International Journal of Information Management 37 (2017) 1488–1498

Contents lists available at ScienceDirect

International Journal of Information Management

journal homepage: www.elsevier.com/locate/ijinfomgt

Research Note

Evaluation module based on Bayesian networks to Intelligent Tutoring Systems

Alan Ramírez-Noriega [a,*], Reyes Juárez-Ramírez [a], Yobani Martínez-Ramírez [b]

[a] Autonomous University of Baja California, School of Chemical and Engineering Science, Calzada Universitaria 14418, Parque Industrial Internacional, Tijuana, Baja California, C.P. 22390, Mexico
[b] Autonomous University of Sinaloa, Fuente de Poseidon y Angel Flores s/n, Col. Jiquilpan, Modulo B2, Los Mochis, Sinaloa, C.P. 81223, Mexico

ARTICLE INFO

Article history:
Received 5 February 2016
Received in revised form 1 April 2016
Accepted 23 April 2016
Available online 24 May 2016

Keywords:
Knowledge representation
Bayesian network
Evaluation
Intelligent Tutoring System

ABSTRACT

Assessing knowledge acquisition by the student is the primary task of an Intelligent Tutoring System (ITS). Assessment is needed to adapt learning materials and activities to student's capacities. In this paper, a proposal to infer the level of knowledge possessed by the student is presented. A general structure of an ITS is shown, an evaluation module based on Bayesian network is proposed. The module mainly based on a test was implemented 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. A network of concepts was used to get the inferences, which contains the relationships between concepts. Evaluation module could infer many questions and concepts through the relations and the probabilistic inference of the Bayesian network. It information easily can be used to reinforce weak topics in order to cover the student's needs. Given the positive evidence is considered that testing the rest of variable examined in the Bayesian network can provide better accurate in the diagnostic of student' knowledge possession.

1. Introduction

Learning can be defined as internal processes of change, as the result of learner's personal experience. Also, it can be defined as the acquisition or adding of something new, which involves any variation or modification previously acquired (Rivas Navarro, 2008).

Teachers guide students during the learning and must perceive the students needs 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 (Carbonell, 1970; Cataldi & Lage, 2010a; Huertas & Juárez-Ramírez, 2013; Santhi, Priya, & Nandhini, 2013) to satisfy that needs. Furthermore, the software system should be adapted to the student's needs.

Adaptability to students needs is a challenge for software engineering (Luckey & Engels, 2013). Adaptability is defined as the software adaptation to individual user characteristics according to user aims (Radenkovic, 2011). Different types of software adaptations are defined (Radenkovic, 2011; Razek & Bardesi, 2013), 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 adjustment of the learning environment can be achieved with artificial intelligence strategies (Razek & Bardesi, 2013), 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 (Huertas & Juárez-Ramírez, 2013; Santhi et al., 2013) in the same way as a teacher does to his students (Cataldi & Lage, 2010a). Carbonell (1970) proposed a generalized architecture for ITS, which considers three core modules (Cataldi & Lage, 2010a; Santhi et al., 2013): 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 (Conejo, Millán, Pérez, & Trella, 2001). ITS must be able to determine accurately and quickly the student cognitive level to decide what is important to teach them. Probability theory has been proposed by some authors for handling the uncertainty in diagnosing student knowledge (Conejo et al., 2001; Huertas & Juárez-Ramírez, 2013; Santhi et al., 2013). The Bayesian Network (BN) theory is proposed, within a framework of probability and artificial intelligence, for modeling the way how an intelligence

* Corresponding author.
E-mail addresses: alan.david.ramirez.noriega@uabc.edu.mx (A. Ramírez-Noriega), reyesjua@uabc.edu.mx (R. Juárez-Ramírez), yobani@uas.edu.mx (Y. Martínez-Ramírez).

http://dx.doi.org/10.1016/j.ijinfomgt.2016.05.007
0268-4012/

system should infer causality (Taborda, 2010). Besides, this theory has representation and behavior similar to people's mind (Rivas Navarro, 2008).

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 (Millán, Descalço, Castillo, Oliveira, & Diogo, 2013a). This paper proposes a general architecture of an ITS to implement a knowledge evaluation module (Ramirez-Noriega, Juarez-Ramirez, Martinez-Ramirez, Jimenez, & Inzunza, 2016). To diagnose the student learning needs efficiently and to reinforce weak topics. The BN theory supports this research. It paradigm is wide used by supporting uncertainty handling, knowledge representation, and diagnostic and pattern recognition (Liu & Wang, 2007; Misirli & Bener, 2014; Santhi et al., 2013).

We selected BN for this study over Fuzzy Cognitives Maps (a similar technique), because they have significant attributes as (Cheah, Kim, Yang, Kim, & Kim, 2008): (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 into six sections. Section 2 defines some related work. Section 3 shows how the evaluation module is integrated into an ITS. Section 4 explains our proposal for knowledge's assessment. Also, it shows how the BN is implemented for knowledge representation and its evaluation; moreover, it presents a question-based evaluation design that we used for student evaluation. Furthermore, this section explains the algorithms employed in the module. Section 5 defines the methodology for experimentation that we employed. Section 6 shows the experiment results and discussions. Lastly, Section 7 presents the conclusions and future work.

2. Related works

In this section presents the related work, emphasizing the use of BN for improving learning and education, taking into account that BN is used to assess the knowledge.

Liu and Wang (2007) proposed a student modeling method built with BN. To evaluate 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 a course of Data Structures, especially on Binary Trees. This work just was a proposal, but they did not implement the model.

Goguadze, Sosnovsky, Isotani, and McLaren (2011) 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, Moradi, and Khantaimoori (2012) worked in predicting the student courses score based on the student's 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.

Millán et al. (2013a) 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 means students need answer four questions as one.

The main model used is similar to the Millán (2000); however, some differences are considered with the previous work. Our model evaluates the fundamentals of algorithms as learning topic. We did a complete analysis of the domain knowledge to represent it. Also, to adapt the questions according to Student's knowledge level, Bloom's taxonomy (De Bruyn, Mostert, & Van Schoor, 2011) was implemented organizing the questions into five levels of complexity. We used own algorithms to switch-on the levels and to select the appropriated question to the student. Finally, It is proposed to integrate the evaluation module to an ITS, considering five factors to inference the knowledge.

