IMPLEMENT ADAPTATION IN A CASE BASED ITS

Nikolaj Troels Graf von Malotky and Alke Martens
University of Rostock
Albert-Einstein-Str. 22, 18059 Rostock, Germany

ABSTRACT

ITSs have the requirement to be adaptive to the student with AI. The classical ITS architecture defines three components to split the data and to keep it flexible and thus adaptive. However, there is a lack of abstract descriptions how to put adaptive behavior into practice. This paper defines how you can structure your data for case based systems in a way that adaptivity is easier to achieve while maintaining the classical splitting of the system and reducing the data footprint. Building a case based system from a collection of exchangeable steps is also possible with this approach. Two variants of adaptivity based on the data structure are explored and both can be used in conjunction.

KEYWORDS

ITS, Adaptation, Case Based System

1. INTRODUCTION

ITSs (Intelligent Tutoring System) can look back at a comparably long tradition. The first ITSs have been developed in the 1970s, which is a long time ago. At this time, the main idea of ITS was to provide a maximum of expert knowledge. Nowadays, given the modernization of technical devices and the digitalization processes, the main task of the ITS and the part which makes it different then all the other teaching and training software types, is the adaptivity. Regarding this aspect, the ITS is a teaching program that has the highest scores (Mendicino 2009) (Beal 2010) (Singh 2011) (Van Lehn 2011). Adaptation as different facets, like adaptation of the content, adaptation of the navigation, adaptation to a given student (Pirolli 2013) or adaptation of help and correction. Not all these facets of adaptation are necessarily realized in an ITS - adaption can exist in multiple ways in such a system. Some ITS manage only initial adaption to a student, some provide for runtime adaptability. However, independent of the way adaptation is realized, the software system itself has to manage different groups of data to achieve this task. Regarding the basic idea of adaption, the groups of data can be categorized according to the questions: What shall be adapted to whom or to which process and in which way. On the software engineering level these ideas are reflected in different designs (Graf von Malotky 2020), but mainly the idea is constructed around the availability of data for steering the adaptation and for adaptation itself. The classical groups of data for an ITS is the domain knowledge, the pedagogical knowledge and the student knowledge (Nkambou 2010). These three groups represent the knowledge we want to teach, the knowledge about the teacher's behavior and the student knowledge. The teaching material consists of the domain knowledge, which is the content we want to teach and the pedagogical knowledge which is needed to make good decisions as a teacher. The student knowledge is needed to track the student, so that the system adapts to his preferences and skills. Authoring teaching material that is marked as domain and pedagogical knowledge, makes it possible that domain experts can edit the domain knowledge only, with less thinking about the pedagogical consequences of the material, achieving the goal of making authoring easier (Murray 1999). It is a more modular approach that sees the domain knowledge as a collection of material that can be used as teaching material, without having a certain place in a defined learning sequence of the student.

Since the definition of the data groups (domain, pedagogical, student) are so generalized, one data group cannot expect specific data elsewhere when only designing them separately. Therefore, it is important that the basic structure of the teaching material and the student knowledge are harmonized with one another. Harmonizing the data groups reduces the amount of data we need to gather about the student, since we already know what data is needed for the adaptation. Instead of using big data to get clues out of a massive amount of

data from the student, we think about what we need from the student while creating the teaching material. There is a large variety of possible ITS types (Graf von Malotky 2019), by focusing on one ITS type, it is possible to specifically define what should be saved in each data group.

It is rare that there is an abstract definition of how a splitting of the teaching material in pedagogical and domain knowledge can be implemented. In this paper it is shown how to structure your data and additional algorithms to accomplish adaptive teaching material consisting of the classical knowledge components in a case based system, since a case based system is inherently already split up into elements which can be filtered and reasoned about (Funk 2002).

2. DATA OF THE STUDENT

Before we can adapt the teaching material to the student, we need to know what is best suited for the student. This can be a combination of several aspects, like for example his personal learning preferences (e.g. more pictures), his level of expertise (e.g. beginner), and also his performance in former cases. Additionally, we have aspects like the performance at runtime. To abstract this, we split the student knowledge data in two: Student style preferences and student skills. We define the style preferences of a student as the attributes of teaching material which will make him more likely to stay motivated and are subjective to the student. A loss in motivation can partly be measured by checking if the student is active and or performing well in a given time period, but it is not very accurate, since there are many other factors that could influence that. Some students tend to be engaged but are less measurable active. Without knowing what the student is thinking and staying away from forcing the student to interact with the system just to check their motivation. This is not a good solution. Additionally, there is still the problem of slowly decreasing performance when the motivation decreases, even though the student could do better. We decided that the student can decide which explanation style is used for the presentation of the domain knowledge. All these explanation styles are generated from the same domain knowledge. This mostly boils down to the presentation style preferred by the student, for example having a style preference for graphical content. If available, the system chooses the domain knowledge with the matching style. If the student has not decided, the system automatically chooses one where the student performed best and makes a switch if the performance does decrease more than a threshold. Since the student cannot game the system by choosing a different style preference (scoring is not affected), we are letting him select and change it at any time.

