Chapter 3

Student Model

Assessment is a necessary part of tutoring. A number of decisions need to be taken about students’ knowledge mastery while tutoring. For taking these decisions intelligently, the knowledge of what each student has learned and the prediction of how that student will perform when solving future problems are required. This calls for forming a model of the student’s knowledge and diagnosing where the student has incomplete and incorrect knowledge.

In an educational environment, a good student model must include all features of the student’s knowledge and preferences related to his learning and performance (Kavcic, 2000). This information is used to adapt the system to the student. However, building this model is a very difficult task. In practice, a partial model is used. It is necessary to take into account (i) what information is included in the model; (ii) how to obtain it; (iii) how the model will represent the information; and finally, (iv) how the model will process and update the information (Carmona & Conejo, 2004).

For defining the scope of knowledge in a course, course document is the formal basis, in particular, the syllabus. A domain expert will be able to represent this course document as knowledge hierarchy with reference to this document. In Section 3.1, the construction of Bayesian Student Model is explained. Determining the probabilistic values is the difficult part in Bayesian Student Model, and the techniques to obtain these values from teacher
beliefs and from observed data are discussed in section 3.2. Finally, in section 3.3, the approach used to update the Bayesian network is discussed.

### 3.1 Bayesian Student Model

In BiTutor, Bayesian network is used to represent and perform an overlay student model. In overlay student model (described in detail in Chapter 2), the student's knowledge is considered as a subset of the expert's knowledge, and the structural model needs to be defined, that is, nodes, relationships, parameters, and the inference. In the following sections, a brief introduction about Bayesian network and the construction of Bayesian student model are discussed.

#### 3.1.1 Bayesian Network

Bayesian networks are widely used for knowledge representation and reasoning under uncertainty in Intelligent-Tutoring Systems. A Bayesian network (Pearl, 1988) is a directed acyclic graph in which nodes represent variables and arcs represent probabilistic dependence among variables. The parameters used to represent the uncertainty are the conditional probabilities of each node given each combination of states of its parents; that is, if \( X_i, i=1\ldots n \) are the variables of the network and \( pa(X_i) \) represents the set of parents of \( X_i \), for each \( i=1\ldots n \), then the parameters of the network are \( \{P(X_i|pa(X_i)) \}, i=1\ldots n \), that is, the set of discrete conditional probability distributions of each variable given its parents. This set of probabilities defines the joint probability distribution for the entire network as,

\[
P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P(X_i | pa(X_i))
\]  

(3.1)
If Bayesian network is used to define a student model, the variables can represent different things depending on the domain. The variables can be rules, concepts, problems, abilities, skills, etc. These variables are linked by relationships between them such as belong-to and pre-requisite. Once the links and the variables have been defined, the conditional probabilities must be specified.

**Evidence**

Evidence (sometimes called finding) on a variable is a statement of the certainties of its states. If the evidence tells us exactly which state the variable is in, it is called *hard* evidence (or instantiation) and we say that the variable is instantiated. Evidence that is not hard is called *soft* evidence. An example of soft evidence on a variable $X_a$ is when one of its descendants has received (hard or soft) evidence, (see Fig. 3.4). Variables that have received soft evidence have vertices that are also double-framed but they are now dashed. The vertices are also colored with a somewhat lighter version of gray.

**Connections and Propagation Rules**

A *serial connection* in a belief network $G = (N,L)$, where $N$ is the set of nodes and $L$ is a set of relations/links between nodes, is a directed sub-graph $G' = (N', L')$ such that $N' = \{a, b, c\} \subseteq N$, and $L' = \{(a, b), (b, c)\} \subseteq L$. The connection can also be written as $X_a \rightarrow X_b \rightarrow X_c$. In Figure 3.1, an example of a serial connection is given. A connection, which is serial, has the property that, if we receive evidence about $X_a$, it will update our knowledge about $X_b$, which then in turn will update our knowledge about $X_c$. This propagation of information is done by marginalization. Similarly, if we receive evidence about $X_a$, it will, through Bayes' Rule update our knowledge about $X_b$ and continue on to $X_c$ (see Figure 3.1.a). If $X_b$ is instantiated, all information will be blocked at $X_b$, e.g. new information about $X_a$ will not reach $X_c$, since $X_b$ is determined. The same is valid if we receive evidence about
Our knowledge about $X_b$ will not be changed (see Figure 3.1.b). To conclude, a serial connection has the property $X_a \perp X_c \mid X_b$.

![Fig 3.1: Serial connection. A vertex colored gray is an instantiated variable. Non-instantiated variables are white. In a serial connection $X_a \rightarrow X_b \rightarrow X_c$, the information cannot pass $X_b$ if it is instantiated.](image)

A *diverging connection* in a belief network $G = (N, L)$ is a directed sub-graph $G' = (N', L')$ such that $N' = \{a, b, c\} \subset N$, and $L' = \{(a, b), (a, c)\} \subset L$. Shortly, it can be written $X_b \leftarrow X_{a} \rightarrow X_c$ as shown in Figure 3.2.a. The *diverging connection* has similar properties to the serial connection. If $X_a$ is not instantiated, new information received about $X_b$ or $X_c$ will update our knowledge about $X_a$. Then the information will continue to $X_c$ and $X_b$, respectively (see Figure 3.2.a). In the case that $X_a$ is instantiated, information will not be able to pass through $X_a$. It is blocked at $X_a$ (see Figure 3.2.b). To conclude, a diverging connection $X_b \leftarrow X_a \rightarrow X_c$ has the property $X_a \perp X_c \mid X_b$.

