Mathematical Problems of Computer Science 44, 116--132, 2015.

Developing Interactive Personalized Tutors in Chess

Sedrak V. Grigoryan and Levon S. Berberyan

Institute for Informatics and Automation Problems of NAS RA
e-mail: addressforsd@gmail.com, levon711@mail.ru

Abstract

We suggest an algorithm and computer software for personalized interactive chess tutoring. The article starts with analyzing some standard approaches to teaching chess, listing some of their pros and cons, especially concerning personalized and interactive learning. Further we specify the algorithm as an approach to tutoring chess providing variable behavior depending on specific student skills. Then we describe the implementation of proposed algorithm, which includes involvement in RGT Solver package and engines interaction tool, providing tutoring handling and progress fixing mechanisms. We also provide a certain valid chess endgame tutoring example.

Keywords: Chess, Interactive Tutors, Personalized Tutoring, RGT Solver Package, Chess Engines Interaction, Personalized Education, Technology Enhanced Learning.

1. Introduction

1.1. There are different approaches to teaching chess. The classical approach includes teacher and implies interaction between the students and the teacher.

a. Some concept of chess is defined and is explained in depth. The depth of explanation depends on a student, because the student also needs to understand the concepts used in definition of what the teacher defines. If the student knows all the required related concepts, then the explanation is not deep, it just defines the teaching concept. For each of the related concepts the following steps are performed if required. Usually if a student has missed an explanation of some concept, teacher will not go back and teach for that concept again in future sessions or levels.

b. Chess exercises and puzzles are suggested to solve. If the student is able to solve the problems then it is considered as already learned. If the student is unable to solve the problem, then it is assumed that the student did not learn the concept and step ‘a’ needs to be performed again.

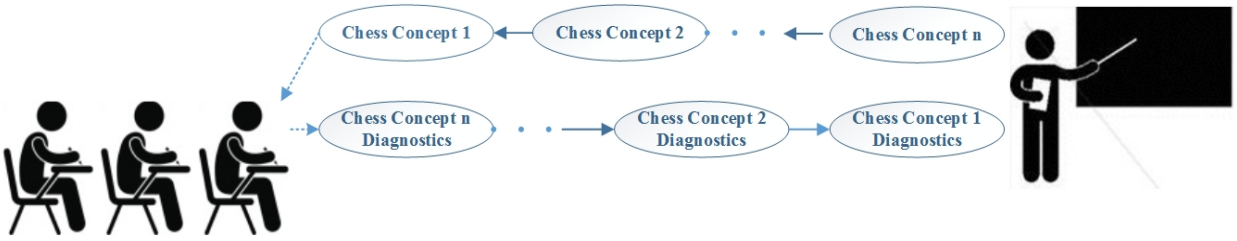

Fig. 1. Chess teaching classic approach (sequential learning, concept by concept, test by test).

1.2. Questioning standard approaches to teaching.

The mentioned mechanism of teaching chess

a. requires teacher's effective involvement in the studying process
b. is not flexible enough
c. is not personalized for each student and does not differentiate between autistic, genius and other levels of students in learning, usually provides the same approach to all of the students.

Other chess learning mechanisms can include chess books [1] or software with static databases, e.g., video lecture course database, where all lectures are included in videos [2], after each video lecture there might be suggested learned concept testing exercises, too. The restrictions of these mechanisms are:

a. they are not interactive

b. mechanisms might be personalized but the student needs to indicate the level of his/her knowledge by him/herself, thus making difficult having results in different performance levels, e.g., for autistics or geniuses

c. the feedback for testing of the suggested learning concept exercises is not well organized

d. they do not provide student performance measurement mechanisms.

Many psychologists and education specialists such as T. Ogneva noted in their works that chess education, as well as other disciplines education must take into account the individual psychological characteristics of a student making the teaching more personalized [15-17]. Taking into account the individual features allows to maximally uncover one's internal resources, to take into account the nuances of student's mental processes.

Among the methods that, in one way or another, take into account the individual characteristics of the students, differentiated, individual and individualized approaches to teaching are classified [18-20]. Differentiated approach involves the allocation of a similar individual, personal qualities of students. Individual approach involves training without contact with other students, but in the same pace for all. The individualized approach is built taking into account the peculiarities of each individual student. The concepts of education individualization and differentiation are revealed in the works of I. Osmolovskaya, I. Unt, A. Kirsanov, A. Granickaya, V. Shadrikov, I. Smirnov and others.

In [13] personalized tutoring approach is discussed, particularly chess tutoring is discussed for ordinary and autistic children tutoring process requirements are discussed, where several requirements are revealed and certain software is suggested as a personalized tutor for chess. Software selection is substantiated. Following tests for personalized tutoring are suggested 1) generation by the models of meanings positions on the board to be commented by the experts for possible corrections of the models, 2) using complementary means to minimize the threat that only a part of the meanings of the expert classifiers were "caught " by the models like the

following tests: * experts generate examples which are tested by the models, * known positions that meet certain criteria of completeness are taken from some repository to compare expert responses with the model ones. Acquisition of concepts can be done by a variety of strategies like the one where tutors sequentially decompose the given concept A, for each decomposition reveal not learned units, decompose each of them to find new unlearned components, etc., and, finally, composing the map of acquisition of A layer by layer move up while achieving the learning of each missed component of the layers. Comparing the suggested approach with the suggestion of knowledge-based chess bishop-pawn endgames tutoring in [14] the following extensions are revealed - considering not only endgames but arbitrary positions including the complex middle game ones, - tutoring learning the strategies of important for applications of other competition problems, particularly the intrusion protection, defense, management ones. More extensions are suggested for the algorithms to enable tutoring math.

We aim to develop certain methods and software modules according to the suggested model of personalized tutoring chess suggested in [13].

1.3. Designing personalized interactive chess tutors. Students want to learn chess concepts provided by their names. Tutoring Software is searching for the concept in its chess concepts graph. If the concept is found, software provides concept explanation by giving the name, relations with other known concepts, regularities, description if defined. Then it provides concept examples. The process is recursive, the related concepts explanations and examples can also be requested. Afterwards software checks if the user understood the concept by chess exercises and puzzles. If the student does not pass checking, expected valid solution is shown, he can request the concept explanation and examples again, so the process can be repeated.

