American Journal of Applied Sciences 6 (5): 913-921, 2009
ISSN 1546-9239

Log data Approach to Acquisition of Optimal Bayesian Learner Model

Choo-Yee Ting and Somnuk Phon-Amnuaisuk
Faculty of Information Technology, Multimedia University, Cyberjaya, Selangor, Malaysia

Abstract: Log data provide valuable insight into observable behavioural patterns, which could be inferred to study a learner's cognitive processes, levels of motivation and levels of knowledge acquisition. To date, most of the research work has been devoted to study the different methods to analyze and interpret log data. Little attention, however, has been given to use log data as a tool to investigate the behaviour of Bayesian learner models. In this light, this article discusses how log data could be employed to investigate the performance of Bayesian learner models. The log data were firstly transformed into a set of structured dataset, which conformed to the INQPRO's learner model. The transformed dataset were then fed into different versions of INQPRO's learner model to obtain their predictive accuracies. From the predictive accuracies, an optimal learner model was identified. Empirical results indicated that the log data approach provides an efficient way to study the behaviour of a Bayesian learner model

Key words: Log data, bayesian learner model, scientific inquiry learning environment

INTRODUCTION

The log data are a chronological record of learners' interactions with a computer-based learning environment. Log data, which may take different forms, can be treated as a crucial instrument to educational psychologies because from which behavioural data can be extracted, analyzed and interpreted. By analyzing log data, insights into valuable information about (1) cognitive process, motivation level and acquisition level of domain knowledge a learner pursues. Consequently, tailored pedagogical support could be provided, (2) how a particular learning environment should be further improved to enhance learners' learning experiences through the graphical user interfaces.

Log data can be obtained via three common methods, namely, eye movement registration, protocol analysis and computer-registered operations[1]. The eye movement registration aims at capturing the part of the visual field the learner is paying attention on whereas the protocol analysis (also refers to as think-aloud approach), is a popular method for obtaining detailed report of real-time cognitive processing. Protocol analysis involves three steps: transcribing, segmenting and encoding[2]. The third type of method for obtaining log data is keeping track of learners' interactions performed on computer programs. Interaction captured may include mouse-click, drag-and-drop, typing and many more. The advantage of computer-registered operations approach is that detailed activities performed by a learner can be captured. However, without a proper reasoning mechanism behind the captured information, reliable interpretation can hardly be acquired.

Analysis of log data can be performed in two modes, namely, real-time and post-hoc[1]. In real-time analysis, the system captures a learner's behaviour up to the point in time that the analysis is performed. Such analysis mode is widely applied to computer-based learning environments. Most of the real-time analysis of log data, however, provides learners with a mere ad-hoc[3] rather than the adaptive support[4]. Conversely, the post-hoc analysis, aims to provide a comprehensive yet informative analysis of the learner's learning progress towards the end of a particular learning session. This mode of analysis is important when the learning environment employs exploratory or discovery learning approach, in which learners are given the freedom to explore and navigate from one interface to another.

Recent advancement in educational technology has shifted towards incorporating Artificial Intelligence (AI) into computer-based learning environments. These learning environments often maintain a learner model for each learner, which consistently inferring the required information from the associated log data. By integrating AI into the learning environment, log data

Corresponding Author: Choo-Yee Ting, Faculty of Information Technology, Multimedia University, Cyberjaya, Selangor, Malaysia

can be interpreted efficiently and subsequently pedagogical support can be tailored to learners. There are three types of interventions as a result of inferring from log data[1]: (1) supervising. The purpose of supervising intervention is to constrain learners within predefined set of allowable interactions, (2) monitoring. Monitoring intervention provide learners with guidance to complete a particular task by minimizing possible errors created by learners, (3) examining. Examining interaction aims at keeping track of actions performed by the learner without any interference at all. Learners experience complete freedom of action while interacting with the learning environment.

Little study, however, has been reported on using log data as a basis to study the predictive accuracies of learner models for a Bayesian Intelligent Tutoring System (ITS). In addition, little information can be acquired on the preprocessing and transformation of log data when it comes to Bayesian ITSs. Without a proper study on a particular Bayesian learner model, at least three problems could occur[5]: (1) knowledge representation issue, (2) tutorial action selection capabilities, (3) real-time inference requirement. Thus, the research work presented within this article aimed at providing a detailed methodological approach that begins with preprocessing of raw log data into dataset and subsequently feeding them into different learner models to overcome the mentioned challenges.

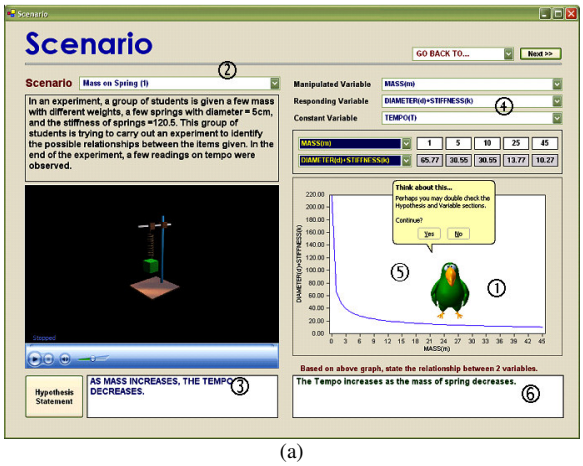
The next section provides an overview of INQPRO, which serves as prerequisite to the subsequent discussion on the preprocessing and transformation of log data. It discusses the interactive components in an interface from which log data are collected. An INQPRO's interface and a partial structure of its associated Decision Network is presented to give reader a sense of how evidence was captured and probabilities were propagated before the mastery levels of two scientific inquiry skills, namely formulate-test-retest hypotheses $\mathcal{H}$ and identifying and controlling variables $\mathcal{V}$, could be obtained.

