Journal of Universal Computer Science, vol. 16, no. 19 (2010), 2801-2820
submitted: 1/3/10, accepted: 29/9/10, appeared: 1/10/10

Authoring of Probabilistic Sequencing in Adaptive Hypermedia with Bayesian Networks

Sergio Gutierrez-Santos
(London Knowledge Lab, Birkbeck College, UK
sergut@lkl.ac.uk)

Jaime Mayor-Berzal, Carmen Fernandez-Panadero
Carlos Delgado Kloos
(Carlos III University of Madrid, Spain
{jmayor, mcfp, cdk}@it.uc3m.es)

Abstract: One of the difficulties that self-directed learners face on their learning process is choosing the right learning resources. One of the goals of adaptive educational systems is helping students in finding the best set of learning resources for them. Adaptive systems try to infer the students' characteristics and store them in a user model whose information is used to drive the adaptation. However, the information that can be acquired is always limited and partial. In this paper, the use of Bayesian networks is proposed as a possible solution to adapt the sequence of activities to students. There are two research questions that are answered in this paper: whether Bayesian networks can be used to adaptively sequence learning material, and whether such an approach permits the reuse of learning units created for other systems. A positive answer to both question is complemented with a case study that illustrates the details of the process.
Key Words: Bayesian networks, adaptive educational hypermedia, sequencing.
Category: L.2.0, L.2.1, L.3.5

1 Introduction

Advancement in communication technologies is shifting the responsibility of learning management from teachers and lecturers to students. The widespread availability of the web, the increasing capabilities of mobile terminals, and the growing number of available learning resources allow students to learn "anytime, anywhere", and to lead the path of their own learning. The ever increasing spread of intelligent communication technologies will continue empowering students to make relevant decisions with regard to what they want to learn. On the other hand, this also means that students share the burden of choosing the best resources to use at any given time.

This is only one of the difficulties that self-directed learners face on their learning process, and it is arguably one of the most important ones. Students need to choose what resources will help them more to achieve their learning objectives but, on the other hand, they do not really know what resources will be better because they have not attained the necessary knowledge to make such judgements. The solution to this "choice" problem in a traditional setting comes in the form of a learning designer or teacher that designs a common course, which is followed by all students. This one-size-fits-all approach is quite limited, as different students will find their learning needs covered with varying degrees of success.

One of the goals of adaptive educational systems is helping students in finding the best set of learning resources for them. For doing so, they try to guide students through the set of available learning activities, adapting different aspects of the system to the particular characteristics of the student, e.g. hiding irrelevant material. An adaptive system tries to infer the students' characteristics and store them in a user model [Woolf, 2008], whose information is used to drive the adaptation. The problem is, however, that the information used in the inference is always limited, and the inference of the characteristics of the user can only be partial.

In this paper the focus is on the adaptation of one aspect of learning: sequencing of learning material. Ideally, a successfully adapted sequence of learning activities results in optimal learning for the student. In a real system, however, knowledge about the student is incomplete and the adaptation can only be approximate, as a recommendation. Due to these difficulties, the use of Bayesian networks to deal with the lack of valid information about the user has been considered. Bayesian networks have been successfully used in other applications of adaptive systems, like user modelling.

Therefore, the first research question this paper tries to answer is whether Bayesian networks can be used to sequence learning material for an adaptive educational hypermedia system, and what are the implications from the authoring point of view. This paper shows how learning resources can be organised and described in a form that allows a Bayesian system to infer which activities are most appropriate for the student at every moment in order to provide a recommendation.

A second research question, which is extremely relevant from the point of view of authoring, is whether the proposed approach permits the reuse of learning units created for other systems. This second question has been answered by developing a new Intelligent Tutoring System (ITS) using web pages and exercises from an already existing ITS which used a different sequencing strategy.

The rest of the paper is structured as follows. The next section frames the paper in the context of both other sequencing initiatives and other applications of Bayesian networks in adaptive systems. Section 3 describes the general methodology developed in this work. This is complemented by details about the technical implementation in Section 4; and a case study in Section 5, in which the methodology is applied to design a new course reusing learning units from another one. Section 6 discusses the data used in the case study, hinting lines for future research. Section 7 closes the paper drawing the main conclusions.

2 Background

2.1 Sequencing in adaptive hypermedia

Most of the approaches to sequencing adaptation are based on rules that determine the next activity or set of activities available for the student. The rules are queried with data from the user model and provide an output that hides or shows the specific activities on the screen for the student to perform. The activities can be organised using UML, which is a well-known description language that can be extended, or some other graph metaphor, specifically designed for the problem at hand.

2.1.1 Deterministic adaptation

UML has been used in CADMOS [Papasalouros et al., 2004] as well as UML-Guide [Dolog, 2004]. The sequencing is usually defined as a UML profile, with a semantic definition for the new elements and several syntactic restrictions regarding how the new elements can be connected. The design of the learning material is usually organised in layers, and one of them is devoted to the design of the sequencing strategy. For example, in [Papasalouros et al., 2004] the sequencing layer is further divided into a structural sub-layer (that connects concepts and resources) and a behavioural sub-layer, that defines several rules that determine whether some elements can be visualised after some others, according to the information stored in the user model. The rules, therefore, are the mechanism that adapts the sequence of learning activities to the student.

