A Novel Intelligent Tutoring System For Learning Programming

Meng Wang
School of Computer Science and Engineering
Beihang University
Beijing, China
wangmeng@nlsde.buaa.edu.cn

Wenjun Wu, Yu Liang
School of Computer Science and Engineering
Beihang University
Beijing, China
wwj@nlsde.buaa.edu.cn, liangyu@nlsde.buaa.edu.cn

Abstract—**The goal of this paper is to propose the concept, structure and implementation of a novel intelligent tutoring system designed for beginners in C language and Python. The system is implemented by adding the functions of code classification, program error repair and personal knowledge tracing to an online programming practice platform. This implementation makes the original platform become more intelligent and help students learn programming better.**

Keywords—intelligent tutoring system; code classification; program error repair; personal knowledge tracing

I. INTRODUCTION

With the rapid development of computer technology, programming has become a necessary skill for a qualified college student, especially for students majoring in computer science. Therefore, the relevant colleges of information technology in the University pay great attention to the cultivation of students' programming ability, and set up many courses related to programming language, such as C language and Python.

In addition to the teaching of theoretical knowledge, programming practice is also a necessary part of this kind of course, and training students through online judge (OJ) system is a widely used way. Through OJs, students can submit their own programs, and then the OJs will compile and execute the source code, and compare the running result with the pre-designed test cases to check the correctness of the code, and provide the feedback to students. However, the function of common OJs at present is too simple, only supports the detection of code correctness, it is difficult to give more personalized feedback.

Therefore, if an OJ can be more intelligent, such as having functions of classifying the answer codes of the problem and visualizing them clearly, and giving corresponding hints or repair methods for program errors, and tracing students' mastery of different knowledge points, the OJ will be more educationally helpful to the beginners. This is why we call our system "a novel intelligent tutoring system". For program beginners, this implementation will make them open their mind, understand their mistakes in time, and have a clear understanding of their learning. As a result, their learning efficiency can be improved. On the other hand, for the teachers, they can know the learning states of different students better by using the feedback of the OJ, which will help them to improve their teaching methods as well.

This paper is structured as follows. The first section contains an introduction to the issues and possible improvements of existing OJ systems, and section II presents the related work about the topic. Section III describes the system architecture, and section IV presents the methods and implementation of the system. Section V shows some figures of the user interfaces and section VI presents the experimental results. Section VII describes the conclusion and future research directions.

II. RELATED WORK

OJ is useful for students majoring in computer science to practice programming. However, most of OJ's judgments on the codes are binary: "accepted" or "not accepted". This may be appropriate for experienced students, but for beginners, it may be more helpful to give more feedback [1].

Intelligent tutoring system (ITS) may solve this problem. ITS refers to any computer program that can be used for learning and contains intelligence [2], which rose in the 1950s and gained the attention of developed countries in the 1970s. Since the 1990s, it has developed comprehensively and rapidly. Traditional ITS consists of four parts: expert model, student model, teaching model and learning environment or user interface [2]. Its biggest advantage is that it can teach students according to their aptitude, and provide personalized guidance for different students, so as to improve students' independent learning ability and learning efficiency. In recent years, ITS has made further development in human-computer interaction, virtual reality and other aspects, and is used more and more widely.

It is of great significance for the program beginners to make traditional OJ have more intelligent functions such as code classification, program error repair, knowledge tracing, etc. In terms of code classification, CodeWebs [3] can classify different sub trees of abstract syntax tree based on the program running results of same input, and the visual system OverCode [4] uses the combination of lightweight program analysis and manual rewriting rules to classify the codes submitted by students. In terms of program repair, Gopinath et al. put

National Key Research and Development Program of China.

978-1-7281-6870-8/20/$31.00 ©2020 IEEE

forward a method in [5] which can repair programs that manipulate complex structured data. Könighofer et al. proposed automatic error location based on symbol execution and model diagnosis as well as automatic program correction based on template in [6]. However, these methods are designed to fix large programs, not small but complex errors occurring in program beginners' codes. In contrast, the method proposed in this paper uses dynamic analysis to achieve scalability, which is more accurate. In addition, there is also a repair method based on program mutation [7] and genetic programming [8], by combining mutation and genetic operators, and then selecting an appropriate repair strategy based on fitness function. However, due to the huge search space of mutation, the efficiency of this method is not high.