OVERVIEW OF INQPRO

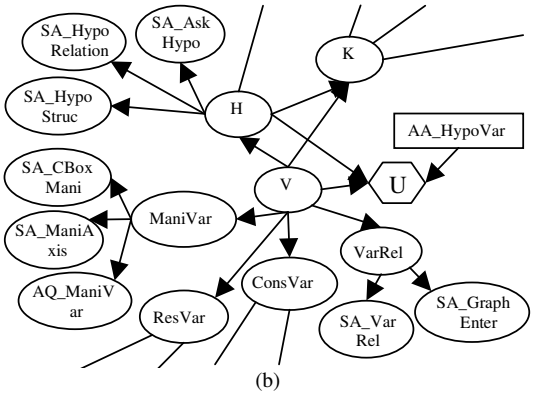
Scientific inquiry skills are among the key emphases in recent science education reform. The importance of scientific inquiry skills can be noted from the recently developed computer-based learning environments such as the KIE[6], SimQuest[7] and SCI-WISE[8]. To capitalize learning experience, exploratory learning approach is employed to provide learners with the freedom to practice on scientific inquiry skills. To be in line with current needs of current science education reform, a prototype named INQPRO is developed in this study. INQPRO consists of seven interfaces, namely, the Scenario (Sce), Hypothesis Visualization (Vz), Verification (Vf), Formula Investigation (Fe), Simulation Experiment (Ex), Data Comparison (Dc) and Feedback (Fd). By actively interacting with the interfaces and Intelligent Pedagogical Agent (hereafter Agent), learners are ultimately expected to command $\mathcal{H}$ and $\mathcal{V}$. Associated with each interface is a static learner model, represented by a Decision Network (DN). The specification of Conditional Probability Tables (CPTs) within the DNs is elicited by domain experts guided by the author of this article. $\mathcal{K}$ is introduced during the modeling of the DNs to encode the interrelation between $\mathcal{H}$ and $\mathcal{V}$.

Figure 1a shows one of the interfaces in INQPRO, namely, Scenario. Upon logging into Scenario, the Agent (Fig. 1a-①) will firstly introduce the learner with the various sections contained within the interface and subsequently with the presentation of a series of activities to be carried out which include selecting a scenario (Fig. 1a-②), formulate a hypothesis statement (Fig. 1a-③), identify variables (Fig. 1a-④), interact with data and graph (Fig. 1a-⑤) and formulate variable relationship statement (Fig. 1a-⑥). All the interaction performed such as button click, drag-and-drop, typing, selecting from list box, answering questions prompted by Agent and the like are logged without the notice of learners. Learners can proceed by two means: default learning path (by clicking the button next) or selective learning path (by clicking the button Go Back To…).

The corresponding DN for the Scenario interface is depicted in Fig. 1b. The DN performs both diagnostic and predictive reasoning[9] to acquire posterior mastery levels of $\mathcal{H}$ and $\mathcal{V}$ in light of receiving evidence, which constitutes a series of observed learner interactions). For instance, by performing diagnostic reasoning the mastery of $\mathcal{H}$ (node H) can be inferred from: (1) the structure of hypothesis statement (node SA_HypoStruc), (2) the position of variables stated in the hypothesis statement (node SA_HypoRelation) and (3) the frequency of explicit help requested by the learner to the Agent (node SA_AskHypo). Similarly, a learner's mastery level for the manipulated variable (node ManiVar) can be inferred from whether or not the learner has correctly selected the manipulated variable (node SA _CBoxMani) from the list down box (Fig. 1a-④) and the x-axis selected (node SA_ManiAxis). Besides, the degree to which a learner is considered to have understood the variables' relationship (node VarRel) can be inferred from the evidence that she has studied the graph (node

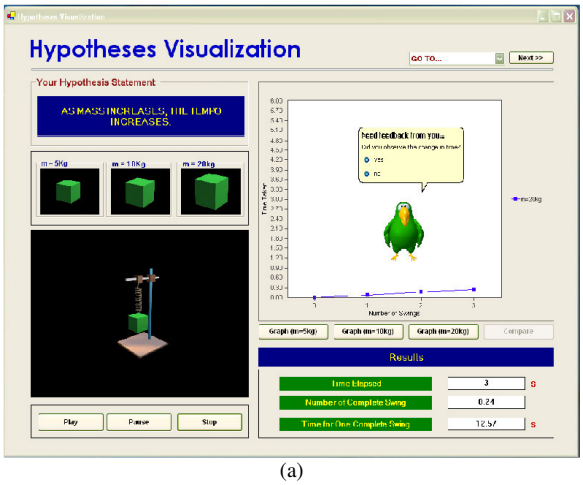


(a)

