Interrogative Sentence Generation and Dialogue Management in Intelligent Tutoring System

Zhixiang ZHANG[a], Xiongwei SHANG[b]

Department of computer engineering, Naval academy of engineering, Wuhan, 430033, China

[a]email: hgzzx@sina.com, [b]email: 971981210@qq.com

Keywords: Tutoring System; Dialogue management; Natural language generation

Abstract. Intelligent tutoring is a hot field in the study of information-based teaching. This paper proposed a series of methods for building ITS in training of operational regulation domain. To make the dialog between system and users more intelligent, a frame-based knowledge representation is proposed, and a template-based generation method for interrogative sentences are used. An algorithm to assess the correctness of the user's answer and to find the error point if not correct is also proposed. The proposed methods make it convenient to build a ITS with high intelligence and high efficience.

Introduction

Intelligent Tutoring System (ITS) is a computer-based tutoring system, trying to satisfying the need for on-demand tutoring among students nowadays[1][2]. The use of Intelligent Tutoring Systems (ITSs) in education has increased considerably over the last years.

There are many operation rules or regulation that should be strictly abided in many industrial domain, such as regulation governing marine engineroom fire fighting, electrician operation rules,et.al. Operators must receive special trainings and strictly abide by operating procedures. Damage controlling(DC) is a typical domain of operational regulation training [3][4]. ITS for such domain such as DC enables and manages the desired learning processes, and makes convenient decisions about different issues such as the materials that are shown at each moment for different profiles, or motivational and emotional aspects.

Knowledge-representation is the field of artificial intelligence that focuses on designing computer representations that capture information about the world that can be used to solve complex problems. There are many knowledge representation formalisms, including semantic nets, Frames, Rules, and ontologies. The integration of Frames, rules, and object-oriented programming was significantly driven by commercial ventures and various research projects. ITSs may choice elaborately one or more represnetations for their domain. For example, [5] uses a domain specific notation to represent both the problem structure and the current state of the resolution process.

The assessment of user's reply during conversations with the system is important in ITSs[6]. The evaluator tries to evaluate the user progress throughout the tutoring process. Every time the user answers the questions, the evaluator captures the data and evaluates the user's performance. Several techniques such as latent semantic analysis (LSA) and those based on machine translation evaluation methods have been proposed to support automatic evaluation of summaries. [7] proposes an ensemble approach that integrates LSA and n-gram co-occurrence for English language examinations. User who failed to answer a question correctly will have to do correction and the agent will provide them with hints or notes for the user reference. [8] studied the choice of hints and prompts.

The main objective of this study is provide a systematic method to implement a lightweighted ITS for training about operation ruls, which can:

- automatically generate interogative sentences based on the instructional knowledge the user is asked to know and the answering history;
- automatically assess the answers of the user;
- iterate throught the steps above, until the user answers the question correctly or he has tried

a predefined times;

- judge the user's action.

The contributions of the paper are: (1) a frame-based representation of instructional knowledge is proposed, extending the classical slot with OR,AND and SEQ aspects; (2) An algorithm to assess the correctness of the user's answer and to find the error point if not correct is also proposed; (3) a integerated dialogue management framework is designed. The proposed methods make it convenient to build a ITS with high intelligence and high efficience for training of operational rules or regulation.

The Model of Instructional Knowledge

There are many knowledge representation methods such as semantic nets, Frames, Rules, and ontologies. In training domain of operational rule/regulation, an instance of operational knowledge describes what to do, how to do and sometimes why to do. Borrowed from the frame-based knowledge representation, an operational knowledge is called frame containing so-called slots, including condition, subject, tools, places, and actions. A frame describes a rule of operation.

For example, in the domain of ship damage controlling, there are rules as follows:

- Frame 1: In condition of oil fire in the main carbin, only the halon fire extinguisher, foam fire extinguisher or carbon dioxide fire extinguishers can be used to fight fire.
- Frame2: In condition of the failure of portable fire extinguisher, you should isolating cabins and Extinguishing Fire first.
- Frame3: The steps of Isolating cabins and Extinguishing Fire are turning off vent valve, closing the oil valve, and closing the hatch.
- Frame4: The fire boundary should be setting at No. 1, No. 2, No. 3 compartments.
- ……

A frame may contains one or more slot with null value, for example, frame 1 only contains "condition" and "tools" slots.

