systems-120526-2.docx

WORD COUNT 7591
TIME SUBMITTED 11-MAR-2016 02:16PM
PAPER ID 21958777

Systems **2015**, *3*, 1-x manuscripts; doi:10.3390/systems30x000x

OPEN ACCESS

systems

ISSN 2079-8954

www.mdpi.com/journal/systems

Review

Mathematical Approaches to User Modeling

Loc Nguyen [1]

[1] Sunflower Soft Company, An Giang, Vietnam; E-Mail: ng_phloc@yahoo.com

Academic Editor:

Received: / Accepted: / Published:

Abstract: User model is description of users' information and characteristics in abstract level. User model is very important to adaptive software which aims to support user as much as possible. The process to construct user model is called user modeling. Within learning context where users are learners, the research proposes a so-called Triangular Learner Model (TLM) which is composed of three essential learners' properties such as knowledge, learning style, and learning history. TLM is the user model that supports built-in inference mechanism. So the strong point of TLM is to reason out new information from users, based on mathematical tools. This paper focuses on fundamental algorithms and mathematical tools to construct three basic components of TLM such as knowledge sub-model, learning style sub-model, and learning history sub-model. In general, the paper is a summary of results from research on TLM. Algorithms and formulas are described by the succinct way.

Keywords: user model; user modeling; adaptive learning; knowledge; learning style; learning history

---

1. Introduction

User model is collection of user's properties such as knowledge, characteristics, interests, background, and demography. Information is wide and importance level of such information varies with regard to context, for example, knowledge, learning styles, goals, and backgrounds are the most important in learning context - the main context of the research. Users also depend on context and so, users are learners and customers in learning context and commercial context, respectively. Terms such as user model, learner model, and student model have the same meaning in the research.

System that builds up and manages user model is called user modeling system (UMS); please imagine that user model is similar to database and user modeling system is similar to database management system (DBMS). User modeling system provides user information to adaptive system so that adaptive system gives the best service to users. In e-learning, adaptive system supplies learner teaching methods, courses, lectures, tests, and exercises which are appropriate to each individual, based on her/his personal information provided by user modeling system. Figure 1 depicts relationship between user modeling system and adaptive system.

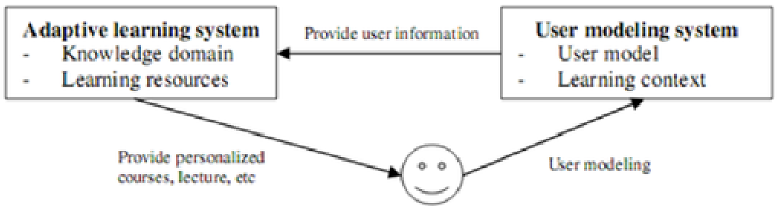

Figure 1. User modeling system and adaptive system.

According to (Fröschl, 2005, pp. 59-74) and (Kobsa, 2007), existing user modeling systems are classified into four groups:

- User modeling shell is now outdated because it is component attached to adaptive system and so it is not specialized user modeling system.
- User modeling server works in specialized and independent manner as database management systems but it is restricted on inferring new potential information from user model.
- Agent-based user modeling system works by interaction of agents, therefore each agent manages a piece of user information. Agents are defined as software modules that operate independently and interact together. Agent-based user modeling system is distributed system.

Each user modeling system has own strong points and drawbacks but in general, these systems focus on information storage and provision; so there is lack of strong inference mechanism. Modern user modeling systems differ from database management system by ability of inference; collective user information is not much enough to need of adaptive systems and so it is required to exploit new potential information from existing information in user model. Moreover, it is necessary to choose essential users' attributes in learning context because information about user is very much; we cannot describe and model all of such information. Therefore, the research has two objectives:

1. The first objective is to build up a user modeling system incorporated with inference mechanism based on solid algorithms and mathematical tools, which aims to draw new potential information about users. The preeminence of this research is to provide mathematical approaches to user modeling study.
2. The second objective is to select and model the most essential characteristics about learners in learning context because the accuracy of inference mechanism depends on the importance level of information. That information is too much or less important will decrease performance of inference process. So the second objective supports the first objective.

2. Architecture of proposed user modeling system

This research proposes a learner model that consists of three essential kinds of information about learners such as *knowledge* (K), *learning style* (LS) and *learning history* (LH) which are fine-tuned from a lot of user information. Such three characteristics are both mutually independent and coherent in order to form a triangle and so this learner model is called *Triangular Learner Model* (TLM). TLM will cover the whole of user's information required by learning adaptation process and give the best support to adaptive learning. The reasons for such assertion are:

- Knowledge, learning styles and learning history are prerequisite for modeling learner.
- While learning history and knowledge change themselves frequently, learning styles are relatively stable. The combination of them ensures the integrity of information about learner.
- User knowledge is domain specific information and learning styles are personal traits. The combination of them supports user modeling system to take full advantages of both domain specific information and domain independent information in user model.

Figure 2 shows the TLM.

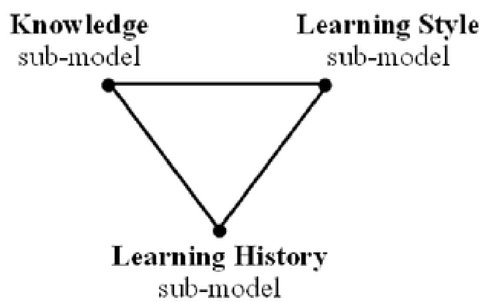

Figure 2. Triangular Learner Model (TLM).

Knowledge, learning style and learning history are *sub-models* of TLM. TLM is designed according to triangular model so that it is easy to describe and construct other user information such as interests and goals. In other words, TLM can be extended to interpret about learner in more detailed by attaching more learners' characteristics such as interests, background, goals, etc. into the learning history sub-model. So learning history sub-model is the most important one, which is used to initialize two other sub-models such as knowledge and learning style. Figure 3 shows extended TLM.

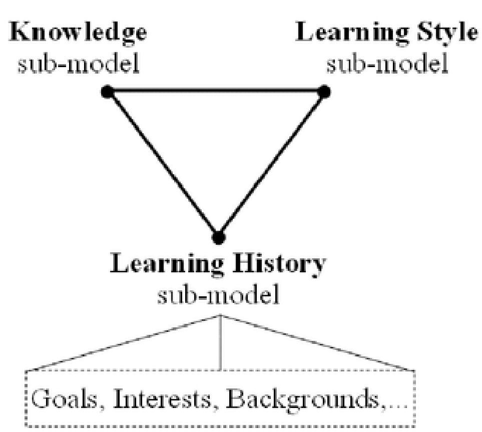

Figure 3. extended Triangular Learner Model.

TLM finishes the second objective and the first objective is achieved by the proposal of a user modeling system called *Zebra*: constructing TLM, managing TLM, and inferring new information from TLM. The core of Zebra is the composition of two engines: *mining engine* and *belief network engine*.

- Mining engine (ME) is responsible for collecting learners' data, monitoring their actions, structuring and updating TLM. ME totally uses mining techniques to perform modeling tasks. ME is very important when it manages whole TLM but it focuses mainly on learning history sub-model. ME also provides input information to belief network engine. ME has three other important functionalities that are to discover some other characteristics beyond knowledge and learning styles (like interests, goals, learning context), to support learning concept recommendation, and to support collaborative learning.
- Belief network engine (BNE) is responsible for inferring new personal traits from TLM by using deduction mechanism available in belief network. This engine applies both Bayesian network and hidden Markov model into its tasks. Two sub-models: knowledge and learning style are managed by this engine. BNE does not work as actively as ME but it is more complicated than ME.