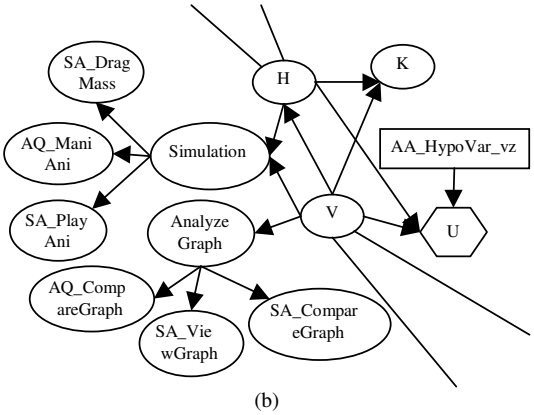


(b)

Fig. 1: (a): The Scenario interface, (b): The partial structure of Scenario DN



(a)



(b)

Fig. 2: (a): The Hypothesis interface, (b): The partial structure of Hypothesis DN

SA_GraphEnter) and typed in the correct statement describing the relationship between variables (node SA_VarRel).

Figure 2a shows another instance of INQPRO's interface, namely, Hypotheses Visualization. The uniqueness of this interface is that it helps learners explicitly visualize their hypotheses through computer simulations. Depending on the hypothesis generated in Scenario interface, the computer simulation presented might not be similar to what the learner has in mind. It is predicted that the naïve concepts can be made explicit which subsequently allows the uncovering of learner's misconceptions can be maximized. Making learner's mental model explicit has been reported to be a vital precursor to mental model restructuring[10]. There are three mass (m = 50 g, m = 100 g, m = 200 g) from which a learner can choose from to investigate the relationships between mass and tempo. Having the mass chosen (Fig. 2a-①), learners will proceed to the computer simulation control section (Fig. 2a-②). A corresponding computer animation will be displayed upon clicking the play button. Conversely, once the stop button is clicked, the graph halted and detail analysis of the simulation is reported in the Results section (Fig. 3b-③).

Figure 2b shwos the partial structure of Hypotheses Visualization DN. To infer the degree to which a learner is considered to have mastered the variables (node Variable_Vz) and hypothesis (node Hypothesis_Vz) relies on whether or not s/he is able to analyze the graph (node AnalyzeGraph) and has understood the purpose of simulation (node UnderstandAnimation). However, both AnalyzeGraph and UnderstandingAnimation cannot be observed directly from the interface. To obtain the posterior probability values for these two nodes, the network performs diagnostic reasoning given the instantiation of evidential nodes (nodes AQ_MassAni, SA_PlayAni, SA_DragMass, SA_ViewGraph, SA_CompareGraph and AQ_CompareGraph in Fig. 2b). The next section

will discuss about the formalism and transformation of log data into dataset which can be fed into different models of INQPRO's dynamic learner model.

LOG DATA FORMALISM

There were three main challenges when employing log data approach in this study: (i) to identify the structure of log data for storing information on learners' interactions, (ii) transformation of log data into dataset which were learner model compliant and (iii) encoding time into the dataset for the purpose of DDN evaluation.

In this research, log data were stored in Microsoft Access 2003 database format. To maintain the uniqueness of the log data, each learner had a database file in which raw and transformed log data was stored. There are two aspects discussed in this section. The first concerning the formulation of log data to capture interactions from INQPRO whereas the second aspect focusing on how raw log data (e.g., Table 1) were pre-processed and transformed into dataset which could be fed into INQPRO's dynamic learner models.

In this study, identifying the appropriate structure for storing interaction behaviour was not a trivial task particularly when information within the log data was meant to study the predictive accuracy of a learner model. The raw log data (e.g., Table 1) consists of the following properties:

- Log data is represented as a database relation $\mathcal{R}$ with n tuples, where n $\mathcal{R}$ {integer>0}. Each $\in$ can be uniquely identified from the others
- Each tuple $\mathcal{P}$ of $\mathcal{R}$ takes the form {$d_{\mathcal{S}}$, $d_{\mathcal{I}}$, $d_{\mathcal{C}}$, $d_{\mathcal{O}}$, $d_{\mathcal{V}}$, $d_{\mathcal{T}}$}, where $d_{\mathcal{S}}$ and $d_{\mathcal{I}}$ denote the name of the learner and interfaces, respectively. There are six interfaces in INQPRO and thus $d_{\mathcal{I}}$ $\mathcal{R}$ {Sce, Vz, Vf, Fe, Ex, Dc}. $d_{\mathcal{C}}$ denotes the interface components the learner interacts with. Examples of interface components can be the graph section, hypothesis section, or variable selection section. $d_{\mathcal{O}}$ represents the $d_{\mathcal{C}}$ that an operation can performed on. The operations in all the interfaces have included drag-and-drop, double click and mouse hovering. $d_{\mathcal{V}}$ denotes the specific value selected by learners after performing $d_{\mathcal{O}}$ on $d_{\mathcal{C}}$. For instance in Table 1, $d_{\mathcal{V}}$takes the value mass = 50 g when the 50 g mass is selected and clicked. Lastly, $d_{\mathcal{T}}$ denotes the time $d_{\mathcal{O}}$ is performed

Table 1: An excerpt of log data, $\mathcal{F}$

Name ($d_{\mathcal{S}}$)	Interface ($d_{\mathcal{I}}$)	Interface component ($d_{\mathcal{C}}$)	…
Ting	Sce	Hypothesis	…
Ting	Sce	Combobox	…
Ting	Sce	Agent	…

This study did not follow the unstructured logging approach discussed in[1]. In[1], log data consisted of have multiple lines consisting symbols like @ and : which editing and filtering processes complicated. Thus, employing database rather than a text file approach is considered to be a better solution as Structured Query Language (SQL) provides efficient way of querying required information. The time in a log data represent the actual time (in the format HH:MM:SS) $d_{\mathcal{O}}$ is performed. Unlike most of the studies (e.g.,[1]) where time represents the total time a particular operation takes to complete. This is to reduce the incurred computational cost needed to compute the time duration for a particular operation.