![Fig 3.2: In a diverging connection $X_b \leftarrow X_a \rightarrow X_c$ the information cannot pass $X_a$ if it is instantiated.](image)
In a Bayesian network, $G = (N, L)$, a **converging connection** is a directed sub-graph $G' = (N', L')$ such that $N' = \{a, b, c\} \subseteq N$, and $L' = \{(a, c), (b, c)\} \subseteq L$. The connection can also be written as $X_a \rightarrow X_c \leftarrow X_b$ as shown in Figure 3.3. A converging connection will in some sense have opposite propagation properties compared to a serial or a diverging connection. Evidence received in $X_a$ or $X_b$ will *only* pass $X_c$, if $X_c$ has received some evidence, hard (see Figure 3.3.b) or soft. We say that the evidence at $X_c$ *opens* for information. If nothing is known about $X_c$, the new information will bounce back (see Figure 3.3.a) and we call the parents independent. Note that it is not necessary for $X_c$ to be instantiated to make the parents dependent, only that we know *something* about $X_c$.

**Fig 3.3:** In a converging connection $X_a \rightarrow X_c \leftarrow X_b$ the information can *only* pass $X_c$ if it has received some evidence.

In the case described in Figure 3.4, there is an opening in the converging connection at variable $X_c$, because it has received soft evidence from its descendants, $X_d$ and $X_e$. Variable $X_c$ did first receive (hard) evidence. To conclude, a converging connection $X_a \rightarrow X_c \leftarrow X_b$ has the property $X_b \perp X_c | X_a$. 

---

*Student Model 52*
3.1.2 Construction of Bayesian Student Model

This section will explain the different types of nodes used in Bayesian student model. Also, the relationships among these nodes are explained. Nodes and relationships are the building blocks of Bayesian student model.

Nodes

BiTutor describes two different types of nodes, namely, the nodes to measure a student's knowledge and the nodes to collect evidence.

Nodes to Measure Student's Knowledge

To measure student's knowledge, BiTutor uses a hierarchical representation of domain knowledge. When converting domain knowledge into a hierarchical representation, dynamic levels of granularity can be identified. In order to keep terminology simple, BiTutor uses terminologies like Course, Topic, Sub-topic, and Concept. A concept is an elementary piece of knowledge, in the sense that it cannot be decomposed into smaller parts. A sub-topic is a collection of concepts and other sub-topics. A topic is a collection of sub-topics. A course is...
a collection of topics. In this case, nodes can be represented as Course, Topic, Sub-topic, or Concept, for easy use. BiTutor calls these nodes as Knowledge Nodes (KN). Each knowledge node is a random variable, which takes three different states, namely, $M$ (Mastery state) if the student has mastered the KN, $PM$ (Partial Mastery state) if the student has partially mastered the KN, and $NM$ (Non-Mastery state) if the student has not mastered the KN. The Expected Mastery Value ($EMV$) of KN is

$$EMV(KN) = \sum x, P(x)$$ (3.2)

The KN takes the value $NM$, if $0 \leq EMV(KN) \leq 0.39$, $PM$ if $0.40 \leq EMV(KN) \leq 0.79$, and $M$ if $0.80 \leq EMV(KN) \leq 1.0$.

**Nodes to Collect Evidence**

These nodes are used to collect the information relevant to the student's state of knowledge. BiTutor uses the test item as the source of evidence. The test item can be a multi-choose or multiple-choice question, which is used to test one or more KNs. To represent an evidence node, the random variable $E$ with Bernoulli distribution is used; $E$ takes two different states, namely, $1$ if the student gives correct answer and $0$ otherwise. The probability law of $E$ is given by

$$P(E = x) = p^x (1 - p)^{1-x}$$ (3.3)

where $p$ represents the probability that the student give correct answer, and $x$ takes values $1$ or $0$. Evidence nodes are not attached to the network until the test items are selected by BiTutor decision system for testing the knowledge node. Once an item is selected for testing the student, it will be taken from the item/question bank and the evidence node will be dynamically added to the network. The selection of the best item to be tested is described in detail in Chapter 4.
Example

BiTutor is developed to teach computer science courses. One of these courses is “Elementary Data Structure”, which is an important course for the computer science/engineering students. Some of the knowledge nodes of the course “Elementary Data Structure” used by BiTutor are shown in Table 3.1.

Table 3.1: Design of a structure of the “Elementary Data Structure” course.