Zebra also provides *communication interfaces* (CI) that allows users and adaptive systems to see or modify restrictedly their TLM. Because user modeling system always interacts with adaptive system and TLM cannot be modified directly from outside, CI operates as communication port. Figure 4 depicts the architecture of Zebra.

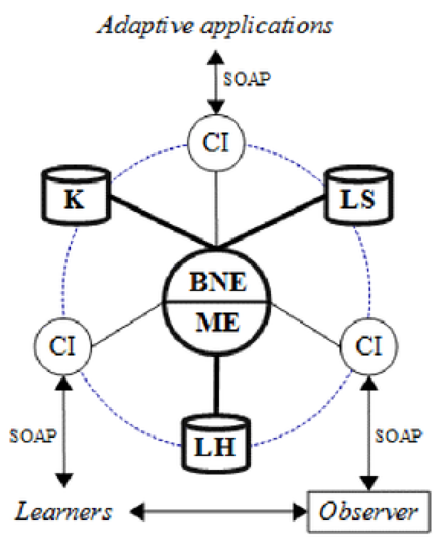

Figure 4. Architecture of Zebra.

Please see (Nguyen, ZEBRA: A new User Modeling System for Triangular Model of Learners' Characteristics, 2009) for *Zebra* architecture in more detailed. Figure 5 shows the adaptive process, as follows:

1. Adaptive system sends user information query to CI via protocols: SOAP, HTTP, RMI, etc.
2. CI analyzes query request into input parameters and sends such parameters to respective ME or BNE.
3. ME or BNE executes query according to specified parameters and sends results back CI.
4. CI sends results back adaptive system.

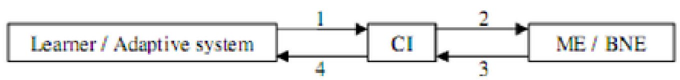

Figure 5. Adaptive process of Zebra.

3. Three sub-models of TLM

The previous section gives us a general architecture of Triangular Learner Model (TLM) and its user modeling system Zebra. TLM is composed of three sub-models such as knowledge sub-model, learning style sub-model and learning history sub-model. This section is the most detailed one that describes thoroughly these sub-models together with their attributes, constructing method and inference mechanism.

3.1. Knowledge sub-model

Knowledge is study result, which is very important to adaptive system and so it is evaluation of adaptive system. A preeminent adaptive system will improve learners' knowledge after they study on on-line course via such adaptive system. The research gives a proposal of knowledge sub-model along with ways to construct and enhance knowledge sub-model.

The research proposes the combination of overlay model and Bayesian network. In overlay model, domain is decomposed into a set of knowledge elements and each element is a lecture, exercise or test and represented by an integer number that measures user knowledge. Each learner is modeled as a subset of masteries over those elements and such subset is called overlay model. Bayesian network (Neapolitan, 2003, p. 40) is a directed acyclic graph (DAG) composed of a set of nodes and a set of directed arcs. Each arc indicates the relationship between two nodes and the strength of this relationship is quantified by conditional probability table (CPT). The inference mechanism in Bayesian network is based on Bayes' theorem. The combination between overlay model and Bayesian network is done through following steps:

1. The structure of overlay model is translated into Bayesian network; each user knowledge element becomes a node in Bayesian network. Evaluation knowledge such as test or exercise becomes evidence node and normal knowledge such as lecture or lesson becomes hypothesis node or hidden node.
2. The aggregation relationship between domain elements in overlay model becomes a conditional dependence assertion signified by CPT of each node in Bayesian network.

Following is an example of combination between overlay model and Bayesian network. Suppose Java course for students is constituted of 6 knowledge elements:

- A top-most course *Java* (J) and three lectures: *control structure* (C), *class & object* (O), *interface* (I).
- An exercise: *set up class Human* (E).
- A test that students need answer accurately more than 10 answers: *Questions* > 10 (Q).

The problem needs solved is that how much knowledge a student gains after she/he finished Java course when student knowledge is marginal posterior probability of node J with $J = 1$, denoted $P(J=1)$. Figure 6 depicts the combination of Bayesian network and overlay model where each arc is weighted.

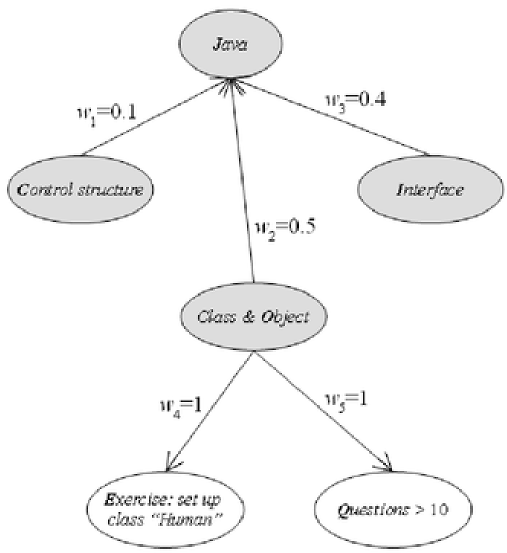

Figure 6. Combination of Bayesian network and overlay model.

Note that J, C, O and I are four hidden nodes (shaded) while E and Q are two evidence nodes (not shaded). Because the strength of each arc quantified by a CPT is not determined yet, it is impossible to calculate the marginal posterior probability. I proposes two-step process to determine CPT:

- Each arc is assigned by the weight representing importance level of a lesson, lecture, test or exercise.
- Sigma formula mentioned below is used to create CPT based on these weights.

Suppose weights of lectures *control structure* (C), *class & object* (O), *interface* (I) are $w_1=0.1$, $w_2=0.5$, $w_3=0.4$, respectively. Lecture *class & object* (O) is the most important, in consecutive, there are *interface* (I) and *control structure* (C). The weights of exercise E and test Q are $w_4=1$ and $w_5=1$, respectively. The CPT of Java course is determined via following Sigma formula:

$$P(J|C,O,I) = w_1 * h_1 + w_2 * h_2 + w_3 * h_3$$

Where,

$$h_1 = \begin{cases} 1 \; if \; C = J \\ 0 \text{ otherwise} \end{cases}, h_2 = \begin{cases} 1 \; if \; O = J \\ 0 \text{ otherwise} \end{cases}, h_3 = \begin{cases} 1 \; if \; I = J \\ 0 \text{ otherwise} \end{cases}$$

Note that all nodes are random binary variables which get values 0 and 1. For example, we have:

$$P(J = 1|C = 1, O = 1, I = 1) = 1 * P(C = 1) + 1 * P(O = 1) + 1 * P(I = 1)$$
$$= 1 * 0.1 + 1 * 0.5 + 1 * 0.4 = 1.0$$

Please see (Nguyen & Do, 2009) for more details in Bayesian overlay model. Figure 7 depicts the Bayesian overlay model specified completely with full of CPT (s).