The raw log data $\mathcal{F}$ depicted in Table 1, however, cannot be directly applicable to a DN or a DDN. Thus, transforming $\mathcal{F}$ into both DN and DDN compliant must be performed. The transformation process has resulted in the following properties:

- $\mathcal{F}$ can be transformed into six relations because INQPRO has six interfaces. Each relation $\mathcal{R}_{\mathcal{I}}$, where $\mathcal{I} \in$ {Sce, Vz, Vf, Fe, Ex, Dc}, corresponds to a DN and an INQPRO's interface
- Each tuple $\mathcal{P}_{\mathcal{I}}$ of $\mathcal{R}_{\mathcal{I}}$ takes the form {d_1, …, d_n} having d_1 denotes the first observable node whereas d_n represents the last observable node of the corresponding DN. The total number of attributes, n, depends on the number of observable nodes in that particular DN. The original log data, $\mathcal{F}$, is transformed into $\mathcal{R}_{\text{Sce}}$ (Table 2) and $\mathcal{R}_{\text{Vz}}$ (Table 3). As shown in Table 2, $\mathcal{R}_{\text{Sce}}$ has four attributes, namely, SA_CBoxMani, SA_ManiAxis, SA_HypoRelation and AQ_ManiVar. Each field in a relation resemble the corresponding observable node in the DN and can take one of the node's states as its value. For instance in Table 2, SA_AskHypo, SA_CBoxMani and AQ_ManiVar have two states, namely, Yes and No, whereas SA_HypoRelation has mastery, partial-mastery and

Table 2: An excerpt of transformed Scenario relation, $\mathcal{R}_{\text{sce}}$

SA_CBox Mani	SA_ManiAxis	SA_Hypo Relation	…
no	no	partial	…
yes	yes	mastery	…

Table 3: An excerpt of transformed Hypothesis Visualization relation, $\mathcal{R}_{\text{Vz}}$

SA_DragMass	SA_Play Ani	AQ_ManiAni	…
yes	yes	non-mastery	…
-		mastery	…

```
Select case
 Case Scenario Interface
     If active section = Hypothesis section Then
       If  1st attempt = non-mastery and
           2nd attempt = partial-mastery Then
           SA_HypoRelation = partial-mastery
       End if
     End if
   :
End case
```

Fig. 3: An excerpt of rules

non-mastery as its states. Conversely, if no evidence can be captured for at a particular interface section (e.g., Hypothesis Formulation Section), the corresponding nodes will be assigned a- and will not be instantiated. As depicted in Table 3, SA_viewGraph is assigned a-indicating that the learner did not view the graph

- Given a relation $\mathcal{R}_{\mathcal{I}}$, the nth tuple of a relation $\mathcal{R}_{\mathcal{I}}$ corresponds to the nth visit to that particular interface $\mathcal{I}$. Referring to Table 1, the learner has exhibited the learning path: Sce $\rightarrow$ Vz $\rightarrow$ Sce $\rightarrow$ Vz. Visiting the Scenario (Sce) interface twice has resulted in two tuples for $\mathcal{R}_{\text{Sce}}$. Similarly, the relation $\mathcal{R}_{\text{Vz}}$ has two records because the Hypothesis Visualization (Vz) was visited twice

Log data often contains multiple entries for a particular interface component. As shown in Table 1, the learner Ting attempted to generate the suitable hypothesis statement for two times, with the first at 15:22:00 while the subsequently one at 15:22:55. Because of multiple entries, instantiation of the variables can be a challenge.

To tackle the problem, a set of rules is consulted before instantiation of variables. Figure 3 shows an excerpt of rules for instantiate of the node SA_HypoRelation. Based on the rules and log data (Table 1), SA_HypoRelation shall be instantiated to partial-mastery.

MATERIALS AND METHODS

Inqpro's dynamic learner models: In this research, discussion on how transformed dataset presented in the previous section could be employed to study the behaviour exhibited by a learner model. To begin with, this article firstly discusses the different versions of INQPRO's learner model and subsequently followed by the method employed to feed dataset into the learner models.

In this research, the INQPRO's dynamic learner model, which takes the form of a Dynamic Decision Network (DDN)[11], was employed to assess the two temporally variable scientific inquiry skills. Employing a DDN is crucial in this research work for three reasons. First, modeling the evolving scientific inquiry skills is difficult. Often, the level of mastery of a scientific inquiry skill at time t depends on its immediate past. Second, freedom in navigating from one interface to another introduces complexity in predetermining a DDN.

A predetermined DDN can easily become computationally intractable as it exhibits 5n state spaces (combination of different navigation paths) with $n \in \{\text{Integer} > 0\}$. Third, employing a static Decision Network will resort to reinterpretation of new evidence over previous evidence[11]. In order to overcome this drawback, a DDN is employed instead of a static Decision network. This research work has employed three different DDN models, $\mathcal{M}_1$, $\mathcal{M}_2$ and $\mathcal{M}_3$, in search of the optimal one. In this subsection, each model will be briefly discussed. Detailed discussion about the models, however, can be found in the author's other work[12,13].