We define the student skill as the part of the domain knowledge the student has already learned. Skills are determined only by the student's performance and are therefore easier to track. The goal is that the student fully learns all the skills available. Which skills they may learn at one point in time, is inferred by the skills already estimated to be obtained and saved in the student knowledge. How good a student is in a certain skill is saved by the skill levels, which are set through the evaluation of their performance in different sessions of the available cases for that skill. Since you cannot game the system to give you higher scores, students have the option to choose which teaching material they want to learn that has their current difficulty level. Asking the student which skills he has obtained opens up the possibility to game the system (d Baker 2006). Even ignoring this fact, it is expected that some student misjudges themselves. The student should not be required to know the correct dependencies of the skills. Even though a skill might seem to be easy for a student, the system expects also the dependencies. Students outperforming by a large margin will be recognized and their skill updates much faster. It may be beneficial to have short estimation tests for expert students, to speed up the process for them to get a higher difficulty level.

Reducing the amount of data gathered improves the systems disk space, performance and protects the student's privacy. Only the parts of data of the student which are required to make good assumptions about his skills and style preferences are necessary. Since we expect the system to handle teaching by cases, we can save a history of all the cases the student interacted with, including all necessary information that are relevant to judge the performance of the student. At the minimum, it is enough data, when you can generate an estimation of how good the student is at the different skills, which has to include the student's history (Zhou 1999). The system cannot easily track the skills of the student directly, but instead the system can easily track what the student does. As an example, how this can be achieved is shown in the following: it is possible to use just three data sets, which are

- Duration of each step of a case (which implies if the case was started and finished)
- Inputs for each step of the case (to check the correctness and the number of inputs)
- Score achieved in each step of case (which is automatically also the progress)

3. BUILDING THE TEACHING MATERIAL

The teaching material consists of instructionally elaborated training cases. This material is split into domain knowledge and pedagogical knowledge, i.e. the underlying expert knowledge and the instructional aspects related to and embedded in the constructed training case respectively. The training cases often are related to real live cases, which can be extended, anonymized and then integrated in the knowledge modules.

In contrast to other teaching and training system (e.g. mathematics or chemistry), we are developing a system based on these cases, which means the idea of the training cases influenced the software system design. In contrast to non case based training, in case based training we always find the above mentioned combination. Always these cases contain aspects of knowledge of the domain, e.g. in clinical medicine, we find overall medical information, medical information about anatomy and symptoms. This allows us to integrate a large amount of pre-existing databases. As for example the medical knowledge domain consists of facts and rules, we can integrate parts of this as universal information parts. Additionally, we have case-related information, e.g. in the medical domain, an x-ray of an elderly male thorax with lung disease after 30 years of smoking is a special picture with a related special diagnosis related, and which cannot be re-used in an arbitrary way. However, this is still not the pedagogical knowledge. The pedagogical knowledge can in such a case consist of certain related facts and special rules, like level required, skills, sequence of steps, but also of question/answer sets of different presentation styles (Graf von Malotky 2017). Moreover, here we have set goals: Which knowledge we want to teach which each case. This means we need different tasks grouped together to teach some predefined skills.

A case can be represented as a graph with each node of it being a step in the case (see figure 1). There is always one starting node (no incoming edges), at least one end node (no outgoing nodes) and all nodes are reachable by edges. To keep it simple, in this model a step is associated with a display of content on the level of the human-computer-interface (e.g. the monitor). The navigation from one step to another (the edges in the graph) are the navigation buttons or menus in human-computer-interface. To graph shows all the potential ways navigating through the pages of a training case. The steps exist at least once and, in our example, cases as graphs are often simple enough to not need to think about complex graph theories. Generally, the complexity is realized in the training case to switch the nodes. Sometimes, content of steps can be designed in a way that a step can re-occur in different branches of the tree (e.g. in the medical domain, some examinations are not dependent on the time in the training case, some can even be repeated).

