched to designers, which can improve the efficiency and success rate of open source project. However, there is little research on designers' task selection behaviors in OSD with quantitative model and algorithm. As regards to selection problem, there are abundant researches such as recommendation and task allocation. Therefore, relative studies on web service recommendation and task allocation are discussed for reference.

2.1. Research on Web Service Recommendation. The ongoing rapid expansion of the Internet greatly increases the necessity of effective recommendation systems for filtering the abundant information [10]. A recommendation system, which is widely applied in web service such as online shopping, e-resource services, and social network activities, aims to provide users with personalized online product or service recommendations to handle the increasing online information overload problem [11]. Collaborative filtering (CF) is widely employed for making web service recommendation [12]. CF-based web service recommendation, which attempts to predict what information will meet a user's needs from the neighborhoods of like-minded people or similar items, aims to recommend users products, services, resources, and so on, to best satisfy their requirements [13]. There are usually millions of customers and products in web service recommendation system, which is similar to OSD system. Hence, the recommendation models such as CF are of great value on task recommendation research of OSD. According to web service recommendation system, users' selection behaviors during OSD process can be described by system's recommendation algorithm.

2.1.1. Recommendation Research on Online Shopping. To attract more customers and provide them good service to select satisfactory products, many e-commerce enterprises, such as Amazon, develop recommendation systems to recommend customers products which they probably need [14].

Rodrigues and Ferreira [15] propose a hybrid recommendation system that combines content-based, collaborative filtering, and data mining techniques, to surpass recommendation difficulties of low efficiency and quality to provide customer right products. In addition, a novel recommendation system using collaborative filtering algorithm is implemented in Apache Hadoop leveraging MapReduce paradigm for Bigdata, and the Amazon dataset is used for the product recommendations [16]. Online personalized product ranking is also extensively discussed in the literature of recommendation systems and considered beneficial to both consumers and e-retailers. Zhang et al. [17] propose a new approach called Ranking with Prediction Uncertainty to improve the accuracy of personalized product ranking based on collaborative filtering techniques. Arguing that current approaches are suboptimal in terms of matching tasks and contributors' individual interests and capabilities, Geiger and Schader [18] advocate the introduction of personalized task recommendation mechanisms in crowdsourcing information systems and contribute to a conceptual foundation for the design of such mechanisms by conducting a systematic review of the corresponding academic literature. Moreover, the multicriteria based CF presents a possibility of providing accurate recommendations by considering the user preferences in multiple aspects. Hence, Nilashi et al. [19] propose new recommendation methods using Adaptive Neuro-Fuzzy Inference Systems (ANFIS) and Self-Organizing Map (SOM) clustering to improve predictive accuracy of criteria CF, in which SOM enables generating high quality clusters of dataset and ANFIS is used for discovering knowledge (fuzzy rules) from users' ratings in multicriteria dataset. In recommendation system, the sparsity problem usually occurs in the transaction data, which makes it difficult to identify reliable neighbors, resulting in less effective recommendations. Therefore, Choi et al. [20] suggest a means to derive implicit rating information from the transaction data of an online shopping mall and then propose a new user similarity function, which computes the user similarity of two users if they rated similar items, to mitigate the sparsity problem.

2.1.2. Recommendation Research on e-Resource Service. Recommendation systems are information filtering tools that aspire to predict the rating for users and items, predominantly from big data to recommend their likes. This makes recommendation system essentially a central part of e-commerce applications. In e-resource service area, the emergence of the online media sharing sites (e.g., YouTube, Youku, and Hulu) has introduced new challenges in program recommendation in online networks, and personalized recommendation services can effectively solve this problem to assist users in classifying users with similar interests.

Katarya and Verma [21] suggest an improved movie recommendation system through data clustering and computational intelligence, applying hybrid of k-means and cuckoo

search to the Movielens dataset. However, there is a bottleneck that the amount of available viewing logs and user friendship networks are too limited to design effective recommendation algorithms. Thus, Li et al. [22] propose a novel recommendation model which turns to the social networks and mine user preferences information expressed in microblogs for evaluating the similarity between online movies and TV episodes, to bridge the gap between domains of movie and TV watchers with social network activities. In addition, García-Cumbreras et al. [23] present a novel application of Sentiment Analysis by categorizing users according to the average polarity of their comments, use these categories as attributes in CF algorithms, and prove this solution can provide a more reliable prediction by generating a new corpus of opinions on movies obtained from the Internet Movie Database. In consideration of annotation information, Wei et al. [24] propose a hybrid movie recommendation approach using tags and rating, to improve recommendation accuracy. On the other hand, music recommendation is a research topic of increasing interest since online music platforms have become rapidly popular. However, some important problems, such as the difficulty of extracting content information from music, must be addressed in order to give reliable recommendations. Sánchez-Moreno et al. [25] propose a recommendation method based on playing coefficients to deal with gray sheep and sparsity problems without needing user attributes, content data, and explicit ratings from users, and the proposal is proved to outperform the methods that make use of user attributes. Meanwhile, content personalization is a long-standing problem for online news services. Bai et al. [26] study the problem of news personalization by leveraging usage information that is external to the news service, propose a novel approach applying user profiles that are built based on the past interactions of the user with a web search engine, and extensively test it on real-world datasets obtained from Yahoo. Resources in cloud computing platforms such as Amazon, Google AppEngine, and Microsoft Azure provide a new space of mobile search to improve the availability of cloud resources. On this aspect, Zhao et al. [27] propose a hybrid filtering mechanism to eliminate irrelevant or less relevant results for personalized mobile search, which combines content-based filtering and collaborative filtering to make use of the user's query history and communication history of social network.

2.1.3. Recommendation Research on Social Network Activities. The rapid growth of social network services has produced a considerable amount of data, called big social data. Big social data are helpful for improving personalized recommendation systems because these enormous data have various characteristics. Therefore, many personalized recommendation systems based on big social data have been proposed.

Seo et al. [28] introduce an appropriate measure to calculate the closeness between users in a social circle, namely, the friendship strength, proposing a friendship strength-based personalized recommendation system that recommends topics or interests users might have in multidomain environments order by analyzing big social data, using Twitter in particular. Based on collaborative filtering methods, Shahmohammadi et al. [29] propose directed proximity measures for activity prediction and recommendation both for pairs of users without any interaction background and also for user pairs with the activity background and perform experiments on the dataset of different Facebook activity networks including like, comment, post, and share networks, showing that the proposed collaborative methods deal with the activity prediction.

In social networks, a commonly adopted recommendation method takes advantage of the tastes of a user's trust neighbors and recommends resources which his/her neighbors have bought or evaluated. It will perform poorly for the inactive users who have few trust neighbors. Social tagging has become increasingly prevalent on the Internet, which provides an effective way for users to organize, manage, share, and search for various kinds of resources. Guo et al. [30] try to find users' similar neighbors using tag information which is not only from users' photos but also from their favorite photos and the common friend information, propose a group recommendation scheme utilizing users' trust neighbors and similar neighbors' tastes, and do the experiments on a real-world Flickr dataset and obtain a promising result especially for inactive users. Zheng and Li [31] investigate the importance and usefulness of tag and time information when predicting users' preference and examine how to exploit such information to build an effective resource-recommendation model, carrying out empirical research with data from a real-world dataset to show that tag and time information can well express users' taste and that better performances can be achieved if such information is integrated into collaborative filtering.

