Models and Methods for Computer Support of Adaptive Training of Algorithmic Tasks Solution

Andrey Chukhray[1] [0000-0002-8075-3664], Olena Havrylenko[2] [0000-0001-5227-9742
Valeriy Mygal[3] [0000-0003-3622-5423], Galyna Mygal[4] [0000-0002-9862-9338]

National Aerospace University "Kharkiv Aviation Institute", Kharkiv, Ukraine
[1] achukhray@gmail.com, [2]o.havrylenko@khai.edu,
[3]valeriymygal@gmail.com, [4]mygal.galina@gmail.com

Abstract. Based on the approach to the rational control of engineering systems in conditions of uncertainty, created by Professor A. Kulik, models and methods of adaptive computer support of getting knowledge and skills in algorithmic academic tasks solving were developed. There were proposed three modes of such systems functioning: demonstration, solving with hints and test, which organized in two stages of training - calculations according to certain algorithm and algorithm development according to certain task statement. The models of algorithmic academic task, the method of calculation and generation of the tasks statements variants, the student model, the method of automatic generation of diagnostic models of missing operands, the data model, the model of tutoring and methods of trainee program diagnosis are described in the paper. Proposed models and methods allow to structure and formalize training process, as well as to improve such algorithms characteristics as occupied memory and calculation time in order to provide adaptive computer support of tutoring.

Keywords: Algorithmic Task, Knowledge and Skills, Diagnostic Model, Student Model

1 Introduction

From the system viewpoint the training management process at secondary schools and at universities are characterized by a number of factors that affect negatively on the quality of the trainees learning. Among them there are disturbances acting on a trainee and a trainer, weak professional and pedagogical preparation of a certain mentors, low basic level of students' knowledge and skills, and lack of students motivation to learn everything carefully and deeply in "toxic" information environment that we can observe in modern society. Additionally, in conditions of "mass production" traditional education forms cannot be really adaptive, thus new information technologies widely used in the training process. Each big modern college and university has their own or adapts one of the existing Learning Management System (or Virtual Learning Environment, which is another name of a such systems), such as, for instance Moodle, ATutor, Google Education 360, etc. They usually are used to organize and guide "trainer-student" and "student-content" interactions, including possibility of distance

CEUR-WS.org/Vol-2740/20200408.pdf

Copyright © 2020 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

education. If we talk about monitoring and guiding students' behavior then China is a good example of using eye-trackers and other devices and software tools for teacher-pupils communication enhancing.

In Ukraine experiment with electronic tutorials implementation in schools was not successful for some technical reasons, but the problems can be solved with system approach and applying of experience such countries like Finland, India, Romania, some USA universities. Promising way is creation of integrated educational environments with adaptive support tutoring tools and versatile extensible content. Such systems should have built-in intelligence to simulate the activity of the best tutors aimed to obtain knowledge and skills.

In the National Aerospace University "KhAI" at the department of Aircraft Control Systems, the development and implementation of the Intelligent Tutoring Systems (ITS) have been successfully providing since 2004. Development is based upon approach proposed by Professor A. Kulik to the rational control of technical objects on the basis of the deep diagnosis and recovery [1].

2 Problem statement

There is specific area of tutoring tasks in secondary, high and professional education that can be defined as Algorithmic Tasks (AT). The essential feature of such tasks is a multi-step process, where each step is an activity (calculation, object operation, choice) and all together they implement some algorithm (in math and computer science sense). In this case we can link each step with certain components of knowledge and skills from the set, which cover all required competence for the AT whole class solution in a given problem area.

In order to develop effective tools for Adaptive Tutorial System (ATS) design and implementation in AT area, the new scientific basis is proposed as combination of modern ITS principles [2-4] and rational control approach for uncertainty conditions [1].

3 Models and methods of adaptive training

Considering an effective teaching trajectory for solving laboratory and practical task in engineering three main stages should be separate: a sample task solving demonstration by tutor, another tasks solving by trainee with the trainer's pedagogical support and, finally, some control task trainee solution without any hints, but with estimation. Thus, ATS should provide three modes of training: "demo", "supported solving" and "test".