In terms of modeling students' knowledge, the most commonly used model is the Knowledge Tracing (KT) model proposed by Corbett and Anderson in [9] in 1994. This model uses the hidden Markov method to infer whether the students have mastered a certain knowledge point by observing the performance of a series of answers [10]. Because the model can accurately infer the learning of student on specific knowledge point and predict the correctness of student's next answer, it is widely used by most ITS. In detail, the KT model assumes that the students' knowledge state is a binary variable: not mastered (0) and mastered (1), then under the whole learning system, students' knowledge mastery is a set of binary variables. The model updates the probability distribution of students' knowledge mastery by observing whether they answer the questions correctly or not. Besides, the KT model also assumes that there are four parameters for each knowledge point: two knowledge parameters and two performance parameters. The two knowledge parameters are initial knowledge rate $P(L_0)$ and learning rate $P(T)$. The initial knowledge rate refers to the probability that students master the knowledge point before learning on the tutoring system. Learning rate refers to the probability that the students master the knowledge point after learning which is not mastered before. The two performance parameters are: guessing rate $P(G)$ and slip rate $P(S)$. Guessing rate refers to the probability that students can answer the questions correctly by guessing even if they do not have the knowledge in advance. The slip rate refers to the probability that students make a wrong answer even though they have mastered the knowledge in advance. In addition, the model assumes that students will not forget, that is, knowledge points will not change from the mastered state to the not mastered state.

In recent years, with the limitation of the traditional Knowledge Tracing model becoming more and more prominent, many extended models have also appeared. In [11], Pardos and Heffernan invented the KT-IDEM model by adding difficulty nodes to the traditional model, which achieves better results than the traditional model in some data sets. Beck et al. proposed the HELP model in [12] by measuring the influence of teacher's help on students' answers. However, the "help" measured by the model is proposed by students to their teachers. The higher the students' knowledge level, the lower their willingness to ask for help, which will affect the accuracy of the model. In addition, many researches try to personalize the parameters in the traditional model. For example, Pardos and Heffernan put forward the Prior Per Student model in [13], which has one more polynomial node representing the future ability of students than the traditional model, and it has been proved more accurate. Yudelson et al. proposed to expand the traditional model based on the specific learning probability of students in [14], and the experiment shows that the method is indeed effective. Besides, Baker and Corbett put forward an innovative method in [15], which can judge whether a student has the behavior of guessing the answer from the context, so as to avoid the influence of recognizability and model degradation caused by uncertainty. The results show that the method improves the accuracy and reliability significantly compared with the traditional model.

III. Architecture of the System

The architecture of the proposed novel intelligent tutoring system is shown in Fig. 1. The front-end of the system is implemented with the Vue.js framework, html, css and some UI libraries. And the back-end is implemented with the Django framework and python. The front and back ends communicate through Ajax. The database of the system is PostgreSql and Object Relation Mapping (ORM) framework is used to operate the data.

The general running mechanism is that after the front-end initiates some requests to the back-end, the back-end queries the corresponding data from the PostgreSQL database, and then returns to the front-end after analyzing and processing the data. The front-end then visualizes the returned data through some UI libraries such as Echarts, iView and Element UI so as to present the feedback to the users in a vivid manner.

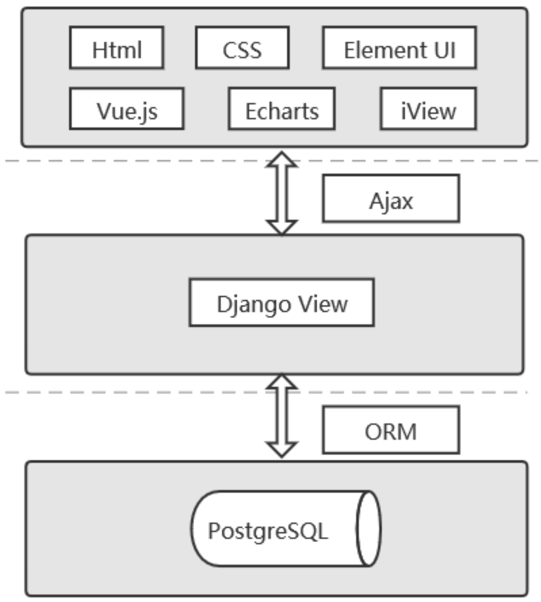

Fig. 1. Architecture of the system

The proposed novel intelligent tutoring system consists of the following components (Fig. 2):

- The front-end module displays the system webpages, including problem page, competition page, code submission page, etc. At the same time, the system administrator can set the system through the management page.
- The back-end module receives requests from the front-end, then executes the relevant queries on the database, analyzes and processes the data and returns it to the front-end for display.
- The PostgreSQL module stores the system data, such as problem data, code data, user data and so on.
- The judging module judges the codes submitted by the user and then stores the results in the database.
- The code classification module obtains the accepted codes of a certain problem from the database and classifies them and stores the results in the database.
- The error repair module repairs the error codes, and gives some feedback such as the hints and repair methods to the users.
- The knowledge tracing module obtains the users' historical data of programming from the database, and then stores the users' probability of mastering the knowledge points into the database after analyzing and processing the data.