With the rapid development of web service technology and cloud computing environments, more and more service providers supply web services with the same features. To solve the service discovery problem, Lin et al. [32] propose a trustworthy two-phase web service discovery mechanism based on QoS (Quality of Service) and CF, which discovers and recommends the needed web services effectively for users in the distributed environment, and also solve the problem of services with incorrect QoS information. In the constantly changing business environment, organizations must exploit effective and efficient methods of preserving, sharing, and reusing knowledge in order to help knowledge workers find task-relevant information. Hence, Lai and Liu [33] propose hybrid recommendation methods based on a knowledge flow model, which integrates KF mining, sequential rule mining, and CF techniques to understand knowledge workers' task-needs and the ways they reference documents, and recommend codified knowledge.

2.1.4. Cold Start Problem of Collaborative Filtering in Recommendation System. Although collaborative filtering (CF) is widely used for recommendation systems, it suffers from complete cold start (CCS) problem where no rating records are available and incomplete cold start (ICS) problem where only a small number of rating records are available for some new items or users in the system.

Wei et al. [34] propose two recommendation models to solve the CCS and ICS problems for new items, which are based on a framework of tightly coupled CF approach and

deep learning neural network, and the experiment results on Netflix movie recommendation show the tight coupling of CF approach and deep learning neural network is feasible and very effective for cold start item recommendation. Meanwhile, Kim et al. [35] propose a collaborative filtering method to provide an enhanced recommendation quality derived from user-created tags, in which collaborative tagging is employed as an approach in order to grasp and filter users' preferences for items, and experimental results show that the proposed algorithm offers significant advantages in terms of both improving the recommendation quality for sparse data and dealing with cold start users as compared to existing work. Recommending items to new users generally creates a sense of belonging and loyalty and encourages them to frequently utilize recommendation systems. Chen et al. [36] propose a cold start recommendation method for the new user that integrates a user model with trust and distrust networks to identify trustworthy users, whose suggestions are then aggregated to provide useful recommendations for cold start new users, and experiments based on the well-known Epinions dataset demonstrate the efficacy of the proposed method. Given that the relational characteristics between items can provide much useful information during the recommendation process, Lv et al. [37] propose an item recommendation method based on a domain ontology and genetic algorithm (GA), obtain data relations between items by using all the item relations in the ontology and GA, and utilize the data relations as the basis of the online top-n item recommendations to solve the cold start problem.

2.2. Research on Task Allocation. It is a kind of passive behaviors for customers to select products or services by recommendation system, which endeavors to provide accurate recommendations by prediction. Nevertheless, customers usually surf the web and select products or services on their own initiative, instead of recommendations by system. In this case, autonomous selection is an important type of task selection in OSD. However, there is little research on autonomous selection of tasks. Compared to autonomous selection, which is performed personally, task allocation is usually brought out from an overall perspective of systems or workflows. In spite of this, the two behaviors are both in consideration of matching attributes between individuals and tasks. Hence, the research on task allocation is considerable reference for autonomous selection study. Some research on task allocation is abstracted as follows.

Ul Hassan and Curry [38] provide a conceptual framework to study the minimum-cost maximum reliability assignment problem with online combinatorial optimization and online learning on spatial crowdsourcing, which provides new insights into the combinatorial assignment strategies when the objective is to maximize reliability and minimize costs. Brahmbhatt and Camorlinga [39] leverage the existing similarity between disease epidemics and distributed system services and evaluate several factors on the SARS pandemics from a CAS perspective, which provides several insights and inspiration used to develop an algorithm for the task assignment problem in a distributed system. To assign workers to tasks effectively, Nembhard and Bentefouet [40] investigate the operational decision-making processes including selecting workers from a pool, grouping workers based on individual characteristics, and assigning groups to tasks and model worker productivity to include skill knowledge obtained by learning-by-doing and learning-by-transfer. To handle scheduling of tasks on heterogeneous systems, Akbari and Rashidi [41] propose an algorithm based on multiobjective scheduling cuckoo optimization algorithm to reduce execution time allowing for maximum parallelization, which is effectively implemented on a large number of random graphs and real-world application graphs with wide range characteristics. On the research of workflow execution dynamics in distributed environments, Yun et al. [42] formulate a generic problem considering both workflow mapping and task scheduling to minimize the end-to-end delay of workflows and propose an integrated solution to improve the workflow performance. Moreover, Shao et al. [43] study knowledge workforce assigning problem in software projects from three essential project management perspectives, timeliness, effectiveness, and efficiency, explore ideal workforce composites focused on productivity and quality with different scenarios of workload ratio, and propose an analytical model and a metaheuristic approach based on particle swarm optimization. In addition, Brown et al. [44] conduct a laboratory experiment to examine how task difficulty and different types of performance feedback (none, individual, and relative) affect individuals' selection and find that participants exhibit a strong better-than-average bias in assessing their relative skills on easy tasks and a moderate worse-than-average bias in assessing their relative skills on difficult tasks. As a multiobjective problem including time, cost, quality, and risk, the optimal allocation of distributed manufacturing resources is a challenging task for supply chain deployment. Zhang et al. [45] present an improved variant of the Teaching-Learning-Based Optimization algorithm to concurrently evaluate, select, and sequence the candidate distributed manufacturing resources allocated to subtasks comprising the supply chain. Yu et al. [46] incorporate the synergy effect between products in supplier selection process and propose a negotiation protocol including combinatorial procurement auction protocol and multibilateral bargaining protocol, by which both the purchasing company and suppliers can express their preferences on the synergy effect between products in negotiation process.

3. Task Selection of Bidirectional Behavior

During OSD process, designers contribute to product design by performing different and complex behaviors, such as task setting, release, selection, execution, interruption, collaboration, and screening, which directly drive the design process and influence product evolution as well as OSC evolution. In the numerous different kinds of behaviors, task selection behavior is a most essential one. Before task execution, designers need to select appropriate tasks which can obviously promote the evolution of OSD. Otherwise, subsequent execution of tasks, which are selected not matched to designers, will encounter more difficulties, thus affecting

quality and efficiency of OSD evolution. As a result, it has been a key issue on how to select tasks that are effectively matched to design agents. There are mainly two ways on task selection in OSD, which are passive selection based on website recommendation and autonomous selection based on task information [10, 38]. Although website recommendation provides much convenience to designers while selecting tasks, it is a type of passive selection behaviors for designers, by which the recommended tasks may not fully meet designers' requirement if the historical data are sparse or cannot reflect designers' preference. On the other hand, autonomous selection based on task information is a type of behaviors on designers' own initiative, but this selection is not effective in the condition of numerous tasks released in OSD process, which are too many for designers to autonomously select best matched tasks. Moreover, designers select tasks often considering both website recommendation and autonomous selection while contributing to OSD. As a result, the bidirectional selection behavior in consideration of both recommendation and autonomous selection is proposed in this paper. To correctly describe the bidirectional behavior, the designer is defined as design agent by the methodology of agent modeling which is proposed in preliminary study [9]. The model of bidirectional selection is built as follows.