3. Evaluation module in ITS

This section explains how the evaluation module (Ramirez-Noriega et al., 2016) can be adapted to the general architecture of the ITS. Besides, It explains a complete Bayesian network to obtain better accurate in the diagnose of the student's knowledge possession. In order to test the network was only used a variable (test questions), but its use is the same.

3.1. ITS characteristics

3.1.1. Knowledge organization in education

Education in institutions is organized by subjects; these subjects are topics related to the chosen career. Besides, subjects consider an order and planning to work with students. The index of contents in a subject represents the topics that students have to learn and its order. The information is organized by chapters or units, topics, and subtopics. Fig. 1 clearly represents a hierarchical structure where are established categories organized into topics and subtopics. This hierarchical order is organized by complexity degrees; first topics are easy to assimilate with small reasoning degree, after topics increase the knowledge level to advance to complex topics.

The previous order is related to Bloom taxonomy (De Bruyn et al., 2011; Rodrigues & dos Santos, 2013), where the first knowledge is simple concepts, and final concepts are a complex process of analysis, synthesis, and evaluation. For instance, in algorithm course the first topics could answer questions such as What is an algorithm? What is a variable?, Which are the kind of algorithms? etc. these questions are simple concepts without reasoning. At the end of the course, the students could develop small programs in an algorithm editor, testing their reasoning and applying all the concepts previously learned.

Graphs can represent this educative organization, so, a computer could easily analyze this information through a knowledge representation approach based on nodes and relations. Fig. 2 represents a tree structure founded in the contents displayed in Fig. 1. The chapter one was only considered to simplify the idea. Tree root is the chapter, the topics arising from the root; finally, subtopics are leaf nodes.

A greater level of granularity can be represented adding concepts of each subtopic as shown in Fig. 3. This structure, in its complete form, should be represented in the student's memory at

1 Algorithms
1.1 Introduction to the algorithms
1.1.1 General concepts
1.1.2 Characteristics
1.1.3 Types of algorithms
1.1.4 Algorithms classification
1.2 Analysis of the problem
1.2.1 Analysis
1.2.2 Examples

Fig. 1. Topics of the algorithm course.

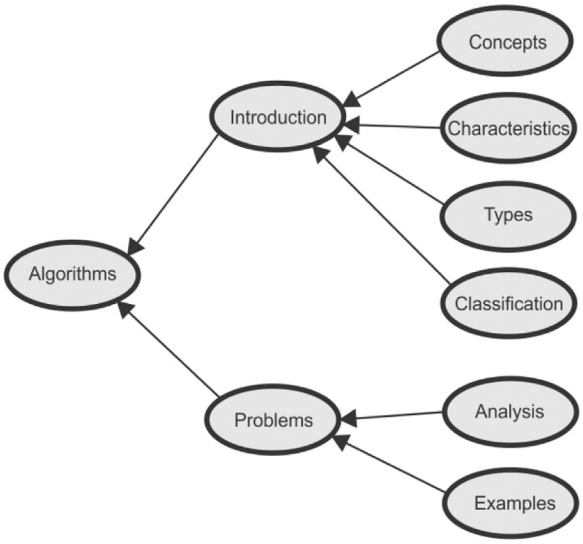

Fig. 2. Graph of the course.

the end of this block. Various theories argue that semantic memory organizes information in a similar structure made up of nodes and relation (Rivas Navarro, 2008). So, it is would be simulated the knowledge in the student's mind.

This representation could serve as means of assessment to determine a student's knowledge, which is used by Intelligent Tutoring Systems (ITS) and some computer-aided systems. Evaluation is probably the most important to help students in the learning process (Brown, 2004). It refers to a comparative measure between what students learned and what they should learn (Millán, Descalço, Castillo, Oliveira, & Diogo, 2013b) employing any process or activity designed to gather information about knowledge, attitudes or abilities of the learner (Kellaghan & Greaney, 2001). However, evaluation mechanisms are only approximations to the reality, and teachers cannot know exactly what the student has learned. This lack of understanding is called uncertainty (Kovalerchuk, 2013). That is, the teacher will always have uncertainty about what the student has learned.

This type of representation is rather used as a technique for representing knowledge (Millán, Loboda, & Pérez-De-La-Cruz, 2010; Peña-Ayala, Sossa-Azuela, & Cervantes-Pérez, 2012; Ramirez & Valdez, 2011), in the following sections is defined as work.

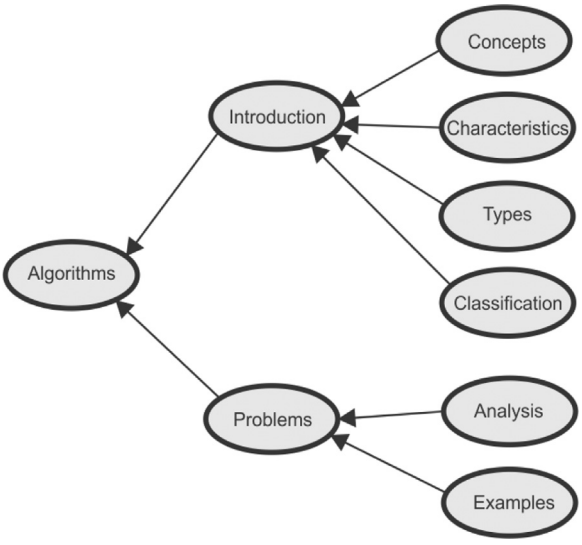

Fig. 3. Extended graph of the course.

3.1.2. Pedagogical strategies

Pedagogical strategies are employed in an ITS, each software decides which elements they use. However, below is defined a generic list of strategies implemented:

- *Tests*: It is an evaluation procedure. Generally, it is a series of questions, problem, or another kind of instrument designed to determine the student knowledge, intelligence, or ability. Sometimes, students cannot repeat the exam and receive a grade.
- *Exercises*: A particular activity performed to develop or maintain skills. It is similar to a test, but the student can repeat the exercises as many times as necessary. Usually, they are in the final of the topic.
- *Activities*: It is an educational task that involves direct experience and participation of the student. It refers a specific action to maintain active to the student, such as chats, surveys, forums, among other. Students usually do not receive a grade for this action.
- *Lessons*: Lessons are lists of basic terms about a topic. It is an activity that students can do to learn something. The information of the lessons contains: text (It can be read), sound (It can be heard), images (It can be seen) and video (It can be seen and heard).