In the ITS development area there are three aspects which are usually considered in modeling and formalization: a task (subject of study), a process (pedagogical techniques and algorithms) and a trainee (student behavior during interaction with ITS). Further some methods and models for these three computer tutoring components will be described in details.

3.1 Generalized task model and task generation method

Let the general model of AT is represented as an ordered sequence:

$$ModelOfCT1 = (condct1,\ objct1_1,\ objct1_2,\ ...,\ objct1_n), \qquad (1)$$

where $condct1$ – given problem statement, $objct1_i$ – an object consisting of seven components: uid, $name$, $type$, $value$, $format$, alg, $note$, where uid – unique ID of the object, $name$ – an object identifier, $type$ – value type (e.g., number, array etc.), $value$ – an object value, $format$ – a value format (e.g., string, float, etc.), alg – a value calculation algorithm, $note$ – an object description.

Then for AT solution a trainee form a tuple of n objects, where $objct1_{n-l},\ objct1_{n-l+1}, ..., objct1_n$ – solution components, $l \in \{1, 2,\ ...,\ k\}, k < n$, the remaining objects are intermediate. It is need to mention, that the same $note$ may be present in different neighbor objects of the same tasks, for example, in the case of a data collection (array, matrix, etc.). In addition, all objects of a data collection can also be characterized by the same algorithm for calculating the value.

Let form a set $OBJ = \{obj1_1,\ obj1_2,\ ...,\ obj1_n\}$ from the values of $objct1_i$ included in sequence (1). As well let put in corresponding to each $obj1_i$ one element of a set $IE\_CT1\_D = \{iect1d_1,\ iect1d_2,\ ...,\ iect1d_n\}$ which consists of window form input elements (fields for entering data).

Then in the demo mode each $iect1d_i$ is filled in with the corresponding $obj1_i$. In the both supported training and test modes another set $OBJS = \{objs1_1,\ objs1_2,\ ...,\ objs1_n\}$ is constructed from the values entered by trainee in the corresponding $iect1d_i$.

To provide possibility of generating source data and solutions, the relationship between $objct1_i$ must be specify. Thus, each $objct1_i$ may be preceded by zero, one or more $objct1_j$, whose values must be calculated by a certain algorithm. To formalize such relationships graph theory can be used and directed graph $G = (OBJ,\ F)$ can be built.

Let each vertex is assigned with ordinal function defined for a graph without loops. Then following subsets can be defined for the graph vertices:

$$\begin{gathered}
B_0 = \{X_i \mid X_i \in E, \Gamma^{-1} X_i = \varnothing\}, \\
B_1 = \{X_i \mid X_i \in E - B_0, \Gamma^{-1} X_i \subset B_0\}, \\
B_2 = \{X_i \mid X_i \in E - (B_0 \cup B_1), \Gamma^{-1} X_i \subset B_0 \cup B_1\}, \\
\dots \\
B_r = \{X_i \mid X_i \in E - \bigcup_{k=0}^{r-1} B_k, \Gamma^{-1} X_i \subset \bigcup_{k=0}^{r-1} B_k\},
\end{gathered} \qquad (2)$$

where r is the smallest integer such that $\forall X_i \in B_r : \Gamma X_i = \varnothing$. Then a function $O(X)$, which is defined by the expression $X_i \in B_k \Rightarrow O(X_i) = k$, is an ordinal function graph without loops, and a subsets $B_k, k = \overline{0,r}$, which form a partition of the original set of vertices G, are levels. To find the graph levels G the method by Demukron can be used [2].

