Research Article

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

Shuo Zhang¹,² and Yingzi Li³

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

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-Cumbrañas 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 multilateral 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 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. 

$$DT = (dt_{ij}) \quad (dt_{ij} \in \{0, 1\})$$

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 $T_j$, $dt_{ij} = 0 \Rightarrow dt'_{ij} = 0$.

$$DS = (ds_{ik}) \quad (ds_{ik} \in \{0, 1\})$$

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 $S_k$, $ds_{ik} = 0 \Rightarrow ds'_{ik} = 0$.

$$TS = (ts_{jk}) \quad (ts_{jk} \in \{0, 1\})$$

denotes task-skill 0-1 matrix, in which $ts_{jk} = 1$ means that it requires skill item $S_k$ to execute $T_j$ while $ds_{jk} = 0$ 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)

$$\text{Sim}(D_i, D_{i'}) = \frac{\sum (dt'_{ij} - DT_{ij}) \cdot (dt'_{i'j} - DT'_{i'j})}{\sqrt{\sum (dt'_{ij} - DT_{ij})^2} \cdot \sqrt{\sum (dt'_{i'j} - DT'_{i'j})^2}}.$$

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

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

$$dt_{ij} = \alpha \frac{rt_{ij}}{at_{ij} + rt_{ij}} + (1 - \alpha) \frac{aw_{ij}/at_{ij} + aw_{ij}/rt_{ij}}{aw_{ij}/at_{ij} + aw_{ij}/rt_{ij}},$$

where $rt_{ij}$ is rated completion time of task $T_j$; $at_{ij}$ is the actual completion time of task $T_j$ by $D_i$; $aw_{ij}$ is rated bonus after
completion of task $T_j$; $aw_{ij}$ is actual bonus after completion of task $T_j$ by $D_i$; $\alpha$ is weight, $\alpha \in [0, 1]$.

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

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

$$f_{dt'_{ij}} = \frac{\sum_{D_j} \text{Sim}(D_i, D_j) \left( dt'_{ij} - DT'_{ij} \right)}{\sum_{D_j} \text{Sim}(D_i, D_j)}.$$  (3)

After the calculation, task $T_j$, whose predicted evaluation value $f_{dt'_{ij}}$ 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)

$$e_{ij} = \frac{aw^T_{ij} \times aw^T_{ij}}{rt_{ij} \times aw^D_{ij}} = \left(\frac{aw^T_{ij}}{rt_{ij}}\right)^2,$$  (4)

s.t. $\sum_{k=1}^l ds_k t_{jk} \geq \Delta^d_k$,

where $aw^D_{ij}$ is cumulative bonus of $D_i$ and $\Delta^d_k$ 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)

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

where $r_{ij}$ is similarity between $D_i$ and $T_j$; $e_{ij}$ is incentive coefficient to execute the task as formula (4); $\beta$ is weight, $\beta \in [0, 1]$.
The calculation of $r_{ij}$ is as formula (6)

$$r_{ij} = \frac{\sum_{k=1}^{l} (ds_{jk} \cdot ts_{jk}) (ds'_{jk} - \overline{DS}^{ij}) (ts'_{jk} - \overline{TS}^{ij})}{\sum_{k=1}^{l} (ds_{jk} \cdot ts_{jk}) (ds'_{jk} - \overline{DS}^{ij})^2 \sum_{k=1}^{l} (ds_{jk} \cdot ts_{jk}) (ts'_{jk} - \overline{TS}^{ij})^2}$$

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

where $\overline{DS}^{ij}$ is mean value of $D_i$ on $S_k$ ($ds_{jk} = 1, ts_{jk} = 1$), $\overline{DS}^{ij} = \sum_{k=1}^{l} ds_{jk} \cdot ts_{jk} / \sum_{k=1}^{l} ds_{jk} \cdot ts_{jk}$, and $\overline{TS}^{ij}$ is mean requirement of $T_j$ on $S_k$ ($ds_{jk} = 1, ts_{jk} = 1$), $\overline{TS}^{ij} = \sum_{k=1}^{l} ds_{jk} \cdot ts_{jk} / \sum_{k=1}^{l} ds_{jk} \cdot ts_{jk}$.

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 $ST_j$ 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 $T_j$ ($i = 0, 1, 2, \ldots, 19$), 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 $T_j$ ($i > 19$), 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 $S1, S2$, and $S3$ 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.

$$H_0: \mu_1 = \mu_2 = \mu_3 \text{ (task selection behavior has no effect on evolution cycle of OSD)}$$
Figure 2: Module task relationship of the scenario.