To successfully complete the case the student has to view or solve steps and come to a step where there are no outgoing edges. Each step can be a passive explanation or an interactive task to be solved by the student. The passive steps are for the preparation of the student for the next task and can additionally feature storytelling aspects to merge the different tasks into one motivating case. The tasks in the case can be embedded in a story which progresses step by step. The story makes the tasks more interesting and motivating.

The domain knowledge contains teaching content visible to the student while the pedagogical knowledge is the added data. For the case it is the graph, the available style preferences and skills and how they are connected to each case. For the step it is its difficulty, the style of the step and whether it can be used as a passive or interactive step. The style for the passive steps can for example be clean text, comic images or 3D-Model instructions. The only “interaction” here is that the student has to read the information. No additional action like selecting, clicking, marking or whatever are included. These types of interactions are realized in the so-called interactive tasks. For the interactive tasks there are the typically used single or multiple choice questions, drag and drop, draw lines, text with drop down fill holes and so on.

In figure 1 you can see an example graph, where passive steps are squares and interactive steps are circles. In this example passive and interactive steps alternate. The passive steps show knowledge embedded in a story which should be learned, while the interactive steps tests the skills of the student, so the student is forced to think about what they are showing, instead of only remembering. The starting (A) and the ending step (K) start and finish the included story. Dependent on the difficulty more or less information in the passive steps is shown, important parts highlighted and kept for later look up. The same goes for interactive steps, where help is reduced with higher difficulty.

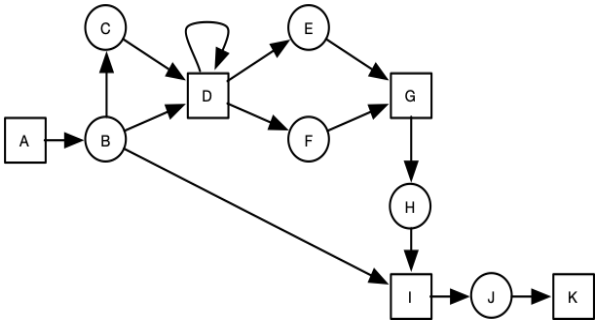

Figure 1. Example case graph with passive elements as squares and interactive elements as circles

With this approach, the idea is to build up a collection of steps that can be reused and exchanged. Each step has enough additional information to make an educated decision which step to use for a given student in his current situation. An idea was to create the graph automatically from the dependencies of the nodes to each other so there would be a dynamic graph, but that creates the problem that each node has to know other nodes or that there exists some sort of general, detailed, wide applicable node classification to use for all steps. To maintain the simplicity and the modularity of the steps the graph of the case is static. This means that the structure is a fixed attribute of the case which is represented by the graph and will not adapt to a student or changing steps. The graph can of course be still changed by authors of teaching content. The case itself is not adaptive, but is built in a way that each step can easily be exchanged at runtime. The adaptation algorithm has to decide which steps to choose. In another variation, the adaptation algorithm can also decide which steps are possible for a given case, but this is not the focus of the paper. This automation needs a lot more understanding of the system, e.g. which content is inside a step and which shall be used for the student according to which rules.