3.1. Passive Behavior of Task Selection Based on Website Recommendation. According to this way, design agent selects task only based on recommendation service of website. In this case, the recommendation algorithm for task selection is key important. There are considerable researches on recommendation algorithms in related areas such as e-commerce [10, 11, 14]. In this paper, collaborative filtering algorithm is applied to recommend tasks to designers, which describes task recommendation mechanism based on a three-dimensional matrix including information on design agent, task, and skills. In the mechanism, the similarity between target agent and other design agent is firstly calculated based on evaluation information of tasks, skill information of agents, and skill demand information of tasks; secondly, some design agents are selected as recommendation agents whose similarities with target agent are relatively higher; thirdly, tasks which are completed by the recommendation agents and have not been selected by target agent are chosen into task recommendation list; then, each evaluation value of the task list by target agent is predicted, by which the task list is ordered; at last, a number of tasks which are selected from highest to lowest based on predicted evaluation values are recommended to target agent.

3.1.1. Three-Dimensional Matrix for Recommendation. The three-dimensional matrix is composed of information on design agent, task, and skill, which includes designer-task 0-1 matrix, designer-task evaluation matrix, designer-skill 0-1 matrix, designer-skill 0-1 matrix, designer-skill matrix, task-skill 0-1 matrix, and task-skill demand value matrix, as shown in Figure 1.

In Figure 1, D_i $(i = 1, 2, \ldots, n)$ denotes design agent, and n is the number of design agents; T_j $(j = 1, 2, \ldots, m)$ denotes the task of OSD, and m is the number of tasks; S_k $(k = 1, 2, \ldots, l)$ denotes the skill item, and l is the number of skill items.

$DT = (dt_{ij})$ $(dt_{ij} \in \{0, 1\})$ denotes designer-task 0-1 matrix, in which $dt_{ij} = 1$ means that D_i has completed T_j while $dt_{ij} = 0$ means that D_i has not selected T_j.

$DT' = (dt'_{ij})$ denotes designer-task evaluation matrix, in which dt'_{ij} represents the evaluation value of T_j by D_i after D_i completes it. If D_i does not select T_j, $dt_{ij} = 0 \Rightarrow dt'_{ij} = 0$.

$DS = (ds_{ik})$ $(ds_{ik} \in \{0, 1\})$ denotes designer-skill 0-1 matrix, in which $ds_{ik} = 1$ means that D_i masters skill item S_k while $ds_{ik} = 0$ means that D_i does not master S_k.

$DS' = (ds'_{ik})$ denotes designer-skill matrix, in which ds'_{ik} represents the level of D_i in S_k. If D_i does not master S_k, $ds_{ik} = 0 \Rightarrow ds'_{ik} = 0$.

$TS = (ts_{jk})$ $(ts_{jk} \in \{0, 1\})$ denotes task-skill 0-1 matrix, in which $ts_{jk} = 1$ means that it requires skill item S_k to execute T_j while $ds_{ik} = 0$ means that it does not.

$TS' = (ts'_{jk})$ denotes task-skill matrix, in which ts'_{jk} is the lower limit for S_k to execute T_j. If the execution of T_j does not require S_k, $ts_{jk} = 0 \Rightarrow ts'_{jk} = 0$.

3.1.2. Recommendation of Similar Design Agents

(1) Similarity between Design Agents. Similarity is a key factor to measure the similar degree between design agents and target agent. Based on corresponding references [32, 35], the similarity between target agent D_i and design agent $D_{i'}$ is calculated as formula (1)

$$\mathrm{Sim}(D_i, D_{i'}) = \frac{\sum \left(dt'_{ij} - \overline{DT_i^{ii'}}\right)\left(dt'_{i'j} - \overline{DT_{i'}^{ii'}}\right)}{\sqrt{\sum \left(dt'_{ij} - \overline{DT_i^{ii'}}\right)^2 \sum \left(dt'_{i'j} - \overline{DT_{i'}^{ii'}}\right)^2}},$$
$$\text{s.t.} \quad \begin{cases} \sum_{j=1}^{m} dt_{ij} dt_{i'j} \geq \Delta_t \\ \sum_{k=1}^{l} ds_{ik} ds_{i'k} \geq \Delta_s, \end{cases} \tag{1}$$

where Δ_t is the lower bound constraint for number of tasks completed by both D_i and $D_{i'}$; Δ_s is the lower bound constraint for number of skill items that both D_i and $D_{i'}$ master; $\overline{DT_i^{ii'}}$ is the mean evaluation value of tasks by D_i, which both D_i and $D_{i'}$ have rated, $\overline{DT_i^{ii'}} = \sum_{j=1}^{m} dt_{ij} dt_{i'j} dt'_{ij} / \sum_{j=1}^{m} dt_{ij} dt_{i'j}$; $\overline{DT_{i'}^{ii'}}$ is the mean evaluation value of tasks by $D_{i'}$, which both D_i and $D_{i'}$ have rated, $\overline{DT_{i'}^{ii'}} = \sum_{j=1}^{m} dt_{ij} dt_{i'j} dt'_{i'j} / \sum_{j=1}^{m} dt_{ij} dt_{i'j}$.

The value of the task evaluated by D_i is calculated as formula (2), which considers factors of time and incentive

$$dt'_{ij} = \alpha \frac{rt_j^T}{at_{ij} + rt_j^T} + (1 - \alpha) \frac{aw_{ij}/at_{ij}}{aw_{ij}/at_{ij} + aw_j^T/rt_j^T}, \tag{2}$$

where rt_j^T is rated completion time of task T_j; at_{ij} is the actual completion time of task T_j by D_i; aw_j^T is rated bonus after

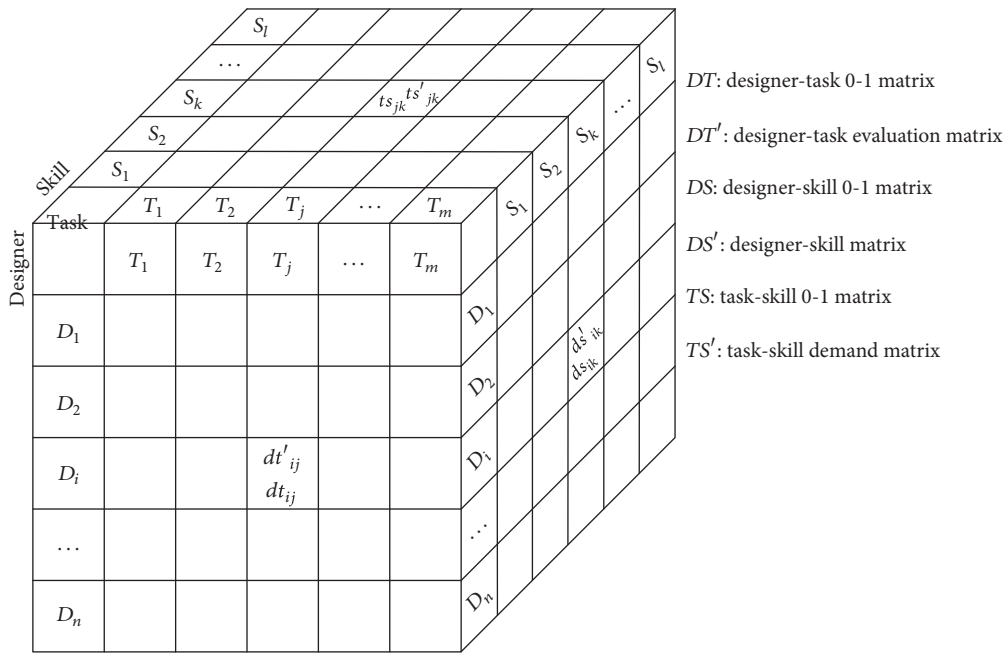

Figure 1: Three-dimensional matrix.

completion of task T_j; aw_{ij} is actual bonus after completion of task T_j by D_i; α is weight, $\alpha \in [0, 1]$.