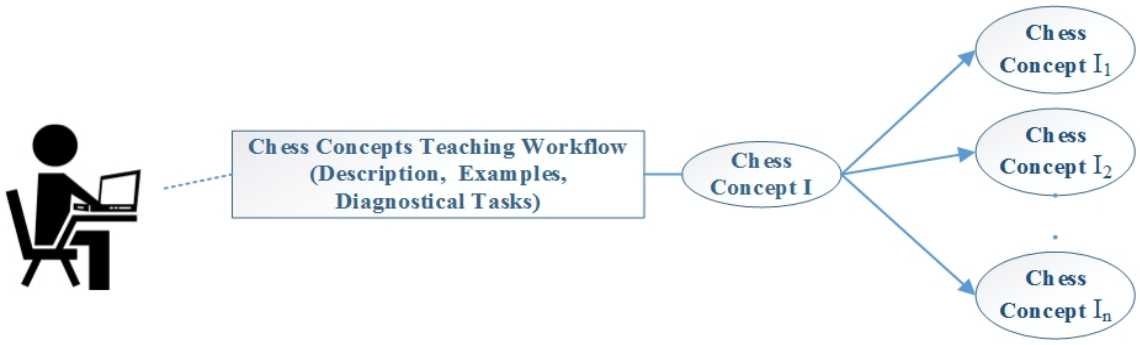

Fig. 2. Suggested chess tutoring approach (interactive personalized software Workflow (Fig 3), random chess concept learning, learning of any required chess concept I, related to concepts I_1, I_2, ..., I_n).

Diagnostics that checks student's understanding of a concept can be requested at any time. So it can also be requested before starting of learning.

Designed software can provide an explanation of any defined concept on board visualization description level. The lowest level of explanation is the explanation of the lowest level concept by visualization on the board. The lower level explanation is out of software chess tutoring scope.

1.4. Expectations of chess tutors

a. provide personalized tutoring, enabling individual approach to any student, regardless of the level of student’s understanding, where autistic students, regular chess students, students with huge speed of performance in learning chess can be involved and each of them will have personalized learning.

b. make interactive platform where any learning concept can be described level by level depending on the student knowledge, where problems and chess puzzles are being suggested and student solutions are checked

c. not involve human teachers, they can still be required in several cases, 1) the teacher will be required for inserting the whole set of chess domain knowledge which will be transferred to students during the teaching process, 2) if the software meets difficulties in explaining and teaching for some chess concepts the final action is to ask the human teacher to explain the concept to the student and improve the definition of that chess concept in the software.

d. provide feedback and analyze the results. If solutions to the suggested exercises do not match the solutions suggested by the software while checking test results for each learned concepts, they will be corrected and explained step by step.

e. rate students and compare them against rated and validated chess players to validate the learning results.

2. An Approach to Chess Tutoring

We are developing a method and software that is tutoring chess. We interpret the whole chess knowledge in a graph where each piece of chess knowledge is a node. A node represents relations with other chess knowledge pieces and defines regularities that identify that relation, e.g., "opposition" concept represented as a node identifies its relation with "king" node and saying that it contains two instances of king and one of the first king instance coordinates X or Y is different from the second king instance with 2, also kings are of different colors.

The algorithm brings up the name of the concept, relations and regularities, also certain examples when tutoring.

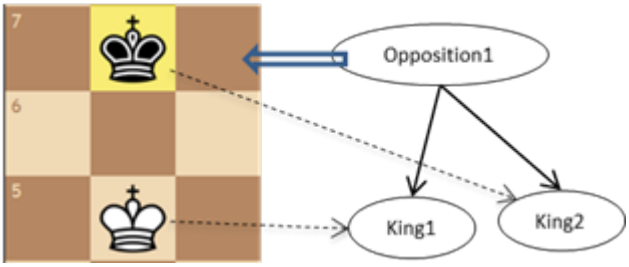

Fig. 3 Opposition chess concept node structure.

The algorithm also has a module which solves a certain chess problem according to the knowledge it has. This module allows comparing the solution of the problem by the student and by the mentioned module for checking the solution result.

The strategy constructing knowledge is described in goals grouped in plans. Strategies are explained using those plans.

For each unit of knowledge examples (chess graph nodes, goals, plans) are added to demonstrate on the board (ideally it will generate situations by itself)

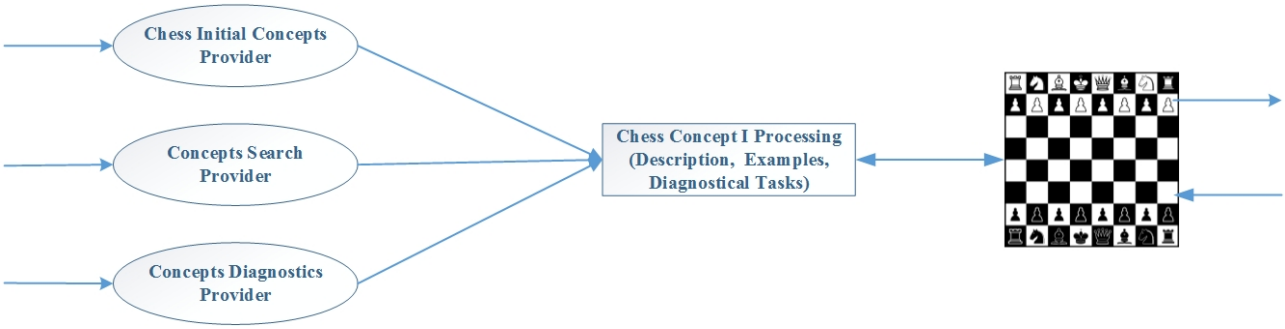

Fig. 4. Chess Concepts Teaching Software Workflow Structure.