All ITS consider the previously explained elements, and maybe any other. However, these minimum elements were defined to related to the Bayesian network. So, the evaluation module can be considered for any ITS.

3.2. Interaction between modules

The classical architecture for a ITS is composed by (Badaracco & Martínez, 2011; Cataldi & Lage, 2010b; Victorio-Meza, Mejia-Lavalle, & Ortiz, 2014):

- *Domain module*: This defines the knowledge domain. It contains knowledge about the subjects that must be learned. The domain model is obtained by abstracting the way in which humans make use of knowledge. This type of model is most effective from a pedagogical point of view. It is called cognitive model.
- *Student module*: It can define the student's knowledge at each point during the work session. This module makes inference taking into account the domain model. An overlay model is used. It model represents the student's knowledge about the domain; the students behavior is compared with an expert domain.
- *Tutoring module*: It defines the teaching and tutorial strategies. This module considers the control on the representation of knowledge to select and sequence the parts that should be supplied to the student, ability to answer students questions about his/her instructional objectives and contents, strategies to determine when the student needs support and how to select the appropriate help.
- *User interface*: It enables student interaction efficiently. It refers to how to present content.

A evaluation module is added in Fig. 4. It has an interaction with other modules, to provide them the probabilistic inference of the Bayesian network. Evaluation module works with the following submodules:

- *Knowledge domain (Domain module)*: It represents the knowledge, represented as concepts, questions, exercises, problems, etc. Structure with nodes and relation is usually used. Evaluation module obtains a copy of the domain knowledge and works with it, updating the student's knowledge with each shown evidence.
- *Current state of knowledge (Student module)*: This submodule represents the knowledge obtained by the student through the ITS.

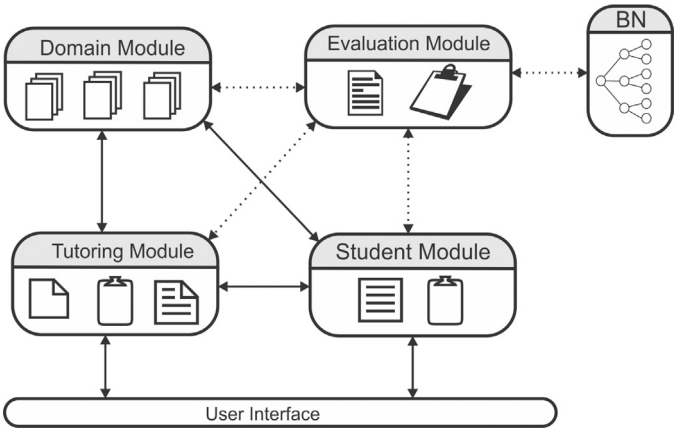

Fig. 4. Relation between modules in ITS.

This submodule is updated progressively according the interaction with the ITS.

- *Student's aims (tutoring module)*: The aims of the topics are checked according to the advance of the student, it is achieved reviewing the current state of knowledge of the student.
- *Content generator (tutoring module)*: Through current state of knowledge is shown to the student the ideal topic, according to their learning needs.
- *Grader (tutoring module)*: This submodule evaluates the student's activities, to define the current state of knowledge.

3.3. Extended Bayesian network

Evaluation module has a Bayesian network to manage the probabilistic inference. The BN has different kind of nodes to get the student knowledge evidence. Fig. 5 displays the structure.

The Bayesian network shows five variables to increase the accuracy to define what the student knows. Each variable is defined as:

- *Tests questions (Tq)*: These questions are based on a test. It is used as the main group of nodes to give evidence to the structure. This work starts testing the network with these nodes.

  This variable was used to testing the Bayesian network. So, in the next sections is explained the behavior of this variable. This variable is represented as a set of elements ($Tq = \{tq_1, tq_2, \ldots, tq_n\}$), each element of Tq is represented as a tuple of four elements in Eq. (1):

$$tq_i = \{q, A, R, CPT\} \tag{1}$$

  where:
  - q: represents a question.
  - A: represents a set of answers to the question.
  - R: represents a set of concepts related to the question.
  - CPT: represents a set of numbers that represent the CPT.
- *Exercises questions (Eq)*: These questions are more informal than test questions, but this variable is based on the same rules than test questions. Exercise question is used to practice and reinforce a topic and their concepts. This variable is represented as a set of elements ($Eq = \{eq_1, eq_2, \ldots, eq_m\}$), each element of Eq is represented as a tuple of four elements in same way like is represented in Eq. (1). The influence of this questions are lower than the test question.
- *Activities (Aq)*: It is refers to activities that the student can do, such as posting on a forum, answer a survey, etc. It is represented as a set $Aq = \{aq_1, aq_2, \ldots, aq_m\}$. Each element of Aq is represented as a tuple of three elements in (2):

$$aq_i = \{a, R, CPT\} \tag{2}$$

  where:
  - a: represents an activity.
  - R: represents a set of topics related to the activity.