(2) Screening Similar Design Agent. Design agent $D_{i'}$, whose similarity $\mathrm{Sim}(D_i, D_{i'})$ with D_i is one of K maximum similarities of the candidates, will be selected into recommendation set $\mathrm{KNN} = \{D_1, D_2, \ldots, D_K\}$.

(3) Prediction for Task Evaluation. Task T_j $(dt_{i'j} = 1, dt_{ij} = 0)$, which is not performed by D_i while completed and evaluated by $D_{i'}$ $(D_{i'} \in \mathrm{KNN})$, will be selected to calculate the predicted evaluation value fdt'_{ij} if performed by D_i, as formula (3)

$$fdt'_{ij} = \overline{DT_i} + \frac{\sum_{D_{i'}}^{\mathrm{KNN}} \mathrm{Sim}\left(D_i, D_{i'}\right)\left(dt'_{i'j} - \overline{DT_{i'}^{ii'}}\right)}{\sum_{D_{i'}}^{\mathrm{KNN}} \mathrm{Sim}\left(D_i, D_{i'}\right)}. \quad (3)$$

After the calculation, task T_j, whose predicted evaluation value fdt'_{ij} is one of the N maximum values of the candidate tasks, will be selected into recommendation task set RT_i, which is recommended to D_i.

(4) Cold Start Problem. If D_i never performs any task before $(\forall dt_{ij} = 0)$, D_i cannot be recommended for any task applying aforementioned algorithms without task evaluation data, which is the so-called cold start problem. To figure out the issue, a model to calculate incentive coefficient in consideration of incentive factor is proposed, which is applied to recommend D_i tasks, as formula (4)

$$e_{ij} = \frac{aw_j^T}{rt_j^T} \times \frac{aw_j^T}{aw_i^D} = \frac{\left(aw_j^T\right)^2}{rt_j^T aw_i^D},$$
$$\text{s.t.} \quad \sum_{k=1}^{l} ds_{ik} ts_{jk} \geq \Delta_s^{dt}, \quad (4)$$

where aw_i^D is cumulative bonus of D_i and Δ_s^{dt} is the lower bound constraint for number of skill items to perform T_j.

After the calculation, task T_j, whose incentive coefficient by D_i is one of the N maximum values of the candidate tasks, will be selected into recommendation task set RT_i, which is recommended to D_i.

3.2. Autonomous Selection according to Task Information. In OSD process, design agent often autonomously selects tasks from task series that OSC releases to public in consideration of some factors such as skill and incentive. To select satisfactory tasks, design agent performs the behavior by fitness f_{ij} as formula (5)

$$f_{ij} = \beta \times r_{ij} + (1 - \beta) \times e_{ij}, \quad (5)$$

where r_{ij} is similarity between D_i and T_j; e_{ij} is incentive coefficient to execute the task as formula (4); β is weight, $\beta \in [0, 1]$.

The calculation of r_{ij} is as formula (6)

$$r_{ij} = \frac{\sum_{k=1}^{l} \left(ds_{ik} ts_{jk}\right) \left(ds'_{ik} - \overline{DS_i^{ij}}\right) \left(ts'_{jk} - \overline{TS_j^{ij}}\right)}{\sqrt{\sum_{k=1}^{l} \left(ds_{ik} ts_{jk}\right) \left(ds'_{ik} - \overline{DS_i^{ij}}\right)^2 \sum_{k=1}^{l} \left(ds_{ik} ts_{jk}\right) \left(ts'_{jk} - \overline{TS_j^{ij}}\right)^2}} \quad (6)$$

$$\text{s.t.} \quad \sum_{k=1}^{l} ds_{ik} ts_{jk} > 0,$$

where $\overline{DS_i^{ij}}$ is mean value of D_i on S_k ($ds_{ik} = 1, ts_{jk} = 1$), $\overline{DS_i^{ij}} = \sum_{k=1}^{l} ds_{ik} ts_{jk} ds'_{ik} / \sum_{k=1}^{l} ds_{ik} ts_{jk}$, and $\overline{TS_j^{ij}}$ is mean requirement of T_j on S_k ($ds_{ik} = 1, ts_{jk} = 1$), $\overline{TS_j^{ij}} = \sum_{k=1}^{l} ds_{ik} ts_{jk} ts'_{jk} / \sum_{k=1}^{l} ds_{ik} ts_{jk}$.

By calculation as formula (5), task T_j, whose fitness f_{ij} is one of the N maximum values of the candidate tasks, will be selected into task set ST_i for autonomous selection.

3.3. Bidirectional Behavior of Task Selection. During OSD process, task array RT_i, which is recommended for D_i based on the three-dimensional recommendation algorithm, may not meet D_i's actual requirement for sparse data or designers' special preference. Meanwhile, D_i probably cannot select optimal task only by autonomous selection because of large number of tasks which cannot be traversed one by one. To figure out the problem, designers often take full account of both website recommendation and autonomous selection to perform task selection behavior, which is called bidirectional behavior of task selection. This kind of selection behavior is proposed to guarantee the tasks more in line with requirements of D_i. According to the methodology of bidirectional selection behavior, the selection process is calculated as follows.

(1) D_i calculates f_{ij} of each task T_j according to formula (5) in RT_i, which is obtained by the model of website recommendation.

(2) D_i integrates RT_i and ST_i which is obtained by the model of autonomous selection into one task array BT_i and rearranges the task series according to f_{ij}.

(3) D_i selects task T_j, whose f_{ij} is one of the N maximum values of BT_i, into task set FT_i for execution.

4. Simulation Study of Bidirectional Selection Behavior

4.1. Simulation Experiment Setting. According to the model of bidirectional selection behavior proposed in Section 3, simulation study is carried out to testify the efficiency of bidirectional behavior on OSD evolution.

Based on the model and the simulation platform [9], the simulation study is carried out according to a cell phone design scenario abstracted from corresponding references [47–49]. In this scenario, cell phone is composed of many components, such as front cover, rear cover, main board, keyboard, battery, screen, receiver, microphone, antenna, Wi-Fi, and camera. These components are independent but relative. For example, front cover, rear cover, and screen are independent in function design, but compatible in assembling, so the relationship among the three parts should be taken into account. Based on the scenario derived from the references, the relationship of module tasks is set in the simulation platform as shown in Figure 2.