The algorithm works like described below:

i. Tutoring chess concepts

It is possible to start learning from the beginning or learn a certain chess concept.

a. If the student selects learning from the beginning the algorithm starts tutoring by definition of chess initial concepts which are board (we identify it by X and Y coordinates), each figure type and its moves, playing sides (black and white), mate, stalemate and other chess rules. Initially the algorithm can start tutoring from the following concepts required by others: 1) X coordinate of the board, 2) Y coordinate of the board, 3) type of each figure 4) playing side colors.

b. If a certain concept learning is requested and this is a piece of knowledge then each relation of the knowledge piece is brought to the student and explained, if the student does not know knowledge pieces related to the tutoring concept, then all those related concepts are explained similarly until the algorithm reaches the concepts which are known by the student, for the tutoring piece also the regularities are defined, e.g., if “pawn” "concept is taught to the student, then it is identified as a related concept to “figure” as pawn is a figure, pawn moves are described, pawn Y coordinate restrictions are indicated. If a plan of a strategy is being taught, then all the required knowledge pieces are taught as described above, then all the related goals and their priorities are taught. If the student is unable to understand the description and regularities of the chess concept after several attempts, then the request is forwarded to the expert for improving the definition of that concept and explanation to the student.

ii. Testing tutoring

After teaching, certain chess knowledge testing is done by asking the student to solve chess problems. The student solution is compared to the problem solving module of the software which solves the problem according to the taught concept, too. The results are compared and evaluated in a tool which compares the games, measures and rates the chess players. If the solution doesn’t match the expected result suggested by the chess solving module then the wrong solution is corrected, the correct approach is explained in details. If the student still does not understand the explained plan, goal or another chess concept, then again the concept is explained iteratively as explained in b section of 1st point.

3. Implementing Chess Tutors

We use Reproducible Game Tree (RGT) Solver package and developed external tool for tutoring the student and measuring the performance. Here we will describe how the algorithm is integrated with RGT Solver and Connection tool and will demonstrate the tutoring process on a certain chess endgame tutoring example.

3.1. Solver as an environment for chess tutoring

i. RGT Solver

RGT Solver [6, 7] is a package aimed to solve problems where space of solution is a reproducible game tree [3, 4, 5]. Chess as an example of RGT problem is widely used by our team for providing experiments in the development or Solver. Overall chess problem definition in the Solver is discussed in [9].

In Solver the knowledge pieces are kept in a graph called Graph of Abstracts (GA) where each node has 3 types of English language derived relations to other nodes: be, have and do [10]. Personalized Planning and Integrated Testing algorithm is developed for optimal strategy searching, where strategies are described by plans [8].

We assume an expert defined the chess game in Solver and it is ready for execution and playing. Once the chess is brought to Solver it can be also used for tutoring. We usually start definition of chess from X, Y coordinates of the board, figure color and figure type concepts. Those are the nuclear concepts which cannot be parsed in the given way of definition. If an expert wants to make deeper level of definition he will have to define lower level nuclear concepts of chess. Currently we only use the mentioned set of nuclears. Next we define figures as a composition of X, Y, figure color and figure type concepts, each action for figures, also mate, stalemate are being defined and whole the required chess strategy related knowledge. As an example node of chess concept king has the following regularities in it: 1) it is a figure, so king has a connection to figure concept (it is a "be" type of language derived connection), it has X, Y coordinates and figure color just like figure general concept, certain figure type concept value which identifies that this is a king (those are "have" type language derived connections) and king move definitions (those are "do" type connections).

ii. Explanations in the Graph of Abstracts

The overall tutoring algorithm is brought in Fig. 5. For any concept "X" Solver node brings the relations to other nodes "Y" and "Z", and each of them is explained respectively if needed. Let's consider "Y" is known by the student and "Z" is unknown, then "Z" is parsed and its relations and regularities are brought for explanation, until we get to the nodes which cannot be parsed to more simple ones anymore.

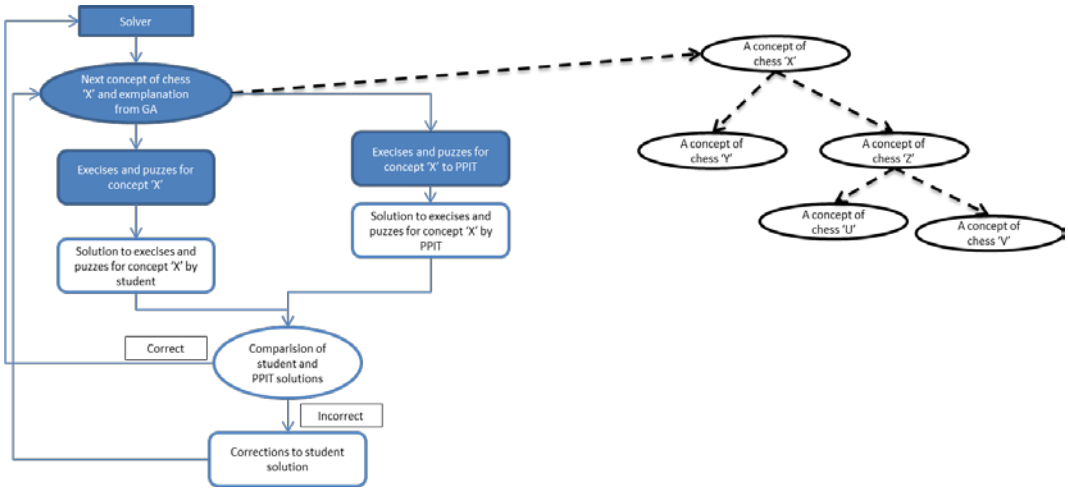

Fig. 5. Algorithm of tutoring for a chess concept.

Generally explanation of any node of GA consists of a) name of the node concept, b) its relations to other concepts (any of "be", "have", "do" relations which are in the definition of that node) and related concept names, c) regularities between related nodes for the certain node, e.g., opposition concept has two kings as instances in it king1, king2 and king1's color is different from king2's color, king1's X or Y coordinate is different from king2's coordinate by 2 and the other coordinate of king1 equals king2's respective one. Those regularities are also brought in the explanation panel of Solver when tutoring, d) certain situations containing the described concept.

i. Explanation of strategies

Other than GA nodes strategy searching plans can be also explained to the student. Plan is a set of goals sorted by their priorities, where each goal is a composition of chess concepts definition precondition and postcondition of the goal, the depth of tree to search for the goal

from the initial situation and criteria to evaluate how good the goal is achieved. Description of plan is done by explaining each goal of plan and stating its priority, for each goal its preconditional and postconditional chess concept node in GA is explained as described above. The depth of search tree is brought and criteria are explained, where each criterion is also a regularity, which may need calculation.