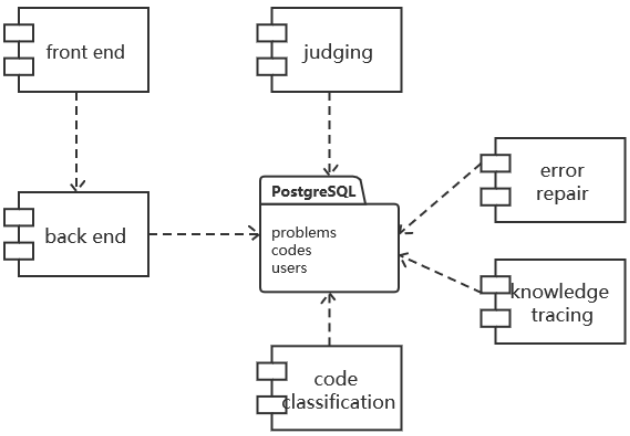

Fig. 2. Components of the system

IV. Theory, Methods and Implementation of the Designed System

First of all, we assume that for a certain problem, the accepted codes for the problem has been stored in the database of the system. Then, by dynamic program analysis, these accepted codes are automatically divided into several classes. In each class, we will select a program as the specification of the class, which matches the rest of the codes in the class. For an existing error program that fails to be accepted, we will run the repair algorithm on the error program and each specification respectively, then choose the most appropriate method to repair the error, and give the corresponding feedback. The operation process is shown in Fig. 3.

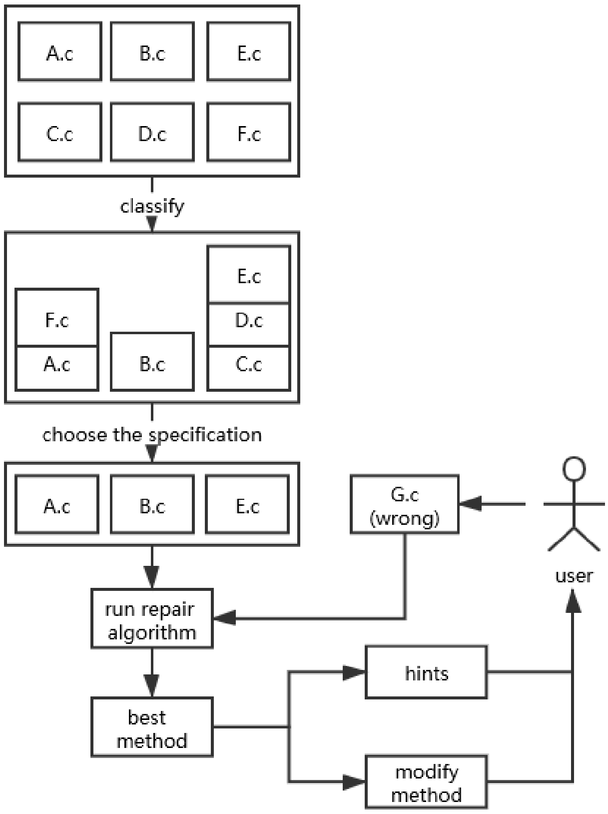

Fig. 3. Process of the code classification and error repair

A. Code Classification

In our system, we use the CLARA engine [16] to classify the codes. The classification algorithm used by the CLARA engine is mainly based on program matching. Specifically, if program P and Q can be matched, then they belong to the same class, and the following conditions need to be satisfied: 1) P and Q have the same control-flow structure, which means the abstract syntax tree structures of program P and Q are the same; 2) There is a bijective relation between the variables of program P and Q, such that the related variables have the same values, in the same order, during the running of program P and Q based on the same set of inputs [16]. By using these rules, the CLARA engine can quickly and accurately analyze the structures of a group of sample programs, so as to judge whether they match and achieve the purpose of classification. We extend CLARA by visualizing the classification result of a given problem and present it as a tree diagram, which will be introduced in the next section.

