Using Bayesian Networks for Knowledge Representation and Evaluation in Intelligent Tutoring Systems

Ramirez-Noriega Alan*, Juarez-Ramirez Reyes*, Martinez-Ramirez Yobani**, Jimenez Samantha*, Inzunza Sergio*

Autonomous University of Baja California *, Autonomous University of Sinaloa **

Abstract. Assessing knowledge acquisition by the student is a main task of an Intelligent Tutoring System. Assessment is needed in order to adapt learning materials and activities to students capacities. To evaluate knowledge acquisition, different techniques can be used, such as probabilistic inference. In this paper we present a proposal based on Bayesian Networks to infer the level of knowledge possessed by the student. We implemented a kind of test to know what student knows. During the test, the software system chooses the new questions based on the responses to the previous ones, that is, the software system makes an adaption in real time. To get the inferences, we use a network of concepts, which contains the relationships between those concepts. This work is focused on the design of the Bayesian Network and the algorithm to do inferences about students knowledge.
Keywords: Knowledge representation, Bayesian Network, Evaluation, Intelligent Tutoring System.

1 Introduction

Learning can be defined as internal processes of change, as the result of learners personal experience. Also, it can be defined as the acquisition or adding of something new, which involves any variation or modification previously acquired [17].

Teachers guide students during the learning and must perceive the students needs in order to improve the teaching. However, in group tutoring environments, one-on-one time dedicated by professors to each student decreases considerably. For that reason, some authors propose the use of a software system [1][2][7][20] to satisfy that needs. Furthermore, the software system should be adapted to the students needs.

Adaptability to students needs is a challenge for software engineering [10]. Adaptability is defined as the software adaptation to individual user characteristics according to user aims [14].Different types of software adaptations are defined [14][16], but this research focuses on the content adaptation; that is, what information is shown to the user according to the software interaction and user characteristics. This adaptation of the learning environment can be achieved with artificial intelligence strategies [16], carrying out intelligence for deducing user needs.

A special type of software that meets the characteristics mentioned above is called the Intelligent Tutoring System (ITS). This can be defined as a software system that uses artificial intelligence techniques to interact with students and teach them [7][20] in the same way as a teacher does to his students [2]. Carbonell [1][2] proposed a generalized architecture for ITS, which considers three basic modules [2][20]:which are the tutoring model, domains model, students model, further of users interface.

An important problem in ITS development is the assessment of student knowledge [4]. ITS must be able to determine accurately and quickly the student cognitive level to decide what is important

Á. Rocha et al. (eds.), *New Advances in Information Systems and Technologies*,
Advances in Intelligent Systems and Computing 444,
DOI 10.1007/978-3-319-31232-3_16

to teach them. Probability theory has been proposed by some authors for handling the uncertainty in diagnosing student knowledge [7][20][4]. The Bayesian Network (BN) theory is proposed, within a framework of probability and artificial intelligence, for modeling the way how an intelligence system should infer causality [21]. Besides, this theory has a representation and behavior similar to peoples mind [17].

Our research project considers the causal relationships to refer to nodes that represent concepts, and they are related to other nodes to obtain the domain knowledge representation [12]. This paper proposes a knowledge evaluation module for ITS, to diagnose the student learning needs efficiently, in order to reinforce topics. This research is supported in the BN theory considering uncertainty handling, knowledge representation, and the wide use in diagnostic and pattern recognition [20][8][9][13].

We selected BN to this study over Fuzzy Cognitives Maps (a similar technique) because they have important attributes as [3]: (1) Forward and backward chaining, (2) efficient evidence propagation mechanism, (3) enough implementation and support tools, (4) mathematical theorems derivable from well-defined basic axiom, and (5) correctness of the inference mechanism is provable.

This paper is organized in six sections. Section 2 defines some related work. Section 3 explains our proposal for knowledges assessment. Also, it shown how the BN is implemented for knowledge representation and its assessment; moreover, it presents a question-based evaluation design that we used for student evaluation. Also this section explains the algorithms employed in the module. Section 4 defines the methodology for experimentation that we employed. Section 5 shows the experiment results and discussions. Lastly, Section 6 presents the conclusions and future work.