$\mathcal{M}_1$ (Fig. 4) resemble the commonly employed DDN model in the existing probabilistic ITS (e.g.,[5]). The main characteristic of this model is that it aggregates static DNs by introducing arcs (the dotted arcs, Fig. 4) among the dynamic nodes between different time-slices. The dynamic nodes are nodes H, V and K (Fig. 4) which evolve across time. The DDN is generated based on the information contained in log data presented in Table 1.

Each time-slice represent the INQPRO's interface navigated by the learner. Because there are four interfaces navigated by the learner Ting, a DDN with four time-slices was generated. By querying the posterior probabilities of $\mathcal{H}$, $\mathcal{V}$ and $\mathcal{K}$ at time t_3, the final mastery levels of the evolving scientific inquiry skills can be acquired.

Figure 5 shows the DDN model $\mathcal{M}_2$. Different from $\mathcal{M}_1$, $\mathcal{M}_2$ has extra nodes, namely, S^{K}, S^{V} and S^{H}. These nodes are static nodes, which are introduced to capture the levels of mastery of $\mathcal{K}$, $\mathcal{H}$ and $\mathcal{V}$ which are initially unknown and their gradual changes. The gradual changes can be captured through the arc that stretches from a static node to its corresponding dynamic node (e.g., $S^{\text{K}} \rightarrow \text{K}_n^{gui}$). It captures the idea that the belief of dynamic node is conditioned upon its static node.