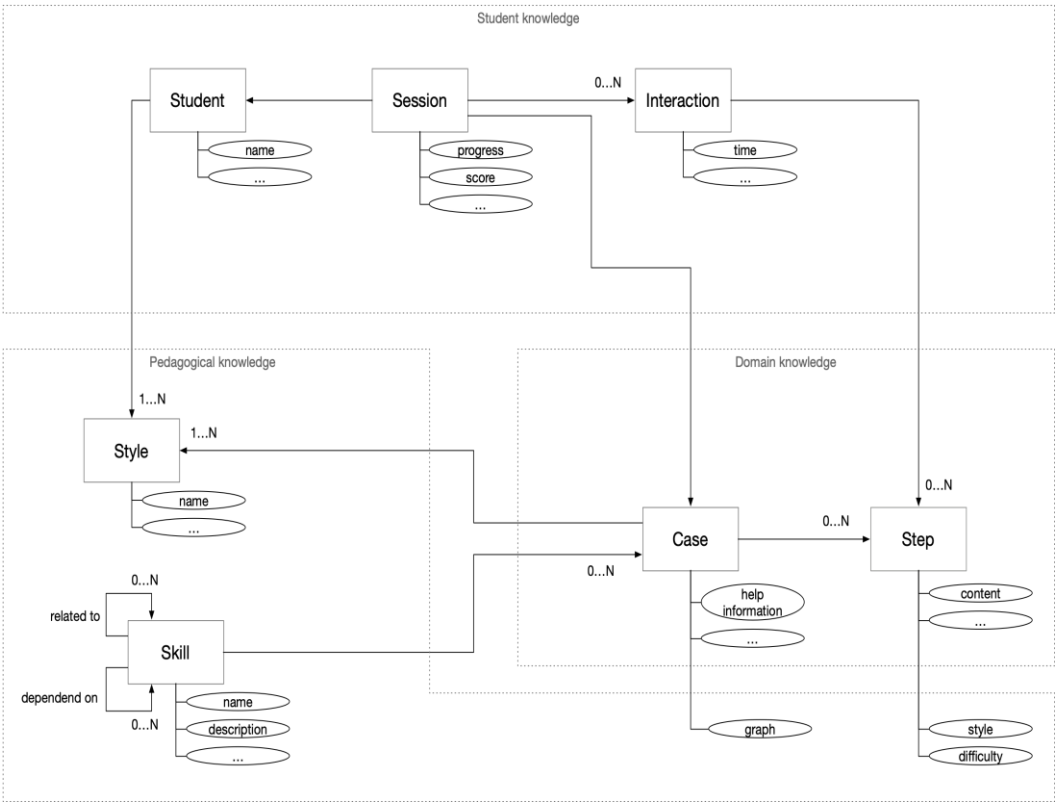

Figure 2. Database structure of a case based ITS

The reduced structure of the general data needed for the system is shown in figure 2. To be reusable we explain the system in an abstract way. There is a separation of domain knowledge and pedagogical knowledge, which allows authors to modify only one of them. The cases are exchangeable, as are their steps. Instead of defining which concrete case is dependent on other cases, this information is extracted into its own part, the skills that the student should learn.

As we already defined the student knowledge, we have to associate the student knowledge with the teaching material. To track the progress in the observed skill of the student, we save the interactions of the student with the steps to evaluate a score. With the score, the progress and the time of each interaction in one case session it is possible to estimate the student's skill.

4. ADAPTATION

Now that the teaching material is structured in a way that it is ready to collect the necessary data about the student, while also being built in a way that the student's data is enough to be used as input to adapt the teaching material, we can use adaptation methods. In its raw form the student knowledge is not informative enough. We additionally need to calculate the student's overall skill levels from the sessions the student completed. The skill level of a student is just a number based on the progress, score, number of interactions and duration of his session history. For the score the difficulty of the steps is included. The difficulty could be a number from 0.0 to 1.0 and be used as a multiplier in the score calculation. To be able to compare the student's skill level to a difficulty/scores and showing the progress, it is important that the algorithm for the skill level does not change and is built together with the adaptation methods. It has to be clear for the system, what a high and low score is. The teaching material also needs additional data if it did change. This includes the dependencies/relations between skills and implies the dependencies/relations between cases, the possible difficulties of each case, the possible styles of each case and the selection of available styles and skills. Two possibilities are explored how to adapt to the student. The first option is that adaption is realized by the selection of domain knowledge that matches the preferences and skills of the student; in the second option the adaptation is realized through adjustments on the shown domain knowledge.

For the first option we need a case to teach the student. Each case has skills and styles associated with it. Dependency of cases to each other is calculated beforehand by the skills each case requires. Therefore, the selection of matching teaching material comes first. We want students which use our system to show matching learning material, but avoid to show the exact same material again. Additionally the teaching material should also take into account what style preferences and skills the student has. So the task is to select teaching material which is less often seen, has matching style and is about skills that are related to and dependent on skills that the student already has.

Currently there is a problem for students, that do not learn in the set speed of existing static teaching material: If they learn slower, they will be presented with the same teaching material repeatedly. That means that some students may pass only because they learned about how the material is structured. They learned the answers but did not understand all necessary parts for it. The difference here is between “putting it into the brain” without grasping them deeper meaning, in contrast to the student’s knowledge construction and deeper learning, which is the target of all good structured learning material. If a student who does not pass a test after reading a book the first time is shown the exact same test over and over again, he will pass it eventually, but maybe does not really understand the underlying content. That problem also applies to digital content, if not prevented with varying content. Instead of showing complete new content we can exchange only some part of the content to adapt on a finer grained level to the learning speed of the student. Adjustments of the teaching material allows not only to have a better matched difficulty but has also the benefit of varying the case if used multiple times by the same student. Creating variations for a case has a higher change of requiring the requested skills, not knowledge about how the case is structured. We can reuse the teaching material by modifying it enough so that the student is more likely to think about the problem instead of remembering the already given solution from previous sessions.