The well-known educational framework AHA! [Stash and de Bra, 2002] uses graphs to define the relationship between concepts, and the edges that connect the concepts represent dependency rules between concepts (e.g. knowledge propagation, or prerequisites). An automatic process transforms these relations into adaptation rules that, among other things, can result in the hiding or annotation of links on the system. Another initiative that uses graph to adapt the sequence of learning activities is Sequencing Graphs [Gutierrez et al., 2004a]. In this case, the nodes in the graph represent learning activities and the edges represent possible transitions, that are activated or deactivated according to rules in the user model.

2.1.2 Evolutionary adaptation

All the former initiatives rely heavily on a correct initial design in order to achieve a proper level of adaptation to students. However, there are several factors that can reduce the effectiveness of an *a priori* design, including unknown factors about the student population (e.g. about what they already know) and changes in the student population over time. There are at least two initiatives that have tried to overcome this limitation taking an evolutionary approach.

The Paraschool system [Semet et al., 2003] takes a stochastic graph as the starting point. This graph is defined by pedagogy experts, and contains learning activities at the nodes. The edges represent possible transitions from the current activity to other activities. These transitions are probabilistic (e.g. the transition from activity A to activity B happens 45% of the time). The system then uses an adaptation of the Ant Colony Optimisation [Dorigo and Stutzle, 2004] heuristic to optimise those probabilities. Students that are successful when performing an activity reinforce those transitions they have recently traversed (i.e. the probability increases); conversely, if students are not successful the probabilities on the last transitions they have travelled are reduced.

SIT [Gutierrez et al., 2006] uses another adaptation of the Ant Colony Optimisation heuristic with a similar goal. In this case, the transitions between activities are activated by rules, that determine which are the next activities to be recommended to the student. In order to guide the student's choice, the success of students is recorded in a similar fashion to Paraschool. This information is then given to students, who can choose their next activity based on the success information of former peers with a similar history.

In this paper we acknowledge that providing a deterministic sequencing of learning activities is difficult, especially in the absence of relevant and precise information about the students, and that a sub-optimal design can have a very negative impact in the usefulness of the system to support learning. Therefore, we explore a probabilistic approach based on Bayesian networks, which have already been used in related applications with a high degree of success, as described in the next section.

2.2 Bayesian networks in adaptive hypermedia

Bayesian networks are directed graphs that represents a set of random variables and their conditional dependencies [Charniak, 1991]. The nodes in a Bayesian network represent random variables, while edges represent conditional dependencies between them. Therefore, nodes which are not connected represent variables which are conditionally independent of each other.

Among many other fields, Bayesian networks have been used in educational adaptive hypermedia systems before. They have been used to create learner models [Conati et al., 2002], to infer the need of help for students [Mavrikis, 2008], to ascertain affective characteristics of learners [Conati and Maclaren, 2005], and for plan recognition and problem solving [Albrecht et al., 1997].

The use of Bayesian networks to sequence the learning materials has been little exploited. Hyperbook [Henze and Nejdl, 1999] is one of the few relevant initiatives. This work presents the Bayesian network as something built into the system and they used a single overall dependency graph for modelling the knowledge of the application domain. In the work presented in this paper, the Bayesian networks is used like those in in [Henze and Nejdl, 1999] to infer the system's belief of a user's knowledge and to take decisions based on this information to select the contents that are more adequate for the students.

The main difference of the proposed approach is the focus on reuse and the separation among the *resources*, the *knowledge model* used to capture dependencies among learning units, and the *conditions* used to retrieve adequate resources.

This separation provides a great flexibility to Intelligent Tutoring System generated with the approach proposed in this paper. First, it allows external resources to be incorporated into a course. Additionally, the learning sequence presented to students is not fixed but allows some room for choice, like a recommender system, to enhance the system flexibility from the point of view of the student.

3 Sequencing design with Bayesian networks

3.1 Conceptual framework

The sequencing of the course has to be designed taking into account several concerns. A course is composed of a set of learning units, and those learning units cover some specific knowledge domain. This *knowledge* is to be acquired by the student, and the goal of the adaptive system is to help in this process so that the learning is as effective as possible.

The learner interacts with the learning units (e.g. theory pages, exercises, etc). From this interaction, information can be extracted to evaluate the student and the course itself. This information is called the *evidence*.

After the student has interacted with a learning unit, the system must decide which learning unit is the most adequate to be performed next (i.e. next *action*). In this context, "most adequate" means the one that maximises the probability that the learner achieves the learning objectives of the course. In many cases, several units qualify as "adequate". To calculate which learning units to show next, *evidence* is obtained from two sources: the student's results, and the historical record of the results that other students have obtained in their past interactions with the current resource. Then, the *utility* of all possible next actions is evaluated.

3.1.1 Knowledge

Knowledge is composed by the concepts and ideas that students possess related to a subject, and it is usually dispersed. Information about the knowledge domain needs to be classified in some way. For the purpose of this paper, it has been assumed that the knowledge is divided in learning objectives (with a scheme similar to the one described in [Fernandez Panadero et al., 2002]). These learning objectives are those that the student must achieve during the course.

The learning objectives are defined according to the following rules: the total body of knowledge is divided in several *main learning objectives*, whose number should not be very high; the main learning objectives are divided into secondary levels and, if necessary, the process is repeated; at each level of the hierarchy, all learning objectives should be independent from each other; the level of achievement for each learning objective will be a number between 0 and 1.