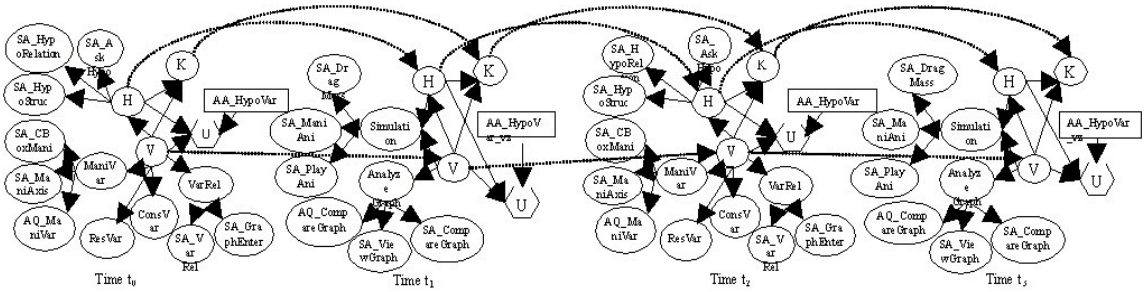


Fig. 4: INQPRO's learner model $\mathcal{M}_1$

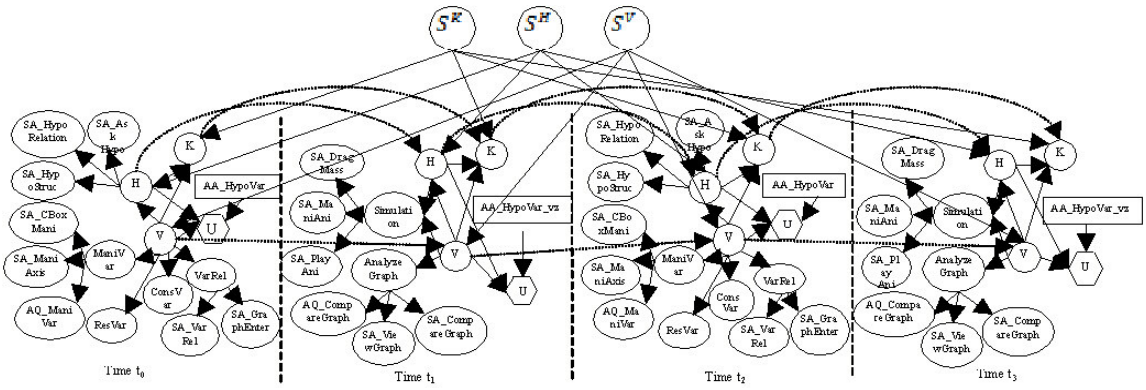


Fig. 5: INQPRO's learner model $\mathcal{M}_2$

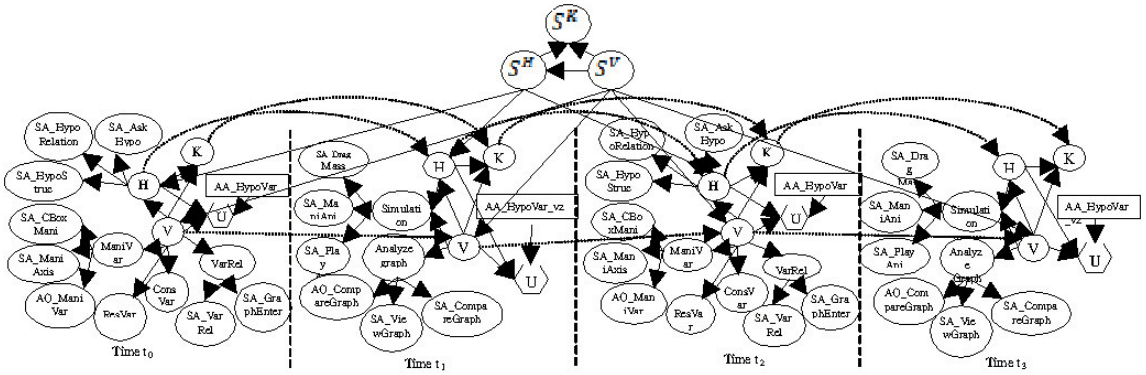


Fig. 6: INQPRO's learner model $\mathcal{M}_3$

Figure 6 shows the third DDN model, $\mathcal{M}_3$, employed in this research work. Similar to $\mathcal{M}_2$, $\mathcal{M}_3$ has three static nodes. However, the only difference is that the causal dependencies between these nodes resemble those between the dynamic nodes in each time-slice: V $\rightarrow$ H, V $\rightarrow$ S and H $\rightarrow$ S. Rather than having the dynamic node (node K) conditioned upon the static node (S^K), the posterior probability of S^K can be retrieved once the probability of S^H and S^V are known.

FEEDING DATASET INTO LEARNER MODELS

Research has shown that there have been three approaches to evaluate a learner model. The first approach is through simulated learners[14]. The challenge of this approach is that the generated simulated learners might not be able to represent the real human learners because it is impossible to consider

all personalz traits. The second approach is via human learners[5,15]. Such approach has been widely accepted and implemented. The approach, however, requires substantial amount of participants each learner model to be evaluated. Not only the number of participants must be sufficient, but the participants must represent three different categories of learners, namely, weak, moderate and advance. Because of the constraints, a fair comparison of all learner models can hardly be made. The third approach is evaluation of learner models through log data, which is implemented and discussed in this research work. Because log data employed for different versions of learner model has been standardized for all the evaluations, a fair comparison between the learner models can be made.

Figure 7 provides a high-level procedure of how the transformed log data (e.g., Table 1) can be used to evaluate different versions of INQPRO's learner model, which take different versions of a DDN. The algorithm takes a log data as input and returns the final mastery levels of scientific inquiry skills. As depicted in Fig. 7, the process begins with feeding L (e.g., Table 1) into the algorithm DDNNodesInstantiation() (Line 1, Fig. 7). Given L, the function RetrieveGUIs() (Line 10, Fig. 7) extracts a sequence of interfaces navigated by a learner and stores them into an array $\mathcal{G}_{DDN}$. A DDN is then generated based on $\mathcal{G}_{DDN}$ via the function GenerateDDN() (Line 11, Fig. 7). Before instantiation

```
1.  Algorithm DDNNodesInstantiation (L)
2.  Input:L = Log data
3.
4.  DDN = DDN generated based on L
5.  G_DDN ← Array of interfaces { G_1,..., G_n} recorded in L
6.  G = An instance of interface in G_DDN
7.  R_G ∈ {R_Sce, R_Vz, R_Vf, R_Fe, R_Ex, R_Dc}
8.  i_G = Number of visits for a particular interface G
9.
10. G_DDN ← RetrieveGUIs (L)
11. GenerateDDN ( G_DDN)
12. Σi_G = 0
13.
14. For each G in G_DDN
15.     i_G = i_G + 1
16.     RetrieveRelation?(R_G)
17.     RetrieveRow ( i_G)
18.     SetNodeEvidence (DDN, G)
19. Next
20. UpdateNetwork ()
21. Output: Posterior probabilities of node H, V, and K
```

Fig. 7: Algorithm for instantiation of nodes in a DDN

of nodes can be done, the each $i_{\mathcal{G}}$ is initialized to zero. For each interface $\mathcal{G}$ in the array $\mathcal{G}_{DDN}$, the corresponding $i_{\mathcal{G}}$ is added with 1. For instance, when the interface Scenario is firstly visited by the learner, i_{Sce} will take the value 1, i_{Sce} will be set to 2 when the similar interface is revisited for the second time. In addition, the value of $i_{\mathcal{G}}$ determines which row of the corresponding relation $\mathcal{R}_{\mathcal{G}}$ is to be retrieved. The next process is to retrieve the corresponding relation $\mathcal{R}_{\mathcal{G}}$. That is, when is $\mathcal{G}$ is assigned the value Scenario, the relation $\mathcal{R}_{Sce}$ will be retrieved. Given the value of $i_{\mathcal{G}}$ and the relation $\mathcal{R}_{\mathcal{G}}$, the corresponding row is retrieved via the function RetrieveRow() (Line 17, Fig. 7). Lastly, the function SetNodeEvidence() (Line 18, Fig. 7) instantiates the nodes of the DDN by using the values from the retrieved row. Once the nodes are instantiated, the DDN can be updated via the function UpdateNetwork() (Line 20, Fig. 7). Updating the DDN allows the posterior probabilities of nodes V, H and K to be revealed.

RESULTS AND DISCUSSION

To demonstrate the importance of log data approach to determine optimal INQPRO's learner model, a two phase empirical study was conducted. Learners participated in both phases of evaluation involved in a series of activities, which had included a session that lasted at most 90 min involving a pretest, a session to INQPRO and a posttest. The results of pretest and posttest were calculated and the learners' interactions with INQPRO were logged. To illustrate the application of log data to finding of optimal learner model, let $\mathcal{F}_{30}$ denotes the first set of log data collected from 30 learners whereas $\mathcal{F}_{46}$ represents the second set of log data collected from 46 learners. There was no similar learner who participated in both phases of evaluation. The matching accuracies of $\mathcal{K}$, $\mathcal{H}$ and $\mathcal{V}$ were computed by comparing the classifications elicited by the DDN models with the results obtained from the pretest and posttest. During the first evaluation phase, $\mathcal{M}_1$ was employed in the INQPRO learning environment.