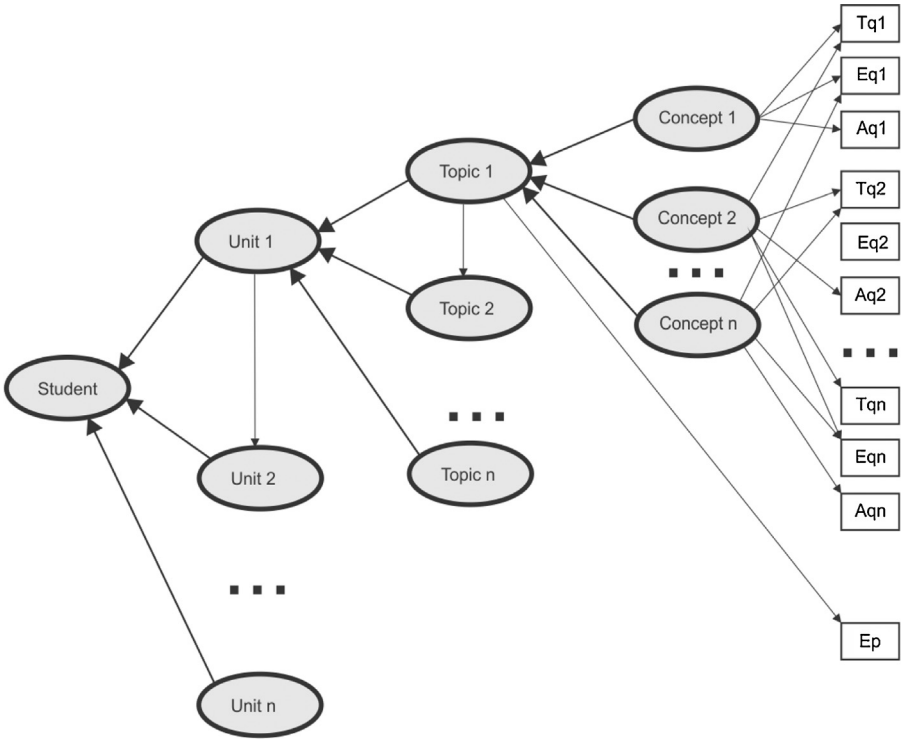

Fig. 5. Schema of Bayesian network.

- *CPT*: represents a set of numbers that represent the CPT.

- *Expected progress (Ep)*: It is referred to the advance in a topic according to the expected progress. This variable considers three states; behind, normal, and forward. Fist state is for a student with low performance, the second state is for regular performance, and the third state considers a student with high performance. It influences the topics mainly and the lesser extent to the concepts and units.
- *Probability for previous topics (Pt)*: It is referred to the probability to know topics and concepts given the records of the student on previous topics. It is represented in the Bayesian network with relations between near topics or units.

According to Fig. 5, the test, exercise, and activity questions are related to the concepts. These variables have major influence over the nodes than other. The variables related with expected progress influence to the topics and units. Finally, probability by previous topics is related to units to units or topics to topics. This last variable has a deeper level of impact to the structure.

The main idea is that the student access to lessons to study a topic, each topic has associated tests, exercise, and activities. Each time the student interact with the tutor, they give evidence of knowledge to the structure, through the relations can be inferred knowledge about specific concepts, topics or units, even when this knowledge is not directly related.

Taking into account the concepts with lower probabilities to be known, the tutor can propose topics related to that weak concepts. Besides, exercises can be shown to the student to reinforce concepts.

The evaluation module based on Bayesian network was testing with one variable. The test questions variable is considered more important than the rest. So, this variable was selected to test the Bayesian network. Next section explains how to work the evaluation module with one variable.

3.3.1. Building CPT

In general, this section explains how building a CPT to this domain. The weight of the relation is the most significant value to build the tables. How the value is obtained is explained in other sections. So, it value is considered as a variable (t).

A child variable (h variable), may have zero or more parents ($h = \{p_1, p_2, \ldots, p_n\}$). Where n refers to the number of parents that could have h. If $n = 0$ then h have not parents ($n = 0 \rightarrow h = \emptyset$). Each parent has two possibles values (states): true (t) and false (f) ($p_n = \{t, f\}$), where t is the influence weight assigned to the child (h variable), f is the complementary value obtained based on the equation $f = 1 - t$. When is mentioned the weight of p_n we refers to the t value to p_n. The values assigned to the parents must be between zero and one ($0 \leq p_n \leq 1$). Besides, the summation of the t value of the parents regarding to its child must be equal to one ($1 = \sum_{i=1}^{n} p_i$).

The total of true values for the CPT of h is calculated with 2^n. For instance, if $n = 1$ then two true values are obtained and two false complementary values.

CPT are built according to the parents number as:

- If $h = \emptyset$, this means, there are not parents. It only establishes the true value in h, due to there are not influence of any parent. So, it obtained $v_\emptyset$, where v represents the CPT. Besides, $CPT_\emptyset = \{v_t\}$. The value of this table is (3);

$$v_t = t \tag{3}$$

- If $h = \{p_1\}$, this means, a parent (p_1) related to the child (h). Combinations cannot be established, only the direct influence (true or false) of the parent v_{p_1}. It is obtained $CPT_{p_1} = \{v_t, v_f\}$. Values to this structure are (4):

$$v_t = t, v_f = 0 \tag{4}$$

- If $h = \{p_1, \ldots, p_n\}$, the combination between parent is established as $v_{p_1,\ldots,p_n}$. It is obtained $TPC_{p_1,\ldots,p_n} = \{v_{t,\ldots,t}, v_{t,\ldots,f}, \ldots, v_{f,\ldots,f}\}$. The values are represented for (5):

$$v_{t,\ldots,t} = t+, \ldots, +t, v_{t,\ldots,f} = t+, \ldots, +0, \ldots, v_{f,\ldots,f} = 0+, \ldots, +0 \tag{5}$$

Although it is represented $f = 0$, and its summation is equal to zero. There are adjust factors to solve that the relation can exist even when all values are zero, a low relation, but finally a relation.

4. Implementing the evaluation module

4.1. Setting the diagnostic of the student using Bayesian networks

BN helps to 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 Millán et al. (2013a), 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 last 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 a 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 (Millán et al., 2013a).
- *The word parameters signify dependence on probability values of the child nodes given their parents.*

4.2. Building the Bayesian network

We use a building process divided into six phases (Ramírez-Noriega, Juárez-Ramírez, Huertas, & Martínez-Ramírez, 2015)

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.

4.2.1. Network construction

Algorithms and Computer Logic course was considered to the experiment. This course is part of the curriculum for the Software Engineering undergraduate program at 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, topics 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). These concepts are represented by the dotted ovals in Fig. 6.

4.3. Creating questions

We generated a total of 73 questions divided into levels, which are represented in Fig. 6, with the small solid circles to the right. The gray cloud represents questions relationships to their corresponding concepts. We assigned values to each question, taking into account the related concepts and the values of each one.