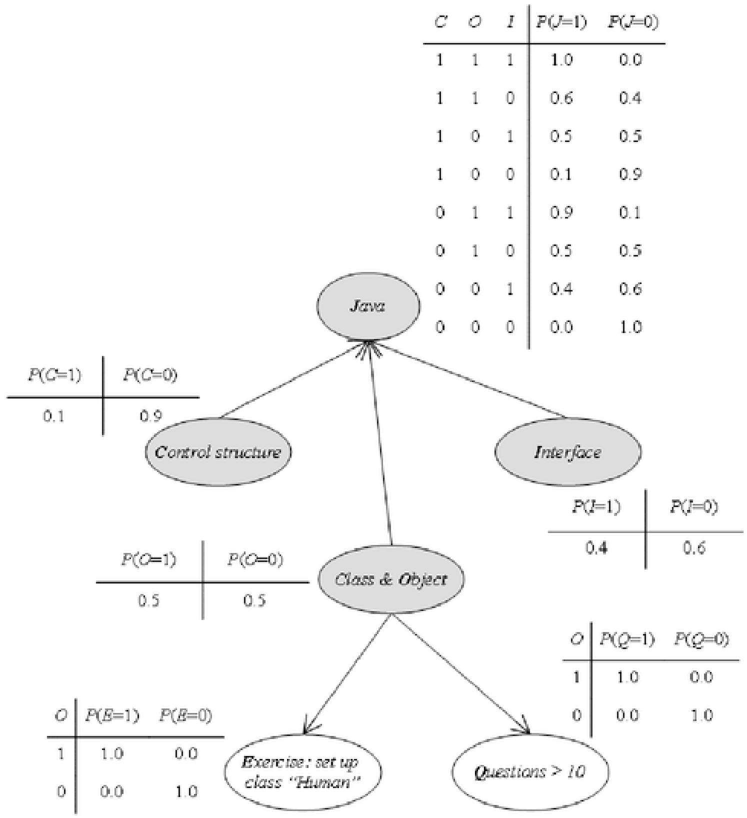

Figure 7. Bayesian overlay model with full of CPT (s).

When Bayesian network is specified completely with full of CPT (s), it is easy to compute the posterior marginal probability $P(J=1)$ representing user's mastery over Java course. Knowledge sub-model is improved by two methods:

- *Parameter learning*: CPT (s) - Bayesian parameters, determined via the combination of weight specification and Sigma formula, needs enhanced after Zebra operates in long time process. This is a kind of parameter learning. The research applies beta density function into parameter learning (Neapolitan, 2003, pp. 293-373) and proposes maximum likelihood estimation (MLE) algorithm for learning parameters.
- *Structure learning*: structures of nodes, arcs and CPT (s) can be very different from Bayesian sub-model derived from the combination of overlay model and Bayesian network. The research proposes an optimal algorithm that applies dynamic Bayesian network into structure learning.

I propose the *maximum likelihood estimation* (*MLE*) *algorithm for learning parameters*. The basic idea of MLE algorithm is to find out the maximum value of likelihood function. Suppose each node in Bayesian network is attached by beta distribution (Neapolitan, 2003) which is specified by two parameters a and b, the likelihood function of beta distribution is:

$$L(a,b)=\prod_{i=1}^{n}f(x_i,a,b)=\prod_{i=1}^{n}\frac{1}{B(a,b)}x_i^{a-1}(1-x_i)^{b-1}=\frac{1}{B^n(a,b)}\left(\prod_{i=1}^{n}x_i^{a-1}\right)\left(\prod_{i=1}^{n}(1-x_i)^{b-1}\right)$$

Where $B(a, b)$ expresses beta distribution (Neapolitan, 2003) and x_i are observations in training data. The logarithm of likelihood function is determined as below:

$$LnL(a,b)=-nln\big(B(a,b)\big)+(a-1)\sum_{i=1}^{n}lnx_i+(b-1)\sum_{i=1}^{n}ln(1-x_i)$$

The goal of MLE algorithm is to find out two parameters a and b so that the logarithmic likelihood function gets maximal. After taking the first partial derivatives of logarithmic likelihood function with subject to a and b, I proves that parameters a and b are satisfied following set of equations:

$$\begin{cases} e^a\sum_{k=1}^{b}\frac{(-1)^k}{k}\binom{b}{k}=\frac{1}{n}\sum_{i=1}^{n}ln(x_i) \\ e^b\sum_{k=1}^{a}\frac{(-1)^k}{k}\binom{a}{k}=\frac{1}{n}\sum_{i=1}^{n}ln(1-x_i) \end{cases} \Leftrightarrow \begin{cases} F_1(a,b)=L_1 \\ F_2(a,b)=L_2 \end{cases}$$

Where,

$$F_1(a,b)=e^a\sum_{k=1}^{b}\frac{(-1)^k}{k}\binom{b}{k} \text{ and } F_2(a,b)=e^b\sum_{k=1}^{a}\frac{(-1)^k}{k}\binom{a}{k}$$

$$L_1=\frac{1}{n}\sum_{i=1}^{n}ln(x_i) \text{ and } L_2=\frac{1}{n}\sum_{i=1}^{n}ln(1-x_i)$$

Note that notation $\binom{a}{k}$ or $\binom{b}{k}$ denotes the number of combinations of a or b elements taken k. The research proposes an iterative algorithm to find out approximate solutions a and b of above set of equations. The iterative algorithm is based on the proposition: "*The range of variables a and b is from 1 to n where n is the whole positive number and not greater than the number of evidences in training data*". Each pair values (a_i, b_i) which are values of parameters a and b are fed to F_1, F_2 at each iteration. Two biases $\Delta_1 = F_1(a, b) - L_1$ and $\Delta_2 = F_2(a, b) - L_2$ are computed. So, the normal bias $\Delta = \sqrt{\Delta_1^2+\Delta_2^2}$ is totally determined. The pair (a, b) whose normal bias Δ is minimum are chosen as the parameter estimators.

I also propose the method to *improve knowledge sub-model by dynamic Bayesian network*. The strong point of static Bayesian network is simple and its inference mechanism is very effective but it cannot model temporal relationships between variables. In e-learning context, it is very necessary to monitor chronologically users' process of gaining knowledge. Dynamic Bayesian network (Neapolitan, 2003, pp. 272-279) consists of a series of G_t where G_t is static Bayesian network at time point t as according to figure 8.

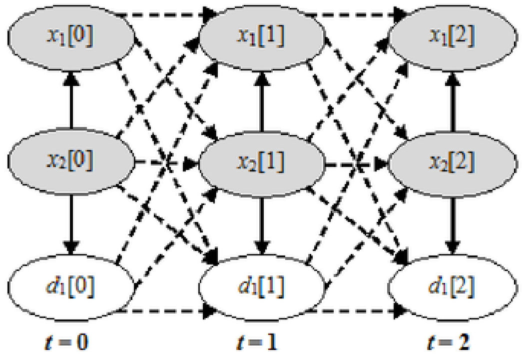

Figure 8. Dynamic Bayesian network with time points t.

The size of dynamic Bayesian network (DBN) gets huge after long time process, which causes a boom of combinations. In order to overcome this drawback of DBN, the research proposes the new algorithm to construct DBN so that the size of DBN are kept intact when the process continues for a long time, based on the Markov property "given previous time point t–1, the conditional probability of current time point t is only dependent on the previous time point t–1, not relevant to any further past time point (t–2, t–3,…, 0)". At every time point t, the algorithm only specifies G_{t-1}, G_t and transition probability $P(G_t \mid G_{t-1})$. The algorithm has a lot of iterations and each iteration includes 6 accumulated steps:

1. *Initializing DBN*: All variables (nodes) and relationships (arcs) among variables of initial Bayesian network G_0 must be specified. The strength of relationship is considered as weight of arc.
2. *Specifying transition weights*: The transition weights expressing conditional transition probabilities of current network G_t given previous network G_{t-1} are specified based on factors *slip* and *guess*.
3. *(Re-)constructing DBN*: The DBN at two consecutive time points t–1 and t is (re-)constructed based on current network G_t, previous network G_{t-1} and transition weights.
4. *Normalizing weights of relationships*: All weights which express relationships among variables inside DBN at current time point t are normalized so that sum of these weights are equal to 1.
5. *Re-defining CPT* (s): All CPT (s) of DBN at current time point t are updated based on normalized weights (in step 4) and posterior probabilities. Note that the posterior probabilities were computed in step 6 of previous iteration.
6. *Probabilistic inference*: The posterior probabilities at current time point t are computed according to Bayesian inference. These probabilities will be used to update CPT (s) in step 5 of next iteration.