Suppose that 0-level objects, which are independent, must be generated, and objects of all other levels, excepted 0-level (i.e. dependent objects) must be calculated. Then it is possible to describe a method for generation and calculation services of ATS, where conjunctive allow is calculated as followed:

1. Select random 0-level object Y, on which the X depends.
2. Form the set of 0-level objects Y, on which the X depends both directly and transitively: $A = \{Y \mid Y \in B_0, X\Gamma^g Y\}, g\epsilon\{1,2,\ldots,N\}$, $\Gamma^1 Y = \Gamma Y, \Gamma^2 Y = \Gamma\{\Gamma Y\}, \Gamma^3 Y = \Gamma\{\Gamma\{\Gamma Y\}\}, \ldots$ Randomly select a single object from the set A: Y = random (A).
3. Generate new value for Y as $\prod_{\text{value}} \mathrm{Y}$ by the algorithm. $\prod_{\text{alg}} \mathrm{Y}$
4. Calculate the values of all dependent objects Y, i.e. $\forall z\epsilon\ \{Z \mid Z = \Gamma^1 Y \cup \Gamma^2 Y \cup \ldots \cup \Gamma^N Y\}$ calculate according to the algorithms $\prod_{value} z$ and $\prod_{alg} z$ and Conjunctive_allowability = Conjunctive_allowability AND Conjunctive_allowability(z).

It should be mentioned that the final solution in such way might not be achieved because of getting inappropriate data on the certain step, so the process of generation should be execute again several times. It could be avoided with setting some restrictions on the 0-level objects Y which randomly chosen on the first step of the considered above loop body.

In the demo mode it is also possible to built-in explanations of the various steps implemented by ATS. For this purposes button "Explain" should be provided to the form for showing a formula or an algorithm of the value calculation. For this purpose the values and algorithms of all objects, on which object Y depends directly, should be selected from the ordered graph G, i.e. objects of the set $\Gamma^{-1} Y$.

3.2 Tutoring process modeling of the supported solving mode

As it was mentioned above (see 3.1) in the supported solution and test mode task model contains two sets of objects: $objct1_i$ calculated by ITS and $objs1_i$ – by a trainee. Then, both sets should be compared.

In the test mode it could be done once at the end of the current AT solving by trainee with corresponding estimation in certain grades for each valid or fault value. Such estimation is not in the sphere of the present research as it is rather from the area of automation testing. It is much more interesting to find out what the reasons for current $objs1_i$ and $objct1_i$ mismatch in the supported solution mode and to figure out how to identify an adaptive training trajectory for a specific trainee.

The pedagogical decisions depend on the issues of fails, which might be some “blind holes” of student’s knowledge or skills, incorrect understanding of certain source materials or just inattentiveness in calculations. So it is necessary to solve the inverse problem: by the given mistakes in student answers (output signals) it is required to define their reasons (inner faults) in order to choose the proper pedagogical intervention for the best training trajectory.

There is a plenty of scientific papers from technical diagnostics considered solutions of inverse problems. However, they are not applicable as is in ITS development because of some specific of the area. Therefore, every learning subject area requires its own way of solving inverse problems for ATS development. The first step of the solution should be analysis aimed at identification and classification of the most probable reasons and consequences of trainees’ mistakes, which tend to appear in solutions of specific classes of AT.

The most common mistakes students admitted in solving AT in the subject area of "Automatic Control Theory", as well as appropriate diagnostic model described in [5]. Such models should be stored in a database and interpreted by the program. Besides this, ATS should be open for adding new classes of mistakes, and some classes of errors could be obtained automatically, for example, in the case of missing operands in calculations.

3.3 The method of missing operands' diagnostic models automatic construction

Within this method is considered a set of the most common binary operators { +, -, *, /, ^}, which are usually used in the solutions of math problems. Moreover, these operators as well could be used for presentation of non-binary operators. For example, the unary minus $-a$ can be easily transformed to a binary minus as $0-a$.

Results of both the left and right missing operands formalization for all five binary operators are shown in tab. 1.

Table 1.

Expression	Left operand missing	Right operand missing
x / y	1 / y	x / 1
a ^ b	1 ^ b	a ^ 1
c * d	1 * d	c * 1
f + g	0 + g	f + 0
k-z	0-z	k-0

It should be mentioned, that operands listed in the table can be both simple and composite. Then, a method of missing operands' diagnostic models automatic construction should allow for every possible formula written using five considered operators to build all of possible formulas in which one either left or right operand is omitted, i.e. the errors are occurring once.