3.1.2 Actions

Actions depend on learning objectives, although some actions may use different types of knowledge. There are three types of actions: *passive*, that do not require the students to use their knowledge (e.g. reading), and are evaluated like done ('1') or not done ('0'); *individual active*, that require use of knowledge (e.g. exercises, search of additional information), and are evaluated between 0 and 1 depending on correctness; and *collective active*, that require interaction with other students (e.g. discussion, collaborative exercises), and are evaluated as well between 0 and 1. It has been assumed that this evaluation is performed by the learning units, that give a mark to the students (e.g. after they provide answers for an exercise).

3.1.3 Evidence

There are three types of evidence. *Input evidence* is necessary to choose the *next* activity to present to students (e.g. knowledge required before starting an activity). *Output evidence* provides information about the *last* activity that was performed by a student (e.g. mark obtained on an exercise). Additionally, there is also mixed evidence, that is useful both as input and output (e.g. level of knowledge for a learning objective).

Some evidences are set at design time, like knowledge required for an activity (although they can be refined over time). Other evidences must be collected after the system has been in use for some time, like how effective a particular exercise is to achieve a learning objective.

3.1.4 Utility

The utility of each action can be calculated from the obtained evidence. In the general case, the utility is calculated as a linear combination of the evidences

$$U = \omega_1 \cdot E_1 + \omega_2 \cdot E_2 + \ldots + \omega_n \cdot E_n$$

where those evidences that are more relevant have higher weights. Choosing the right set of evidences and weights is usually a complicated process. In this case, a pragmatic approach has been taken: To calculate the utility of an activity, the Bayesian network has been evaluated as if all the answers had been correct (i.e. if the evaluation was 1). The activity that shows a higher increase in the knowledge level would be selected like the preferred next activity.

3.2 General description of the process

The first step in the process of authoring an adaptive hypermedia system in which activities are to be sequenced using Bayesian networks is an analysis of the domain. The result of this analysis is twofold. First, it classifies the activities in relation to the learning objectives. Second, it determines the relations among learning objectives, in the form of prerequisites and subordinated knowledge. A Bayesian network can be created with this information, using expert knowledge to determine the conditional probabilities. The evaluation of this network gives an estimation of the level of knowledge on some learning objectives as a function of other learning objectives. Those learning objectives at the root of the knowledge hierarchy (i.e. no other learning objective depends on them) are called *primary learning objectives*, and are usually –but not necessarily– associated with high-level content structures (e.g. topics in a course or chapters of a book).

The knowledge of the student for every learning objective is described using three knowledge levels, i.e. low, medium, and high. For each learning objective, the student will have a certain probability of each level of knowledge. For example, for a certain objective (e.g. “knowledge about hard disk scheduling”) a student might have a probability of 0.1 of having a low level of knowledge, 0.6 for medium, and 0.3 for high.

The evaluation of the network for the students provides a fuzzy representation of their accomplishment of the learning objectives, in a similar way to some of the user-modelling initiatives discussed in Section 2.2. In order to go one step further and being able to use this information for sequencing the activities, two additional data structures are needed.

The first one connects the actions (i.e. learning activities) with the learning objectives. This data structure defines how the outcome of the activities influences the knowledge levels in the nodes of the Bayesian network, e.g. how the

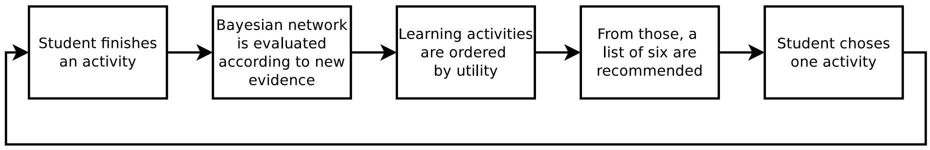

Figure 1: General procedure for sequencing learning activities using Bayesian networks

probability of "high knowledge about hard disk scheduling" is lowered if the activity is not completed successfully.

The second structure is a map between activities and knowledge objectives, i.e. nodes on the Bayesian network. For each activity, a certain knowledge level can be defined as a prerequisite. Prerequisites can be minimum knowledge (to prevent difficult activities from frustrating beginners), maximum knowledge (to prevent easy activities from boring advanced students), and a combination of the two. This structure acts as a filter that enables only those activities that are adequate to students according to their knowledge level.

Once these data structures are in place, the general procedure for sequencing learning activities is depicted in Figure 1. When a student finishes with an activity, the knowledge levels in the nodes of the Bayesian network are updated appropriately as explained. Then, all activities are filtered by minimum and maximum level of knowledge to find all possible next activities. This is potentially a huge number of activities, so there are two additional restrictions: all activities must be relevant for the current primary learning objective, and the maximum number of activities is six. The former restriction reduces confusion by ensuring that all activities are related, and the latter prevents the system from showing too many activities to students and overwhelming them [Schwartz, 2004]. The number was chosen from previous experience with a former tutoring system [Gutierrez et al., 2004b]. An additional activity needs to be added to allow students to change of main learning objective.

4 Implementation details

The aforementioned sequencing strategy has been implemented as a module of a platform for the development of Intelligent Tutoring Systems called SIT. This section provides some additional details about SIT and the implementation.