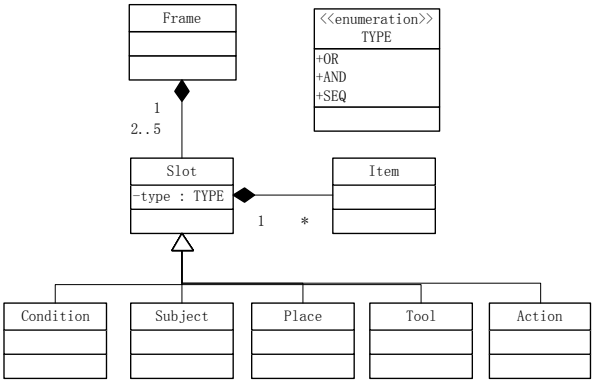

Fig 1. The model of instructional knowledge

According to the contents, every slots can be of type OR, AND or SEQ.

For slots of type OR, the correct answer has many alternatives. For example, in frame 1, halon fire extinguisher, foam fire extinguisher or carbon dioxide fire extinguishers are all correct answers;

For slots of type AND, the correct answer contains many items. For example, in frame 4, the places contians three compartments. The missing of each compartment results in a uncorrect answer;

Type SEQ is similar with the type AND, except for that the slots of type SEQ are forced to be in some order. For example, in frame 3, the actions should be turning off vent valve first, then closing the oil valve, and closing the hatch at last. The missing of each steps or the steps out of order will result in uncorrect answers.

We use the Object-Oriented concept to describe instructional knowledge. The model the instructional knowledge is described as Figure 1.

As shown in figure 1, a frame contains 2-5 slots, and a slot contains many items. A slot has a

property called “type”,which is enumerational.

Template-based Interrogative Sentence Generation

There are many technologies for generation of natural language[9][10][11]. As a frame has at most five slots *condition, subject, tools, places,* and *actions,* at most five interrogative sentences can be generated corresponding to each slots.

The interrogative templates in terms of *condition* are (1) In what *condition*, <*subject*> should do <*actions*> using <*tools*> at <*places*>? And (2) In what condition except <A>, should <*subject*>do <*actions*> using <*tools*> at <*places*>?

The interrogative templates in terms of *subjects* are (1) In condition of <*condition*>, who should do <*actions*> using <*tools*> at <*places*>?(2)In condition of <*condition*>, except <A>,who should do <*actions*> using <*tools*> at <*places*>?

The interrogative templates in terms of *tools* are (1)In condition of <*condition*>, what tools should the <*subject*> use to do <*actions*> at <*places*>? And (2) In condition of <*condition*>, except <A>,what tools should the <*subject*> use to do <*actions*> at <*places*>?

The interrogative templates in terms of *places* and *actions* can be designed in the same way.

In templates above, the user is asked using (1) when he/she has not tried the question ever; once he/she has provided partial answer, template (2) is used. In template (2), the phase “except <A>” must be determined.

Given a frame and the slot where the aimed interrogative sentence is about, assuming that the corresponding slot with type mtype has n items, there are 3 related data structures: solutions $S=\{s_1,s_2,\ldots s_n\}$, user’s answer $A=\{a_1,a_2,\ldots a_n\}$, and answer history $F=\{f_1,f_2,\ldots f_n\}$.

$$f_i=\begin{cases}0 & a_i \neq s_i\\ 1 & a_i = s_i\end{cases} \quad (1)$$

f_i indicates the correctness of the user’s answer about ith item.

There are different strategies to generate interrogative sentences for different types.

Here we assume the slot be action. For other slot, the method is the same.

1.Type SEQ

Assuming the action slot be of type SEQ, the contents can be described as an ordered list { s1,s2,…sn } with the length of n. If the user has answered the first k items correctly, to induce the user to achieve the right solution, the interrogative sentence in terms of tools should be “In condition of <condition>, except <X> what should the <subject> do in sequence at <places> using <tools>”. Assuming the previous answer is $\{ a_1,a_2,\ldots a_n \}$, X can be generated as $X= < s_1,s_2,\ldots s_k >$ where $k=\min\{i|a_{i+1} != s_{i+1}\}$.