2 Related works

In this section we present the related work, emphasizing the use of BN for improve learning and education. Taking into acount that BN are used to assess the knowledge.

Liu et al. [9]proposed a student modeling method built with BN. To assess the students performance, they adopted a logistic model with three parameters to calculate the conditional probability distribution of the testing item. They were focused on course of Data Structures, especially on Binary Trees. This work just was a proposal, but they did not implementation of the model.

Goguadze et al. [6] presented the design and evaluation of a Bayesian approach for modeling student misconceptions in the domain of decimals. The results showed that the models predictions reach a high level of precision, especially in predicting the presence of student misconceptions. They did not explain precisely how the BN was built.

Torabi et al. [22] worked on predicting the student courses score based on the students educational history. They proposed a BN model for the inference process. The results show that applying their proposed method has primary effects on the quality of the students learning and can be used as a helpful tool for them.

Millan et al. [12] developed, integrated and evaluated a Bayesian student model. This work is focused on the mathematics area, specifically on first-degree equations. They used twelve concepts to assess the knowledge. Each concept is evaluated in batches of four questions or exercises, this mean you need answer four questions as one.

The main model used is similar to the Millan et al. [11]. But, some differences are considered with the previous work. Our model evaluates the fundamentals of algorithms as learning topic. We did a complete analisis of domain knowledge for represent it. Also, to adapt the questions according Student's knowledge level, Bloom's taxonomy [5] was implemented organizing the questions into five levels of complexity. Finally, we used own algorithms to switch-on the levels and to select the appropriated question to the student.

3 Implementing a module for knowledge representation

3.1 Setting the diagnostic of the student using Bayesian Networks

BN help us determine the students cognitive degree; this means we can assess areas of superior knowledge and areas in development. Firstly, we define essential elements for using BN to diagnose students problems. The elements are variables, links between variables and parameters. This work is based on the method used by Millan [12], where they consider the following aspects:

- Variables for measuring students attained knowledge: we use three levels of granularity. The concepts are found in the lower level. These represent the smallest unit to dividing the knowledge. The topics are the next level, these are clusters of concepts. Finally, the las level is represented by units that involve topics.
- Variables for gathering evidence: These are multiple choice questions and can be right or wrong.
- Links between variables to measure the knowledge: Dominating knowledge has causal influence on knowing preceding, immediate levels in the related granularity hierarchy. Regarding links between the nodes and the questions, we consider that knowledge has a causal influence on correctly answering the questions [12].
- The word parameters signify dependence on probability values of the child nodes given their parents

3.2 Building the Bayesian Network

We use a building process divided into six phases [15]: (1) Defining a knowledge domain: selecting the work area. (2) Developing a hierarchical scale of knowledge: classification of the knowledge in different levels. (3) Building the Bayesian Network: creating nodes and establishing dependence relations between them. (4) Designing the conditional probability tables (CPT): assigning probabilities to nodes, according to relationships with parents. (5) Designing questions to evaluate knowledge: creating a bank of questions and assigning relations with the concepts contained in nodes. (6) Creating the CPT for the questions: assign probabilities to the question nodes according to their parents.

The phases 1, 2, 3, and 5 are used for creating the network structure, and the phases 4 and 6 to calculate the estimated probability values for each node. The phases 4 and 6 could be combined in a single step, but they were divided into two phases allowing us better organization and clarity.

For our experiment we considered the course Algorithms and Computer Logic, which is part of the curriculum for the Software Engineering undergraduate program in the Autonomous University of Sinaloa, Mexico. This course has a set of programming topics. Following the methodology presented above, we created the BN, which contains the course structure ordered hierarchically as concepts, themes and the unit name.

We considered the first unit that contains introductory topics, such as: program, programmer, data, algorithm concepts, variables (concept, assignment, and types), constant, algorithms characteristics, algorithms types (qualitative and quantitative), and algorithm classification (sequential, conditional, and repetitive).