For each taught planning algorithm and related chess knowledge Solver suggests testing chess problems. Solution to the problem suggested by the student is being compared with the solution suggested by the PPIT algorithm generated by the same chess knowledge or the same plan. Moves comparison is done by the Connection Tool and if they do not match, that means the student did not understand the explained plan or concept correctly, the correct solution of the problem is demonstrated to the student, the tutoring knowledge is explained again and a new problem for the same knowledge is suggested to the student to solve.

ii. Improving explanations

If the explanation is bad and the student is not able to understand the concept, the problem is forwarded to the expert who improves the definition of the expected chess knowledge to make it understandable and clear and explains the student if needed.

Regular improvement of Solver and PPIT are achieved by Connection Tool. The tool provides an ability to execute chess powerful engines, which allow checking how well the strategy plan is and request for correction if needed. Also Connection Tool enables rating the student, also creating certain games and suggesting more chess problems by the requests.

3.2. Handling the interactions between the student and tutors

During the study a new software tool was also designed, that allows to establish a connection between chess engine, for example Solver, and player, to handle chess game and fix its results. The tool has a modular construction.

Engines Interaction Tool allows to run chess engines, it creates a session for started engines and controls their lifecycle. It also creates a session for regular player and provides a simple input/output interface for interaction with the chess engine using text commands. Designed tool has possibility to start chess games between the player and the chess engine from scratch or from defined board position to the end of game or for defined count of game moves. It allows to get info about game states and game results. It also has an ability to connect to board GUI programs to visualize the defined cases. The tool has a mechanism of communication between the chess player and the engine based on using of communication protocol, the rules of which allow to send different types of data.

In context of tutoring concept defined in this work Engines Interaction Tool is able to show information about requested for learning chess regularity, show examples of the regularity using chess board visualization. Then the tool allows to check the understanding of the material using the requested tasks concerning the regularity by organizing chess game or some part of it and comparing the expected results for solving the tasks and the results of chess player. Afterwards some chess games can be handled between learning player and some chess engines to check advantages of learning the regularity for playing chess game as whole from start to finish.

To implement the listed functionality Engines Interaction Tool has the following structure, shown in Fig 6.

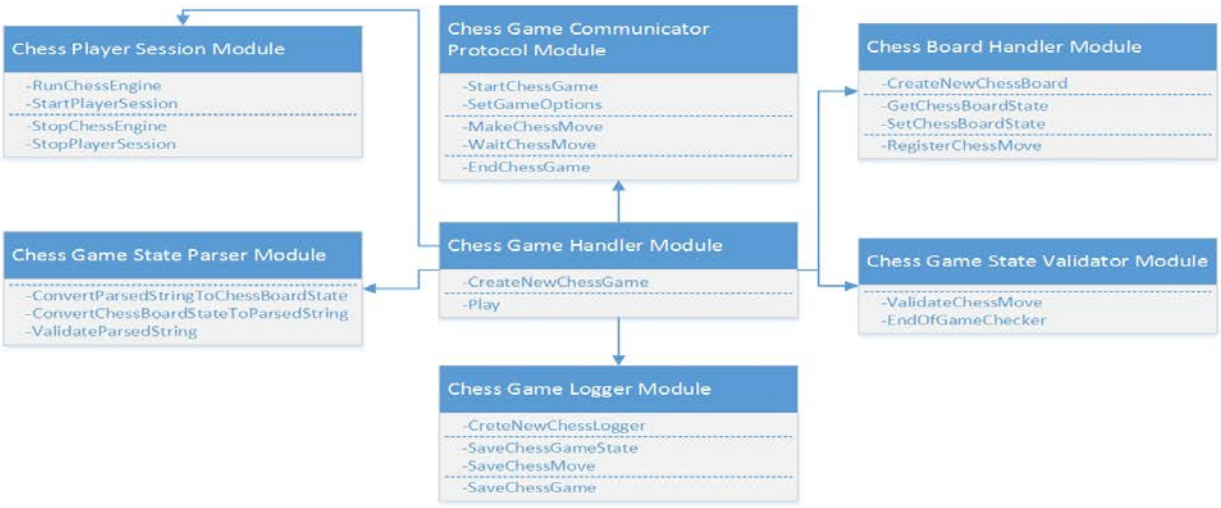

Fig. 6. Engines Interaction tool modular construction.

ChessPlayerSession module allows to create a new session for chess player. Chess player can be represented by chess engine or a person. Using this module the chess engine can be started as a new process. The module also handles running chess engine process destroying.

Sending game data between the chess engine and the player can be performed using one of chess game communication protocols. As an example of one of the most used such protocols UCI [11] was implemented as a part of corresponding tool module named ChessGameCommunicationProtocol.

ChessBoardHandler module is designed to represent chess board. It allows to create new chess boards for game, to set the chess board to some state or to get chess board current state. To change the state of board also registering new moves method can be used.

ChessGameStateParser module is used to save any intermediate game state using some notation, restore it specific condition, described using that notation. One of the most used chess game state notations is Forsyth-Edwards Notation (FEN) [12]. FEN record describes chess game particular board position. ChessGameStateParser module contains FEN parser implementation, so a board state can be converted to FEN string and vice versa.

ChessGameStateValidator module is designed to check if the specified state of game is valid in terms of chess. It also checks the meaning of state for game, so it can detect the end of game. Designed tool also has a module, that allows to conduct chess games named ChessGameHandler. So using this module a new chess game can be started and run until the end or being stopped. The game can be also started from the defined intermediate board position.

ChessGameLogger module allows to save different conditions of game, moves, results and a game as a whole.

To fix the player's results ChessGameLogger module is used to retrieve game data, analyze its contents and provide some progress measurement details.

The designed tool is flexible, so other chess game protocols, other ways of engine running or connected, other parsing notations, etc., can also be implemented and used if necessary.

3.3. Tutoring in endgames

For the validation of Solver development of the given tutoring algorithm and development of the Interaction Tool we discuss an example of tutoring a student to a chess endgame “mate by one rook” assuming that the student already knows at least the basic rules of chess, i.e., figures, their moves, mate, stalemate, etc.