<table>
<thead>
<tr>
<th>Course</th>
<th>Topics</th>
<th>Subtopics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis of Algorithms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Arrays</td>
<td></td>
<td>Singly Linked Lists</td>
</tr>
<tr>
<td>Elementary Data Structure</td>
<td>Linked List</td>
<td>Doubly Linked Lists</td>
</tr>
<tr>
<td>Elementary Data Structure</td>
<td>Multiply Linked Lists</td>
<td></td>
</tr>
<tr>
<td>Applications</td>
<td>Applications</td>
<td></td>
</tr>
<tr>
<td>Stack Operation</td>
<td>Stack Application</td>
<td></td>
</tr>
<tr>
<td>Stacks</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Queue</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trees and Binary Tree</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Modeling Relationships or Links

The causal relationships among nodes are, as follows: belong-to or aggregation relationships between knowledge nodes at different level of granularity, and pre-requisite relationships between the knowledge nodes, which are the very common relationships in education field. Carmona et al. (2005) have defined two different layers to represent these relationships. Nouh et al. (2007a) have defined both relationships in one layer only.

Modeling Belong-to Relationships

Belong-to relationships are recognized between a knowledge node and its decomposed knowledge nodes. For example, the relationship is recognized between a course and its topics and between a sub-topic and its concepts.

Millán and De-la Cruz (2002) have defined two different methods to model the causal relationships between mastering the knowledge node (KN) and its decomposed knowledge
nodes $KN_i$ to $KN_n$ (see Fig. 3.5). Method (I) *converging connection*: mastering $KN_i$ to $KN_n$ have causal influence in mastering $KN$; Method (II) *diverging connection*: mastering $KN$ has causal influence in mastering $KN_i$ to $KN_n$. The rules of propagation of these methods are discussed earlier in this chapter.

To compare the two methods, let us analyze the independent relationship implied by each method, and then determine the number of parameters required for each method. For Method (I), the parameters required are the prior probabilities of knowing each $KN_i$, $\{P(KN_i), i=1,\ldots,n\}$, and the conditional probabilities $P(KN|KN_i,\ldots,KN_n)$.

![Diagram showing converging and diverging connections](image)

**Fig 3.5: Two methods in modeling relationship among the different levels of hierarchy.**

Therefore, the total number of parameters required is $(n-1)(p-1) + (p-1)p^{(n-1)}$ where $p$ is the number of states in each knowledge node. In addition, if nothing is known about $KN$ expect what may be inferred from knowledge of its parents $KN_i (i=1,\ldots,n)$, then the parents are independent: evidence about mastering a $KN_i (i=1,\ldots,n)$ changes the probability of mastering its child $KN$. Evidence about mastering $KN$ changes the probability of mastering its parents $KN_i (i=1,\ldots,n)$, and open communication between them.

For Method (II), the parameters required are the prior probability of knowing $KN$, and the conditional probabilities $\{P(KN_i|KN), i=1,\ldots,n\}$. The total number of parameters...
required is \((p-1) + p(p-1)(n-1)\). In addition, evidence about mastering \(KN_i\) 
\((i = 1, \ldots, n)\) changes the probability of its parent \(KN\), which in turn changes the probabilities 
of the other children \(KN_j (i \neq j)\). Evidence about mastering \(KN\) changes the probabilities of its 
children \(KN_i (i = 1, \ldots, n)\), and blocks the communication between them. Thus, the main 
differences are that in Method II, evidence about mastering a \(KN\) affects the probability of 
mastering the rest of the \(KNs\) of the same level, \(KN_j (i \neq j)\) and that the evidence about \(KN\) 
opens (Method I) or blocks (Method II) the communication between \(KN_i (i = 1, \ldots, n)\). It is 
not clear which of the two Methods in models belong-to relationships is better (Millán & 
De-la Cruz, 2002).

Examples of both methods can be found in the ITS literature. Method (I) has been 
chosen by Van Lehn and his team for the ANDES (Conati et al., 1997), and also MEADA 
(Trella et al., 2005). Method (II) has been chosen by Mislevy and Gitomer in their 
HYDRIVE system (Mislevy & Gitomer, 1996), and also by Murray in his Desktop 
Associate (Murray, 1999). BiTutor has chosen Method I for the following reasons:

i. In models with Method (I), the students learn in an incremental way. That is, when a 
student learns a topic, the usual way is to study each of the parts that compose the 
topic. On the other hand, in Method (II), information about the student’s mastering 
state of a sub-topic will tend to affect the tutor belief of his mastery of other sub-topics 
which belong to the same topic. This is incompatible to the incremental method of 
learning.

ii. As for parameter specification, Method (I) requires an exponential number of 
parameters instead of the polynomial number required by Method (II). From the 
teacher’s point of view, it is easier to calculate the probability from historical data
using Method I. Moreover, since a KN is composed to limited number of KNs, the
task of parameter specification is manageable.

For example, from "Elementary Data Structure" course, the relationships between
Analysis of Algorithms, Array, Linked List, Stack, Queues, and Trees and Binary tree etc.
nodes (topic) and Elementary Data Structure node (course) are represented as belong to
relationships. When a student wants to master "Elementary Data Structure" course, he has
to master all the topics which belong to this course. Similarly, if the student wants to master
the Stack topic, he has to master all subtopics, and concepts which belong to it.