DBN is evolutional and inference mechanism gets more precise after each iteration; please see (Nguyen, A New Algorithm for Modeling and Inferring User's Knowledge by Using Dynamic Bayesian Network, 2014) for more details. At step 2, the concepts *slip* and *guess* are defined:

- *Slip* is the probability that learner does know a particular subject but there is solid evidence convincing that she/he does not understand it, $P(not\ X_t \mid X_{t-1})$.
- *Guess* is the probability that learner does not know a particular subject but there is solid evidence convincing that she/he does understand it, $P(X_t \mid not\ X_{t-1})$.

- *Transition weight* is derived from concepts *slip* and *guess* according to formula: $(1 - slip)\frac{1}{1+guess}$. Please see (Nguyen, A New Algorithm for Modeling and Inferring User's Knowledge by Using Dynamic Bayesian Network, 2014) for proof of this formula.

Figure 9 shows the flow chart of the proposed algorithm to construct DBN.

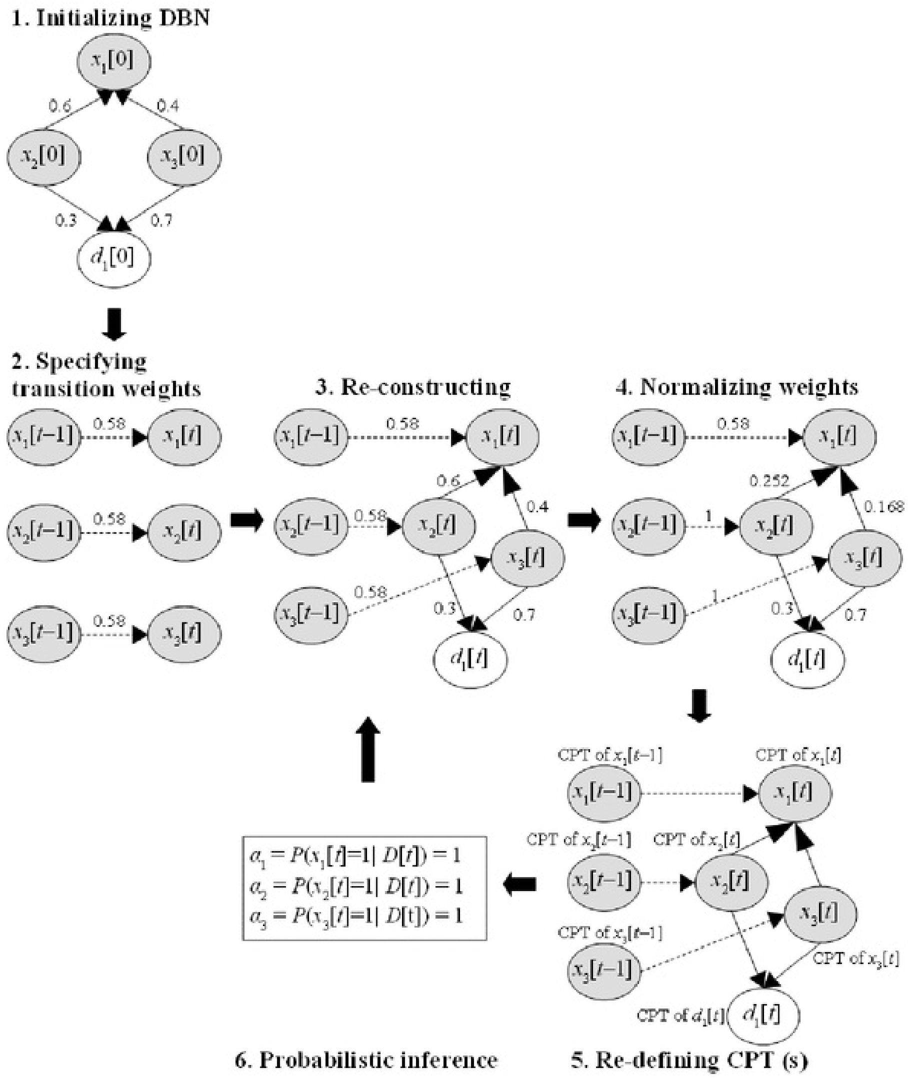

Figure 9. Algorithm to construct dynamic Bayesian network.

In general, the knowledge sub-model is the combination of overlay model and Bayesian network. The structure of knowledge sub-model is designed by specialist according to overlay model and its implementation conforms overlay model (Nguyen & Do, Combination of Bayesian Network and Overlay Model in User Modeling, 2009) (De Bra & Calvi, AHA! An open Adaptive Hypermedia Architecture, 1998) (De Bra, Smits, & Stash, The Design of AHA!, 2006) but it applies Bayesian network into inference mechanism. Moreover, the research proposes two methods to improve knowledge sub-model such as learning parameters via MLE algorithm, learning structure and modeling knowledge sub-model chronologically by DBN.

3.2. Learning style sub-model

Learning styles (Stash, 2007, p. 93) are defined as the composite of characteristic cognitive, affective and psychological factors that serve as relatively stable indicators of how a learner perceives, interacts with and responds to the learning environment. There are many psychological theories relevant to learning style such as Dunn and Dunn, Witkin, Riding, Myers-Briggs, Kolb, Honey-Mumford, Felder-Silverman. The research chooses hidden Markov model (Schmolze, 2001) to represent the psychological theory Honey-Mumford and Felder-Silverman. According to Honey-Mumford (Honey & Mumford, 2000) and Felder-Silverman (Felder & Silverman, 1988), learning styles are classified into dual pairs as follows (Stash, 2007, pp. 105-106):

- Verbal / Visual: Verbal person prefer to perceive materials as text and visual person prefer to perceive materials as pictures.
- Activist / Reflector: Activists are open-mined and comprehend new information by doing something with it. Reflectors prefer to think about new information first before acting on it.
- Theorist / Pragmatist: Theorists think things through in logical steps and assimilate different facts into coherent theory. Pragmatists have practical mind and prefer to try and test techniques relevant to problems.

Hidden Markov model has inference mechanism consisting of 5 components $\langle S, \Theta, A, B, \prod\rangle$. A finite set of states $S = \{s_1, s_2, \ldots, s_n\}$. A stochastic process P is a set of states $P = \{x_1, x_2, \ldots, x_k\}$ với $x_i \in S$. Process P must meet fully the Markov property, namely, given previous state x_{k-1} of process P, the conditional probability of current state x_k is only dependent on the previous state x_{k-1}, not relevant to any further past state $(x_{k-2}, x_{k-3}, \ldots, 0)$. In other words, $P(x_k \mid x_{k-1}, x_{k-2}, x_{k-3}, \ldots, x_0) = P(x_k \mid x_{k-1})$. In hidden Markov model, S is hidden set. Initial state distribution for every state, $\prod = (\pi_1, \pi_2, \ldots, \pi_n)$ where $\pi_i = P(s_i)$ is initial probability for state s_i. The transition probability matrix $A = (a_{ij})$ where a_{ij} is the probability of transitioning from any given state s_i to some next state is s_j. We have $\forall s_i \in S, \sum_{s_j \in S} a_{ij} = 1$. A finite set of possible observations $\Theta = \{\theta_1, \theta_2, \ldots, \theta_m\}$. Observation probability matrix $B = (b_{ij})$ where b_{ij} is the probability that state s_i occurs given an observation θ_j.