3.3.1. A winning strategy in Rock against King endgames

1. Put mate
2. Avoid stalemate
3. Escape rook from attack
4. Push king to the edge (without putting rook under attack)
5. Make a waiting move when preOpposition appears
6. Bring white king closer to the opponent king

Each step defines a goal and its priority. Tutoring starts with showing the plan to the student. Then each of plan goals is explained and an example is demonstrated on the board.

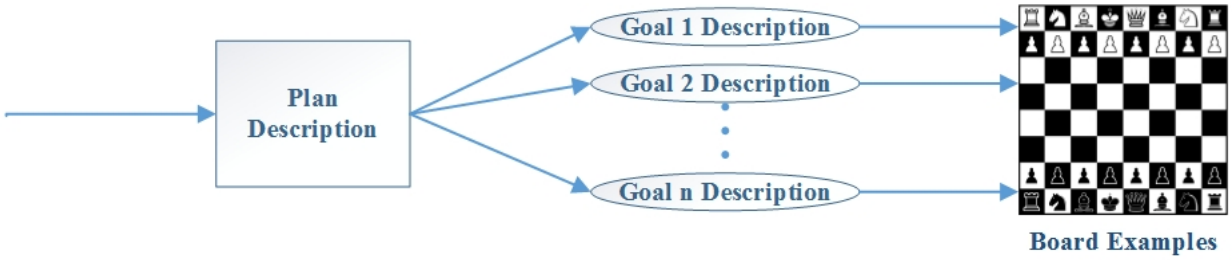

Fig. 7. Chess Concept Tutoring Process.

3.3.2. Tutoring Rock against King endgames.

1. First goal is "put mate" for which preCondition is any situation, and postCondition is a situation where mate exists, the depth is 1. Since we consider a student who already knows chess rules, then explanation of this goal does not need to go deep, it just shows the goal, its tree depth, preCondition and postCondition patterns (in general cases the program will explain "mate" and each of its component concepts if needed). For the mentioned goal a situation is suggested to the student to solve. The student solves the problem and if that is correct the next goal explanation is started, if the student makes a wrong move comparing to Solver execution of the goal, then the goal is explained again and again a situation is suggested to solve. The comparison of student to Solver suggested moves, as well as chess game playing ability are provided by Interaction Tool.

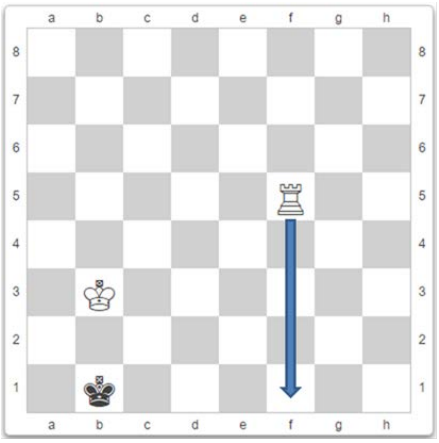

Fig. 8 Chess goal of "put mate".

2. "Avoid stalemate" goal is explained similar to 1st goal since there is no much difference in knowledge levels used in those goals, preCondition of this goal is again any situation and the postCondition is a situation where no stalemate appears. The depth of search is 1. Again stalemate is known from chess rules, and if no, then it is explained. We consider a student who knows, then Solver just brings the info in the goal and suggests a situation to make a move or moves which are good for this goal. Again this is done using Interaction Tool.

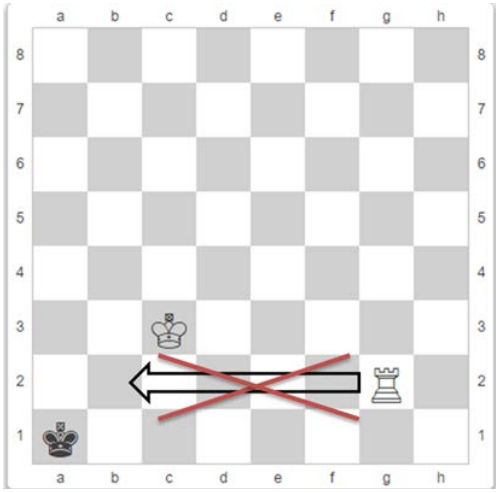

Fig. 9 Chess goal “avoid stalemate”.

3. “Escape rook from attack” is a goal which preCondition is “rook under attack” abstract, indicating the goal is applicable for situations where the rook is under the attack (in the example we consider it under opponent king’s attack). The postCondition is a situation where rook is not under attack and the vertical or horizontal coordinate of the rook is not changed depending on what plan we execute: pushing king by vertical or by horizontal. It has a search depth of 1 and the evaluator will have one criterion defined which calculates the distance of the rook and opponent king by vertical/horizontal direction. The explanation is started from overall definition of the goal, where we can consider the student does not know the concept “rook under attack” which is a concept derived from “field under attack” concept indicating that there is a rook instead of field, which is shown to the student. If the student does not know “field under attack” concept, it is explained by its specifications, where there are different types of attacks, such as “field under attack of king” which is needed in this very case and other similar specifications. “Field under attack” concept is a virtual abstract in GA and “field under attack of king” is its specification (GA node types are described in [6]). “Field under attack of king” is also a virtual abstract which needs to be explained by its all specifications where each possible attack of king is being demonstrated, overall there are 8 specifications of this concept to explain. We consider the student already knows field is under attack of king abstract, as this is a part of basic chess rules defining king moves and attacks. After explanation of “rook under attack” an example for this concept is demonstrated on the board. Next postCondition is explained, where “rook under attack” concept is used again. For the goal depth is indicated and unchanged horizontal or vertical coordinate of the rook is shown as regularity. The criterion is also named to the student indicating that goal achievement is considered better when the distance of the rook from opponent king is maximal by showing the name of the criterion, its regularity and demonstration of two different examples mentioning which is better for this criterion. Again a situation is suggested to solve and solution is compared with Solver execution of the same goal.