For instance, the question which algorithms must decision-making based on a condition? It is related to the topic algorithms and their types. In this case, the answers are known, 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,

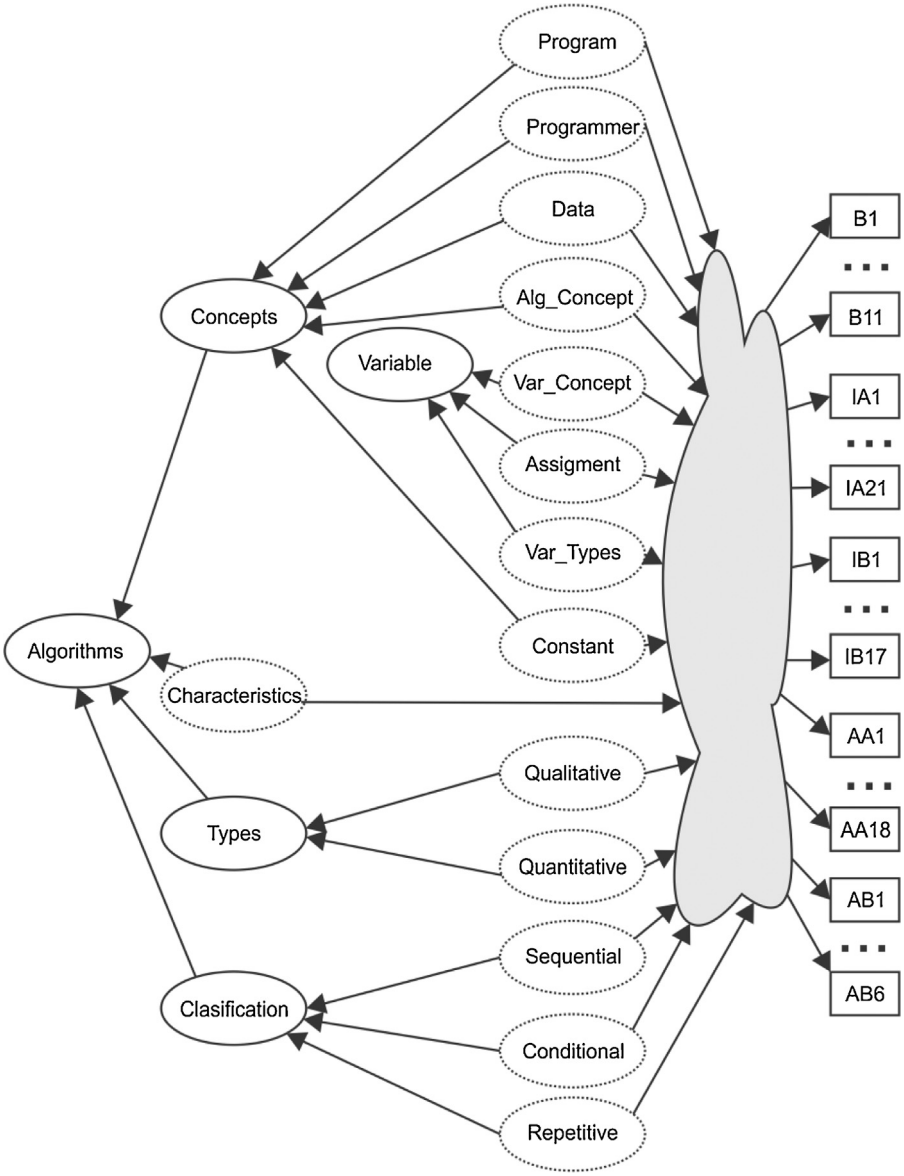

Fig. 6. Bayesian network with question.

Table 1
Levels of complexity

Blooms taxonomy	Name of level	Number of question
Knowledge	Basic (B)	11
Comprehension	IntermediateA (IA)	21
Application	IntermediateB (IB)	17
Analysis	AdvancedA (AA)	18
Synthesis	AdvancedB (AB)	6
	Total	73

depending on who creates the questions and responses. In order to prove the correct formulation of the questions and responses, we apply a survey with professors of the Software Engineering curricular block, who give feedback on the formulation: readiness, clarity, and so on.

Particularly, professors who teach the Algorithms and computer logic course collaborated on the survey. The feedback suggested improving the following aspects:

- Some concepts were not covered in the classroom.
- Some questions and answers were ambiguous.
- Some questions were located in a wrong category.

Attending to this feedback, we improved the set of questions and answers to increase reliability. Questions are organized based on Blooms Taxonomy (De Bruyn et al., 2011). We only work with the first five levels. Table 1 illustrates the organization.

According to Table 1, basic level questions (knowledge) recognize and retrieve relevant information from long-term memory to use in the short-term memory. The intermediateA level questions (comprehension) are questions that require building significance from educational material (De Bruyn et al., 2011; Rodrigues, Bez, & Flores, 2013).

The intermediateB level (application) contains questions where a learning process is applied. The advancedA level (analysis) divides the knowledge in parts and reasoning is required. Finally the advancedB level (synthesis), this level joins elements to form a whole. Students reasoning is deepened; questions are slightly more complicated than the advancedA level. The module aims for students to achieve their maximum potential to use at this level (De Bruyn et al., 2011; Rodrigues et al., 2013).

4.4. 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.

4.4.1. 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. 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) little knowledge, (2) great 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 in 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 reasons mentioned above.

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 to 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 student's 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

```
1:  QuestionsRecord = Structure for control the questions by level
2:  while not FinishTest() do
3:    ReachLevel();
4:    ShowQuestion();
5:    EvaluateQuestion();
6:    AddtoQuestionsRecord();
7:    if question = true then
8:      if three questions are corrects in a row and level in {B, IA, IB, AA} then
9:        GoNextLevel();
10:     end if
11:     if question = true and level = AB then
12:       FinishTest();
13:     end if
14:   end if
15:   if question = false then
16:     if two question are corrects in a row then
17:       if level in {IA, IB, AA} then
18:         GoDownLevel();
19:       end if
20:       if level = B then
21:         FinishTest();
22:       end if
23:     end if
24:     if level = AA then
25:       GoDownLevel();
26:     end if
27: end if
28:   if five questions are showed in the level then
29:     DiscontinuousQuestions();
30:   end if
31: end while
```

Thus, software controls the student's movement through levels, deciding if the student goes down a level, proceeds to the next level, or ends the exam.