The main objective of SIT is to be a platform for the development of Intelligent Tutoring Systems. The platform takes care of administrative tasks (from managing user accounts to retrieving and forwarding learning content

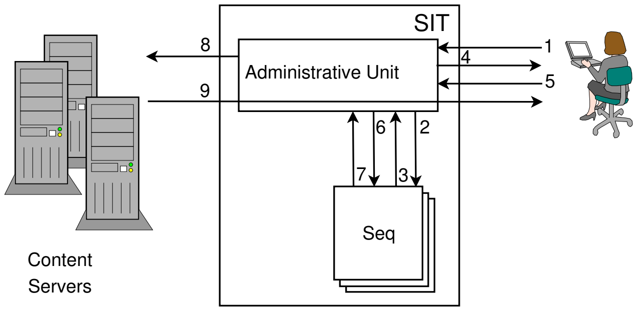

Figure 2: Working cycle of SIT

to users), thus alleviating a big load of work from the tutoring system developer, who is able to concentrate on the pedagogical [Engvig, 2002], cognitive [Love and Guthrie, 2003] and instructional [Clark and Mayer, 2002] aspects of the system. It has been shown [Murray, 1999, Woolf and Cunningham, 1987] that the other tasks amount to more than half of the resources (e.g. time, developers) dedicated to the creation of an ITS. SIT tries to allow teachers and designers to focus on learning and not on technology.

The architecture of SIT is oriented to the reuse of learning resources already available on the Internet. These resources are shown to the student using a web browser, and they can be either static (e.g. images, static HTML pages, etc) or dynamic (e.g. JSP pages, etc) content. Resources are sequenced for students by the use of sequencers, i.e. modules that process the interaction of the student with the system and make recommendations on the next activities that students should try. A central administrative unit acts as the central hub that connects the student, the sequencer, and the web resources.

The working cycle of SIT is shown in Figure 2. The cycle starts when students requests a new learning activity (1), usually after finishing with the former one. The administrative unit collects the data (if any) from the last activity performed (e.g. results of an exercise). All data collected is sent to the sequencer (2), which uses that information to update the student model and choose some activities as candidates to be the next activity (3). This information is sent back to the students through the administrative unit (4). The students can then choose the next activity they prefer to try next (5). This information is given to the sequencer (6-7), and the activity is offered to the students (8-9), closing the cycle.

A sequencer was created for SIT that implements the sequencing strategy described in the former section. This module uses three XML files to describe the Bayesian nets of knowledge, activity evaluation, and activity selection; and implements the same lifecycle as other SIT sequencers.

5 Case study

The methodology presented in Section 3 has been applied to the design of a support system for a course on operating systems. The starting point was a set of learning resources, including web pages with the theory of the course, examples, and exercises. This learning material was reused from a already existing course in Operating Systems (described in [Prieto-Linillos et al., 2008]).

The learning units (e.g. theory web pages, web-based exercises) were reused, but the sequencing strategy was designed using the methodology presented in Section 3. In other words, the activities were already present, but it was necessary to design the learning objectives and the evidence. This section presents the process that was followed and the lessons that were learnt about the process.

5.1 Learning objectives and course structure

The first step was to define learning objectives. The key point here is to identify the granularity level of these objectives because a Bayesian network will be defined for each one of them in order to guide the decision process.

In this case study, the selection of learning objectives matches the structure of the course [Fernandez Panadero et al., 2002]. The main learning objectives are equivalent to the highest level hierarchy of the learning units in the Operating Systems course defined in [Prieto-Linillos et al., 2008]. In this case, the learning material was classified into five main sections: "processes", "cache memory","main memory" (i.e. RAM, ROM), "secondary access memory" (i.e.disk), and "file systems". A Bayesian network was created for each one of them. This Bayesian network are represented by an XML file (in XBN format[1]) as depicted in Figure 3

This design decision does not affect to the granularity of the learning resources. Each learning unit can be further divided into secondary and tertiary nodes, where appropriate. For instance, the objective associated with the content"file systems" was divided into: "access permissions", "inodes", "free space management"; and "free space management" was further divided into "bit vectors", and "linked lists". This new division of the content is no longer reflected as a new Bayesian network but as different nodes in the network represented by the node "file systems" as depicted in Figure 4.

The resulting Bayesian network is a directed acyclic graph whose nodes represent observable variables. These variables are related with a specific unit of learning and can have different states with different probability functions associated with them. For example the node "linked list" reflects the knowledge of a specific student about this subject and has three possible states (low, medium,

[1] XBN is a well-known XML format for description of Bayesian networks, see *http://xml.coverpages.org/xbn.html* (accessed June 2010).

State of knowledge	Low	Medium	High	Total
Default	0.33	0.33	0.33	1
Actual	0.98	0.01	0.01	1

Table 1: Learning units for learning objective “file systems”, with initial values. In the absence of former knowledge about the student, all states have the same probability.

and high) that represent different levels of knowledge as it shown in Figure 3. Each state has an associated probability and the sum of the probabilities associated with every state node must be one as shown in Table 1.

Edges represent conditional dependencies among units of learning; nodes which are not connected represent variables which are conditionally independent of each other. Each node is associated with a probability function that takes as input a particular set of values for the node’s parent variables and gives the probability of the variable represented by the node. For example, the node “free space management” has two parents with three different states for each one of them: low, medium, high represented respectively by 0, 1, 2. Then the probability function associated with this node could be represented by a table of nine entries, one entry for each of the possible combinations of its parent’s states being.