Given a stochastic process, hidden Markov model finds out hidden states from observations. The research applies hidden Markov model into discovering users' learning styles via observations about theirs learning process. Psychological theories Honey-Mumford and Felder-Silverman is constructed by hidden Markov model with 5 components $\langle S, \Theta, A, B, \prod\rangle$. Suppose three pairs of learning styles are chosen such as *Verbal / Visual, Activist / Reflector, Theorist / Pragmatist*, we build up three models $\Delta_1 = \langle S_1, \Theta_1, A_1, B_1, \prod_1\rangle$, $\Delta_2 = \langle S_2, \Theta_2, A_2, B_2, \prod_2\rangle$ and $\Delta_3 = \langle S_3, \Theta_3, A_3, B_3, \prod_3\rangle$, followed by 5 steps:

1. Defining *states*: Each state is identical to a leaning style. I propose a method to determine user's learning styles by finding out such states. We have $S_1 = \{verbal, visual\}$, $S_2 = \{activist, reflector\}$, $S_3 = \{theorist, pragmatist\}$.
2. Defining *initial state distributions*: Uniform probability distribution is used for each $\prod_i$. For example, we have $\prod_1 = \{0.5, 0.5\}$, $\prod_2 = \{0.5, 0.5\}$, $\prod_3 = \{0.5, 0.5\}$ which implies $P(verbal) = P(visual) = 0.5$, $P(activist) = P(reflector) = 0.5$ and $P(theorist) = P(pragmatist) = 0.5$.

3. Defining *transition probability matrices*: Suppose that learners tend to keep their styles; so the conditional probability of a current state on previous state is high if both current state and previous state have the same value and otherwise.
4. Defining *observations*: There is a relationship between learning object learned by users and their learning styles. Suppose each learning object such as lecture, for example has three attributes such as format attribute, type attribute and interactive attribute. Format attribute indicating the format of learning object has three values: *text*, *picture*, and *video*. Type attribute telling the type of learning object has four values: *theory*, *example*, *exercise*, and *puzzle*. Interactive attribute indicates the "interactive" level of learning object. The more interactive learning object is, the more learners interact together in their learning path. This attribute has three values corresponding to three levels: *low*, *medium*, *high*. Whenever a student selects a learning object (LO), observations are raised according to the attributes of learning object. We must account for the values of the attributes selected. For example, if a student selects a LO which has *format* attribute being *text*, *type* attribute being *theory*, *interactive* attribute being *low*, there are considerable observations: *text*, *theory*, *low* (interaction). So, it is possible to infer that she/he is a theorist. The dimension *Verbal* / *Visual* is involved in format attribute. The dimensions *Activist* / *Reflector* and *Theorist* / *Pragmatist* relate to both *type* attribute and *interactive* attribute. In other words, observations are identified with these attributes and so we have Θ_1 = {*text*, *picture*, *video*}, Θ_2 = {*theory*, *example*, *exercise*, *puzzle*, *low* (interaction), *medium* (interaction), *high* (interaction)}, Θ_3 = {*theory*, *example*, *exercise*, *puzzle*, *low* (interaction), *medium* (interaction), *high* (interaction)}.
5. Defining *observation probability matrices*: Different observations (attributes of LO) effect on states (learning styles) in different degrees. The research specifies these degrees by weights. The larger the weight is, the higher the effectiveness of observation is. As example, for verbal person, the weights of observations: text, picture, video are in descending order because they prefer to text learning material. Otherwise, for visual person, the weights of observations: text, picture, video are in ascending. Weights are normalized into conditional probabilities and observation probability matrix is consists of these probabilities.

After hidden Markov models are built up, Viterbi algorithm (Schmolze, 2001) is applied into finding out users' learning styles from their learning material choices and their learning processes; please see (Nguyen, A New Approach for Modeling and Discovering Learning Styles by Using Hidden Markov Model, 2013) for more details.

3.3. Learning history sub-model

Learning history sub-model is at the bottom of Triangular Learner Model (TLM) because it is the most important one which is used to construct knowledge sub-model and learning style sub-model. Learning history is defined as a transcript of all learners' actions such as learning materials access, duration of computer use, doing exercise, taking an examination, doing test, communicating with teachers or classmates. This sub-model has four main functions:

1. Providing necessary information for two remaining sub-models: learning style sub-model and knowledge sub-model described in previous sub-sections so that they perform inference tasks.

For example, knowledge sub-model needs learning evidences like learner's results of test, frequency of accessing lectures, etc. so as to assess learner's mastery of concrete knowledge item or concept.

2. Supporting learning concept recommendation.
3. Mining learners' educational data in order to discover other learners' characteristics such as interests, background, and goals.
4. Supporting collaborative learning through constructing learner groups.

Mining engine structures and manages learning history sub-model by data mining techniques. I focuses on function 2, 3 and 4 here. By function 1, learning history sub-model is often archived in XML file format or relation database.

The research proposes an interesting method for function 2, *learning concept recommendation based on sequential pattern mining*. Learning activities such as doing exercises and accessing learning materials are represented as sequential database (or learning sequences) and sequential patterns are mined from this database (Han, Pei, & Yan, 2005, p. 184). Sequential pattern represents the learning path that learners should follow in their learning process. For example, if learner is studying concept A and the sequential pattern is $B \rightarrow C \rightarrow A \rightarrow E$, adaptive learning system will recommend her/him to study from concept B to C, A and E in successive. I propose a special technique that breaks mined sequential pattern so as to perform recommendation task.

Suppose there are domain concepts (subjects) in Java course: **d**ata type, **p**ackage, class & **O**OP, **s**election structure, **v**irtual machine, **l**oop structure, **c**ontrol structure, and inter**f**ace which in turn denoted as *d*, *p*, *o*, *s*, *v*, *l*, *c*, *f*. The term "class & OOP" is abbreviation of "class and object-oriented programming". Study activities that students access learning material relating such domain concepts in sessions are represented as the learning sequential database shown in table 1.

Table 1. Learning sequential database.

Student	Session	Concept accessed		ID	Learning sequences	Length
1 1	Aug 5 10:20:01 Aug 5 10:26:12	*o* *d*		1	$\langle od \rangle$	2
2 2 2	Aug 6 08:20:01 Aug 6 14:15:01 Aug 6 15:00:00	*s*, *c* *o* *f*, *l*, *d*		2	$\langle (sc)o(fld) \rangle$	6
3 3 3	Aug 7 12:30:00 Aug 7 13:35:00 Aug 7 15:50:00	*o*, *v* *f* *p*		3	$\langle (ov)fp \rangle$	4
4 4 4	Aug 8 07:14:20 Aug 8 07:40:25 Aug 8 10:17:20	*o* *f*, *d* *p*		4	$\langle o(fd)p \rangle$	4
5	Aug 8 10:26:15	*p*		5	$\langle p \rangle$	1

Suppose sequential pattern $\langle ofp \rangle$ is mined from learning sequential database, we have learning path: "*class & OOP*"$\rightarrow$ "*interface*" $\rightarrow$ "*package*". If a student is studying the concept "*class & OOP*", the adaptive learning system should recommend which next concepts in above patterns. So the pattern will be broken into association rules with their confidences. For example, breaking above pattern ⟨*ofp*⟩ follows steps:

1. Breaking entire ⟨*ofp*⟩ into large itemsets such as *o*, *f*, *p* and determining all possible large 2-sequences which are 2-arrangment of all large itemsets following the criterion: order of 2-sequences must comply with the order of sequential pattern. There are three large 2-sequences: ⟨*of*⟩, ⟨*op*⟩, ⟨*fp*⟩ broken from the pattern ⟨*ofp*⟩. Thus, we have three rules derived from these large 2-sequences in form: "*left-hand large itemset* → *right-hand large itemset*", for example, rule "$o \rightarrow f$" derived from 2-sequence ⟨*of*⟩.
2. Computing the confidences of such rules and sorting them according to these measures. The confidence of a rule is the ratio of the support of 2-sequences to the support of left-hand large itemset, $confidence(x \rightarrow y) = support(\langle xy \rangle) / support(x)$. Strong rules are defined as rules whose confidences are greater than or equal to a pre-defined threshold *min_conf*. Strong rules are also called **sequential rules** and their confidences are known as sequential confidences. Only sequential rules are used for learning concept recommendation. Given *min_conf*=0.5, table 2 shows sequential rules extracted from the sequential pattern ⟨*ofp*⟩.