3.3 Creating questions

We generated a total of 73 questions divided into levels. The questions are similar to the next example: Which kind of algorithms make decisions based on a given condition? This question is related to the topic of algorithms and their types. The answers are known (multiple choice) and their values are expressed in percentages (Algorithm concept 10%, Sequential algorithms 30%, Conditional algorithms 30%, Repetitive algorithms 30%). The value of the first response (algorithm concept 10%) means that the question is 10% related to the concept of Algorithms. That is, if the student knows the concept of Algorithm, then he has a 10% probability of answering the question correctly. The other three answers are the same, but with different values.

As we indicated earlier, values are assigned based on the expert proposal when s/he creates the exam. There is the possibility that experts will assign different values to questions and answers, depending on who creates the questions and responses. In order to prove the correct formulation of the questions and their answers we apply a survey with professors of the Software Engineering curricular block, who gave feedback on the formulation as: readiness, clarity, and so on.

Particularly, professors who teach the Algorithms and Computer Logic course collaborated on the survey. The feedback suggested us to improve the following aspects: (1) some concepts were not covered in the classroom. (2) Some questions and answers were ambiguous. (3) Some questions were located in a wrong category.

Attending to this feedback, we improved the set of questions and answers to increase reliability. Questions were organized based on Blooms Taxonomy [5]. We only work with the first five levels. Table 1 illustrates the organization.

Table 1. Levels of complexity

Bloom's taxonomy	Name of level	Number of question
Knowledge	Basic (B)	11
Comprenhension	IntermediareA (IA)	21
Application	IntermediateB (IB)	17
Analysis	AdvancedA (AA)	18
Synthesis	AdvancedB (AB)	6
	Total	73

According to Table 1, the questions (knowledge) of basic level recognize and retrieve relevant information from long-term memory to use in the short-term memory. The questions of intermediate level (comprehension) are questions that require building significance from educational material [5] [18].

The intermediate level (application) contains questions where a learning process is applied. The advanced level (analysis) divides the knowledge in parts and reasoning is required. Finally the advanced level (synthesis), this level joins elements to form a whole. Students reasoning are deepened; questions are slightly more complicated than the advanced level. The module aims for students to achieve their maximum potential to use at this level [5][18].

3.4 Knowledge Evaluation System

The software module is based on two algorithms, in order to adapt to the student. The first is based on level selection, and the second on question selection. Below each one is detailed.

Level selection algorithm Each question is randomly selected, starting from the basic level. The student can move through the levels, according to the knowledge of the course concepts. The algorithm 1 moves the student through the levels of questions. We use a data structure to save answered questions and their answer (right or wrong) by each level. Thus, we have a record of questions (*QuestionsRecord*). A loop is kept until the software decides to finish the exam. The exam may finish for four reasons: 1) low knowledge, 2) high knowledge, 3) do not approve a level, and 4) excess of questions. The last one limits the exam duration to 25 questions and avoids showing all 73 questions.

The *ShowQuestion*() procedure in algorithm 1 is not defined until this section, because it belongs to the questions selection and needs another algorithm; therefore, the section below is dedicated to that. The *EvaluateQuestion*() procedure defines if the question is right or wrong. The *AddtoQuestionsRecord*() procedure permits saving questions, answers, and level within the defined structure (*QuestionsRecord*()).

The *GoDownLevel*() and *GoNextLevel*() procedures allow movement through levels, either up or down, as the case may be. The *FinishTest*() procedure is used to finish the exam for one of the aforementioned reasons. The *ReachLevel*() procedure determines if a student should return to a previous level, and makes decisions based on question number. The *DiscontinuousQuestions*() procedure refers to question evaluation when a student cannot answer two or more questions in row, either right or wrong. The algorithm considers a level passed if most questions are correct.