4.4.2. Selecting questions

The ranges of the node must be classified as known, unknown, or indeterminate to select a question. Probability values are between 0 and 1, by a probability axiom (Russell & Norving, 2009).

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

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

Table 2
A priori and a posteriori probability.

Questions	A priori probability	A posteriori 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

question shown; it can only be right or wrong. This evaluation is evidence to update the network values using the *UpdateNodeValues*( ) procedure.

4.4.3. 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 the Elvira software (Cano, 2001); in addition, the simulated information is supported by other BN simulation software called GENIE (Decision Systems Laboratory, 2015).

Table 2 has 17 questions at the IntermediateB level with related concepts, for now, do not interest know how questions are related. A priori probability is found in column 2, but does not yet have evidence. Column 3 displays the a posteriori probability after two questions are answered. Finally, the probability values are shown in column 4 after four questions answered.

This example supposes that the start level is IntermediateB, and in the beginning, does not have 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. In other words, the question node value must be greater or equal to 0.3 and less to 0.7. The simulation has just started; therefore, all questions are candidates to be selected. Question 7 was selected randomly; this was shown to the student and was 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 were increased their values while others 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 the 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 responded questions. Some questions already discarded increase their probability values, while others 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. Ten questions are discarded because the module infers that the student knows the answers. We 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 ten known questions with only four pieces of evidence. In this scenario, we do not consider the rules for level moves, to exemplify best the BN inference. In summary, the evaluate 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.

5. Experiment to test the evaluation module

This investigation considered four tests to prove the evaluation module, each defined below:

- *Concepts inference*: It determines how many and which questions the system deduces as known or unknown, according to student evidence.
- *Testing question inference*: This proves software accuracy in the inference. This with the next method: once those students completed the software exam, those questions that the module software infers as right or wrong are selected, in order to make a written test. The same student that answered the software exam also answered the written exam; the student's 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*: It determines which kind of exam is faster, whether the computer-based exam or written exam. The computer-based 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 evaluation 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 judgment.

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 the 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 in 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 chosen 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 tiny 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. ($AS \geq 8.5$), Regular Students (RS) with average greater or equal to 7.0 and less to 8.5. ($7.0 \leq RS < 8.5$), and Irregular Students (IS) with average less to 7.0. ($IS < 7$).

Finally, the advanced group has 21 students; the regular group has 26 students, and the irregular group has 15 students. Students were randomly selected with a simple program that sorted student names alphabetically according to 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.

6. 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, the minimum value was 0.6, and the maximum value was 4.6. Fig. 7 summarizes the data. 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 evaluation module had success in 75.6% of the cases. In other words, when the module inferred a question as known or unknown, this hit almost in three on four occasions, compared with the written exam. This section gave us a standard deviation of 12.5 and a median of 78.4. The maximum and minimum were 43% and 91% respectively (Fig. 8).

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 that 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.

Time spent by students in the computer-based exam is lower than the students that do the written test. They answered the same questions in less time and made better use of the resource to achieve the same results. This means, they are efficient and use 36.8% of the time utilized in a traditional exam.

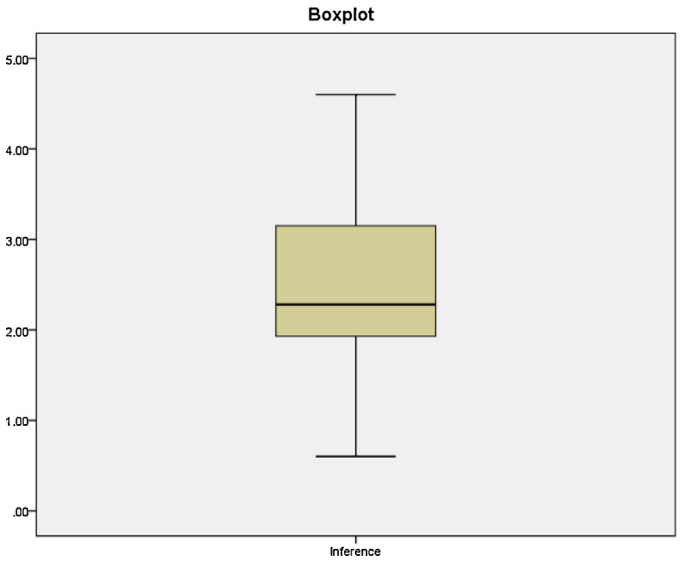

Fig. 7. Boxplot of inferred questions.

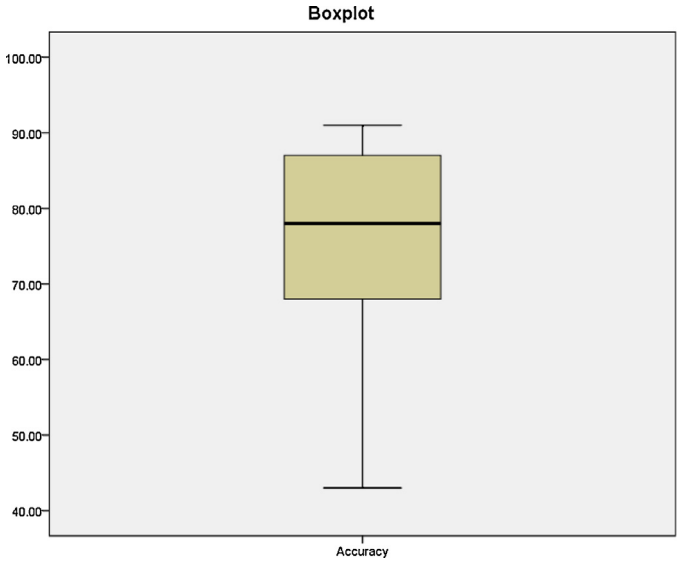

Fig. 8. Boxplot of inference accuracy.