Each node is represented in FileSystems.xbn as an element *<VARIABLE>* as shown in in Figure 3. Connections between different nodes are represented as an *<STRUCTURE>* element containing several *<ARC>* elements. In each arc two attributes *PARENT* and *CHILD* reflect the relation between the nodes. Finally the distribution of probability associated with the Bayesian network is reflected in the element *<DISTRIBUTIONS>* with a *<DIST>* element for each node that shows conditional dependencies from this node *<PRIVATE NAME=”LinkedList”>*, and with its parents represented as conditional elements *<CONDELEM>* inside a conditional set element *<CONDSET>*.

There were no restrictions to define the number of parent nodes for each unit of learning and hence on the structure of contents. However, in an effort to keep the complexity of the network low an additional rule was followed: each resource (activity, theoretical content, etc.) should influence only one node in the Bayesian network, when it is possible, but a node could have several resources associated with it. In most cases this could be achieved, but there is a minority of activities that influence more than one node.

At the end of this stage, the course has a Bayesian network for each learning objective. These learning objectives correspond to the units of learning in the highest level of the hierarchy of the course, the main topics. Each Bayesian network will have one main objective and therefore a single leaf node, but can

```
<?xml version="1.0"?>
<ANALYSISNOTEBOOK NAME="Notebook.bndefault" ROOT="bndefault">
 <BNMODEL NAME="bndefault">
    (...)
  <VARIABLES>
    <VAR NAME="LinkedLists" TYPE="discrete" XPOS="1835" YPOS="2217">
     <FULLNAME>LinkedLists</FULLNAME>
     <STATENAME>low</STATENAME>
     <STATENAME>medium</STATENAME>
     <STATENAME>high</STATENAME>
    </VAR>
      (...)
    <VAR NAME="File_Systems" TYPE="discrete" XPOS="4207" YPOS="7329">
    (...)
    </VAR>
  </VARIABLES>
  <STRUCTURE>
    <ARC PARENT="Bit_Vectors" CHILD="Free_Space_Mng"/>
    <ARC PARENT="LinkedLists" CHILD="Free_Space_Mng"/>
    <ARC PARENT="Inodes" CHILD="File_Systems"/>
    <ARC PARENT="Free_Space_Mng" CHILD="File_Systems"/>
    <ARC PARENT="Permissions" CHILD="File_Systems"/>
  </STRUCTURE>
  <DISTRIBUTIONS>
    <DIST TYPE="discrete">
      <CONDSET>
        <CONDELEM NAME="Bit_Vectors"/>
        <CONDELEM NAME="LinkedLists"/>
      </CONDSET>
      <PRIVATE NAME="Free_Space_Mng"/>
      <DPIS>
        <DPI INDEXES="0 0 ">0.98 0.01 0.01 </DPI>
        <DPI INDEXES="0 1 ">0.65 0.25 0.1 </DPI>
        (...)
        <DPI INDEXES="2 2 ">0.01 0.34 0.65 </DPI>
      </DPIS>
    </DIST>
    (...)
    <DIST TYPE="discrete"><PRIVATE NAME="LinkedLists"/>
      <DPIS>
        <DPI INDEXES="default">0.333333 0.333333 0.333333 </DPI>
        <DPI>0.98 0.01 0.01 </DPI>
      </DPIS>
    </DIST>
  </DISTRIBUTIONS>
 </BNMODEL>
</ANALYSISNOTEBOOK>
```


Figure 3: FileSystems.xbn: XML File to define the Bayesian network for the objective File Systems

have multiple root nodes. These parent nodes represent the learning units of a lowest level in the hierarchy of course contents that contribute to achieving this learning objective. For example, units of learning related with bit vectors, linked lists, permissions, inodes. and free space management contribute to the knowledge of “file systems”.

Secondary LG	Tertiary LG	Theory	Exercises
Permission	-	1	1
Inodes	-	1	1
Free space	Bit vectors Linked lists	1 1	2 1

Table 2: Learning units for learning objective "file systems"

5.2 Classification of resources

Once that the learning objectives and the course structure have been defined and represented in the Bayesian networks, it is necessary to classify the resources and associate them with different nodes in the network.

The resources are the activities that will be presented to the student. These activities provide the evidence about the student's level of knowledge for each unit of learning. Resources can be passive actions (such as theory pages or examples) or active individual actions like exercises. These exercises could be parametrised, so that every time the student interacts with one of them the resulting exercise is different [Prieto-Linillos et al., 2008].

Interaction with learning resources has an impact in the student level of knowledge. This impact should be measured to determine the degree to which the student's state of knowledge associated with this learning unit (node) has changed.

In this case study each node has associated at least one theoretical and one practical resource from the secondary level to the bottom of the hierarchy. For the sake of clarity and simplicity, the result for the main objective "file systems" will be shown here. The resulting network has only six nodes (cf. the network for the learning objective "main memory" was the biggest one, with fifteen nodes), and three levels of learning objectives (c.f. four levels in the case of "main memory"), being adequate to illustrate the main aspects of the design.