Developed method consists of two parts:

1. Formula translation using Dijkstra's method in reverse polish notation (RPN).
2. Using the modified values in calculating RPN is the formation of a desired plurality of diagnostic models.

It should me mentioned that multiple errors are also possible. However, such errors are rare as the probability of double errors can be considered as the product of single errors probabilities. At the same time, even for quite simple formulas taking into account multiple errors requires a large number of calculating options, which significantly complicates the diagnostic software of ATS. Therefore, a multiple occurring errors diagnostic software development is unprofitable and inappropriate.

In the case of simultaneous work-out of different diagnostic models ATS should ask the student additional questions and offer him several options for calculating an incorrect value or provide the ability to enter his own variant.

3.4 Trainee's model for the supported solving mode

It is obvious that the of knowledge and skills components play the major role in the trainee's model as the adaptive training sequence for a particular student depends on the components' mastering degree. Consequently, there is another inverse problem: based on the results of student's work with ATS to measure the values of the knowledge components mastering and their links with objects $objct1_i$. As the most reasonable and less demanding way of the given problem solution in this research probabilistic approach had been chosen.

When adding tasks and components of knowledge and skills it is necessary to check whether these components exist in the system, in order to be used as a priori probabilities of owning one or another component of the new tasks of the posterior probabilities for the tasks that the student has already decided.

As knowledge and skills components are defined by names, it is reasonable to search for the similar components to avoid duplication and semi-duplication of knowledge components in ATS. For this purposes different metrics and edit distance calculation methods could be used [2]. But first of all data model of information flows in ATS should be developed based on the entities presented in above considerations.

3.5 Pedagogical activities choice method for the supported solving mode

The choice of pedagogical activities should be carried out on the basis of information received from the student model, and the current step implemented by a trainee during solution of the AT with number i .

In a case of incorrect step i of a task j and following hints based on activated diagnostic model, the following actions are executed: in the student model after Bayesian network for the step i is inserted as many temporary layers (Bayesian networks) as wrong steps done. Thus if left first layer is represented as a graph, then all layers are isomorphic to original graph, but the a priori probabilities of the next layer are assigned with posteriori probabilities of the previous layer. Fig. 4 shows an example of Bayesian networks combination for the case where the object $O1$ is correct after $s + 1$ attempts and then trainee come to the next step: entering the object $O2$.

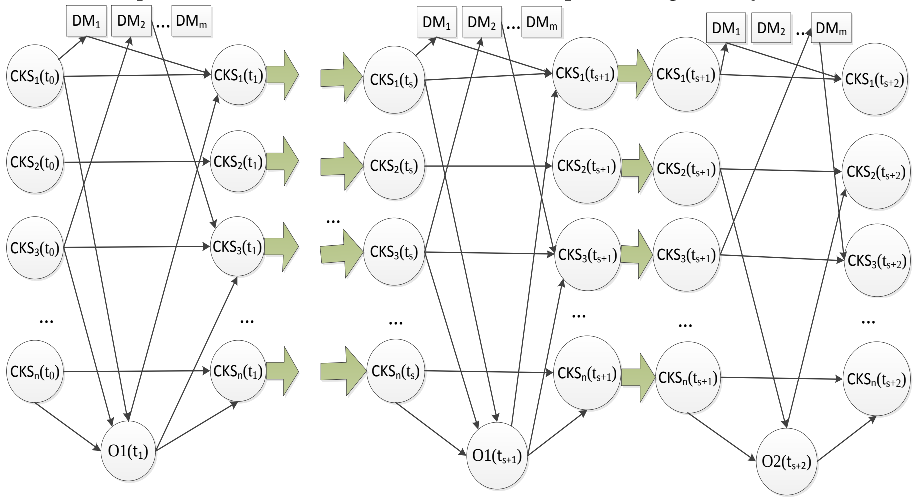

Fig. 1. Bayesian network with temporary layers

Besides this, it is necessary provide ATS self-learning by means of diagnostic of the answers in the form of an algorithm or a program, created by trainee, but differ from a reference solution. The general classifier for the trainee's program can be represented in form of binary tree with the nodes of decisions: succeed/failed, similar/new, better/worse. If a trainee's program that has passed through testing on validness surpasses existing program (alg component of $objct1_i$ in (1)), then ATS can use this program as a new reference one. Wherein the program can be estimated by criteria such as speed, memory used, commands count.