B. Error Repair

After the classification, a program set is generated, which consists of specifications of each class. Then we use the modified CLARA engine to repair the error programs. Fisrt, the engine run the repair algorithm on the wrong program and each specification respectively, and then a series of corrections and repair costs will be generated each time. The corrections here include adding, deleting and modifying some variables or expressions in the error program, and they will not change the control flow structure. As for the repair cost, it is obtained by calculating the tree edit distance of different abstract syntax trees before and after the error program is modified. After the repair algorithm works on all specifications and the error program one by one, the repair engine will choose the best repair method with the lowest cost to modify the error program, and give corresponding feedback by describing the error location and specific modification.

C. *Knowledge Tracing*

Another function of the system is knowledge tracing. We use the KT model, which is introduced in related work, to trace students' mastery of different knowledge points.

As shown in Fig. 4, students' mastery of knowledge points is constantly changing in the process of answering questions. Specifically, K represents the knowledge node, with two states: mastered (1) and not mastered (0). Q represents the question node, with two states: right answer (1) and wrong answer (0). $P(L_0)$, P(T), P(G) and P(S) are the four parameters in KT model. According to the model, we process students' answer sequences and present the result as a curve graph, which is easy to understand.

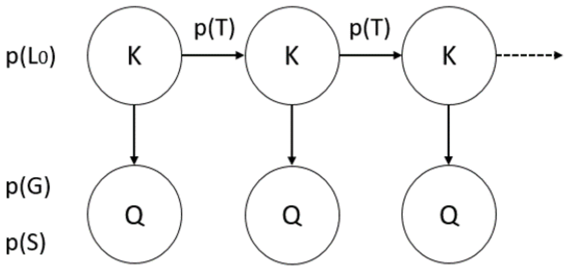

Fig. 4. Knowledge tracing model

As for the implementation, the front-end of the system adopts the Vue.js framework. Vue.js is a progressive framework and it can be used to build beautiful user interfaces. Its design idea is bottom-up and incremental development, which is more open and flexible in actual use. In addition, the front-end also uses the component libraries based on Vue.js such as iView, element UI and Echarts.

The back-end of this system is developed in python. As a high-level programming language, python is easy to use, supports object-oriented programming, provides dynamic data types and various library functions to complete complex programming tasks. In addition, python programs have good scalability and portability.

The overall framework of the back-end is Django. Django is an open source framework that can be used to quickly build high-performance and elegant websites. It follows the Model-Template-View (MTV) development mode, which makes the development easier.

The database used is PostgreSQL. PostgreSQL database is an object-relational database, which supports rich data types, such as common integer, boolean, character types, and large objects stored in binary form, including pictures, audio, video, as well as JSON type, array type and custom type data. At the same time, the database is a complete transaction security database, which supports foreign keys, subqueries, data integrity checks, views, triggers and stored procedures.

As for the background data operations, the system adopts the Object Relation Mapping (ORM) framework. The ORM framework associates the objects in the program with the database by describing the mapping relationship between the objects and the database. For example, ORM associates the class name in the program with the table name in the database, and associates the class property with the table field in the database. The advantage of using ORM framework is that we can add, delete, query and modify the data in the database without caring about which database is used at the bottom of the system, only need to operate the objects in the program.

V. User Interface of the System

This part shows some user interfaces of the novel intelligent tutoring system.

A. *Code Classification*

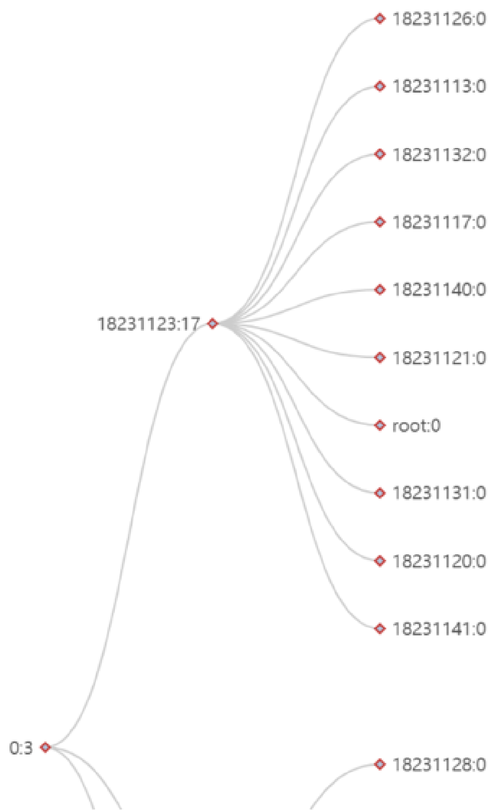

Fig. 5. Visualization of code classification result

The results of the code classification are shown in Fig. 5. The tree type in Echarts is used to achieve visualization. The tree extends from left to right. The leftmost node is the root node, which represents the problem, while other nodes represent the accepted codes of the problem. Assuming that the number of layers of the root node is 0, then each node of the first layer represents the specification of each class of the problem's accepted codes, and the child nodes of the specification belong to the same class.