**Modeling Pre-requisite Relationships**

For tutoring by human, the pre-requisite relationship is clearly a very important one, both
for instructional planning purposes and for gathering information about the current state of
student's knowledge (Reye, 2004). If \( A \) is a pre-requisite of \( B \), knowing \( A \) must have causal
influence in knowing \( B \), so the correct direction of the link is \( A \rightarrow B \), this means, firstly, a
student's lack of knowledge of \( A \) implies lack of knowledge of \( B \). Secondly, evidence of a
student's knowledge of \( A \) can be taken as evidence for revising our belief that the student
also has knowledge of \( B \). There are two different types of pre-requisite relationships,
namely, pre-requisite relationship in the same level, and pre-requisite relationships between
two different levels. Figure 3.6 shows a domain knowledge model of a course divided in
topics, subtopics, and concepts, where belong-to relationships are represented in blue color,
and pre-requisite relationships are represented in green color.
Pre-requisite Relationships in the Same Level

BiTutor uses this relationship in the topic, and subtopic levels only in the hierarchy representation. For example, in Figure 3.6, a topic $T_1$ is a pre-requisite for $T_2$ that means a student must reach master level on topic $T_1$ before he/she starts to learn topic $T_2$. Again, in Elementary Data Structure, understanding “Stack Operations” subtopic is a pre-requisite for understanding the “Stack Application” subtopic. If we encounter a student who does not know “Stack Operation”, and then it is more likely that the student does not know about “Stack Application”. Note, BiTutor assumed no pre-requisite relationship for the concept level.

Pre-requisite Relationships in Different Levels

BiTutor uses this relationship between the topic and subtopic levels only in the hierarchy representation. For example, in Figure 3.6, a subtopic $ST_2$ is a pre-requisite for $T_n$, that means, a student must reach master level on subtopic $ST_2$ before he/she starts to learn topic $T_n$. Again, in Elementary Data Structure, understanding “Stack Implementation using Linked List” subtopic, which belongs to the “Stack Implementation” subtopic, which comes
under “Stack Operations” subtopic, and this belongs to Stack topic, needs to understand the “Singly Linked List” subtopic, which belongs to “Linked List” topic.

When the teacher wants to create a Bayesian network for a particular course, all the nodes in the different levels of the hierarchy must be added, and all types of relationships (pre-requisite and belong to) must also be added. Using a single Bayesian network to represent the domain knowledge for a particular course will be a very complex task from teacher's point of view and system efficiency. Teachers will be very comfortable to work with each topic as a separate unit, by applying directed-dependent separation (or d-separation refers definition 3.1) (Pearl, 1998). Bayesian network for a particular course could be decomposed into a set of Bayesian networks, each Bayesian network containing one topic, its related subtopics and concepts, pre-requisite and belong to relationships, and its related questions, which are hidden as discussed in Chapter 4.

**Definition 3.1 (Decompose of Bayesian network using d-separation)**

Given a Bayesian network on a particular course, and three disjoint subsets of knowledge nodes ($K_{Na}$, $K_{Nb}$, and $K_{Nc}$) of the Bayesian network's $KN$, $K_{Na}$ are said to d-separate $K_{Nb}$ and $K_{Nc}$ if and only if along every undirected chain from each node in $K_{Na}$ to each node in $K_{Nb}$, $K_{Nc}$ is not a head-to-head node in the chain.

**Example**

From the syllabus book, the list of topics can be obtained, and its related subtopics and concepts can be obtained from the list of reference books for the particular course. For example, consider the topic “Stack”, different types of nodes for this topic are shown in Table 3.2.
Table 3.2: Knowledge hierarchy on “Stack” topic.

<table>
<thead>
<tr>
<th>Name of the knowledge node</th>
<th>Node ID</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stack</td>
<td>C_1</td>
<td>Topic</td>
</tr>
<tr>
<td>Stack Operations</td>
<td>C_2</td>
<td>Subtopic</td>
</tr>
<tr>
<td>· Push and Pop Operations</td>
<td>C_4</td>
<td>Concept</td>
</tr>
<tr>
<td>Stack Implementations</td>
<td>C_5</td>
<td>Subtopic</td>
</tr>
<tr>
<td>· Stack Implementation using Arrays</td>
<td>C_6</td>
<td>Subtopic</td>
</tr>
<tr>
<td>· Single Dimension Array</td>
<td>C_8</td>
<td>d-separation</td>
</tr>
<tr>
<td>· Push and Pop in Array Stack</td>
<td>C_9</td>
<td>Concept</td>
</tr>
<tr>
<td>Stack Implementation using Linked List</td>
<td>C_7</td>
<td>Subtopic</td>
</tr>
<tr>
<td>· Singly Linked List</td>
<td>C_{10}</td>
<td>d-separation</td>
</tr>
<tr>
<td>· Push and Pop in Linked Stack</td>
<td>C_{11}</td>
<td>Concept</td>
</tr>
<tr>
<td>Stack Applications</td>
<td>C_3</td>
<td>Subtopic</td>
</tr>
<tr>
<td>· Recursive Programming</td>
<td>C_12</td>
<td>Concept</td>
</tr>
<tr>
<td>· Expression</td>
<td>C_13</td>
<td>Subtopic</td>
</tr>
<tr>
<td>· Balancing</td>
<td>C_14</td>
<td>Concept</td>
</tr>
<tr>
<td>· Evaluation</td>
<td>C_15</td>
<td>Concept</td>
</tr>
<tr>
<td>· Conversion</td>
<td>C_16</td>
<td>Concept</td>
</tr>
</tbody>
</table>