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

<table>
<thead>
<tr>
<th>Task ID</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
</table>

Table 2: Partial parameters of simulation experiment.

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Parameter setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scenario</td>
<td>SI, S2, S3</td>
</tr>
<tr>
<td>Community scale (number of agents)</td>
<td>200</td>
</tr>
<tr>
<td>Task selection method for each scenario</td>
<td>SI: passive behavior of task selection by recommendation</td>
</tr>
<tr>
<td></td>
<td>S2: autonomous selection behavior</td>
</tr>
<tr>
<td></td>
<td>S3: bidirectional selection behavior</td>
</tr>
<tr>
<td>Simulation time</td>
<td>300 (100/scenario)</td>
</tr>
</tbody>
</table>

H1: \( \mu_1 \neq \mu_2 \neq \mu_3 \) (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 SI) is significantly longer than that of S2 and S3. Data analysis of evolution process in SI 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 SI’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 SI 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.
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 S3. 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 ($T_2$, $T_6$, $T_0$, $T_1$, $T_6$, $T_14$, $T_15$, $T_17$, $T_{18}$, and $T_{19}$) which are shown as gray blocks, and 18 collaboration tasks ($T_{35}$, $T_{82}$, $T_{96}$, $T_{122}$, $T_{131}$, $T_{147}$, $T_{160}$, $T_{174}$, $T_{218}$, $T_{279}$, $T_{419}$, $T_{476}$, $T_{485}$, $T_{543}$, $T_{581}$, $T_{638}$, 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

<table>
<thead>
<tr>
<th>Sample</th>
<th>N</th>
<th>Mean</th>
<th>Std. deviation</th>
<th>Std. error</th>
<th>95% confidence interval for mean</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Lower bound</td>
<td>Upper bound</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>100</td>
<td>391.520000</td>
<td>23.3949317</td>
<td>2.3394932</td>
<td>386.877938</td>
<td>396.162062</td>
<td>334.0000</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>373.100000</td>
<td>25.8732341</td>
<td>2.5873234</td>
<td>367.966189</td>
<td>378.233811</td>
<td>293.0000</td>
</tr>
<tr>
<td>3</td>
<td>100</td>
<td>364.810000</td>
<td>22.5266150</td>
<td>2.2526615</td>
<td>360.340231</td>
<td>369.279769</td>
<td>310.0000</td>
</tr>
<tr>
<td>Total</td>
<td>300</td>
<td>376.476667</td>
<td>26.3800901</td>
<td>1.5230552</td>
<td>373.479401</td>
<td>379.473932</td>
<td>293.0000</td>
</tr>
</tbody>
</table>

(b) Test of homogeneity of variances

<table>
<thead>
<tr>
<th>Levene statistic</th>
<th>df1</th>
<th>df2</th>
<th>Sig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.025</td>
<td>2</td>
<td>297</td>
<td>.360</td>
</tr>
</tbody>
</table>

(c) ANOVA

<table>
<thead>
<tr>
<th>Sample</th>
<th>Sum of squares</th>
<th>df</th>
<th>Mean square</th>
<th>F</th>
<th>Sig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Between groups</td>
<td>37381.487</td>
<td>2</td>
<td>18690.743</td>
<td>32.521</td>
<td>.000</td>
</tr>
<tr>
<td>Within groups</td>
<td>170695.350</td>
<td>297</td>
<td>574.732</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>208076.837</td>
<td>299</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(d) Multiple comparisons

<table>
<thead>
<tr>
<th>(I) group</th>
<th>(J) group</th>
<th>Mean difference (I-J)</th>
<th>Std. error</th>
<th>Sig.</th>
<th>95% confidence interval</th>
<th>Lower Bound</th>
<th>Upper Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>18.4200000*</td>
<td>3.3903741</td>
<td>.000</td>
<td>11.747800</td>
<td>-25.092200</td>
<td>574.732</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>26.7100000*</td>
<td>3.3903741</td>
<td>.000</td>
<td>20.037800</td>
<td>-33.382200</td>
<td>14.962200</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>-26.7100000*</td>
<td>3.3903741</td>
<td>.000</td>
<td>-33.382200</td>
<td>-14.962200</td>
<td>14.962200</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>-8.2900000*</td>
<td>3.3903741</td>
<td>.015</td>
<td>-14.962200</td>
<td>-1.617800</td>
<td>14.962200</td>
</tr>
</tbody>
</table>

*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_{581}, T_{131}, T_{147}, T_{160}, T_{476}, T_{543}, T_{419}, 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


Submit your manuscripts at https://www.hindawi.com