B. *Error Repair*

As is shown in Fig. 6, the above is the error code submitted by the user, and the following is the system's feedback after the repair. This page has two buttons: hint and repair. As the name implies, the hint button can be used to view the error program modification hint, which is not an explicit answer, while the repair button can be used to view the specific repair methods.

Fig. 6. Error repair feedback

C. *Knowledge Tracing*

An example of learning curves generated by the KT model is shown in Fig. 7, which represents the student's learning trace of a certain knowledge point. The abscissa of the graph represents the programming exercises done by the student in the process of learning, and the ordinate represents the specific value of the probability that the student grasps the knowledge point, ranging from 0 to 1. Each point on the graph represents the posteriori probability of the student's mastery of the knowledge point after completing the corresponding exercise. The last point can be the final probability of the student's mastery of the knowledge point.

Through the learning curve, students can clearly understand their learning process, so as to practice more pertinently in the following learning. In addition, teachers can find the learning characteristics of each student, so as to improve the teaching methods and achieve personalized teaching.

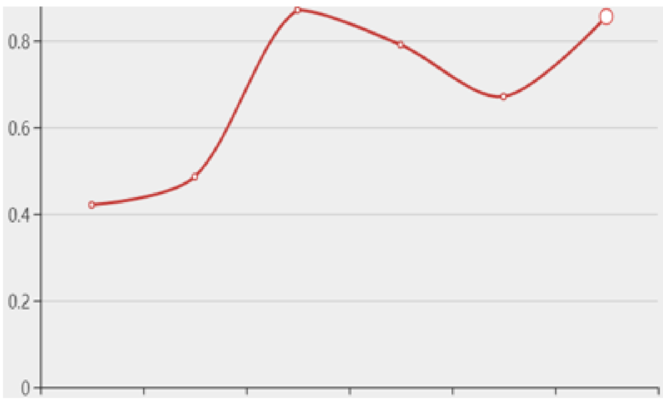

Fig. 7. Knowledge point learning curve

VI. Experimental Results

In order to evaluate the effectiveness of our system, we obtained the dataset from an introductory python programming course offered at EduCoder. This dataset includes 6885 programs submitted by 1037 students through six assignments. The collected programs include not only correct submissions but also the wrong versions. Because the selected assignments are for program beginners, the programs are simple, and the accuracy of program classification are very high. Thus, we focus on the analysis of the efficiency of error program repair.

We use our ITS to fix students' wrong programs and the repair results are in Table I. Similarly, because of the simplicity of the assignments and programs, the system can easily fix the most of wrong programs with a high time efficiency.

TABLE I. Program Repair Results

Assignment	# Wrong program	Fix rate	Time
1	714	97.34%	1.6s
2	74	94.59%	1.8s
3	64	93.75%	1.7s
4	221	90.05%	2.1s
5	640	85.16%	4.0s
6	327	88.38%	3.3s

VII. Conclusion and Future Work

In this paper, we introduced a novel intelligent tutoring system for learning programming, which has the functions of code classification, program error repair and knowledge tracing. The code classification function can help students know different solutions of the same problem, and the error repair function can help students find out the error of their codes in time. As for the knowledge tracing function, it can help students have a clear understanding of their learning.

The main contribution of the paper is that we designed and implemented a novel ITS for learning programming, which uses the improved CLARA engine and KT model. Our ITS is more intelligent and educationally helpful than the existing OJ systems, and it can help beginners to improve their programming skills better.

In the future, we can make improvements in the following aspects:

First, the error repair function needs to be further optimized. At present, the feedback generated by the system is similar to the intermediate language and the readability is poor. This can be improved by using deep learning and natural language processing methods to produce more concise and readable feedback.

Second, the function of knowledge tracing can be more accurate. KT model is currently used in the system, but there are many defects in the model, such as it is only suitable for a single knowledge point, and it does not consider the personalized factors of different students. Therefore, the model can be improved later, such as using deep knowledge tracing model [17], automatic temporal cognitive framework [18], personalized factor model [19] or introducing multiple knowledge points and personalized factors of different students into the KT model, so as to describe students' learning more accurately.