The teacher can easily draw the Bayesian network using the different types of knowledge nodes in the hierarchical domain as the random variables. Figure 3.7 illustrates the Bayesian network for “Stack”, where the green color relationships denote the prerequisite relationships, and the blue relationships denote the belong-to relationships. The Bayesian networks for other topics of “Elementary Data Structure” course can similarly be constructed. Appendix A shows the Bayesian networks for the topics “Array” and “Linked List” of “Data Structure”, which are used to demonstrate BiTutor.
Bayesian student modeling satisfies Wenger’s (1987) three levels of information that the student modeling might address in an Intelligent Tutoring System. The behavioral level is worried with the correctness of the student behavior as compared with the expert model. In this case, it is the comparison of the student response to the expert’s solution to problems. This can be done easily with the object test item. Using the multiple-choice questions, the student’s selection can be used as evidence of his mastery level(s) of the relevant knowledge node(s). The epistemic level is worried with the student’s particular knowledge states. The probability distribution of the topic node and its belong-to subtopic in the Bayesian network represent the information about the epistemic level. The individual level addresses detailed assertions about the individual that transcend particular problem states. The concept nodes in the Bayesian network represent the components of knowledge that
differentiates individual students. The suitable tutoring actions can be taken based on information at individual level (to be described in Chapter 4).

3.2 Completion of Bayesian Network Parameters

A decision theoretic perspective given by Henrion et al. (1991) state that "It is clear that no scheme for reasoning and decision making under uncertainty can avoid making assumptions about prior beliefs and independence, whether these assumptions are implicit". Knowledge, whether expert judgment or from empirical data, may be expressed in which direction is most natural, generally, but not necessarily in causal direction. Probabilities algorithms use this form to reason in whichever direction required: causal or diagnostic.

![Bayesian Network Diagram](image)

**Legend**
- **NM**: Non-Mastery state
- **PM**: Partial Mastery state
- **M**: Mastery state
- **C₆**: Stack Implementation using Array, Subtopic
- **C₈**: Single Dimension Array, Concept
- **C₉**: Push and Pop in Array Stack, Concept

**Tables**

<table>
<thead>
<tr>
<th>Given</th>
<th>C₆</th>
<th>NM</th>
<th>PM</th>
<th>M</th>
<th>NM</th>
<th>PM</th>
<th>M</th>
<th>NM</th>
<th>PM</th>
<th>M</th>
<th>NM</th>
<th>PM</th>
<th>M</th>
<th>NM</th>
<th>PM</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>C₆</td>
<td>NM</td>
<td>0.800</td>
<td>0.550</td>
<td>0.550</td>
<td>0.350</td>
<td>0.100</td>
<td>0.100</td>
<td>0.350</td>
<td>0.100</td>
<td>0.001</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C₆</td>
<td>PM</td>
<td>0.100</td>
<td>0.350</td>
<td>0.100</td>
<td>0.550</td>
<td>0.800</td>
<td>0.550</td>
<td>0.100</td>
<td>0.350</td>
<td>0.001</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C₆</td>
<td>M</td>
<td>0.100</td>
<td>0.100</td>
<td>0.350</td>
<td>0.100</td>
<td>0.100</td>
<td>0.350</td>
<td>0.550</td>
<td>0.550</td>
<td>0.800</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Fig 3.8:** A subset of the Bayesian network on "Stack" with the tables of the probability values.

Figure 3.8 shows a subset of the Bayesian network on mastery of the subtopic "Stack Implementation using Array", which is under "Stack Implementation". As explained in...
nodes section (3.1.1), each node has three knowledge states: non-mastery, partial mastery, and mastery. The granularity of the Bayesian network depends on the number of nodes and its states. The number of entries in the conditional probability table grows exponentially if the model becomes fine grained.

Values at the root nodes are known as unconditional probabilities while the other nodes are conditional probabilities. In case of BiTutor, the concept nodes are unconditional probabilities and other nodes (subtopics and topic) are conditional probabilities. To use the Bayesian network, the random variable must be initialized with prior probability values. These values may be based on teacher's belief or from observed data.

3.2.1 Prior Probability from Teacher's Belief

A variety of techniques for estimating prior probabilities have been developed by decision analysts (Morgan & Herion, 1990; Spetzler & Holstein, 1975). Methods are available for assessing continuous distributions and discrete distributions. The simplest method is one where the assessor needs to make only qualitative judgment which is more apparent, the event of interest or some reference event of agreed probability. For providing the reference event, a popular method could be the use of Probability Wheel. Probability Wheel is a simple graphical device which consists of a colored disk divided into various sectors. The angle of these sectors visually represents the probability of the reference event. The angle is adjusted to accommodate the probability adjustments needed. Using this wheel, a probability can be obtained without explicitly mentioning a number. The teachers may give numeric probabilities directly once they gain experience with assigning probability. For extremely low or high probabilities, techniques using odds or log-odds scales are found to be useful. (Winterfeldt & Edwards, 1986).
The probability wheel can also be used to estimate the probabilities of the three mastery states for various difficulty categories of a concept. Table 3.3 illustrates the probability values given by a teacher.