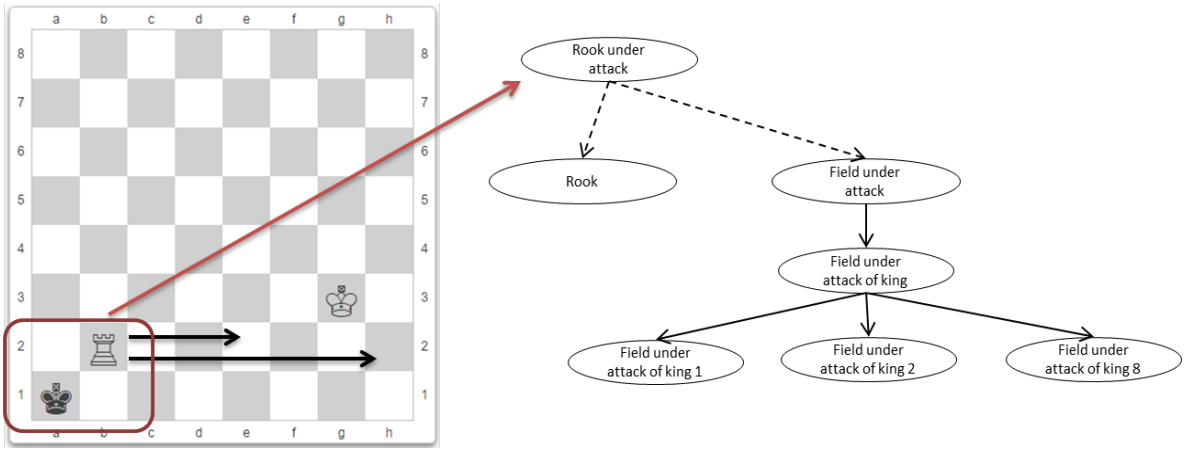

Fig. 10. Chess concept “rook attack” and goal “escape rook attack”.

4. “Push king to the edge (without putting rook under attack)”, where preCondition can be any situation, so nothing to explain and postCondition is “rook is not under attack” which is already explained in 3rd goal, search depth is 2. The evaluator has two criteria which are brought to the student. First is: moves of opponent king are closer to the edge are better, where Solver uses definition of “edge” concept here which is explained to the student as lines of board (defined as set in GA) where either x = 1 or x = 8 or y = 1 or y = 8. The second criterion for this goal evaluator is the number of actions opponent king can do, and the better action is the action

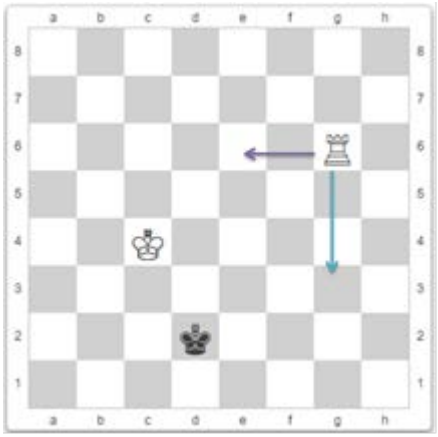

Fig. 11. Goal “push king to the edge”.

which allows fewer number of actions by opponent king. King moves are known by the student and the criterion just shows that minimal number of opponent king moves are better. One more time a situation to solve the goal is suggested and compared to the Solver solution for the same situation.

5. “Make a waiting move when perOpposition appears” goal is defined: preCondition is preOpposition situation. preOpposition is a concept to be taught. In Solver we defined it as a concept that contains two kings, king1 and king2 and is a virtual abstract in GA. Its explanation is done by tutoring of each specification of the given virtual abstract, where each specification identifies specific relations between two kings where opponent kings

appear, e.g., one of perOpposition specifications, we call it perOppositionByVertical1 identifies the following relations for the kings which are shown during the explanation process of preOpposition virtual abstract king1.x = king2.x + 2, king1.y = king2.y - 1. Similarly another preOpposition situation is explained to the student perOppositionByVertical2, where regularties shown to the user are king1.x = king2.x + 2, king1.y = king2.y + 1. Other preOpposition specifications are explained to the student, too and for each of them certain situations are braught as examples. The postCondition of this goal is a situation where the own king position is not changed and vertical/horizontal (depending on the direction of pushing king) coordinate of the rook is not changed, so only regularities defining unchanged king position and rook vertical/horizontal coordinate and explaining situations are shown to the student. Searching depth of goal is 1. The evaluator again has one criterion, indicating the distance between opponent king and own rook shall be maximal, which is shown to the student, too. Usually a chess knowing student would not need to be taught for the regularity of “king position is not changed” and many similar concepts at all, but if he/she needs it, the regularity will be shown and a certain example is brought.

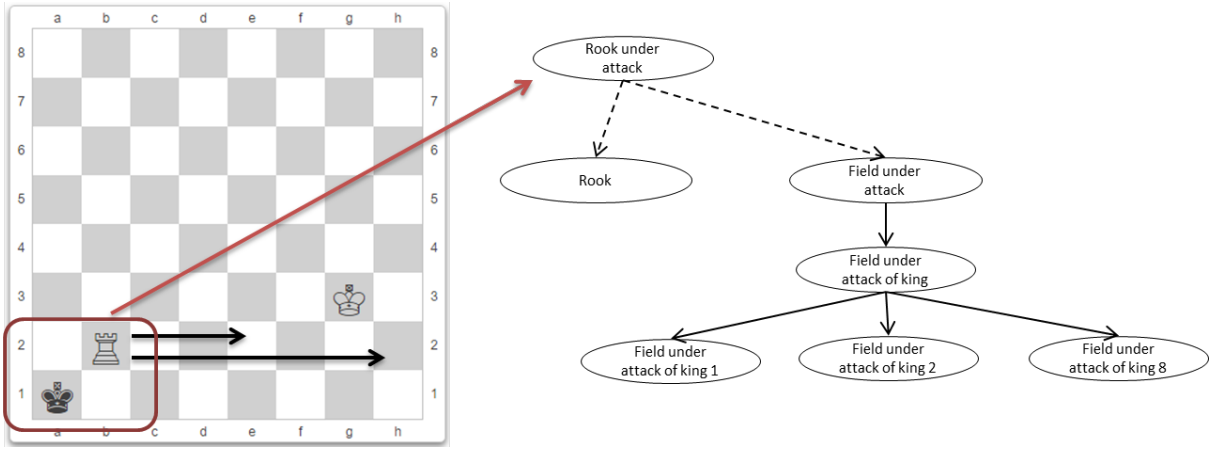

Fig. 12. ”preOpposition” chess concept and “make waiting move” goal.