Figure 2 is mainly composed of two icons, which are *Task* and *Info*, respectively. *Task* denotes two types of tasks which are initial module task and collaboration task in OSD process. Initial module task, shown as T_i ($i = 0, 1, 2, \ldots, 19$), is the basic composition of OSD project, which is divided and packaged according to product function as well as skill requirement, such as front cover, rear cover, and main board of a cell phone. Meanwhile, collaboration task, shown as T_i ($i > 19$), is generated if design agent asks for collaboration while executing initial module tasks. During OSD process, each task, no matter initial module task or collaboration task, can be selected by design agents autonomously. *Info* shows the information of tasks that are connected directly, such as task triggering time and precedence relationship between tasks. In addition, Table 1 lists some important parameters, which quantify task requirement and rewards for design agents. In this scenario, the number of skill items is set as 3.

Besides, some key parameters in the simulation are initialized as shown in Table 2, in which *S*1, *S*2, and *S*3 are three comparison scenarios corresponding to passive behavior of selection by recommendation, autonomous selection behavior, and bidirectional selection behavior, respectively. According to the parameter settings, each scenario is simulated 100 times, obtaining 100 data samples for comparative analysis.

4.2. ANOVA of Three Task Selection Behaviors. In this section, Analysis of Variance (ANOVA) is carried out to analyze if task selection behavior influences OSD process and bidirectional behavior is best effective of the three. In this way, the analysis can be applied as assistant decision of task selection in OSD process. The hypothesis for ANOVA is as follows.

H_0: $\mu_1 = \mu_2 = \mu_3$ (task selection behavior has no effect on evolution cycle of OSD)

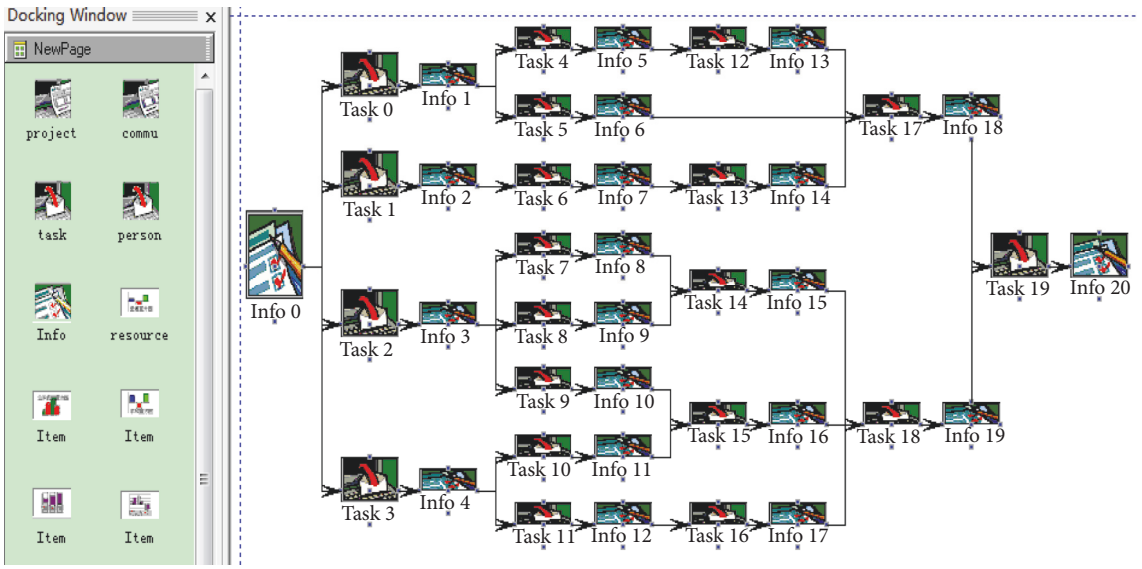

Figure 2: Module task relationship of the scenario.

Table 1: Parameters of task requirement and rewards for design agent.

Task ID	0	1	2	3	4	5	6	7	8	9
	6.26	5.148	27.617	4.263	25.522	16.358	26.001	12.857	14.106	7.669
Matrix of skill demand	3.048	8.416	10.769	8.104	11.663	15.208	10.85	22.561	28.413	0.138
	17.775	13.931	20.516	6.793	28.474	3.164	3.596	5.155	3.253	16.934
Rated bonus	16.6	17.1	16.5	18.1	21.2	17.5	9.6	9.1	10	18.1
Task ID	10	11	12	13	14	15	16	17	18	19
	15.629	20.203	7.088	16.334	23.616	9.666	19.539	13.936	13.301	22.179
Matrix of skill demand	23.029	28.172	19.139	3.754	14.665	1.196	21.834	25.224	3.539	12.78
	22.335	26.397	24.824	2.134	21.044	22.271	21.92	21.208	29.694	7.338
Rated bonus	12.1	12.3	14.3	13.6	18.1	15.5	28	20.1	20.5	20

Table 2: Partial parameters of simulation experiment.

Parameter name	Parameter setting
Scenario	$S1$, $S2$, $S3$
Community scale (number of agents)	200
Task selection method for each scenario	$S1$: passive behavior of task selection by recommendation $S2$: autonomous selection behavior $S3$: bidirectional selection behavior
Simulation time	300 (100/scenario)

H_1: $\mu_1 \neq \mu_2 \neq \mu_3$ (task selection behavior has significant effect on evolution cycle of OSD)

According to the hypothesis, ANOVA is applied with SPSS to process the data of three scenarios which are extracted from simulation experiments, as shown in Table 4.

The results of ANOVA show that task selection behavior has significant effect on evolution cycle of OSD. $\mu_1 = 391.52$ (mean value of evolution cycle in $S1$) is significantly longer than that of $S2$ and $S3$. Data analysis of evolution process in $S1$ shows that task recommendation does not do well at the beginning of OSD process because there is not enough information on task evaluation for recommendation, which is the main reason that results in longer evolution cycle. $\mu_2 = 373.1$ (mean value of evolution cycle in $S2$) is significantly shorter than $S1$'s cycle but longer than that of $S3$. The main reason is that the scale of design tasks in $S2$ is not quite large which is not difficult for design agent to autonomously select compatible tasks. $\mu_3 = 364.81$ (mean value of evolution cycle in $S3$) is significantly shorter than that of $S2$ and $S3$. The main reason is that design agent can select more matched tasks in combination with website recommendation and autonomous selection. As a result, compared with $S1$ and $S2$, $S3$ is more effective to promote evolution process of OSD.

Simulation study shows that task selection behavior of design agent has significant effect on evolution process of OSD. In this experiment with moderate task scale, the bidirectional behavior of task selection is more effective to shorten evolution cycle. However, it should be testified which selection behavior is more effective if task scale is much larger in further study.

Table 3: Task selection of A_{13}.