Table 3.3: Seven difficulty levels of a typical knowledge node.

<table>
<thead>
<tr>
<th>Difficulty level</th>
<th>Non-Mastery</th>
<th>Partial Mastery</th>
<th>Mastery</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very easy</td>
<td>0.001</td>
<td>0.009</td>
<td>0.990</td>
</tr>
<tr>
<td>Easy</td>
<td>0.010</td>
<td>0.090</td>
<td>0.900</td>
</tr>
<tr>
<td>Fairly easy</td>
<td>0.050</td>
<td>0.150</td>
<td>0.800</td>
</tr>
<tr>
<td>Average</td>
<td>0.100</td>
<td>0.200</td>
<td>0.700</td>
</tr>
<tr>
<td>Fairly difficult</td>
<td>0.200</td>
<td>0.300</td>
<td>0.500</td>
</tr>
<tr>
<td>Difficult</td>
<td>0.300</td>
<td>0.400</td>
<td>0.300</td>
</tr>
<tr>
<td>Very difficult</td>
<td>0.400</td>
<td>0.500</td>
<td>0.100</td>
</tr>
</tbody>
</table>

Other teachers can also do the same practice and their average results can be used in making judgment for the prior probability for the different levels of difficulty. While constructing the Bayesian network, the prior probability values for non-mastery, partial mastery and mastery of the concept can be selected from the Table – based on its difficulty category. Alternatively, teachers can make use of the visual tools available for probabilistic models. One such user interface tool for graphical probabilistic models was proposed by Wang and Drudzdlz (2000). The flexible navigation and visualization of probability distributions help to detect unspecified probabilities and inconsistency in responses.

An extensive literature on human judgment has identified some of the factors distorting human judgments about uncertain events. Cognitive biases and mental heuristics are identified as some of the factors distorting the human judgments about uncertain events. (Kahneman et al., 1982). One common bias (Morgan & Henrion, 1990) is the tendency to underestimate uncertainty, assessing probabilities that are nearer to 1 or 0 than is appropriate. To counteract these effects of these biases, some methods have been developed.
by decision analysts: De-biasing techniques include attempts to make all assumptions explicit, and attempts to consider extreme possibilities and unexpected outcomes.

3.2.2 Prior Probability from Observed Data

To obtain the prior probabilities from empirical/observed data, there are two techniques: parametric and non-parametric. Parametric statistical techniques are based on very precise and restrictive assumptions about the characteristics of the measurement populations and the measurement samples being investigated (DeGroot & Schervish, 2002). Required features of the populations studied, such as the nature of their parameters and the shapes of their distributions, are stated by these assumptions. These assumptions also indicate the type of the samples that must be taken.

The probabilistic characteristic of a random phenomenon is sometimes difficult to define. For example, the appropriate probability model needed to describe these characteristics is not readily amenable to theoretical deduction or formulation. In particular, the functional form of the required probability distribution may not be easy to derive or ascertain. The basis or properties of the physical process may suggest the form of the required distribution, under certain circumstances. For example, assume that a process is composed of the sum of many individual effects. Then, the Gaussian distribution may be appropriate on the basis of the central limit theorem (Ang & Tang, 1975). However, there are occasions where the required probability distribution has to be determined empirically. Suppose, if the frequency diagram of a set of data can be constructed, a visual comparison on a density function with the frequency diagram can be done to determine the required distribution model. Therefore, non-parametric techniques are testing hypotheses (Devore, 2000) about population distributions rather than about population parameters.
In BiTutor, parametric technique is used to obtain the prior probability of values of the Bayesian network based on students’ response to test item, as described below.

**Procedure 3.1 (Obtaining prior probability values based on students’ response to test items)**

i. For each concept, provide some questions.

ii. Record student’s response for each question that he has answered.

iii. Compute average number of students for each mastery state for each concept.

Consider the test item (question) given in Figure 3.9, which is used to test the mastery of the concept “Expression Conversion” in “Stack” topic. A class of 61 MCA (Master of Computer Applications) students has answered this question during the mid-semester exam. Their responses are noted down and based on that calculations are done. As mentioned earlier in section, we assume the mastery states to be NM, if $0 \leq E(KN) \leq 0.39$, PM if $0.40 \leq E(KN) \leq 0.79$, and M if $0.80 \leq E(KN) \leq 1.0$. In the figure given below, the correct solution is given and so the mastery state is M. Likewise, evidence is collected from all the students and their mastery states are summarized in Table 3.4. From the table, we can assign the prior probability for the concept “Expression Conversion” in “Stack” topic for various mastery states: $P(KN="NM") = 0.148$, $P(KN="PM") = 0.524$, and $P(KN="M") = 0.328$. 