Finally, the *ExtraQuestion*() procedure handles levels with five evaluated questions, when, according the software rules, an extra question is needed to decide if the student should go up or down a level, or finish the exam. Thus, the software controls the students movements through levels, deciding if the student goes down a level, proceeds to the next level, or ends the exam.

Algorithm 1 Algorithm for the levels

```
$QuestionsRecord$=Structure for control the questions by level
while not $FinishTest()$ do
    $ReachLevel()$;
    $ShowQuestion()$;
    $EvaluateQuestion()$;
    $AddtoQuestionsRecord()$;
    if $question=true$ then
        if three questions are corrects in a row and $level$ in $\{B, IA, IB, AA\}$ then
            $GoNextLevel()$;
        end if
        if $question=true$ and $level=AB$ then
            $FinishTest()$;
        end if
    end if
    if $question=false$ then
        if two question are corrects in a row then
            if $level$ in $\{IA, IB, AA\}$ then
                $GoDownLevel()$;
            end if
            if $level=B$ then
                $FinishTest()$;
            end if
        end if
        if $level = AA$ then
            $GoDownLevel()$;
        end if
    end if
    if five questions are showed in the level then
        $DiscontinuousQuestions()$;
    end if
end while
```

Selecting questions To select a question, node ranges must be classified as known, unknown, or indeterminate. Probability values are between 0 and 1, by a probability axiom [19].

This research defines the next classification: (1) Known Question (KQ): A node that has a value greater or equal to 0.7 and less or equal to 1 ($0.7 <= KQ <= 1$). (2) Unknown Question (UQ): A node that has a value greater or equal to 0 and less to 0.3 ($0 <= UQ < 0.3$). (3) Indeterminated Question (IQ): A node that has a value greater or equal to 0.3 and less to 0.7 ($0.3 <= IQ < 0.7$).

The $ShowQuestion()$ procedure chooses the question and shows it. The question probability value may be in the indeterminate range to be selected. The software module has not been able to classify those questions due to the ranges. Thus, the known and unknown concepts are discarded. Moreover, this procedure displays the selected question. The $EvaluateQuestion()$ procedure evaluates the question shown; it can only be right or wrong. This evaluation is evidence to update the network values using the $UpdateNodeValues()$ procedure.

Question selection based on probability values Table 2 contains questions and values to decide which question will be the next to be shown. It is simulated by Elvira and Genie software. This table has 17 questions at the IntermediateB level with related concepts, for now, do not interest know how questions are related.

We Suppose that the start level is IntermediateB. In the beginning have not evidence. This means that the student has not answered a question. The software always proceeds to select a question randomly, according to the probability value, this must be in the indeterminate range ($0.3 <= IQ < 0.7$). The simulation just started; therefore, all questions are candidates to be selected. Supossing

Table 2. A priori and a posteriori probability

Questions	A priori probability	A posteori probability (2Q)	A posteriori probability (4Q)
1	0.6125	0.7453	0.7711
2	0.5562	0.7183	0.7318
3	0.5375	1.000	1.000
4	0.5188	0.7437	0.7699
5	0.5188	0.7437	0.7699
6	0.5188	0.7437	0.7699
7	0.5012	1.000	1.000
8	0.5094	0.5668	0.7774
9	0.5094	0.5668	1.000
10	0.5094	0.5668	0.7774
11	0.5094	0.5668	0.7774
12	0.6125	0.6125	0.6148
13	0.6125	0.6125	1.000
14	0.6125	0.6125	0.748
15	0.6125	0.6125	0.6148
16	0.4756	0.4756	0.4466
17	0.550	0.6922	0.7092

that question 7 was selected randomly; this is shown to the student and is correctly answered. The probability values must be updated according to the evidence; these values are not displayed in Table 2 to simplify. Afterward, another question was selected in the indeterminate range, this was the question 3, and was answered correctly.

Column 3 in Table 2 presents the probability values for each question after two questions were correctly answered. Some questions increased their values while other did not, since they are not related to the question concepts 3 and 7. According to column 3, questions 1, 2, 4, 5, and 6, are now in the known questions range, therefore, are discarded from selection. The other questions (8, 9, 10, 11, 12, 13, 14, 15, 16, and 17) are candidates to be selected and shown to the student.