Order	Simulation step	Task recommendation				Autonomous selection			Final selection		
		Task number	Task ID	fdt'_{ij}	f_{ij}	Task number	Task ID	f_{ij}	Task number	Task ID	f_{ij}
1	0	4	3	1.389110	0.304965	3	2	0.730274	3	2	0.730274
			1	1.239946	0.376856		1	0.376856		0	0.686295
			2	1.024853	0.730274		3	0.304965		1	0.376856
			0	0.784246	0.686295						
2	149	1	6	0.050432	0.269533	1	6	0.269533	1	6	0.269533
3	157	1	14	0.113611	0.269073	1	14	0.269073	1	14	0.269073
4	176	2	484	0.008573	0.240730	1	485	0.24942	2	485	0.249420
			485	0.005935	0.249420					484	0.240730
5	188	2	13	0.076980	0.128854	2	15	0.222744	2	15	0.222744
			15	0.060707	0.222744		13	0.128854		13	0.128854
6	254	1	17	0.125698	0.271574	1	17	0.271574	1	17	0.271574
7	286	1	18	0.100443	0.208817	1	18	0.208817	1	18	0.208817
8	315	1	19	0.104335	0.182663	1	19	0.182663	1	19	0.182663

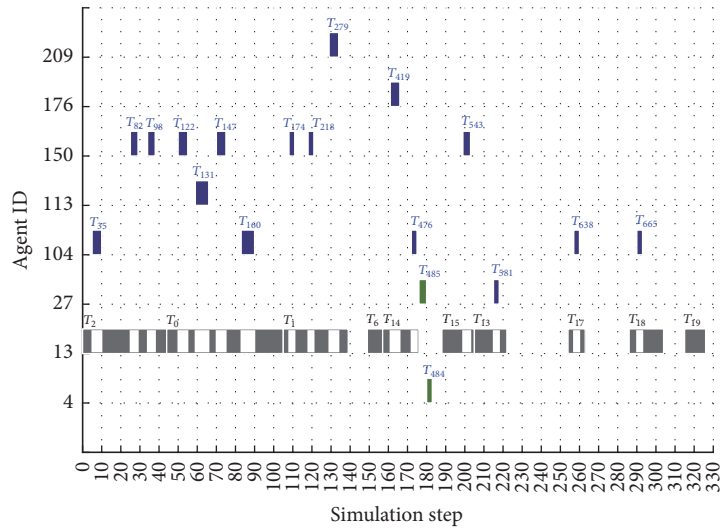

Figure 3: Task selection and execution flow of A_{13}.

4.3. Simulation Analysis of Bidirectional Selection Process. According to previous analysis, bidirectional behavior of task selection is testified to be the best of three behaviors of task selection based on simulation experiment setting. To describe and analyze designer's bidirectional behavior of task selection integrating website recommendation and autonomous selection during OSD process, a data set is extracted from data sample of *S*3. The data set records the task selection behaviors of design agent A_{13} in OSD process. The task selection of A_{13} is shown as Table 3 and Figure 3.

As shown in Table 3, A_{13} performs bidirectional behaviors of task selection for 8 times during this OSD process. In each time, the bidirectional selection behavior contains 3 phases: website recommendation, autonomous selection, and integrated selection. Firstly, the system recommends A_{13} some tasks which are sorted by website recommendation according to formula (3); secondly, A_{13} chooses some tasks autonomously from OSS and sorts them according to formula (5); meanwhile, corresponding values of tasks that are recommended are also calculated based on formula (5); at last, A_{13} integrates the tasks that are recommended or autonomously chosen, sorts them by values based on formula (5), and selects tasks in descending order, which is also the order of task execution subsequently. Table 3 shows the detailed data for bidirectional selection behavior of A_{13}.

As shown in Figure 3, vertical axis denotes design agent ID, and horizontal axis denotes simulation step of OSD. During the whole process, A_{13} selects and executes 28 tasks, including 10 module tasks (T_2, T_0, T_1, T_6, T_{14}, T_{15}, T_{13}, T_{17}, T_{18}, and T_{19}) which are shown as gray blocks, and 18 collaboration tasks (T_{35}, T_{82}, T_{98}, T_{122}, T_{131}, T_{147}, T_{160}, T_{174}, T_{218}, T_{279}, T_{419}, T_{476}, T_{485}, T_{484}, T_{543}, T_{581}, T_{638}, and T_{665}) which are shown as blue or green blocks (blue ones represent online collaboration tasks; green ones represent offline collaboration tasks) [9].

Table 4: ANOVA of task selection behaviors.

(a) Descriptives

					Sample			
	N	Mean	Std. deviation	Std. error	95% confidence interval for mean		Minimum	Maximum
					Lower bound	Upper bound		
1	100	391.520000	23.3949317	2.3394932	386.877938	396.162062	334.0000	451.0000
2	100	373.100000	25.8732341	2.5873234	367.966189	378.233811	293.0000	432.0000
3	100	364.810000	22.5266150	2.2526615	360.340231	369.279769	310.0000	423.0000
Total	300	376.476667	26.3800901	1.5230552	373.479401	379.473932	293.0000	451.0000

(b) Test of homogeneity of variances

Sample			
Levene statistic	df1	df2	Sig.
1.025	2	297	.360

(c) ANOVA

Sample					
	Sum of squares	df	Mean square	F	Sig.
Between groups	37381.487	2	18690.743	32.521	.000
Within groups	170695.350	297	574.732		
Total	208076.837	299			

(d) Multiple comparisons

Dependent variable: sample						
LSD						
(I) group	(J) group	Mean difference (I-J)	Std. error	Sig.	95% confidence interval	
					Lower Bound	Upper Bound
1	2	18.4200000*	3.3903741	.000	11.747800	25.092200
	3	26.7100000*	3.3903741	.000	20.037800	33.382200
2	1	−18.4200000*	3.3903741	.000	−25.092200	−11.747800
	3	8.2900000*	3.3903741	.015	1.617800	14.962200
3	1	−26.7100000*	3.3903741	.000	−33.382200	−20.037800
	2	−8.2900000*	3.3903741	.015	−14.962200	−1.617800

*The mean difference is significant at the .05 level.

During collaboration process, A_{13} collaborates with A_{150} for 7 times (T_{82}, T_{98}, T_{122}, T_{147}, T_{174}, T_{218}, and T_{543}), A_{104} for 5 times (T_{35}, T_{160}, T_{476}, T_{638}, and T_{665}), A_{27} for 2 times (T_{485} and T_{581}), A_4 (T_{484}), A_{113} (T_{131}), A_{176} (T_{419}), and A_{209} (T_{279}) for 1 time, respectively. In this simulation, design agent's bidirectional selection behavior as well as execution behavior in OSD process is described in detail.

Compared to Table 3, tasks (T_{35}, T_{82}, T_{98}, T_{122}, T_{131}, T_{147}, T_{160}, T_{174}, T_{218}, T_{279}, T_{419}, T_{476}, T_{543}, T_{581}, T_{638}, and T_{665}) that are blue blocks are not in selection list of Table 3, indicating that they are not selected by A_{13}. This is because that these tasks are generated by A_{13} itself while executing module tasks but encountering exception for collaboration, corresponding to the blank blocks of module tasks in Figure 3, and that they are not released to public but directly sent to corresponding design agents for online collaboration.

The simulation describes the process of designer's bidirectional selection behavior in microperspective and proves that the model of bidirectional selection is effective in OSD process by simulation analysis with the table and figure aforementioned.