Table 4 shows the matching accuracies elicited by $\mathcal{M}_1$. The low accuracy for $\mathcal{V}$ at the pretest was largely due to the misclassification of learners into partial-mastery level while in actual fact these learners were graded as non-mastery by the pretest. Such difference was largely because learners did learn about variables while they were attempting the pretest.

Table 4: Accuracies given by $\mathcal{M}_1$ using $\mathcal{F}_{30}$

#(%) matched classification		
	Pretest (n = 30)	Posttest (n = 30)
$\mathcal{K}$	24(80.0)	25(83.3)
$\mathcal{H}$	10(33.3)	20(66.7)
$\mathcal{V}$	4(13.3)	22(73.3)

Table 5: Matching accuracies given by $\mathcal{M}_2$ and $\mathcal{M}_3$ using $\mathcal{F}_{30}$

# (%) matched classification				
	$\mathcal{M}_2$		$\mathcal{M}_3$	
	Pretest	Posttest	Pretest	Posttest
$\mathcal{K}$	22(73.3)	4(13.3)	22(73.3)	25(83.3)
$\mathcal{H}$	17(56.7)	20(66.7)	20(66.7)	20(66.7)
$\mathcal{V}$	15(50.0)	22(73.3)	15(50.0)	22(73.3)

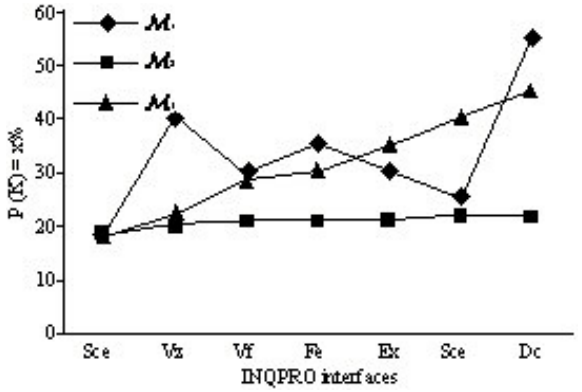


Fig. 8: The mastery level of $\mathcal{K}$ as modeled by $\mathcal{M}_1$, $\mathcal{M}_2$ and $\mathcal{M}_3$

The similar set of log data was given to six domain experts to further investigate the behaviour. Despite the promising results demonstrated at the posttest section in Table 1, the experts rejected $\mathcal{M}_1$. The experts argued that the mastery levels of the skills should not differ greatly from one interface to another (Fig. 8). Thus a low overall average (58.53%) was given by experts to $\mathcal{M}_1$[11].

Table 5 displays the matching accuracies of $\mathcal{M}_2$ and $\mathcal{M}_3$ using the similar set of log data. Although an increment of 36.7% can be observed for $\mathcal{V}$ at the pretest when $\mathcal{M}_2$ was employed, $\mathcal{K}$ dropped to 13.3%. When the probabilities of $\mathcal{K}$ are plotted on graph, only a small amount of changes can be observed ($\mathcal{M}_2$, Fig. 8). Such phenomenon is due to the fact that probabilities cannot propagate from the children nodes of S^K - S^K itself.

Table 6: Comparison of matching accuracies given by $\mathcal{M}_1$, $\mathcal{M}_2$ and using $\mathcal{F}_{46}$

Percentage of matched classification (n = 46)						
	$\mathcal{M}_1$		$\mathcal{M}_2$		$\mathcal{M}_3$	
	Pre test	Posttest	Pre test	Post test	Pre test	Posttest
$\mathcal{K}$	21.7	52.2	21.8	15.2	93.5	78.9
$\mathcal{H}$	73.9	73.9	41.3	71.8	97.8	76.1
$\mathcal{V}$	13.1	67.4	36.9	67.4	52.2	67.4

That is, S^K is d-separated from the parent nodes of K. As shown in Table 5, $\mathcal{M}_3$ performed better than $\mathcal{M}_2$. The matching accuracy for $\mathcal{K}$ with respect to the posttest has increased from 13.3% ($\mathcal{M}_2$)-83.3% ($\mathcal{M}_3$). With the results obtained from the first phase of evaluation, it was concluded that $\mathcal{M}_3$ has somehow depicted the expected modeling behaviour with the accuracies elicited by domain experts, pretest and posttest.

The second phase of evaluation was conducted to further investigate the performance of $\mathcal{M}_3$. $\mathcal{M}_3$ was integrated into INQPRO in the second phase of evaluation. Results given by $\mathcal{M}_1$, $\mathcal{M}_2$ and $\mathcal{M}_3$ are shown in Table 6. The results for $\mathcal{M}_1$ and $\mathcal{M}_2$ were obtained by feeding them with the pre-processed log data $\mathcal{F}_{46}$. $\mathcal{M}_3$, as hypothesized, has again outperformed $\mathcal{M}_1$ and $\mathcal{M}_2$.