Table 2. Sequential rules.

sequential rule	confidence	recommended concept
$o \rightarrow f$	0.75	"*interface*"
$o \rightarrow p$	0.50	"*package*"

If student choose the concept (itemset) *x*, system will find the rules whose *left-hand itemset* must contain *x* and whose confidence is highest. After that, concepts in the *right-hand itemsets* (concepts) of this rule are recommended to learner. For example, if learner is studying concept "*class & OOP*", system find out the rule $o \rightarrow f$ which contains concept *o* (*class & OOP*) and gains highest confidence (0.75); so concept *f* (*interface*) is recommended to student.

This methodology is similar to mining association rules but it achieves high performance and precise prediction since it proposes learner learning path derived from result of sequential pattern mining process. Please see (Nguyen & Do, Learning Concept Recommendation based on Sequential Pattern Mining, 2009) for more details in learning concept recommendation based on sequential pattern mining.

The research proposes an interesting method for function 3 in mining engine, *discovering user interests by document classification*. Interest is personal trait not modeled in standard Triangular Learner Model (TLM) but it is described in extended TLM because of its importance in adaptive process. Learning history sub-model is responsible for discovering user interests by mining techniques. I only propose a mining technique so-called document classification but it assures that there will have many other mining techniques and so this is an open research. The proposed technique is based on two points of view:

- The series of user access in her/his history are modeled as documents. So user is referred indirectly to as "document".

- User interests are classes such documents are belong to.

The proposed technique includes four following steps:

1. Documents in training corpus are represented according to *vector model*. Each element of vector is product of term frequency and inverse document frequency. However the inverse document frequency can be removed from each element for convenience.
2. *Classifying training corpus* by applying classification methods such as decision tree, support vector machine, and neural network. Classifiers must be constructed by these methods; for example, classification rules are drawn from decision tree and weight vectors are drawn from support vector machine. Classification rules and weight vectors are typical classifiers.
3. Mining user's access history to *find maximum frequent itemsets*. Each itemset is considered an *interesting document* and its member items are considered as terms. Such interesting documents are modeled as vectors.
4. Applying classifiers (see step 2) into these interesting documents (see step 3) order to choose which classes are most suitable to these interesting documents. *Such classes are user interests*.

In general this methodology is based on the point of view that learner is modeled indirectly as a "document" and the class of this document is her/his interest. The method is the combination of classification technique and information retrieval technique. Please see (Nguyen, Discovering User Interests by Document Classification, 2010) for more details in discovering user interests by document classification.

The research also proposes methods to *construct user groups or user communities*. User model in the research aims to individualization, which is appropriate to personalized adaptive system; in addition, there is a demand to provide adaptation to a group or community of users. Consequently, all users in the same group will profit from the same learning materials, teaching methods, etc. because they have the common characteristics. Of course, individual or community (group) adaptation has particular features and demands but the preeminent adaptive system should support both of them. Group adaptation has more advantages than individual adaptation in some situations:

- Common features in a group which are the common information of all members in such group are relatively stable, so it is easy for adaptive systems to perform accurately adaptive tasks.
- If a new user logins system, she/he will be classified into a group and initial information of her/his model is assigned by common features in such group.
- In the collaborative learning, users need to learn or discuss together. It is very useful if the collaborative learning is restricted in a group of similar users. Therefore, it is convenient for users that have common characteristics (knowledge, goal, interest, etc.) to learn together because they do not come up against an obstacle when interacting together.

Because it is necessary to specify user groups, the research proposes clustering technique to determine such groups, especially clustering for overlay model and Bayesian network. Please see (Nguyen, User Model Clustering, 2014) for clustering techniques along with similarity and dissimilarity measures. The research focuses on clustering techniques for overlay model and Bayesian network.

There are many applicable clustering techniques (Han & Kamber, Data Mining, 2006, pp. 443-483) based on dissimilarity measure (distance) of two user models but there is a problem: "how to compute such dissimilarity measure in case that user model is an overlay model which is in form of domain graph". The research proposes special technique based on graph depth.

Suppose there are two overlay models $U_1=G_1=<V_1, E_1>$ and $U_2=G_2=<V_2, E_2>$ where V_i and E_i are set of nodes and set of arcs, respectively. Graphs G_1 and G_2 are in form of tree in which each directed arc represents an aggregation relationship of two nodes. The set of nodes $V_i = (v_{i1}, v_{i2},\ldots, v_{in})$ is also considered as a vector whose elements are numbers representing user's masteries of knowledge items. Let $depth(v_{ij})$ is the depth level of node j of graph model G_i. Note that the depth level of root node is 1. Figure 10 depicts a graph model in form of tree.

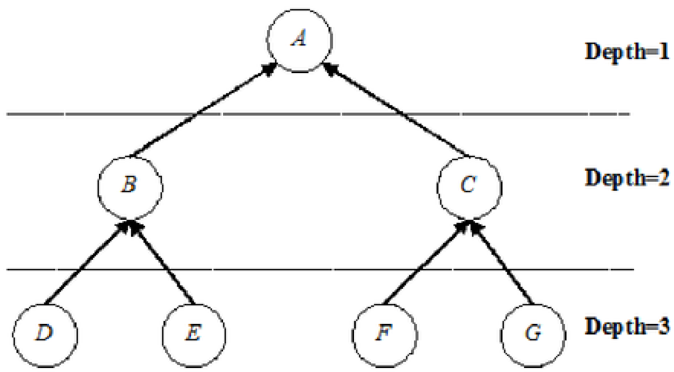

Figure 10. Graph and depths.

The dissimilarity (or distance) of two graph models G_1 and G_2 is defined as below:

$$dissim(G_1, G_2) = distance(G_1, G_2) = \sum_{j=1}^{n} \left| \frac{v_{1j} - v_{2j}}{depth(v_{1j})} \right|$$

The meaning of this formulation is "the high level concept is the aggregation of low level concepts". In case that arcs in graph are weighted, let $weight(v_{ij})$ be the weight of arc from node j (of graph model i) to its parent. When we consider that $weight(v_{ij})$ is the weight at node v_{ij}, the dissimilarity (or distance) of two graph models G_1 and G_2 is re-defined as below:

$$dissim(G_1, G_2) = distance(G_1, G_2) = \sum_{j=1}^{n} \left| \frac{v_{1j} - v_{2j}}{depth(v_{1j})} weight(v_{1j}) \right|$$

In case that graph model is Bayesian network, the dissimilarity-calculation formula is more complicated. Bayesian network is considered as graph and the graph depth has the same definition but the weight of arc is replaced by conditional probability table (CPT), as shown in figure 11. So the value at each node is replaced by marginal posterior probability.