5. Conclusions

5.1. Contributions. It is a tough issue for designers to select tasks completely fit to them in OSD process, which also directly impacts the efficiency of OSD evolution. Thus, designers' bidirectional behavior of task selection integrating passive selection of website recommendation and autonomous selection is modeled in this paper. Passive selection behavior based on website recommendation is modeled with application of collaborative filtering algorithm to recommend tasks to designers by predication, which describes recommendation process based on a three-dimensional matrix including information on design agent, task, and skills; autonomous selection behavior is described in consideration of factors such as skill and incentive; the bidirectional selection model integrates the aforementioned two selection

algorithms to describe designers' selection behaviors who usually select tasks in consideration of both OSD recommendation and autonomous selection. By simulation comparison of bidirectional selection, passive selection based on website recommendation, and autonomous selection with ANOVA, analysis is carried out to show that task selection behavior has significant effect on OSD evolution process and that bidirectional selection behavior is more effective to shorten evolution cycle according to the experiment settings on community scale and task number. In addition, the simulation study shows the process of task selection in microperspective and testifies the model of bidirectional selection.

5.2. Implications. In OSD (e.g., SourceForge, NetBeans) [50, 51], there are millions of projects/products/tasks provided to designers for selection and contribution. For designers, it is really difficult to find right one from mass abundant tasks and also costs time, which may block their contributions to OSD. For OSD, it is a key issue how to guarantee the projects/products/tasks matching to designers. Therefore, the bidirectional selection method is proposed to solve the problems.

On the one hand, it will provide designers much convenience to select right tasks by OSD recommendation. Besides, it will provide designers more personalized services on project/product/task selection including both recommendation and autonomous selection. On the other hand, it will provide OSD managers useful advice on how to match designers and projects/products/tasks in consideration of factors such as skill and incentive, by bidirectional selection simulation, which will help coordinate resources of OSD and promote the development of projects/products/tasks effectively. In addition, the findings prove that agent-based modeling and simulation can be taken as an approach to study the open source design process. In this paper, designer's selection behavior is abstracted by agent modeling, which represents both his/her subjective factors (e.g., incentive) and objective factors (e.g., skill) while making decisions. Meanwhile, the selection process is described by agent simulation, which describes not only individual's decision process, but also OSD's evolution process. As a result, the proposed bidirectional selection method can be applied in practice benefiting both designers and OSD.

5.3. Future Work. Although the bidirectional selection method is proved to provide much convenience in task selection of OSD, there are some limitations to solve. First, the algorithm cannot work effectively if the number of designers and tasks is too huge, because the paper does not provide a corresponding method to solve this problem. Second, more factors should be considered in the bidirectional selection algorithm to help designers select more proper tasks and provide more effective advice on matching tasks and designers as well as promoting OSD evolution.

In future work, intelligence algorithms such as genetic algorithm will be added to solve the problem of large number of tasks and designers. Besides, more simulation experiments, in consideration of more complex factors, such as designers' preference, community scale, and task number, will be designed and carried out to study which selection scheme is more effective to product design as well as OSD evolution.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Authors' Contributions

Shuo Zhang and Yingzi Li contributed equally to this work.

Acknowledgments

This contribution is based upon the research work supported by the National Natural Science Foundation of China no. 71601078, Humanities and Social Sciences of Education Ministry in China no. 16YJC630060, the Social Science Fund of Beijing no. 16GLC070, and the Fundamental Research Funds for the Central Universities no. 2016MS72.

References

[1] Don Tapscott and A. D. Williams, *Wikinomics: How Mass Collaboration Changes Everything*, Portfolio Hardcover, 2006.

[2] E. V. Hippel, *Democratizing Innovation: The Evolving Phenomenon of User Innovation*, vol. 55, MIT Press, pp. 63-78, 2005.

[3] J. H. Panchal and M. Fathianathan, "Product realization in the age of mass collaboration," in *Proceedings of ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pp. 219–229, 2008.

[4] K. Crowston and J. Howison, "Hierarchy and centralization in free and open source software team communications," *Knowledge, Technology & Policy*, vol. 18, no. 4, pp. 65–85, 2006.

[5] C. Kohtala, "Addressing sustainability in research on distributed production: An integrated literature review," *Journal of Cleaner Production*, vol. 106, pp. 654–668, 2015.

[6] G. Müller-Seitz and G. Reger, "Networking beyond the software code? an explorative examination of the development of an open source car project," *Technovation*, vol. 30, no. 11-12, pp. 627–634, 2010.

[7] B. Rolandsson, M. Bergquist, and J. Ljungberg, "Open source in the firm: Opening up professional practices of software development," *Research Policy*, vol. 40, no. 4, pp. 576–587, 2011.

[8] R. J. Davies et al., "A user driven approach to develop a cognitive prosthetic to address the unmet needs of people with mild dementia," *Pervasive & Mobile Computing*, vol. 5, no. 3, pp. 253–267, 2009.

[9] S. Zhang, Y. Li, and X. Zhang, "Agent behavior-based simulation study on mass collaborative product development process," *Mathematical Problems in Engineering*, vol. 2015, Article ID 689383, 10 pages, 2015.

[10] L. Lü, M. Medo, C. H. Yeung, Y. Zhang, Z. Zhang, and T. Zhou, "Recommender systems," *Physics Reports*, vol. 519, no. 1, pp. 1–49, 2012.

[11] J. Lu, D. S. Wu, M. S. Mao, W. Wang, and G. Zhang, "Recommender system application developments: a survey," *Decision Support Systems*, vol. 74, pp. 12–32, 2015.

[12] J. Liu, M. Tang, Z. Zheng et al., "Location-aware and personalized collaborative filtering for web service recommendation,"

IEEE Transactions on Services Computing, vol. 9, no. 5, pp. 686–699, 2016.

[13] Y.-Y. Shih and D.-R. Liu, "Product recommendation approaches: Collaborative filtering via customer lifetime value and customer demands," *Expert Systems with Applications*, vol. 35, no. 1-2, pp. 350–360, 2008.

[14] R. Zhang and T. Tran, "An information gain-based approach for recommending useful product reviews," *Knowledge and Information Systems*, vol. 26, no. 3, pp. 419–434, 2011.

[15] F. Rodrigues and B. Ferreira, "Product recommendation based on shared customer's behaviour," *Procedia Computer Science*, vol. 100, pp. 136–146, 2016.

[16] P. A. Riyaz and S. M. Varghese, "A scalable product recommendations using collaborative filtering in hadoop for bigdata," *Procedia Technology*, vol. 24, pp. 1393–1399, 2016.

[17] M. Zhang, X. Guo, and G. Chen, "Prediction uncertainty in collaborative filtering: Enhancing personalized online product ranking," *Decision Support Systems*, vol. 83, pp. 10–21, 2016.

[18] D. Geiger and M. Schader, "Personalized task recommendation in crowdsourcing information systems—current state of the art," *Decision Support Systems*, vol. 65, no. 1, pp. 3–16, 2014.

[19] M. Nilashi, O. B. Ibrahim, and N. Ithnin, "Hybrid recommendation approaches for multi-criteria collaborative filtering," *Expert Systems with Applications*, vol. 41, no. 8, pp. 3879–3900, 2014.