Fig. 9 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). The 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 to pay special attention on; 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 in Fig. 9 indicated with dotted lines. Instead of serving only a student, we could focus on content for a student's group with similar problems and help more students. Through this procedure, we can detect the concepts less acquired and then reinforce these weak points.

Inference of knowledge was obtained with the evaluation module. Fig. 10 shows the nodes and the relations used by the evaluation module. These are represented by gray rectangles. According to the results, it is obtained a 75.6% of accuracy of the knowledge inferred, this is taking into account just the tests questions variable. Considering four factors more of the interaction student with the ITS will increase the accuracy to infer knowledge.

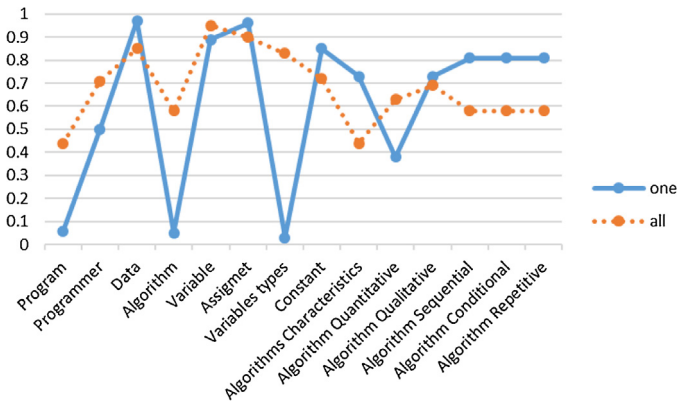

Fig. 9. State of the concepts.

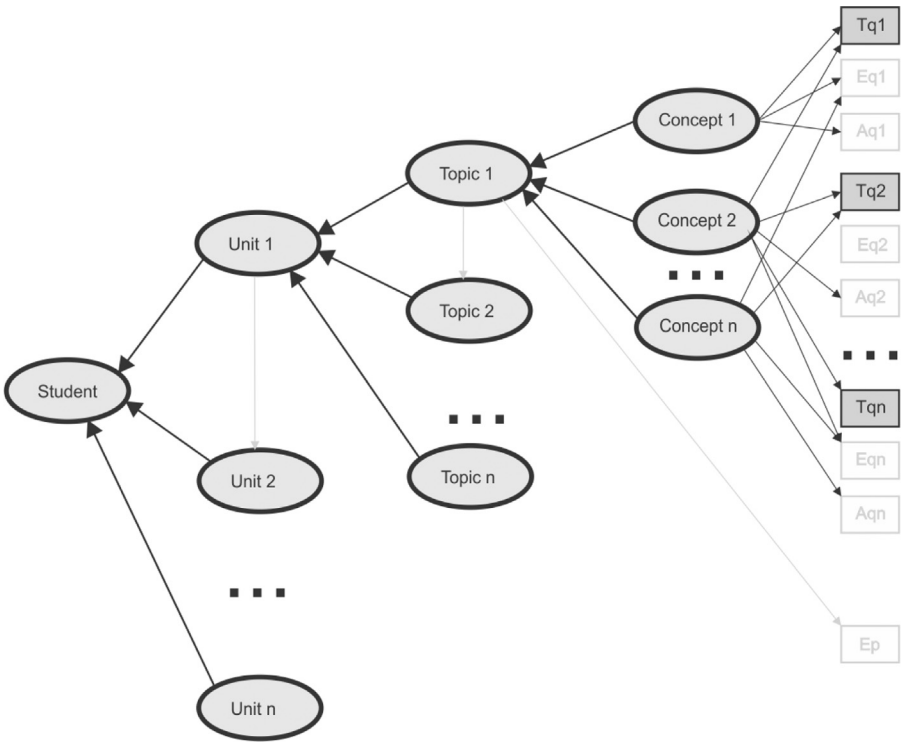

Fig. 10. Schema of Bayesian network.

7. Conclusions and future work

This article explains an evaluation module to be used in an ITS. It is explained how the module was done, how it works, and how is integrated into an ITS. A Bayesian network is used as an inference engine to decision-making to the student learning.

Evaluation module was tested with a group of undergraduate students. Results show that it is more efficient and effective than the computer exam and a traditional paper exam. Also, it is found that students can answer exams 2.7 times faster than traditional exams. Besides, the software system can infer 2.3 known or unknown concepts per student answer. Furthermore, this study proved that those concepts determined as known 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 Bayesian Networks are an appropriate model for assessing student cognitive levels. Besides, the architecture to implement the evaluation module is defined.

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 values) 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, the 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 will be used the proposed five-factor.

References

Badaracco, M., & Martínez, L. (2011). An intelligent tutoring system architecture for competency-based learning. In *Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics) 6882 LNAI (PART 2)* (pp. 124–133). http://dx.doi.org/10.1007/978-3-642-23863-5_13

Brown, S. (2004). Assessment for learning. *Learning and Teaching in Higher Education*, (1), 81–89. http://dx.doi.org/10.1187/cbe.11-03-0025

Cano, A. (2001). *Elvira system*. http://leo.ugr.es/elvira/

Carbonell, J. R. (1970). AI in CAI: An artificial intelligence approach to computer assisted instruction. *IEEE Transaction on Man Machine System*, *11*, 190–202.

Cataldi, Z., & Lage, F. J. (2010a]). Modelado del Estudiante en Sistemas Tutores Inteligentes. *Revista Iberoamericana de Tecnologia en Educación y Educación enTecnología*, *5*, 29–38.

Cataldi, Z., & Lage, F. J. (2010b]). Dimensiones para Evaluacion de Sistemas Tutores Inteligentes. In *XII Workshop de Investigadores en Ciencias de la Computación* (pp. 715–719).