Student Model 67
Write in step wise, the equivalent post fix of the infix expression \( a + b \times c - d \)?

**Solution:**

<table>
<thead>
<tr>
<th>Expression</th>
<th>Sub expression chosen based on rules of hierarchy, precedence and associatively</th>
<th>Postfix expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a + b \times c - d)</td>
<td>(b \times c)</td>
<td>(1): (bc^*)</td>
</tr>
<tr>
<td>(a + (1) - d)</td>
<td>(a + (1))</td>
<td>(2): (abc^* + d)</td>
</tr>
<tr>
<td>(2 - d)</td>
<td>((2) - d)</td>
<td>(2): (abc^* + d)</td>
</tr>
</tbody>
</table>

Hence \(abc^* + d\) is the equivalent post fix expression of \(a + b \times c - d\).

Fig 3.9: A solution to item testing “Expression Conversion” in “Stack” topic.

Table 3.4: Students’ response to the question shown in Figure 3.9.

<table>
<thead>
<tr>
<th>No. of Student</th>
<th>Non-Mastery</th>
<th>Partial Mastery</th>
<th>Mastery</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage</td>
<td>14.8%</td>
<td>52.4%</td>
<td>32.8%</td>
<td>100.00%</td>
</tr>
</tbody>
</table>

### 3.2.3 Conditional Probabilities from Teacher's Belief

Associations such as conjunctive or disjunctive relationships among the diverse influences are incorporated in the conditional probability assignments. The specification of the parameters becomes difficult if there is more number of associations for a knowledge node. For example, in the network given in Figure 3.6, for node \(T_2\) the parameter needed is the conditional probability \(P(T_2|T_1, ST_1, ST_2)\). But the fact that different types of relationships (pre-requisite and belong to) are mixed in the conditioning distribution makes this probability difficult to estimate, and even in some cases it seems that the meaning of such events is unclear (for example, we would need to provide the probability of knowing a topic \(T_2\) given that its parts \(ST_1\) and \(ST_2\) are known and its pre-requisite \(T_1\) is unknown).
In the case of BiTutor, weighted approach is used to lessen the difficulties in computing the conditional probabilities. While constructing the network, it will be easy for the staff to mention the weights for the relationship rather than to set the CPT values manually. The manner in which the weights can be fed to BiTutor is described in Chapter 5.

**Definition 3.2 (Weighted Approach)**

For each $\text{KN}_q$ node with parents $\text{pa}(\text{KN}_q)$ in the Bayesian network, the conditional probability is computed as following:

If the states of $\text{KN}_q$ and $\text{pa}(\text{KN}_q)$ are same, then

$$P(\text{KN}_q | \text{pa}(\text{KN}_q)) = \sum_{\text{pa}(\text{KN}_q)} (W - (c - 1)\kappa)$$

else

$$P(\text{KN}_q | \text{pa}(\text{KN}_q)) = \sum_{\text{pa}(\text{KN}_q)} \kappa$$

(3.4)

where $\kappa$ is an empirical constant, $c$ is the number of states, and $W$ is the weight of the parent to child node with respect to the type of relationships (pre-requisite and belong-to).

Since in BiTutor model, both the relationships are supported in the same layer, the weight $W$ is represented as $W = (\alpha W_1, \beta W_2)$, where $W_i = (W_{i1}, W_{i2}, \ldots, W_{in})$ are the weights in a belong-to relationship, $n$ is the number of belong-to relationships, and $\Sigma W_{il} = 1$, $W_2 = (W_{21}, W_{22}, \ldots, W_{2m})$ are the weights in a pre-requisite relationships, $m$ is the number of pre-requisite relationships, and $\Sigma W_{2l} = 1$, and $\alpha$ and $\beta$ are two constants such that $0 < \alpha$, $\beta < 1$, $\alpha \leq \beta$ and $\alpha + \beta = 1$. $\alpha$ is always kept less than $\beta$ since the pre-requisite relationships are to be given more preference than the belong-to relationships. As pre-requisite must be satisfied before starting a particular knowledge node, they should be given more preference.

In Figure 3.8, the CPT of C9 is computed using the weighted approach (Equation 3.4). Here, $\kappa = 0.05$, $\alpha = 0.6$, $\beta = 0.4$, $W_{8,6} - 1$ is a pre-requisite relationship between C8 and C6.
and $W_{i,j} = I$ is a belong-to relationship between $C_j$ and $C_i$. Another example of the CPT for a knowledge node with 3 parent knowledge nodes is shown in Appendix B.

### 3.2.4 Conditional Probabilities from Observed Data

When there is sufficient data from past students’ responses to items that test the mastery level of several related knowledge nodes, statistical estimates for the conditional probabilities of these knowledge nodes are possible.

### 3.3 Inference

The basic task for any probability inference system is to compute the posterior probability distribution for a set of query nodes, given values for some evidence nodes. This task is called belief updating or probabilistic inference. Inference in Bayesian networks is very flexible, as evidence can be entered about any node and beliefs in any other nodes are updated.