6. “Bring white king closer to the opponent king, but avoid opposition” goal does not have any related concept for explanation in preCondition, while for postCondition “opposition” concept is being explained. Opposiotn explanation is similar to preOpposition concept with the difference that specifications of Opposition virtual abstract are two, oppositionByVertical and oppositionByHorizontal, searching depth is 1. The evaluator has one criterion, which defines the distance of the king from the opponent king is minimal. We can calculate this by the following formula “$(king.cordX-opponentKing.cordX)^2 + (king.cordY-opponentKing.cordY)^2$”, and the criteria are named to the student. The formula is shown as the regularity calculating the minimal distance if needed (here also usually the student would not need this regularity for calculation of two kings distance).

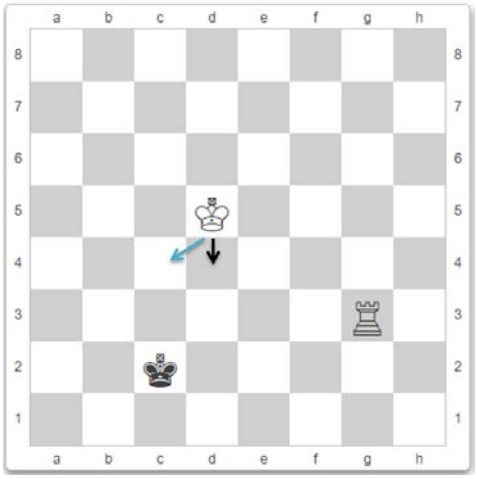

Fig. 13. “bring king closer to opponent king” goal.

3.3.3. Examining tutoring Rock against King

After tutoring the student for the “mate by one rook” strategy this is being tested on a situation, where each move of student is compared with each move of PPIT executed in Solver. After each move Interaction Tool provides opponent’s move after student making correct move according to the plan. If the student suggested move does not match PPIT suggested move, then the PPIT action is explained in details for the PPIT plan execution algorithm. The correctness of PPIT execution of “mate by one rook” is demonstrated in [8]. If the student is able to solve the situation as PPIT does, then the concept is counted as learned. If the student was not able to learn the plan, then explanation process is done again. If the same problem with understanding appears several times (the number is configurable depending on the student performance and abilities) a request is sent to the expert that Solver was unable to explain. Meantime while PPIT and student are solving “mate by one rook” situations powerful chess engines are executed by the Interaction tool to measure the performance of PPIT and student, too. If PPIT solution to the same problem is bad, then a note is sent to the expert about the notices issue and situation where it appears.

Similar to the described example more endgame plans and relevant chess knowledge pieces can be taught to the students in the interactive manner described above without effective input by the expert during explanation process. The demonstrated example proves that the developed algorithm and software tools are able to tutor students to chess problems and concepts, particularly chess endgames can be taught and executed and PPIT solution and concepts understanding by the student can be measured to improve the tutoring performance.

3. Conclusion

1. Method and software for tutoring chess are designed and involved within the RGT Solver package, external tools are implemented and integrated providing students with the following advantages.

a. The software provides personalized tutoring mechanism for different types and levels of students and their performances.

b. The software is interactive making level by level tutoring of chess concepts, their testing, also feedback provision and correction by detailed description.

c. Analyzing games by currently high rated chess engines, provides students' performance measurement mechanisms by rating the students and validating the learning results by checking student learned concepts and tutoring software performance.

2. The designed algorithm and software are validated by testing them on tutoring for the chess endgame.

3. For future improvement of suggested chess tutoring mechanism, chess tutoring protocol concept exploitation is in progress.

4. The developed education concept can be represented as a part of technology enhanced learning (TEL). TEL research covers Learning Management Systems as a learning tool and also non-learning tool as a part of heterogeneous learning environment. Education personalization approach, which we use in specific field (chess), is object of interest in nowadays TEL research. The arranged concept can be bound with Contextualized Attention Metadata (CAM) which is defined by user attention in a given context. To implement this idea integration with the CAM framework can be conducted. We can use as an example the framework described in [21] which has an additional advantage of being able to gather CAMs produced by any tool or computer system. As a middleware level to access to central attention repository the framework has two dynamic services provides possibilities to: 1. define attention data that we want to collect; 2. integrate with the service dedicated to receive and retrieve traces produced by computer systems.

Binding with the CAM framework can allow to use the attention metadata for making personalized recommendations for learners. The solution described in [22] can be used so as it recommends documents to students according to their current activity that is tracked in terms of semantic annotations associated to the accessed resources. Also personalized search tool [24] can be integrated with the suggested concept.

Integration also can take into account the architecture that facilitates sharing and reusing of learner profiles [23] for the education concerning other RGT class tasks.

Further development can be cooperated with other e-learning researches like the one presented in [25].

5. As a practical application it is possible to collaborate with the systems that provide virtual chess education like [26] directed by T. Ogneva. The integration with such system can allow them to use the advantages of suggested personalized chess learning concept to be more relevant in context of TEL systems.

Acknowledgements

Authors express their deep gratitude to Edward Pogossian for supervising this research.

Rerefences

[1] G. Levenfish, *Book For a Beginner Chess Player*, (in russian), Moscow, Russia, 1957.

[2] (2015) Chesscademy chess learning website. [Online]. Available: https://www.chesscademy.com

[3] E. Pogossian, V. Vahradyan and A. Grigoryan, "On competing agents consistent with expert knowledge", *Lecture Notes in Computer Science, AIS-ADM-07: The International*

Workshop on Autonomous Intelligent Systems - Agents and Data Mining, pp. 229-241, St. Petersburg, Russia, June 6-7, 2007.

[4] E. Pogossian, A. Javadyan and E. Ivanyan, "Effective discovery of intrusion protection strategies", *The International Workshop on Agents and Data Mining, Lecture Notes in Computer Science*, St. Petersburg, Russia, vol. 3505, pp. 263-274, 2005.

[5] E. Pogossian, "Effectiveness enhancing knowledge based strategies for SSRGT class of defense problems", *NATO ASI 2011 Prediction and Recognition of Piracy Efforts Using Collaborative Human-Centric Information Systems*, Salamanca, Spain, p. 16, 2011.