There were nine learning units related to "file systems": four of them were theory pages (passive actions), sometimes including examples; and another five were exercises (active individual actions). These learning resources were classified into the secondary and tertiary levels of the hierarchy related with the learning objectives they were relevant for. In this case, each activity influenced only one node. The classification of learning resources is shown in Table 2.

Every learning objective has an associated activity to obtain evidence about the level of knowledge. The resulting Bayesian network is shown in Figure 4.

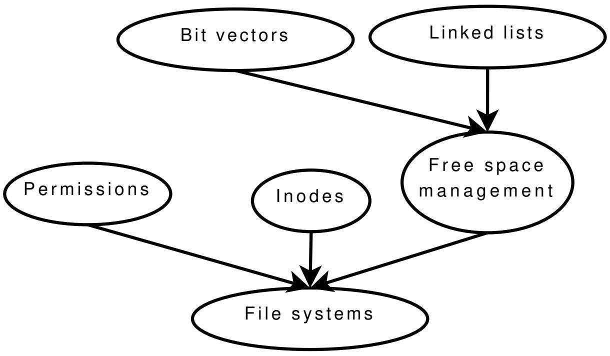

Figure 4: Bayesian network for learning objective "file systems"

The networks for all main learning objectives are combined into one, that comprises the whole course. From that point of view, the course can be thought as a Bayesian network whose objective is the knowledge on operating systems, and where the secondary objectives are those topics like "file systems". In order to achieve these learning objectives the course provides several resources.

The mapping between the activities and nodes that are associated with them is made in a configuration file called *eval_act_objective.xml*. For example, a set of activities (theory, examples and exercises) have been created to provide evidence of the student knowledge about the main topic. This set of activities and the main objective with which they are related are stored in the file *eval_act_FileSystem.xml* as shown in Figure 5

This file contains an element *<OBJECTIVE>* that indicates the main learning objective with which the subsequent exercises are related. A set of exercises is represented by the elements *<EXER>*.

Each specific activity has an attribute *url* that indicates the resource locator for this exercise and an element *<UPDATE>* that allows the designer to specify the *condition* that must be met to consider the state of knowledge the student has changed. For example, this condition could involve reading a theoretical content or having passed an exercise with a given rating. This file only provides an identifier for the condition. The parameters and values for these conditions are specified in a different configuration file.

For example, when students answer correctly to the questions in an exercise of medium-high difficulty about access permissions in file systems, it can be assumed that their knowledge on this subject has improved; therefore, the network node corresponding to this matter (permissions-node) must be updated accordingly to this new evidence.

In this case, it is assumed that the student's knowledge about access permis-

State of knowledge	Low	Medium	High	Total
Before exercise	0.98	0.01	0.01	1
Effect of the exercise	-0.035	+0.02	+0.015	0
After exercise	0.945	0.03	0.025	1

Table 3: Actualization of state of student's knowledge for the node "permissions"

```
<ACTIVITIES>
  <OBJECTIVE name="File_Systems">
  <EXER url="http://chopin.gast.it.uc3m.es:8080/data/files accessPermissions.jsp">
    <!-- 1 File Systems Exercise, SF.PA.P1 -->
    <UPDATE nodename="Permissions" cond1="answer_evaluation">
      <STATES state="low" inc="0.02"/>
        <STATES state="medium" dec="0.035"/>
      <STATES state="high" inc="0.015"/>
    </UPDATE>
  </EXER>
  (...)
  <EXER> (...) </EXER>
  (...)
</OBJECTIVE>
</ACTIVITIES>
```

Figure 5: eval_act_FileSystems.xml: XML File to eval the Bayesian network for the objective File Systems

sions in file systems has been increased. As a result of this fact, the likelihood that the student has a high or medium level of knowledge will be increased and the likelihood that their knowledge is low will be decreased. This can be expressed in the configuration file using the element *<STATE>* and their attributes *state* to indicate a concrete state of the node, *inc* to represent an increase in the likelihood of this state, and *dec* to indicate decrease. The sum of the probability of all states before and after the exercise should continue to be one as shown in Table 3

5.3 Sequencing of resources

The students interact with the learning units one at a time. When the student finishes one learning unit (e.g. submits the answers to the exercise), the system processes the answers and presents a set of possible next learning units. These units correspond to active and passive actions, and help the students to develop their knowledge. Activities are selected for presentation according to the level on each objective. Instead of showing only one activity (i.e. the one with the highest utility), it was decided that several activities would be shown so that students could choose. This decision was motivated by three factors: first, in many cases the utility of two or more activities is so similar that it makes little sense to make a strict decision in favor of one over the other; second, having the possibility of choosing the next activity gives the students a sense of ownership of their learning process, resulting in higher motivation and better learning; last, choosing several possible activities instead of one makes the system more robust against mistakes made by the network designer.

Two additional design decisions had to be taken at this point, as the initial trials with the system showed that the number of learning units that could be chosen was extremely large at the beginning. At that point, the system has very little or no information at all about the students, so most activities have a similar utility for them. Showing too many activities can be counterproductive [Schwartz, 2004], so a limitation was set on the number of activities that could be shown at any time (those whose predicted utility was the highest).