[20] K. Choi, Y. Suh, and D. Yoo, "Extended collaborative filtering technique for mitigating the sparsity problem," *International Journal of Computers Communications & Control*, vol. 11, no. 5, p. 631, 2016.

[21] R. Katarya and O. P. Verma, "An effective collaborative movie recommender system with cuckoo search," *Egyptian Informatics Journal*, 2016.

[22] H. Li, J. Cui, B. Shen, and J. Ma, "An intelligent movie recommendation system through group-level sentiment analysis in microblogs," *Neurocomputing*, vol. 210, pp. 164–173, 2016.

[23] M. Á. García-Cumbreras, A. Montejo-Ráez, and M. C. Díaz-Galiano, "Pessimists and optimists: improving collaborative filtering through sentiment analysis," *Expert Systems with Applications*, vol. 40, no. 17, pp. 6758–6765, 2013.

[24] S. Wei et al., "A hybrid approach for movie recommendation via tags and ratings," *Electronic Commerce Research & Applications*, vol. 18, pp. 83–94, 2016.

[25] D. Sánchez-Moreno, A. B. Gil González, M. D. Muñoz Vicente, V. F. López Batista, and M. N. Moreno García, "A collaborative filtering method for music recommendation using playing coefficients for artists and users," *Expert Systems with Applications*, vol. 66, pp. 234–244, 2016.

[26] X. Bai, B. Barla Cambazoglu, F. Gullo, A. Mantrach, and F. Silvestri, "Exploiting search history of users for news personalization," *Information Sciences*, pp. 125–137, 2017.

[27] F. Zhao, F. Yan, H. Jin, L. T. Yang, and C. Yu, "Personalized mobile searching approach based on combining content-based filtering and collaborative filtering," *IEEE Systems Journal*, pp. 1–9, 2015.

[28] Y. D. Seo, Y. Kim, E. Lee, and D. Baik, "Personalized recommender system based on friendship strength in social network services," *Expert Systems with Applications*, vol. 69, pp. 135–148, 2017.

[29] A. Shahmohammadi, E. Khadangi, and A. Bagheri, "Presenting new collaborative link prediction methods for activity recommendation in Facebook," *Neurocomputing*, vol. 210, pp. 217–226, 2016.

[30] C. Guo, B. Li, and X. Tian, "Flickr group recommendation using rich social media information," *Neurocomputing*, vol. 204, pp. 8–16, 2016.

[31] N. Zheng and Q. Li, "A recommender system based on tag and time information for social tagging systems," *Expert Systems with Applications*, vol. 38, no. 4, pp. 4575–4587, 2011.

[32] S. Y. Lin et al., "A trustworthy QoS-based collaborative filtering approach for web service discovery," *Journal of Systems & Software*, vol. 93, no. 4, pp. 217–228, 2014.

[33] C. H. Lai and D. R. Liu, "Integrating knowledge flow mining and collaborative filtering to support document recommendation," *Journal of Systems and Software*, vol. 82, no. 12, pp. 2023–2037, 2009.

[34] J. Wei, J. He, K. Chen, Y. Zhou, and Z. Tang, "Collaborative filtering and deep learning based recommendation system for cold start items," *Expert Systems with Applications*, vol. 69, pp. 29–39, 2017.

[35] H. N. Kim. et al., "Collaborative filtering based on collaborative tagging for enhancing the quality of recommendation," *Commerce Research & Applications*, vol. 9, no. 1, pp. 73–83, 2010.

[36] C. C. Chen, Y. H. Wan, M. C. Chung, and Y. C. Sun, "An effective recommendation method for cold start new users using trust and distrust networks," *Information Sciences*, vol. 224, no. 2, pp. 19–36, 2013.

[37] G. Lv, C. Hu, and S. Chen, "Research on recommender system based on ontology and genetic algorithm," *Neurocomputing*, vol. 187, pp. 92–97, 2016.

[38] U. Ul Hassan and E. Curry, "Efficient task assignment for spatial crowdsourcing," *Expert Systems with Applications*, vol. 58, no. C, pp. 36–56, 2016.

[39] P. Brahmbhatt and S. G. Camorlinga, "Epidemiology-based task assignment algorithm for distributed systems," *Procedia Computer Science*, vol. 95, pp. 428–435, 2016.

[40] D. A. Nembhard and F. Bentefouet, "Selection, grouping, and assignment policies with learning-by-doing and knowledge transfer," *Computers and Industrial Engineering*, vol. 79, pp. 175–187, 2015.

[41] M. Akbari and H. Rashidi, "A multi-objectives scheduling algorithm based on cuckoo optimization for task allocation problem at compile time in heterogeneous systems," *Expert Systems with Applications*, vol. 60, no. C, pp. 234–248, 2016.

[42] D. Yun, C. Q. Wu, and Y. Gu, "An integrated approach to workflow mapping and task scheduling for delay minimization in distributed environments," *Journal of Parallel and Distributed Computing*, vol. 84, pp. 51–64, 2015.

[43] B. B. M. Shao, P.-Y. Yin, and A. N. K. Chen, "Organizing knowledge workforce for specified iterative software development tasks," *Decision Support Systems*, vol. 59, no. 1, pp. 15–27, 2014.

[44] J. L. Brown, S. Farrington, and G. B. Sprinkle, "Biased self-assessments, feedback, and employees' compensation plan choices," *Accounting Organizations & Society*, vol. 54, pp. 45–59, 2016.

[45] W. Zhang, S. Zhang, S. Guo, Y. Yang, and Y. Chen, "Concurrent optimal allocation of distributed manufacturing resources using extended teaching-learning-based optimization," *International Journal of Production Research*, vol. 55, no. 3, pp. 718–735, 2016.

[46] C. Yu, T. N. Wong, and Z. Li, "A hybrid multi-agent negotiation protocol supporting supplier selection for multiple products with synergy effect," *International Journal of Production Research*, pp. 1–20, 2016.

[47] Q. Le and J. H. Panchal, "Modeling the effect of product architecture on mass-collaborative processes," *Journal of Computing & Information Science in Engineering*, vol. 11, no. 1, pp. 333–359, 2011.

[48] K. Hölttä, E. S. Suh, and O. D. Weck, "Tradeoff between modularity and performance for engineered systems and products," in *Proceedings of the International Conference on Engineering Design: Engineering Design & the Global Economy Engineers (ICED '05)*, Australia, 2005.

[49] K. Hölttä-Otto and O. de Weck, "Degree of modularity in engineering systems and products with technical and business constraints," *Concurrent Engineering Research and Applications*, vol. 15, no. 2, pp. 113–125, 2007.

[50] O. Temizkan and R. L. Kumar, "Exploitation and exploration networks in open source software development: an artifact-level analysis," *Journal of Management Information Systems*, vol. 32, no. 1, pp. 116–150, 2015.

[51] T. Zhang, J. Chen, G. Yang, B. Lee, and X. Luo, "Towards more accurate severity prediction and fixer recommendation of software bugs," *Journal of Systems and Software*, vol. 117, no. C, pp. 166–184, 2016.