The fourth column in Table 2 presents questions probability values, after questions 9 and 13 are answered correctly, as well as the previously answered questions. Some questions already discarded increase their probability values, while other that had not been considered to the known questions range, now are considered (questions 8, 9, 10, 11, 14, and 17). Hence, those questions are not considered candidates for being selected and shown to the student.

After 4 answered questions only questions 12, 15, and 16 are available. 10 questions are discarded because the module infers that the student knows the answers. We only consider correctly answered questions for this example. If we had incorrectly answered questions, we probably would have to discard those questions that are in the unknown questions range, but the operation would be the same, only taking those questions in the indeterminate range.

We can see the inference level of the BN, inferring 10 known questions with only 4 pieces for evidence. In this scenario, we do not consider the rules for level moves, to exemplify best the BN inference. In summary, the module algorithm works by taking into account the level where the student is, the previously answered questions chosen based on probability, and a random factor.

4 Experimental evaluation

This investigation considered 4 tests to prove module work, each defined below:

- Concepts inference: determines how many and which questions the module software deduce as known or unknown, according student evidence.
- Testing question inference: proves software accuracy in the inference. This with the next method: once that students completed the software exam, those questions that the module software infers as right or wrong are selected, in order to make a written exam. The same student that answered the software exam also answered the written exam; the students answer was compared to determine the inference effectiveness. This test expects that inferred questions as correct or incorrect will have the same result in the written exam.
- Time comparison: determines which kind of exam is faster, if the software exam or written exam. The software exam of a student was taken as a base; this exam was answered in writing by other students with the same academic level.
- Determining the student knowledge: to reach this, the software module does values analysis of the concepts, determining those concepts that are in the known and unknown range. Here we can define a new scale to achieve high or low accuracy; this depends on teacher judgement.

This study selected students majoring in software engineering to test the project. They are third semester students from the Autonomous University of Sinaloa. They are evaluated with a minimum grade of 0 and maximum grade of 10.

These students were selected because of the appropriateness of the exam for them. The groups are in a major focused mainly on the software system development; therefore, they already should have a sufficient basis to answer the exam with acceptable skill. Sampling by conglomerate combined with simple random sampling was utilized to select the section of the population.

First, sampling by conglomerate was selected because we needed to group students according to their grades to compare results of students with similar averages (9 or 10) to maintain a balanced match. Second, simple random sampling was used due to a very small population (62 students).

Student classrooms and schedules were known and selected students were available to be examined. This method consists of making a prospect list and selecting them randomly.

Students were separated into groups according to their grades, as Advanced Students (AS) with averages greater or equal to 8.5. ($AS >= 8.5$), Regular Students (RS) with average greater or equal to 7.0 and less to 8.5. ($7.0 <= RS < 8.5$), and Irregular Students (IS) with average less to 7.0. ($IS < 7$).

Finally, the advanced group have 21 students, the regular group have 26 students and the irregular group have 15 students. Students were randomly selected with a simple program that sorted student names alphabetically according the classification. Each student had an index of 0 to 20 for the advanced group, of 0 to 25 for the regular group, and of 0 to 14 for the irregular group. Seven random students were selected per group. A total of 21 students were selected of 62 possible, equivalents to 33.8% of the population.

5 Results and discussion

We highlight the big amount of inferred questions with few answered questions. Taking into account the 21 evaluated students, we obtained an average of 2.3 inferred questions for each student-answered question. We got a standard deviation of 0.90 and a median of 2.38. This allows us to evaluate more knowledge with fewer questions.

We tested concept inference with a written exam with questions inferred by the software. This exam was applied to the same student. According to the result, the software module had a success in 75.6% of the cases. In other words, when the module inferred a question as known or unknown, this hit almost in 3 on 4 occasions, compared with the written exam. This section gave us a standard deviation of 12.5 and a median of 78.4.