This change improved the usability of the system, but an additional measure was still needed. In the initial steps, many activities had very similar utilities, so it was not possible to discriminate. This had sometimes the effect of selecting a set of activities to be presented that contained very different activities (different topic, different difficulty level, etc). This is potentially confusing for students, even more so because it tends to happen at the beginning, when students still have little information to choose the next activity they want to perform. Therefore, it was decided that learning units would be chosen from only one main learning objective at any time. On the other hand, a special activity called "Change topic" was created, so that the students could change learning objectives if they so desired. Therefore, the first time the students interact with the course (i.e. no prior information on the system), they are asked about which topic they find more interesting; the answer to this question selects the first learning objective from which learning units will be selected from.

This solution limited the choice of activities for the student, so that the selection of activities was not extremely confusing; but, on the other hand, it still granted the students access to the whole set of learning units.

The conditions to be met by an educational resource to be elected as the next activity are captured in a configuration file called *next_act_objective.xml*. There must be one of these files for each Bayesian network, i.e. for each learning objective and each main topic as shown in Figure 6.

The structure of this file is very similar to *eval_act_objective.xml* in Figure 5. There are a set of resources included as <*EXER*> elements. In this case this element has two new parameters: *name* to easily identify the resource and *type* to determine when the resource consists of theoretical information, exercises or exercises' solutions.

The tag <*CONDITION*> is used to know whether the resource can be chosen as the next activity. To determine this point several attributes are available:

```
<ACTIVITIES>
  <OBJECTIVE name="File_Systems">
  (...)
  <EXER name="Inodes" url="http://chopin.gast.it.uc3m.es:8080/data/files/
   inodes.jsp" type="activity">
    <CONDITION (...) > (...) </CONDITION>
      (...)
      <CONDITION nodename="Inodos" state="medio" maxlevel="0.75"
                                          minlevel="0.3" >
      <PARAMS name="min_n_directories" value="7"/>
      <PARAMS name="max_n_directories" value="11"/>
      <PARAMS name="min_n_plain_files" value="7"/>
      <PARAMS name="max_n_plain_files" value="11"/>
      <PARAMS name="min_n_symbolic_links" value="7"/>
      <PARAMS name="max_n_symbolic_links" value="11"/>
      <PARAMS name="min_n_physical_links" value="7"/>
      <PARAMS name="max_n_physical_links" value="11"/>
    </CONDITION>
      (...)
      <CONDITION (...) > (...) </CONDITION>
  </EXER>
  (...)
</OBJECTIVE>
</ACTIVITIES>
```

Figure 6: *next_act_FileSystems.xml*: XML File to define conditions to select next resources depending on knowledge level for the objective File Systems

nodename to identify the node where the conditions will be checked, *state* to indicate which state is being assessed and to indicate the probability levels *maxlevel*, *minlevel* between which this resource can be selected. If these conditions are met this learning resource may be selected as the next activity.

The tag <*PARAMS*> indicates the input parameters to be applied in every exercise. The attributes indicate the parameter name (*param*) and its value (*value*). These parameters can increase or decrease the difficulty of the exercise and are the input parameters of the exercises.

6 Discussion and Future Work

The approach presented in this paper is based on one type of evidence: the actions of the students. This was a pragmatic decision that tried to minimise the amount of work to be done by the designer. However, there are several evidences that have not been used. Some of them could play an important role in the sequencing of activities and open interesting lines for future work.

Time is a factor that has been neglected but could potentially be used to produce more precise inferences in two ways. First, time used by students for finishing activities can be used to modify the amount of knowledge gained by the students (e.g. if too little time was used, little was learnt). Second, time can be used to implement a “forgetfulness” factor. For instance, if the learner has not interacted with a learning objective for a long time, it can be assumed that this knowledge would need to be refreshed. Therefore, some activities related to it can be given a higher priority.

Another type of evidence that has not been used are personality traits and emotional factors [Porayska-Pomsta et al., 2008]. Motivation and satisfaction can potentially be inferred from the number of activities performed by the students, and from how frequently they use the system; e.g. a motivated student will connect to the system often and interact with many activities, while a student that logs off every time he fails on exercises can be inferred to be a student with a low-tolerance to frustration.

Finally, it is important to note that learning activities are not independent from each other. An appropriate sequence of activities can have a more positive impact on learning than just the combination of them. In the present work, the sequencing adaptation has considered only the next step of the sequencing. However, the Bayesian sequencing subsystem can consider several possible "future steps", calculating the utility of all those paths to select the best next activities. This adds an additional layer of complexity, but could lead to better adaptation of the sequencing. Additionally, inference about the utility of future activities can be combined with actual results of other students to make a more precise inference using swarm intelligent techniques [Gutierrez Santos et al., 2006]. This line of work has been already pursued in the past for other probabilistic approaches to content sequencing [Semet et al., 2003].

7 Conclusions

A strategy for authoring a flexible sequencing mechanism for educational hypermedia systems has been presented, based on Bayesian networks. This paper has described the methodology and its implication from the point of view of authoring of hypermedia systems. A case study has also been provided.

The main advantage of this approach is its ability to work with and adapt to incomplete or fuzzy information about the student, compared to other more deterministic approaches. This makes this approach very appropriate for authoring sequencing strategies in hypermedia systems with low interaction with students.

Acknowledgment

Research partially funded by the following projects: project "Learn3: Towards Learning of the Third Kind" funded by the Spanish Ministry of Science and Innovation under grant No. TIN2008-05163/TSI and project E_MADRID: "Investigacion y Desarrollo de Tecnologias para el e-Learning en la Comunidad de Madrid" funded by the Madrid Regional Government under grant No. S2009/TIC-1650.