The most common exact inference methods are *variable elimination* (Shafer & Shenoy, 1990), which eliminates (by integration or summation) the non-observed non-query variables one by one by distributing the sum over the product; and *clique tree propagation* (Jenson, 1988; Jenson et al., 1990; Lauritzen & Spiegelhalter, 1988), which caches the computation so that many variables can be queried at one time and new evidence can be propagated quickly. All these methods have complexity that is exponential in the network’s treewidth. The most common approximate inference algorithms are stochastic MCMC simulation, and *mini-bucket elimination* which generalizes loopy belief propagation, and variation methods (Russell and Norvig, 2003). Clique tree propagation is used in BiTutor.
3.3.1 Updating Bayesian Network

A Concept is the smallest unit of knowledge in BiTutor. Each item can test the student’s mastery of one or more knowledge nodes. The item is used to collect the evidence. Presently, in BiTutor, the evidence is collected from multiple choice questions. There are two types of response from the student for a test item, a correct response \((x = 1)\) or an incorrect \((x = 0)\) response. Apart from student’s response to the question, his mastery state and the item difficulty level are also considered when BiTutor instantiated the knowledge node.

BiTutor defines the following formula to calculate the new expected mastery state for the student in that particular knowledge node.

\[
EMV^k = \begin{cases} 
\min(EMV^{k-1} + \phi(\frac{b}{L}), 1) & x = 1 \\
\max(0, EMV^{k-1} - \phi(\frac{L - 1 - b}{L})) & x = 0 
\end{cases}
\]  

(3.5)

where \(0 \leq \Theta \leq 1\) is a constant corresponding to the rate of learning update, \(k\) is the index of the updating iteration, \(b, \ 0 \leq b \leq L\) is the difficulty level and \(L\) is the total number of difficulty levels. \(EMV\) is the expected mastery value for a student in knowledge node, bounded between \(0\) and \(1\). After the new expected mastery is computed for the tested concept, the knowledge node takes one value from its states based on the value of expected mastery. The knowledge node is instantiated and the evidence is propagated through Bayesian network using any inference methods. Table 3.5 shows the example of new student mastery state after a student has responded correctly \((x=1)\) to a question with 5 difficulty levels, assuming that current \(EMV=0.30, EMV=0.65,\) and \(\Theta=0.25\).
Table 3.5: Effects of a correct answer \((x=1)\) on the mastery state of two different EMVs.

<table>
<thead>
<tr>
<th>Description</th>
<th>Level</th>
<th>New EMV</th>
<th>Remarks</th>
<th>New EMV</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very easy</td>
<td>1</td>
<td>0.35</td>
<td>Remains in NM</td>
<td>0.65</td>
<td>Remains in PM</td>
</tr>
<tr>
<td>Easy</td>
<td>2</td>
<td>0.40</td>
<td>Promotes to PM</td>
<td>0.70</td>
<td>Remains in PM</td>
</tr>
<tr>
<td>Average</td>
<td>3</td>
<td>0.45</td>
<td>Promotes to PM</td>
<td>0.75</td>
<td>Promotes to PM</td>
</tr>
<tr>
<td>Difficult</td>
<td>4</td>
<td>0.50</td>
<td>Promotes to PM</td>
<td>0.80</td>
<td>Promotes to M</td>
</tr>
<tr>
<td>Very difficult</td>
<td>5</td>
<td>0.55</td>
<td>Promotes to PM</td>
<td>0.85</td>
<td>Promotes to M</td>
</tr>
</tbody>
</table>

3.3.2 Posterior Belief of Student's Mastery State

Until the decision to stop tutoring is encountered, the evidence collection sequence, instantiation of evidence variables, belief propagation, and drawing conclusions for the query variables are repeated. Bayesian network allows one to calculate the conditional probabilities of the query nodes in the network given that the values of some of the nodes have been observed. Figure 3.10 is a Bayesian network for “Stack” topic represented using BNJ (Bayesian Network Tools in Java, 2004). Each random variable has three states, namely NM (Non-Mastery), PM (Partial Mastery), and M (Mastery). The initial probability values for each variable are shown in this Figure.
In Figure 3.11, the posterior probability values (belief) of the student’s mastery for the concept “Push and Pop Operations” C4 associated with the sub-topic “Stack Operations” C2 of the topic “Stack” C1 are given. The evidence is collected from the item that tested the student’s mastery on the concept C4. As the student responds correctly to the item, the tutor’s belief for the student’s mastery of that concept has increased.

Some inferences that can be drawn from the Bayesian network:

i. Belong-to relationship: C4 belongs to C2, which belongs to C1. As the student has mastered C4, the tutor’s belief on the student’s mastery of C2 and C1 has also increased.

ii. Pre-requisite relationship: C4 is a pre-requisite relationship for “Stack Implementations” C5. As the student has mastered C4, the tutor’s belief on student’s
mastery of C5 also increased and C5 can be tutored to the student. Since C2 is updated, “Stack applications” C3 whose pre-requisite is C2 also gets updated.

iii. Independent nodes: “Balancing” C14 which belongs to “Expression” C13 does not depend on C4. So, there is no change in belief about the mastery state for C14.

Fig 3.11: State of Bayesian network after instantiating the node “Push and Pop Operations” C4.