At last, peer assessment [20] can also be added to the system. As a result, students can learn from each other and make progress together.

Acknowledgment

This work is supported in part by the National Key Research and Development Program of China (Funding No. 2018YFB1004502), the National Natural Science Foundation of China (Grant No. 61532004) and the State Key Laboratory of Software Development Environment (Funding No. SKLSDE-2017ZX-03).

References

[1] Mani A, Venkataramani D, Petit Silvestre J, et al. Better feedback for educational online judges[C]//Proceedings of the 6th International Conference on Computer Supported Education, Volume 2: Barcelona, Spain, 1-3 April, 2014. SciTePress, 2014: 176-183.

[2] Freedman R, Ali S S, McRoy S. Links: what is an intelligent tutoring system?[J]. intelligence, 2000, 11(3): 15-16.

[3] Nguyen A, Piech C, Huang J, et al. Codewebs: scalable homework search for massive open online programming courses[C]//Proceedings of the 23rd international conference on World wide web. ACM, 2014: 491-502.

[4] Glassman E L, Scott J, Singh R, et al. OverCode: Visualizing variation in student solutions to programming problems at scale[J]. ACM Transactions on Computer-Human Interaction (TOCHI), 2015, 22(2): 7.

[5] Gopinath D, Malik M Z, Khurshid S. Specification-based program repair using SAT[C]//International Conference on Tools and Algorithms for the Construction and Analysis of Systems. Springer, Berlin, Heidelberg, 2011: 173-188.

[6] Könighofer R, Bloem R. Automated error localization and correction for imperative programs[C]//Proceedings of the International Conference on Formal Methods in Computer-Aided Design. FMCAD Inc, 2011: 91-100.

[7] Debroy V, Wong W E. Using mutation to automatically suggest fixes for faulty programs[C]//Software Testing, Verification and Validation (ICST), 2010 Third International Conference on. IEEE, 2010: 65-74.

[8] Forrest S, Nguyen T V, Weimer W, et al. A genetic programming approach to automated software repair[C]//Proceedings of the 11th Annual conference on Genetic and evolutionary computation. ACM, 2009: 947-954.

[9] Corbett A T, Anderson J R. Knowledge tracing: Modeling the acquisition of procedural knowledge[J]. User Modeling and User-Adapted Interaction, 1994, 4(4):253-278.

[10] Trifa A , Hedhili A , Chaari W L . Knowledge tracing with an intelligent agent, in an e-learning platform[J]. Education and Information Technologies, 2019, 24(1):711-741.

[11] Pardos Z A , Heffernan N T . KT-IDEM: Introducing Item Difficulty to the Knowledge Tracing Model[M]// User Modeling, Adaption and Personalization. Springer Berlin Heidelberg, 2011.

[12] Beck J E, Chang K M, Mostow J, et al. Does Help Help? Introducing the Bayesian Evaluation and Assessment Methodology[C]// International Conference on Intelligent Tutoring Systems. 2008.

[13] Pardos Z A, Heffernan N T. Modeling Individualization in a Bayesian Networks Implementation of Knowledge Tracing[J]. 2010.

[14] Yudelson M V, Koedinger K R, Gordon G J. Individualized Bayesian Knowledge Tracing Models[C]// International Conference on Artificial Intelligence in Education. 2013.

[15] Baker R S, Corbett A T, Aleven V. More Accurate Student Modeling through Contextual Estimation of Slip and Guess Probabilities in Bayesian Knowledge Tracing[J]. Lecture Notes in Computer Science, 2008, 5091:406-415.

[16] Gulwani S, Radiček I, Zuleger F. Automated clustering and program repair for introductory programming assignments[C]//Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation. ACM, 2018: 465-480.

[17] Piech C, Bassen J, Huang J, et al. Deep knowledge tracing[C]//Advances in neural information processing systems. 2015: 505-513.

[18] Pu Y, Wu W, Jiang T. ATC Framework: A fully Automatic Cognitive Tracing Model for Student and Educational Contents[J].

[19] Liang Y, Wu W, Wu L, et al. Inferring How Novice Students Learn to Code: Integrating Automated Program Repair with Cognitive Model[C]//CCF Conference on Big Data. Springer, Singapore, 2019: 46-56.

[20] Han Y, Wu W, Ji S, et al. A Human-Machine Hybrid Peer Grading Framework for SPOCs[J]. International Educational Data Mining Society, 2019.