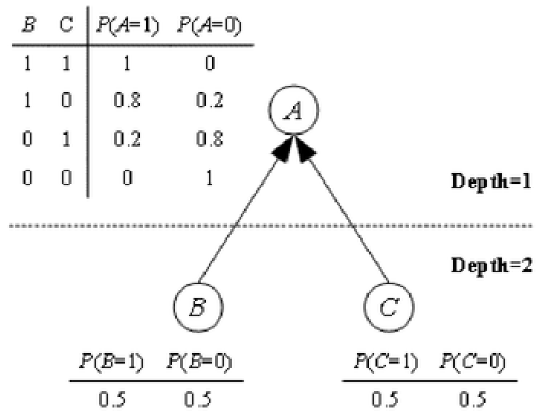

Figure 11. Bayesian network, its CPT (s) and depths.

Let $P(v_{ij})$ be the marginal probability of node j in network G_i, the dissimilarity (or distance) between two Bayesian network G_1 and G_2 is defined as below:

$$dissim(G_1, G_2) = distance(G_1, G_2) = \sum_{j=1}^{n} \left| \frac{P(v_{1j}) - P(v_{2j})}{depth(v_{1j})} \right|$$

4. Evaluation of TLM

In general, this research is fundamental research. Thus, approaches, architectures, models and mathematical formulas are proposed in the coherence of fundamental methodologies. This research is not experimental research and testing technique on testing data is not appropriate to evaluate this research. So there are three ways to evaluate this research:

- The correctness of formulas is proven by mathematical tools and logical reasoning.
- The feasibility of models and architectures is authenticated via the computer software which implements Zebra and TLM. The computer software is available at http://www.locnguyen.net/st/dissertations/zebra
- The effectiveness of adaptive learning is proven via approaches described in this section. Therefore subsection 4.1 is the evaluation on knowledge sub-model with support of Computerized Adaptive Testing (CAT). Subsection 4.2 is the evaluation on the effectiveness of adaptive learning model, especially, the whole TLM and modeling system Zebra.

4.1. Computerized Adaptive Testing (CAT)

The purpose of knowledge sub-model is to improve learners' knowledge. Assessing users' knowledge is the same to evaluating knowledge sub-model. The computer-based tests have more advantages than the traditional paper-based tests when there is the boom of internet and computer. Computer-based testing allows students to perform the tests at any time and any place and the testing environment becomes more realistic. Moreover, it is very easy to assess students' ability by using the Computerized Adaptive Testing (CAT). The CAT is considered as the branch of computer-based testing but it improves the accuracy of test core when CAT systems try to choose items (tests, exams, questions, etc.) which are suitable to students' abilities; such items are called adaptive items. CAT is based on

18

Item Response Function (IRF) which is function of a true ability θ of given examinee with three parameters of item i such as a_i, b_i, and c_i, as follows:

$$IRF(\theta) = c_i + \frac{1 - c_i}{1 + exp\left(-a_i(\theta_j - b_i)\right)}$$

Where a_i, b_i, and c_i are discriminatory parameter, difficult parameter, and guessing parameter, respectively. The important problem in CAT is how to estimate students' abilities so as to select the best items for students with note that students are examinees. There are some methods to solve this problem such as Bayesian approach (Linden & Pashley, 2002, pp. 3-7) but I propose a new method to compute ability estimates by maximization likelihood estimation (MLE). Given N items, I find out the following formula to determine the ability estimate $\hat{\theta}$ of examinee.

$$\hat{\theta} = \hat{b} = \frac{ln(\sum_{i=1}^{N} r_i) - ln(\sum_{i=1}^{N}(q_i - r_i))}{\hat{a}} + \theta_0$$

Where q_i is the number of possible responses of item i, r_i is the correct response of examinee to item i and θ_0 is the examinee's initial ability. The notation "*ln*" denotes natural logarithm function. Additionally, discriminatory parameter estimate $\hat{a}$ is arbitrary.

I prove that the ability variance of examinee $Var(\hat{\theta})$ is determined based on discriminatory parameter estimate $\hat{a}$ as follows:

$$Var(\hat{\theta}) = \sigma_{\hat{\theta}}^2 = \frac{\pi^2}{3\hat{a}^2}$$

Suppose there are k examinees $u_1, u_2, \ldots, u_k$ who have k ability estimates $\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_k$ after they have done a number of test items. Let $\bar{\hat{\theta}}$ be statistical sample mean (Montgomery & Runger, 2003, p. 190) of examinees' ability estimates, we have:

$$\bar{\hat{\theta}} = \frac{1}{k}\sum_{i=1}^{k} \hat{\theta}_i$$

The ability variance $Var(\hat{\theta})$ is now considered as statistical sample variance (Montgomery & Runger, 2003, p. 191), which is calculated as follows:

$$Var(\hat{\theta}) = \frac{1}{k-1}\sum_{i=1}^{k}\left(\hat{\theta}_i - \bar{\hat{\theta}}\right)^2$$

Let a^* be the so-called *discriminatory estimate*, it implies that

$$a^* = \frac{\pi}{\sqrt{3Var(\hat{\theta})}} = \sqrt{\frac{(k-1)\pi^2}{3\sum_{i=1}^{k}\left(\hat{\theta}_i - \bar{\hat{\theta}}\right)^2}}$$

Based on the proposed MLE method for estimating examinee's ability, the new version of CAT algorithm is also invented. Such new version of CAT algorithm is called *advanced CAT algorithm* which includes 4 steps as follows:

1. Suppose there are k examinees $u_1, u_2, \ldots, u_k$ who have k ability estimates $\theta_1, \theta_2, \ldots, \theta_k$ after they have finished a number of items in previous test. Suppose there are n items in the test pool and each item i has individual parameters such as a_i, b_i, and c_i. Items available in test pool must be evaluated. Let $\bar{\theta}$ be ability mean of such k examinees. Let a^* be discriminatory estimate that is calculated at step 3 in previous test. The best items are the ones whose parameters a_i and b_i are

nearest to a^* and $\bar{\theta}$, respectively. The best items are the most suitable to examinees' current ability estimates.

2. Such best items are given to examinees and examinees make responses to these items.
3. New ability estimates of examinees are computed based on responses to all of the chosen items and current abilities θ_1, θ_2,…, θ_k. Concretely, the k ability estimates $\hat{\theta}_1$, $\hat{\theta}_2$,…, $\hat{\theta}_k$ are re-calculated according to the aforementioned formula

$$\hat{\theta} = \frac{ln(\sum_{i=1}^{N} r_i) - ln(\sum_{i=1}^{N}(q_i - r_i))}{\hat{a}} + \theta_0$$

Moreover, discriminatory estimate a^* is re-computed based on new estimates $\hat{\theta}_1$, $\hat{\theta}_2$,…, $\hat{\theta}_k$ according to the aforementioned formula

$$a^* = \sqrt{\frac{(k-1)\pi^2}{3\sum_{i=1}^{k}\left(\hat{\theta}_i - \bar{\hat{\theta}}\right)^2}}$$

Let θ_1, θ_2,…, θ_k are current abilities of k examinees u_1, u_2,…, u_k. We assign $\theta_1 = \hat{\theta}_1$, $\theta_2 = \hat{\theta}_2$,…, $\theta_k = \hat{\theta}_k$ in order to update current abilities θ_1, θ_2,…, θ_k.