The second option is the adaptation of teaching material itself. It allows the ITS to be more suited to the liking of the student to increase motivation or to address problems with improper difficulty settings. Adjustments on the case by exchanging steps to select a matching style, get the correct difficulty and the less seen steps. By allowing to have exchangeable tasks in the case we can adapt the content and difficulty of the case without changing what the goal of teaching of the case is. Another way that such a system can adapt the difficulty is to show less additional information from the case itself. To give a more concrete example of how the rules could look like the following list of rules could help. The system searches through the cases and selects the steps to build a case that matches as many as possible of the existing rules preferred in the given order (most preferred are on the top). A threshold for skills can be set to define how good a student has to be in one skill to progress to the next.

- Cases that can be built from steps matching difficulty to the student's skill level
- Cases that can be built with steps that match the student's style preference
- Cases that are related to cases already in the student's history (ordered by highest score)
- Cases that are directly dependent to cases already in the student's history above score threshold (ordered by highest score)
- Cases with no progress (unseen case)
- Cases that can be built with steps without scores (unseen steps)
- Cases that can be built with steps with score below threshold
- Cases with incomplete progress
- Cases with a higher than threshold score but are completed long time ago (ordered by days since last session)

5. CONCLUSION AND OUTLOOK

There are multiple ways of achieving adaptivity in an ITS. Both shown methods of adaptation that were explored in this paper have their benefits and they can both be implemented side by side. With the presented structure this is possible in a reusable way for a case based system which can then be fitted with more details of the given domain. The presented steps to create an adaptive case based system allow for a general way to achieve the set goals for an ITS without demanding domain specific details. The data structure respects the subjective preferences of a student as well as his objective performance goals for skills while maintaining a minimal data footprint. Both of which can be fitted to many available domains. The presented adaptive methods work on the defined data structure and can easily be implemented with very simple algorithms and grow more complex as the system gets more sophisticated. We have explored this in the context of our DigiCare Project and the training cases in our project are developed based on the above mentioned algorithm.

DigiCare is a project about supporting the Healthcare and Healthcare management students at the University of Applied Sciences Neubrandenburg and at the University of Rostock. Together with the DZNE (German Center for Neurodegenerative Illnesses), a threefold approach is realized in the funding period of 2019 to 2022. The first steps have been the collection and digitalization of teaching and training material, which has gained double speed in the pandemic situation starting in 2020. Currently, large parts of the curriculum are digitalized and recorded with the purpose of distance education. On the long run, the University of Applied Sciences Neubrandenburg will keep open the opportunity for the students to combine distance and presence studs. The second step has been to integrate the system SCARLET (Nicolay 2020) as part of the lectures. SCARLET is a software which allows for the interactive annotation of lectures by students, via using hashtags. An automatic analysis of the lecture slides based on the LDA (Latent Dirichlet Algorithm) allows to grasp the underlying content structure of the slides. The resulting model shows the required domain knowledge, which should be mediated to the students. A mapping of the student’s hashtag annotations with the resulted graph structure allows the students or even a supervisor to see how close the students’ understanding is to the intended understanding. The third step is the Intelligent Tutoring System, as sketched above. The ITS in DigiCare has been developed for the purpose that in healthcare, student have to work with real life cases from very early stages of their professional development. However, given the traditional lectures at the University, this can only take place on a very abstract level. Moreover, in the pandemic situation, students are not allowed for practical parts of their study. Thus, to allow them at least a small glance into the patient situations, the ITS was developed. Our ITS consists of three main aspects:

- It is a case-based ITS, thus is contains only patient related training cases. The training cases are based on real existing cases, e.g. patients with dementia. These training cases, same as the underlying knowledge structures of the expert knowledge in the ITS are provided by the DZNE partner in the project, in close co-operation with the University partners.
- The case-based ITS contains also a natural language and dialog component, which is described in several other publications (Sosnowski 2020)(Abuazizeh 2020). The main underlying idea is here, that patients with dementia show a very good observable behavior in communicative situations. Examples reach from 'answering aggressively' to 'not answering at all'. The communication training is perceived to be one important part of the education, which admittedly cannot be reached in the traditional educational formats like lectures. Thus, an ITS can be at least a bit helpful in this context.
- There exists a bunch of material, which can be re-used in different training cases, and which represents main parts of the necessary education in health care, e.g. how to fill in medical records, which rules apply during patient contact etc. This content can be re-used in diverse settings and allow the student to close the gap between theoretical and practical knowledge. In the above-mentioned graph, these step types are examples of re-occurring or re-usable steps.