Cheah, W.-P., Kim, K.-Y., Yang, H.-J., Kim, S.-H., & Kim, J.-S. (2008). Fuzzy cognitive map and Bayesian belief network for causal knowledge engineering: A comparative study. *KIPS Transactions: Part B*, *15B*(2), 147–158. http://dx.doi.org/10.3745/KIPSTB.2008.15-B.2.147

Conejo, R., Millán, E., Pérez, J., & Trella, M. (2001). Modelado del alumno: un enfoque bayesiano. *Revista Iberoamericana de Inteligencia Artificial*, *12*, 50–58. http://dx.doi.org/10.4114/ia.v5i12.707

De Bruyn, E., Mostert, E., & Van Schoor, a. (2011). Computer-based testing – The ideal tool to assess on the different levels of Bloom's taxonomy. In *14th international conference on interactive collaborative learning, ICL 2011-11th International Conference Virtual University, VU'11 (September) (2011)* (pp. 444–449). http://dx.doi.org/10.1109/ICL.2011.6059623

Decision Systems Laboratory. (2015). *GeNIe & SMILE*. https://dslpitt.org/genie

Goguadze, G., Sosnovsky, S., Isotani, S., & McLaren, B. M. (2011). Evaluating a Bayesian student model of decimal misconceptions. In *Proceedings of the 4th international conference on educational data mining* (p. 5).

Huertas, C., & Juárez-Ramírez, R. (2013). Developing an intelligent tutoring system for vehicle dynamics. *Procedia – Social and Behavioral Sciences*, *106*, 838–847. http://dx.doi.org/10.1016/j.sbspro.2013.12.096

Kellaghan, T., & Greaney, V. (2001). *Using assessment to improve the quality of education*. Paris, France: United Nations Educational, Scientific and Cultural Organization.

Kovalerchuk, B. (2013). *Quest for rigorous intelligent tutoring systems under uncertainty: Computing with words and images*. pp. 685–690.

Liu, Z., & Wang, H. (2007). A modeling method based on Bayesian networks in Intelligent Tutoring System. *Structure*, 967–972.

Luckey, M., & Engels, G. (2013). *High-quality specification of self-adaptive software systems*. pp. 143–152.

Millán, E., Loboda, T., & Pérez-De-La-Cruz, J. L. (2010). Bayesian networks for student model engineering. *Computers and Education*, *55*(4), 1663–1683. http://dx.doi.org/10.1016/j.compedu.2010.07.010

Millán, E., Descalço, L., Castillo, G., Oliveira, P., & Diogo, S. (2013a]). Using Bayesian networks to improve knowledge assessment. *Computers & Education*, *60*(1), 436–447. http://dx.doi.org/10.1016/j.compedu.2012.06.012

Millán, E., Descalço, L., Castillo, G., Oliveira, P., & Diogo, S. (2013b]). Using Bayesian networks to improve knowledge assessment. *Computers & Education*, *60*(1), 436–447. http://dx.doi.org/10.1016/j.compedu.2012.06.012

Millán, E. (2000). *Sistema bayesiano para modelado del alumno*. Ph.D. thesis.

Misirli, A. T., & Bener, A. B. (2014). Bayesian networks for evidence-based decision-making in software engineering. *IEEE Transactions on Software Engineering*, *40*(6), 533–554. http://dx.doi.org/10.1109/TSE.2014.2321179

Peña-Ayala, A., Sossa-Azuela, H., & Cervantes-Pérez, F. (2012). Predictive student model supported by fuzzy-causal knowledge and inference. *Expert Systems with Applications*, *39*(5), 4690–4709. http://dx.doi.org/10.1016/j.eswa.2011.09.086

Radenkovic, B. (2011). Web portal for adaptive e-learning. In *10th international conference on telecommunication in modern satellite cable and broadcasting services (TELSIKS) 2011* (pp. 365–368). http://dx.doi.org/10.1109/TELSKS.2011.6112072

Ramírez-Noriega, A., Juárez-Ramírez, R., Huertas, C., & Martínez-Ramírez, Y. (2015). 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).

Ramirez, C., & Valdez, B. (2011). Memory map of a knowledge representation model used for intelligent personalization of learning activities sequences. In *Proceeding 10h IEEE I. C. on cognitive informatics & cognitive computing* (pp. 423–431).

Ramirez-Noriega, A., Juarez-Ramirez, R., Martinez-Ramirez, Y., Jimenez, S., & Inzunza, S. (2016). Using Bayesian networks for knowledge representation and evaluation in intelligent tutoring systems. In *WorldCist'16-4th world conference on information systems and technologies*.

Razek, M. A., & Bardesi, H. J. A. (2013). Adaptive course for mobile learning. *Proceedings – 5th International Conference on Computational Intelligence, Communication Systems, and Networks, CICSyN 2013*, 328–333. http://dx.doi.org/10.1109/CICSYN.2013.45

Rivas Navarro, M. (2008). *Procesos cognitivos y aprendizaje significativo*. Madrid: BOCM. Servicio de Documentación y Publicaciones.

Rodrigues, A. N., & dos Santos, S. C. (2013). A systems approach to managing learning based on the revised blooms taxonomy. In *43rd Frontiers in education conference*.

Rodrigues, F. H., Bez, M. R., & Flores, C. D. (2013). Generating Bayesian networks from medical ontologies. In *2013 8th computing Colombian conference, 8CCC* http://dx.doi.org/10.1109/ColombianCC.2013.6637538

Russell, S., & Norving, P. (2009). *Artificial intelligence: A modern approach* (3rd ed.).

Santhi, R., Priya, B., & Nandhini, J. (2013). *Review of intelligent tutoring systems using Bayesian approach.* , arXiv preprint arXiv:1302.7081.

Taborda, H. (2010). Modelos bayesianos de inferencia psicológica: Cómo predecir acciones en situaciones de incertidumbre? *Universitas Psychologica*, *9*(2), 495–507.

Torabi, R., Moradi, P., & Khantaimoori, A. R. (2012). Predict student scores using Bayesian networks. *Procedia – Social and Behavioral Sciences*, *46*, 4476–4480. http://dx.doi.org/10.1016/j.sbspro.2012.06.280

Victorio-Meza, H., Mejia-Lavalle, M., & Ortiz, G. R. (2014). Advances on knowledge representation of intelligent tutoring systems. In *2014 international conference on mechatronics, electronics and automotive engineering* (pp. 212–216). http://dx.doi.org/10.1109/ICMEAE.2014.17 http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7120873