There is one mistake for each four questions explained by the following reasons: a) Students may have doubts the first time the exam was answered, implying that a second time the students were already academically prepared. Nevertheless, they were not told they would, do a second exam. b) Students may have the knowledge to answer questions, but conducted poor analysis. c) Finger errors, i.e., knowing the answer, but select the incorrect option due a distraction, may have occurred. d) A wrong approach to the question may have caused confusion, although questions were reviewed by some teachers before being applied to the student. e) Students may not have taken the exam, either written or by software, leading to inaccuracies.

Students times answering the exam in software are lower than the students that do the written exam. They answered the same questions in less time and make best use of the resource to achieve the same results. This means they are efficient and use 36.8% of the time generally utilized in a traditional exam.

Figure 1 shows the results of a selected student (solid line) and all the students (dotted line). This line graph evaluated concepts on the X axis. On the Y axis are the values of each concept (between 0 and 1). Analysis shows the student has a high probability of knowing most concepts. This section could be interpreted by the teacher. Teachers can define their own ranges to analyze results. For this case, we considered the same previously defined rules-known, unknown, and indeterminate questions.

Concepts where the student performed poorly are most important; those concepts require special attention. We can see that this student was confused by questions about Program, Algorithm, and Variables Types. We can measure the knowledge possessed by the group calculating the average of concepts, as we see on the figure 1 indicated with dotted lines. Instead of serving only a student, we

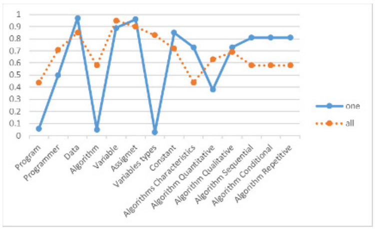

Fig. 1. State of the concepts

could focus on content for a students group with similar problems and help more students. Through this procedure we can detect the concepts less acquired and then reinforce these weak points.

6 Conclusions and future work

Using the software module based on BN we found that it is more efficient and effective than the computer exam and a traditional pen and paper exam. Also, we found that students can answer exams 2.7 times faster than traditional exams or computer exam but without intelligence; in addition, the software system can infer 2.3 known or unknown concepts per student answer. Also, this study proved that those concepts determined as know or unknown have 75.6% of probabilities of being right.

Our testing showed software module development based on BN reflected student deficiencies and skills. For this reason, we can say that Bayesians Networks are an appropriate model for assessing student cognitive levels.

Our proposal involves two levels of adaptation:

1. Exam integration. The selection of questions is based on the student responses; this is the adaption. The software module takes questions that are not related (or are related to low value) to the question with concepts already evaluated; this means that the system only takes the questions that are not classified as known or unknown according to probability values.

   Also, software takes into account question complexity and shows questions to students according to knowledge level. Difficulty of questions is increased or decreased based on the correctness of responses. Thus, the software provides a real-time adaptation in questionnaire construction.
2. By knowing current student knowledge levels, learning material can be adapted. Considering this Kind of adaptation, we have a basis to implement this approach inside an automated tutoring system.

For future work, we are considering including the evaluation module in an Intelligence Tutoring System. Inferring current student knowledge, the software can reinforce topics with low levels of understanding. Moreover, in order to gain accuracy in evaluation, other variables can be considered even if it is the first interaction with the student. Examples of these variables are grades in previous and current courses, the general average, student behavior, related concepts, and so on.