If the student's program does not pass, at least one test on validness, then the next diagnostic task is solved: search for an error location. To do this, it is needed to compare two abstract syntax tree (AST) obtained by parsing the standard program and the program developed by trainee. First way of comparison AST is to use search method in width; second – calculation of the distance between the trees in the metric Sasha-Zhang and edit track. In this tree nodes must be symbols, and operands and operators.

In addition, cognitive images of dynamic information about student's actions and state can be used [6] to estimate stability of the system trainee-ATS for making decision about showing the student concrete command and operand, where he failed, or just inform about error and give him another try.

4 Conclusions

Presented above models and methods allow to develop ATS provided individual effective training strategy for each student solving AT in different problem areas.

Based on the developed models and methods, software for teaching students has been created and is being successfully operated at the Aircraft Control Systems department (KhAI) [7,8] and University of Applied Sciences of Upper Austria [9,10]. As well software for preparing of secondary school students to the final testing from mathematics was implemented in Kharkov Regional Center for Quality Assessment [11] and the British Center for Innovation in Mathematics Education and China Beijing Center for development educational programs [12].

References

1. Kulik, A.: Rational intellectualization of the aircraft control: Resources-saving safety improvement Studies in Systems, Decision and Control, 105, pp. 173-192 (2017).
2. Chukhray A.: Methodology for learning algorithms. National Aerospace University «KhAI», Kharkiv (2017).
3. VanLehn, K. Intelligent tutoring systems for continuous embedded assessment. In C. A. Dwyer (Ed.), The future of assessment: Shaping teaching and learning, pp. 113-138. Erlbaum, NewYork, (2008).
4. Baker, R.S, Corbett, A.T., Aleven, V.: More Accurate Student Modeling Through Contextual Estimation of Slip and Guess Probabilities in Bayesian Knowledge Tracing. In: Proceedings of the 9th International Conference on Intelligent Tutoring Systems, pp. 406 – 415 (2008).
5. Diagnostic models of intelligent tutor system for teaching skills to solve algebraic equations A. Kulik, A. Chukhray, M. Chukhray, https://online-journals.org/index.php/i-jet/article/view/68, last accessed 2016/24/02.
6. Mygal, V., Mygal, G.: Problems of Digitized Information Flow Analysis: Cognitive Aspects Information & Security: An International Journal (2019). DOI: 10.11610/isij.4312
7. Gaydachuk, D., Havrylenko, O., Martínez Bastida, J.P., Chukhray, A. Structural diagnosis method for computer programs developed by trainees. CEUR Workshop Proceedings, Vol. 2387, pp. 485-490, (2019).
8. Martínez Bastida, J.P., Havrylenko, O., Chukhray, A. Developing a self-regulation environment in an open learning model with higher fidelity assessment Communications in Computer and Information Science, Vol. 826, pp. 112-131, Springer (2018).
9. Chukhray, A., Kalinichenko, V.: An approach to development of intelligent computer system for SQL tutoring. In: Proceedings of the East-West Fuzzy Colloquium, pp. 251 – 256, IPM, Zittau, Germany (2010).
10. Vagin, I., Havrylenko, O., Bastida, J.P.M., Chukhray, A. Computer intelligent tutoring system “SQLTOR” CEUR Workshop Proceedings, Vol. 2387, pp. 525-530, (2019).
11. Chukhray A., Sidorenko, A.L.: Computer system for interactive testing of knowledge and skills of students. Vestnik TIMO, № 4, pp. 15 – 19, (2008).
12. Chukhray, A.G., Pedan, S.I., Vagin, E.S.: Development of a set of interactive web-tests in mathematics . Radio-electronic and computer systems, №1 (42), pp. 103 – 107, KhAI, Kharkov (2010).