In conclusion, although there can only be a DDN model to be integrated into INQPRO at one evaluation phase, employing log data approach allows different models to be evaluated with the assumption that learner's interaction patterns unchanged over a period of time. From the results obtained from the evaluation phases, it is concluded that $\mathcal{M}_3$ is the optimal learner model for INQPRO.

CONCLUSION

Researchers in the field of cognitive psychology often rely on log data analysis to study human properties and behaviours. Log data have also been employed by researchers to study the effectiveness of software, so that the functionalities contained with it can be tailored to wide range of users. Log data analysis, however, has not been widely employed to evaluate the appropriateness of a Bayesian learner model employed within a learning environment. As an attempt to contribute to field of user modeling particular towards the Bayesian Intelligent Tutoring Systems, this article presented a detailed discussion on how log data could be firstly pre-processed, transformed and fed into the proposed DDN models, before the obtain the optimal learner model was

obtained. A detailed discussion on instantiation of nodes of a DDN is presented via an algorithm. It begins with feeding the algorithm with raw log data, to extracting related information from the transformed log data and finally with instantiation of the DDN. In this study, the log data gathered from the first phase of empirical evaluation was firstly pre-processed and subsequently transformed to fit DDN. Similar set of the transformed log data were then fed into $\mathcal{M}_2$ and $\mathcal{M}_3$. The empirical results suggested $\mathcal{M}_3$ as the most suitable learner model. This study continued with $\mathcal{M}_3$ in the second phase of evaluation. When the transformed log data obtained from second phase of evaluation were fed into $\mathcal{M}_1$ and $\mathcal{M}_2$, as hypothesized, $\mathcal{M}_3$ has again outperformed the other DDN models.

REFERENCES

1. Hulshof, C.D., 2004. Log File Analysis. In: Encyclopedia of Social Measurement, Kempf-Leonard, K. (Ed.). Elsevier, pp: 577-583. URL: http://www.elsevier.com/wps/find/bookdescription.cws_home/702429/description.
2. Ericsson, K.A and H.A. Simon, 1993. Protocol Analysis: Verbal Reports as Data. MIT Press, Cambridge, MA. URL: http://www.jstor.org/pss/3151491.
3. Reid, D., J. Reid, J. Zhang and Q. Chen, 2003. Supporting scientific discovery learning in a simulation learning environment. J. Comput. Assist. Learn., 19: 9-20. DOI: 10.1046/j.0266-4909.2003.00002.x.
4. de Jong, T., 2006. Computer simulations: Technological advances in inquiry learning. Science, 312: 532-533. DOI: 10.1126/science.1127750.
5. Murray, C., K. VanLehn and J. Mostow, 2004. Looking ahead to select tutorial actions: A decision-theoretic approach. Int. J. Artificial Intell. Educ., 14: 235-278. URL: http://iospress.metapress.com/content/rld5ak9x82rpa9xn/.
6. Linn, M.C., 2004. Designing the knowledge integration environment. Int. J. Sci. Educ., 22: 781-796. URL: http://www.ingentaconnect.com/content/routledg/tsed/2000/00000022/00000008/art00002.
7. Veermans, K. and W.R. Van Joolingen, 2004. Combining heuristics and formal methods in a tool for supporting simulation-based discovery learning. In: Proceeding of Intelligent Tutoring Systems, 217-226. DOI: 10.1007/b100137.
8. Reiser, B.J., I. Tabak, W.A. Sandoval, B. Smith, F. Steinmuller and T.J. Leone, 2001. BGuILE: Stategic and Conceptual Scaffolds for Scientific Inquiry in Biology Classrooms. In: Cognition and Instruction: Twenty Five Years of Progress, Carver, S.M. and D. Klahr (Eds.). Mahvah, Erlbaum, NJ. URL: http://www.developmentalpsychologyarena.zcom/books/Cognition-and-Instruction-isbn9780805838244.
9. Pearl, J., 1988. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, San Mateo, CA: Morgan Kaufmann. URL: http://portal.acm.org/citation.cfm?id=52121.
10. Chi, M.T. and R.D. Roscoe, 2002. The Processes and Challenges of Conceptual Change. In: Reconsidering Conceptual Change: Issues in Theory and Practice, Limon, M. and L. Mason (Eds.). Netherlands: Kluwer. DOI: 10.1007/0-306-47637-1_1.
11. Schafer, R. and T. Weyrath, 1997. Assessing Temporally Variable User Properties with Dynamic Bayesian Networks. (eds A. Jameson, C. Paris and C. Tasso) Proceedings of the User modeling. DOI: 10.1.1.48.3325.
12. Ting, C.Y. and M. Reza Beik Zadeh. A decision-theoretic approach to scientific inquiry exploratory learning environment. Lecture Notes Comput. Sci., 4503: 85-94. DOI: 10.1007/11774303.
13. Ting, C.Y. and S. Phon-Amnuaisuk. Modeling and intervening across time in scientific inquiry exploratory learning environment. J. Educ. Technol. Soc., (In-Press).
14. Eva Millán and J.L. Pérez-de-la-Cruz, 2004. A bayesian diagnostic algorithm for student modeling and its evaluation. J. User Modeling User-Adapted Interaction, 12: 281-330. DOI: 10.1023/A:1015027822614.
15. Bunt, A. and C. Conati, 2003. Probabilistic student modelling to improve exploratory behaviour. J. User Modeling U ser-Adapted Interaction, 13: 269-309. DOI: 10.1023/A:1024733008280.