The ITS is developed based on software engineering aspects and is based on the framework for Intelligent Tutoring Systems. This component based generative framework is the first all-purpose framework for ITS, as it has been shown that especially ITS lack general and re-usable structures.

Our future work will be to refine the existing ITS, to close the gap between the dialogue and the other content of the ITS. Currently, the dialog is perceived to be one step in the abovementioned graph. Thus, knowledge gain in the course of the dialogue is not yet part of the adaptation process. Additionally, the next steps will also be to allow for even more adaptability and flexibility in the training cases.

ACKNOWLEDGEMENT

This paper is part of the research project "DigiCare". This joint research project "DigiCare" is supported by the European Social Fund (ESF), reference: ESF/14-BM-A55-0018/19, and the Ministry of Education, Science and Culture of Mecklenburg-Vorpommern, Germany. We thank all our cooperation partners!

REFERENCES

Abuazizeh, M.D., Kirste, T. and Yordanova, K., 2020, June. Computational state space model for intelligent tutoring of students in nursing subjects. In *Proceedings of the 13th ACM International Conference on PErvasive Technologies Related to Assistive Environments* (pp. 1-7).

Beal, Carole R. et al., 2010, Evaluation of AnimalWatch: An intelligent tutoring system for arithmetic and fractions, *Journal of Interactive Online Learning*, volume 9, issue 1, pages 64-77, ISSN 1541-4914.

d Baker, R.S. et al., 2006, June. Adapting to when students game an intelligent tutoring system. In *International conference on intelligent tutoring systems* (pp. 392-401). Springer, Berlin, Heidelberg.

Graf von Malotky, Nikolaj Troels et al., 2017, Synthesis of pedagogical annotations, *9th annual International Conference on Education and New Learning Technologies (EDULEARN)*, EDULEARN17 Proceedings, ISBN 978-84-697-3777-4, ISSN 2340-1117, DOI 10.21125/edulearn.2017, pages 3655-3661.

Graf von Malotky, Nikolaj Troels, Martens, Alke, 2019, Analyzing the usage of the classical ITS software architecture and refining it, *15th international conference on intelligent tutoring systems (ITS2019)*, pages 40-46, DOI 10.1007/978-3-030-22244-4_6, Springer, Cham, Online ISBN 978-3-030-22244-4, Print ISBN 978-3-030-22243-7.

Graf von Malotky, Nikolaj Troels, Martens, Alke, 2020, General ITS software architecture and framework, *16th International Conference on Intelligent Tutoring Systems (ITS2020)*, Springer.

Funk, P. and Conlan, O., 2002, September. Case-Based Reasoning to Improve Adaptability of Intelligent Tutoring Systems. In *ECCBR Workshops* (pp. 15-24).

Mendicino, Michael et al., 2009, Comparison of Traditional Homework with Computer Supported Homework, *Journal of Research on Technology in Education* 41(3), pages 331–359.

Murray, T., 1999, Authoring intelligent tutoring systems: An analysis of the state of the art. *International Journal of Artificial Intelligence in Education* (IJAIED), 10, pp.98-129.

Nicolay, Robin, Martens, Alke, 2020, Synthesis of Knowledge Tracing Models Using Natural Language Processing on Lecture Content, *EDULEARN20 Proceedings*, pp. 1659-1666.

Nkambou, Roger et al., 2010, Advances in intelligent tutoring systems, *Springer Science & Business Media*, volume 308, ISBN 3642143628.

Singh, Ravi et al., 2011, Feedback during web-based homework: the role of hints, *International Conference on Artificial Intelligence in Education*, Springer, Berlin, Heidelberg.

Sosnowski, T., Yordanova, K., 2020, June. A probabilistic conversational agent for intelligent tutoring systems. In *Proceedings of the 13th ACM International Conference on PErvasive Technologies Related to Assistive Environments* (pp. 1-7).

Van Lehn, Kurt, 2011, The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems, *Educational Psychologist* 46.4, pages 197-221.

Zhou, Yujian Zhou, Evens M. W., 1999, A practical student model in an intelligent tutoring system, Proceedings 11th *International Conference on Tools with Artificial Intelligence*, pp. 13-18, doi: 10.1109/TAI.1999.809759.