References

1. Carbonell, J.R.: AI in CAI: an artificial intelligence approach to computer assisted instruction. IEEE transaction on Man. Machine System 11, 190–202 (1970)
2. Cataldi, Z., Lage, F.J.: Modelado del Estudiante en Sistemas Tutores Inteligentes. Revista Iberoamericana de Tecnologia en Educación y Educación enTecnología 5, 29–38 (2010)
3. Cheah, W.P., Kim, K.Y., Yang, H.J., Kim, S.H., Kim, J.S.: Fuzzy Cognitive Map and Bayesian Belief Network for Causal Knowledge Engineering: A Comparative Study. The KIPS Transactions:PartB 15B(2), 147–158 (2008)
4. Conejo, R., Millán, E., Pérez, J., Trella, M.: Modelado del alumno : un enfoque bayesiano. Revista Iberoamericana de Inteligencia Artificial 12, 50–58 (2001)
5. De Bruyn, E., Mostert, E., Van Schoor, a.: Computer-based testing - The ideal tool to assess on the different levels of Bloom's taxonomy. 2011 14th International Conference on Interactive Collaborative Learning, ICL 2011 - 11th International Conference Virtual University, VU'11 (September), 444–449 (2011)
6. Goguadze, G., Sosnovsky, S., Isotani, S., McLaren, B.M.: Evaluating a Bayesian Student Model of Decimal Misconceptions. In: Proceedings of the 4th International Conference on Educational Data Mining. p. 5 (2011)
7. Huertas, C., Juárez-Ramírez, R.: Developing an Intelligent Tutoring System for Vehicle Dynamics. Procedia - Social and Behavioral Sciences 106, 838–847 (2013)
8. Kammerdiner, A.: Bayesian networks Bayesian Networks. In: Floudas, C.A., Pardalos, P.M. (eds.) Encyclopedia of Optimization SE - 32, pp. 187–196. Springer US (2009)
9. Liu, Z., Wang, H.: A Modeling Method Based on Bayesian Networks in Intelligent Tutoring System. Structure pp. 967–972 (2007)
10. Luckey, M., Engels, G.: High-Quality Specification of Self-Adaptive Software Systems pp. 143–152 (2013)
11. Millán, E.: Sistema bayesiano para modelado del alumno. Ph.D. thesis (2000)
12. Millán, E., Descalço, L., Castillo, G., Oliveira, P., Diogo, S.: Using Bayesian networks to improve knowledge assessment. Computers & Education 60(1), 436–447 (2013)
13. Misirli, A.T., Bener, A.B.: Bayesian networks for evidence-based decision-making in software engineering. IEEE Transactions on Software Engineering 40(6), 533–554 (2014)
14. Radenkovic, B.: Web portal for adaptive e-learning. Telecommunication in Modern Satellite Cable and Broadcasting Services (TELSIKS), 2011 10th International Conference on pp. 365 – 368 (2011)
15. Ramírez-Noriega, A., Juárez-Ramírez, R., Huertas, C., Martínez-Ramírez, Y.: A Methodology for building Bayesian Networks for Knowledge Representation in Intelligent Tutoring Systems. In: Congreso Internacional de Investigación e Innovación en Ingeniería de Software 2015. pp. 124–133. San Luís Potosí (2015)
16. Razek, M.a., Bardesi, H.J.a.: Adaptive course for mobile learning. Proceedings - 5th International Conference on Computational Intelligence, Communication Systems, and Networks, CICSyN 2013 pp. 328–333 (2013)
17. Rivas Navarro, M.: Procesos cognitivos y aprendizaje significativo. BOCM, Madrid (2008)
18. Rodrigues, F.H., Bez, M.R., Flores, C.D.: Generating Bayesian networks from medical ontologies. 2013 8th Computing Colombian Conference, 8CCC 2013 (2013)
19. Russell, S., Norving, P.: Artificial Intelligence: A Modern Approach. 3rd edit. edn. (2009)
20. Santhi, R., Priya, B., Nandhini, J.: Review of intelligent tutoring systems using bayesian approach. arXiv preprint arXiv:1302.7081 (2013)
21. Taborda, H.: Modelos bayesianos de inferencia psicológica: Cómo predecir acciones en situaciones de incertidumbre? Universitas Psychologica 9(2), 495–507 (2010)
22. Torabi, R., Moradi, P., Khantaimoori, A.R.: Predict Student Scores Using Bayesian Networks. Procedia - Social and Behavioral Sciences 46, 4476–4480 (2012)