4. Algorithm terminates if stopping criterion is met; otherwise going back step 1.

Moreover, I suggest the stopping criterion based on ability error, thus, the process of testing stops if the ability error is approximated to zero. The ability error measures difference of student's ability between two successive testing times. In other words, the process of testing ends only when student's knowledge becomes saturated (she/he cannot do test better or worse) and such knowledge is her/his actual knowledge.

4.2. Evaluation of adaptive learning model

User modeling system is the heart of adaptive learning system. There are a lot of theories and practical methods including methodologies and approaches in this research to build up adaptive system and user modeling system. Each method has strong points and drawbacks and so it is very useful to evaluate these methods in order to determine which model is appropriate to which situation because each method tailors to concrete conditions and contexts. For example, studying via internet website is different from studying at a course with support of network. This subsection focuses on how to evaluate adaptive learning system with regard to user modeling system in e-learning or distance learning context when there is no separation between adaptive learning system and user modeling system (Nguyen L. , Evaluating Adaptive Learning Model, 2014). We can consider the corporation between adaptive system and user modeling system as an integrated model called adaptive learning model. This research proposes three criterions of evaluation:

1. Criterion α so-called *system criterion* tells us how adaptive learning system works with/without user modeling system. For example, when modeling server applies Bayesian network into building up learner model, criterion α measures the performance of adaptive system with or without the support of Bayesian network. Criterion α is determined based on statistical hypothesis testing or linear regression. In general, this criterion answers two following questions:
   a. How adaptation is performed in adaptive system with/without the support of modeling server.

   b. Whether the whole user knowledge is computed more accurately with the support of user model, for example Bayesian network.

2. Criterion β so-called *academic criterion* tells us how well modeling server helps users to study. This criterion surveys users' study result. The higher criterion β is, the better study result is. Criterion β is determined based on cumulative function of standard normal distribution.
3. Criterion γ so-called *adaptation criterion* or satisfaction criterion measures the quality of adaptation function of learning system with the support of modeling server. After every student gives feedbacks or comments on adaptive system, these feedbacks are collected and analyzed; hence, criterion γ is calculated based on these feedbacks in order to estimate level of students' satisfaction from adaptive system. The higher criterion γ is, the better quality of adaptation is.

5. Conclusion and future trend

In general, the research aims to build up a user modeling system in learning context. Firstly, learner model so-called Triangular Learner Model (TLM) consisting of 3 sub-models such as knowledge, learning style and learning history which are associated together so as to form a triangular.

- Knowledge sub-model is the combination of overlay model and Bayesian network together with weight specification. Knowledge sub-model is evolved via techniques such as parameter learning via EM algorithm, parameter learning via MLE algorithm, structure learning via dynamic Bayesian network.
- Learning style sub-model is constructed by hidden Markov model which is applied into psychological theories so as to discover user's learning styles.
- Learning history sub-model is the most important one which archives learners' study activities and has XML file format. This sub-model is used to construct knowledge sub-model and learning style sub-model and to discover extended features such as user interests and user group.

User modeling system *Zebra* manages TLM and performs inference mechanism by two engines:

- Mining engine manages whole TLM but focuses on learning history sub-model and uses mainly data mining techniques.
- Belief network engine manages and performs inference mechanism on knowledge sub-model and learning style sub-model (*LS*) by applying Bayesian network and hidden Markov model.

Zebra system in the future will support two ubiquitous environment (Heckmann, 2005) in which the user modeling system interacts with users at anywhere and is totally transparent. It means that users need not make sense information technology infrastructure and the architecture of user modeling system. So, users only take advantages of profits from ubiquitous service. Zebra user modeling system will be integrated into ubiquitous environment.

Acknowledgments

I would like to show my gratitude to all those who gave me the possibility to complete the research. I am grateful to Prof. Dr. Dong, Thuy T. B. for providing invaluable help and encouragement in my work. If it were not for your help, I would not complete my research in time. I have furthermore to thank Department of Information System, Faculty of Information Technology, University of Science for conveniences during the time of research.

I also want to express my warmest thanks for my family and friends who have supported me in difficult times.

Author Contributions

I published 17 papers and articles relevant to the research. Moreover the user modeling system Zebra mentioned in the research is implemented as computer software associated to the book, available at http://www.locnguyen.net/st/dissertations/zebra

systems-120526-2.docx

ORIGINALITY REPORT

41%

SIMILARITY INDEX

PRIMARY SOURCES

1	www.standresjournals.org Internet	2652 words — 33%
2	Nguyen, Loc. "A user modeling system for adaptive learning", 2014 International Conference on Interactive Collaborative Learning (ICL), 2014. CrossCheck	127 words — 2%
3	Studies in Computational Intelligence, 2010. CrossCheck	94 words — 1%
4	Nguyen, Loc. "User Model Clustering", Journal of Data Analysis and Information Processing, 2014. CrossCheck	90 words — 1%
5	file.scirp.org Internet	85 words — 1%
6	Loc Nguyen. "Combination of Bayesian Network and Overlay Model in User Modeling", Lecture Notes in Computer Science, 2009 CrossCheck	49 words — 1%
7	"Evaluating adaptive learning model", 2014 International Conference on Interactive Collaborative Learning (ICL), 2014. CrossCheck	40 words — < 1%
8	Minati, Gianfranco, and Eliano Pessa. "Special Issue on Second Generation General System Theory", Systems, 2014. CrossCheck	18 words — < 1%

9 pirsquared.org — 15 words — < 1%
Internet

10 Zhao Zhi-Jin. "Discrete channel modelling based on genetic algorithm and simulated annealing for training hidden Markov model", Chinese Physics, 06/2007 — 13 words — < 1%
CrossCheck

11 www.columbia.edu — 13 words — < 1%
Internet

12 Loc Nguyen. "Combination of Bayesian Network and Overlay Model in User Modeling", International Journal of Emerging Technologies in Learning/18630383, 20091201 — 12 words — < 1%
Publications

13 Leana Golubchik. "Estimating software component reliability by leveraging architectural models", Proceeding of the 28th international conference on Software engineering - ICSE 06 ICSE 06, 2006 — 11 words — < 1%
CrossCheck

14 Zapico-Valle, J.L.. "A new method for finite element model updating in structural dynamics", Mechanical Systems and Signal Processing, 201010 — 11 words — < 1%
CrossCheck

15 www.archive.org — 11 words — < 1%
Internet

16 Jianjun, Y.. "Gaussian Sum PHD Filtering Algorithm for Nonlinear Non-Gaussian Models", Chinese Journal of Aeronautics, 200808 — 10 words — < 1%
CrossCheck

17 www.intechopen.com — 9 words — < 1%
Internet

18 Ronald Armstrong. "Developing and Assembling the Law School Admission Test", Interfaces, 03/2005 — 8 words — < 1%
CrossCheck

19	Kit Po Wong. "A general method for electricity market price spike analysis", IEEE Power Engineering Society General Meeting 2005, 2005 CrossCheck	8 words — < 1%
20	www.nhm.ac.id Internet	8 words — < 1%
21	m.scirp.org Internet	8 words — < 1%
22	Lee-Ling Ong. "Resource constrained particle filtering for real-time multi-target tracking in sensor networks", 2009 IEEE International Conference on Pervasive Computing and Communications, 03/2009 CrossCheck	8 words — < 1%
23	www.docstoc.com Internet	8 words — < 1%
24	Loc Nguyen. "A Proposal of Discovering User Interest by Support Vector Machine and Decision Tree on Document Classification", 2009 International Conference on Computational Science and Engineering, 08/2009 CrossCheck	5 words — < 1%

EXCLUDE QUOTES	OFF	EXCLUDE MATCHES	OFF
EXCLUDE BIBLIOGRAPHY	ON		