References

[Albrecht et al., 1997] Albrecht, D., Zukerman, I., Nicholson, A., and Bud, A. (1997). Towards a bayesian model for keyhole plan recognition in large domains. In *In Proceedings of the Sixth International Conference on User Modeling, UM97*.

[Charniak, 1991] Charniak, E. (1991). Bayesian networks without tears: making bayesian networks more accessible to the probabilistically unsophisticated. *AI Magazine*, 12.

[Clark and Mayer, 2002] Clark, R. and Mayer, R. E. (2002). *e-Learning and the Science of Instruction: Proven Guidelines for Consumers and Designers of Multimedia Learning*. Pfeiffer.

[Conati et al., 2002] Conati, C., Gertner, A., and VanLehn, K. (2002). Using bayesian networks to manage uncertainty in student modeling. *User Model. User-Adapt. Interact.*, 12(4):371–417.

[Conati and Maclaren, 2005] Conati, C. and Maclaren, H. (2005). Data-driven refinement of a probabilistic model of user affect. In *User Modelling*, pages 40–49.

[Dolog, 2004] Dolog, P. (2004). Model-driven navigation design for semantic web applications with the uml-guide. *Engineering Advanced Web Applications*.

[Dorigo and Stutzle, 2004] Dorigo, M. and Stutzle, T. (2004). *Ant Colony Optimization*. MIT Press.

[Engvig, 2002] Engvig, M. (2002). *Best Practices in eLearning Case Study*. Themo.

[Fernandez Panadero et al., 2002] Fernandez Panadero, M. C., Pardo, A., Fernandez Panadero, J., and Marin Lopez, A. (2002). A mathematical model for reusing student learning skills across didactical units. In *Int. Conf. on Frontiers in Education*.

[Gutierrez et al., 2004a] Gutierrez, S., Pardo, A., and Kloos, C. D. (2004a). An adapting tutoring system based on hierarchical graphs. In *Adaptive Hypermedia'04*, volume 3137 of *Lecture Notes in Computer Science*. Springer.

[Gutierrez et al., 2004b] Gutierrez, S., Pardo, A., and Kloos, C. D. (2004b). An adapting tutoring system based on hierarchical graphs. In *Adaptive Hypermedia*, volume 3137 of *Lecture Notes in Computer Science*. Springer.

[Gutierrez et al., 2006] Gutierrez, S., Pardo, A., and Kloos, C. D. (2006). Finding a learning path: Toward a swarm intelligence approach. In *Web Based Education*. ACTA Press.

[Gutierrez Santos et al., 2006] Gutierrez Santos, S., Pardo, A., and Delgado Kloos, C. (2006). Some ideas for a collaborative search of the optimal learning path. In *Adaptive Hypermedia*, pages 430–434.

[Henze and Nejdl, 1999] Henze, N. and Nejdl, W. (1999). Bayesian modeling for adaptive hypermedia systems. In *Workshop Adaptivitat und Benutzermodellierung in Interaktiven Softwaresytemen*.

[Love and Guthrie, 2003] Love, P. G. and Guthrie, V. L. (2003). *Understanding and Applying Cognitive Development Theory*. Jossey-Bass.

[Mavrikis, 2008] Mavrikis, M. (2008). Data-driven prediction of the necessity of help requests in iles. In *Adaptive Hypermedia*, pages 316–319.

[Murray, 1999] Murray, T. (1999). Authoring intelligent tutoring systems: An analysis of the state of the art. *Int. J. of Artificial Intelligence in Education*, 10:98–129.

[Papasalouros et al., 2004] Papasalouros, A., Retalis, S., and Papaspyrou, N. (2004). Semantic description of educational adaptive hypermedia based on a conceptual model. *Educational Technology and Society*, 1.

[Porayska-Pomsta et al., 2008] Porayska-Pomsta, K., Mavrikis, M., and Pain, H. (2008). Diagnosing and acting on student affect: the tutor's perspective. *User Model. User-Adapt. Interact.*, 18(1-2):125–173.

[Prieto-Linillos et al., 2008] Prieto-Linillos, P., Gutierrez-Santos, S., Pardo, A., and Delgado Kloos, C. (2008). Guaranteeing the correctness of an adaptive tutoring system. In *Adaptive Hypermedia and Adaptive Web-Based Systems*.

[Schwartz, 2004] Schwartz, B. (2004). *The Paradox of Choice: Why More Is Less.* Ecco.

[Semet et al., 2003] Semet, Y., Yamont, Y., Biojout, R., Luton, E., and Collet, P. (2003). Artificial ant colonies and e-learning: An optimisation of pedagogical paths. In *10th International Conference on Human-Computer Interaction*.

[Stash and de Bra, 2002] Stash, N. and de Bra, P. (2002). AHA! A general-purpose tool for adaptive websites. In *World Wide Web Conference*.

[Woolf, 2008] Woolf, B. P. (2008). *Building Intelligent Interactive Tutors: Student-centered strategies for revolutionizing e-learning.* Morgan Kaufmann.

[Woolf and Cunningham, 1987] Woolf, B. P. and Cunningham, P. (1987). Building a community memory of intelligent tutoring systems. In *National Conference on Artificial Intelligence (AAAI)*, pages 82–89.