[6] K. Khachatryan and S. Grigoryan, "Java programs for presentation and acquisition of meanings in SSRGT games", *Proceedings of SEUA Annual conference*, pp. 127-135, Yerevan, Armenia, 2013.

[7] K. Khachatryan and S. Grigoryan, "Java programs for matching situations to the meanings of SSRGT games", *Proceedings of SEUA Annual conference*, pp. 135-141 Yerevan, Armenia, 2013.

[8] S. Grigoryan, "Structuring of goals and plans for personalized planning and integrated testing of plans", *Mathematical Problems of Computer Science*, vol. 43, pp. 62-75, Yerevan, Armenia 2015.

[9] K. Khachatryan, S. Grigoryan and T. Baghdasaryan, "Experiments validating the Be-Have-Do meaning presentation model and matching algorithm for competing and combating problems", *International Conference in Computer Sciences and Information Technologies*, pp. 155-159, Yerevan, Armenia, 2013.

[10] E. Pogossian, "On modeling cognition". *International Conference in Computer Sciences and Information Technologies*, pp. 194-198, Yerevan, Sept.26-30, 2011.

[11] UCI (universal chess interface), *CCC*, R. Huber and S. Meyer-Kahlen, November 28, 2000.

[12] Forsyth-Edwards Notation, Portable Game Notation Specification and Implementation Guide, S. Edwards, 1994.03.12.

[13] E. Pogossian and E. Arakelova, "Tools for testing and correction of the completeness of knowledge acquisition by autistic children", *International Conference in Computer Sciences and Information Technologies*, pp. 159-165, Yerevan, Armenia, 2011.

[14] D. Gadwal, J. Greer and G. McCalla. "Tutoring bishop-pawns endgames: an experiment of using knowledge-based chess as a domain for intelligent tutoring", 34 p., preprint S7N0W0, University of Saskatchewan, Canada, 1992.

[15] T. Ogneva, "Some psychological aspects of teaching children the chess game", Chess and Education, 2004.

[16] V. Anisheva, "Methodical aspects of individualized chess initial training for primary school children", Ph.D. thesis, 165 p., RGAFK, Moscow, 2002,

[17] D. Martinovic and I. Markovic, "Piece values and piece exchange in chess: individualized instruction of schoolchildren using tests with three levels of difficulty", *Mathematics, Computing Education XVIII Conference*, 2011.

[18] A. Kirsanov, *Individualization of Educational Activity As a Pedagogical Problem*, Kazan University Publishing, Kazan, 1982.

[19] I. Unt, *Individualization and Differentiation of Teaching*, Education Publishing, Moscow, 1990.

[20] A Granickaya, *Learn to Think and Act*, Adaptive learning in school, Education Publishing, Moscow, 1991.
[21] V. Butoianu, P. Vidal, K. Verbert, E. Duval and Julien Broisin, “User context and personalized learning: a federation of contextualized attention metadata”, *Journal of Universal Computer Science*, vol. 16, no. 16, pp. 2252--2271, 2010.
[22] J. Broisin, M. Brut, V. Butoianu, F. Sedes and P. Vidal, “A personalized recommendation framework based on CAM and document annotations”, *Proceedings of the 1st Workshop on Recommender Systems for Technology Enhanced Learning*, pp 2839–2848, 2010.
[23] T. Ramandalahy, P. Vidal and J. Broisin, "Opening learner profiles across heterogeneous applications", *Advanced Learning Technologies, Ninth IEEE International Conference*, pp. 504-508, 2009.
[24] J. Broisin and P. Vidal, “A management framework to recommend and review learning objects in a web-based learning environment”, *Advanced Learning Technologies Sixth International Conference*, Kerkrade, Netherlands, pp 41--42, 2006.
[25] J. Broisin and P. Vidal, “E-learning research at the IRIT laboratory”, *International Conference in Computer Sciences and Information Technologies*, Yerevan, Armenia, 2015.
[26] (2015) Chess tutoring website. [Online]. Available: http://virtualchess.ru

Submitted 11.08.2015, accepted 20.11.2015

Ինտերակտիվ անձնավորված շախմատային ուսուցման ծրագրի կառուցում

Ս. Գրիգորյան և Լ. Բերբերյան

Ամփոփում

Առաջարկվում է ալգորիթմ և համակարգչային ծրագիր անձնավորված ինտերակտիվ շախմատային ուսուցման համար: Աշխատանքը սկսվում է որոշ ստանդարտ շախմատային ուսուցման մոտեցումների վերլուծությամբ՝ նշելով դրանց առավելություններն ու թերությունները, հատկապես անձնավորված և ինտերակտիվ ուսուցման վերաբերյալ: Ապա ներկայացվում է ալգորիթմը՝ որպես շախմատային ուսուցման առաջարկվող մոտեցում, որը, կախված աշակերտի ունակություններից, ցուցաբերում է տարբեր մոտեցումներ, հետո նկարագրվում է առաջարկվող ալգորիթմի իրականացումը, որը ներառում է դրա ներգրավումը RGT Solver փաթեթը եւ շախմատային շարժիչների հաղորդակցման գործիքը՝ ներկայացնելով ուսուցման կառավարման և ընթացքի կարգավորման եղանակները: Ցույց է տրվում նաև շախմատային վերջնախաղի ուսուցման հիմնավոր օրինակ:

Построение интерактивной персонализированной программы шахматного обучения

С. Григорян и Л. Берберян

Аннотация

Предлагается алгоритм и программное обеспечение для персонализированного интерактивного шахматного обучения. Статья начинается с анализа некоторых стандартных подходов обучения шахматам, перечисляя некоторые их достоинства и недостатки, в особенности, касающиеся персонализированного и интерактивного обучения. Далее определяется алгоритм как подход к обучению шахматам, предоставляющий вариативное поведение в зависимости от конкретных навыков студента. Затем описывается применение предложенного алгоритма, которое включает интеграцию с пакетом RGT Solver и инструментом контроля взаимодействия шахматных программ, предоставляя механизмы контроля обучения и фиксирования результатов прогресса. Также приводится конкретный валидный пример обучения эндшпилю.