2. Type AND

Assuming the action item be type AND, the contents can be described as an disordered list $\{ s_1,s_2,\ldots s_n \}$ with the length of n. If the user achieves the answer $A=\{ a_1,a_2,\ldots a_n \}$, the interrogative sentence in terms of tools should be “In condition of <condition>, except <X> what should the <subject> do else at <places> using <tools>”. Assuming the previous answer is $\{ a_1,a_2,\ldots a_n \}$, X can be generated as $X=\{si| si \notin \{ a_1,a_2,\ldots a_n \}\}$.

3. Type OR

Compared with the type SEQ and AND, there are two sub-strategies for the processing of type OR. Because the items of type OR have many alternatives, the simple strategy is to check whether the user knows one correct solution, while the sophisticated strategy is to check whether the user knows all the alternatives. So the interrogative sentence can be generated regardless of the answer history, or in the same way as of type AND.

For an interrogative sentence, assuming that the corresponding slot with type mtype has n items, there are 3 related data structures: solutions $S=\{ s_1,s_2,\ldots s_n \}$, user’s answer $A=\{ a_1,a_2,\ldots a_n \}$, and answer history $F=\{ f_1,f_2,\ldots f_n \}$.

To the dialogue system, the text produced by the Human-Machine-Interface (HMI) does not automatically have any meaning that might be useful in that domain. So a Natural Language

Understanding(NLU) component called semantic parser parsed the text semantically and produces an interpretation of the meaning of that utterance. There are many technologies aiming NLU[12][13].The interpretation here is A={ a_1,a_2,…a_n }.

Dialogue Management

Spoken dialogue between people and machines is increasingly common. A spoken dialogue system is intended to engage effectively in dialogue with people. The system may fully control the path of the dialogue (system initiative) or it may permit callers to interrupt and volunteer information (mixed initiative)[14]. We emphasis the judgement of user's answer, and inducing the user toward the path to correct solution. So the system initiative strategy is used.

The first module is *Judge*, which judges whether the user's answer is correct according to the slot's property S and type, the user's answer A and the abstraction F, and returns a boolean value indicate the correctness of the answer, and the first error position index when the answer is not correct.

```
int Judge(S,A,F,type, &index)
{
   if (type == SEQ || type == AND)
   {
      index=the first i for F_i==0
      if (index==N) return 1;
      else    return 0;
   }
   if (type == OR)
   {
      m=the first i for F_i==1
      index=0;
      if (m==N) return 0;
      else    return 1;
   }
}
```

DM_Slot is the most important component. It generates interrogative sentence first, and then sends the sentence to HMI, waits for user's reply; parses the user's reply, updates the abstraction F; at last it calls the module *Judge* to determine the correctness of the answer, and gives hints or prompts if the answer is uncorrect.. The above steps are repeated until the user reaches a correct solution or he has tried a predefined times. *DM_Slot* returns a boolean indicating whether the user reaches the correct solution, and two viariables N_p and N_h indicating the times of hinting and prompting if *DM_Slot* return true, which is used by Assess to compute a overall score measuring the degree of the user.

```
bool DM_Slot(slot, &N_p, &N_h)
{
   count=0;
   while(count<LIMITCNT)
   {
      GenINS(slot);//gerate interrogative sentence;
      //Send the sentence to TTS module, wait for user's reply;
      //parse the user's reply, update slot.F
      if( Judge(slot.S,slot.A, slot.F, slot.Type,index) ==1)
         return true;
      count++;
      //Given Hints or Prompts acoording to the value of index
      // N_p++ or N_h++
   }
   return false;
}
```

DM is the main procedure. It selects a slot from a frame using SelectSlot, and then calls DM_Slot to manage the dialogue in terms of the slot. If DM_Slot returns true, which means the user has given a correct answer, Assess is called for the overall score. If DM_Slot returns false, which means the user fails to reach a correct answer, the overall score is zero.

```
int DM(frame)
{
   slot =SelectSlot(frame);
   if( DM_Slot(slot, Np, Nh))
      score=Assess(Np,Nh);
   else
      score=0;
   return score;
}
```

Module *SelectSlot* select a slot from frame according to different strategies. For example, selectSlot may iterate through all the 5 slots, or slect a slot randomly, or select a slot according to the user's context.

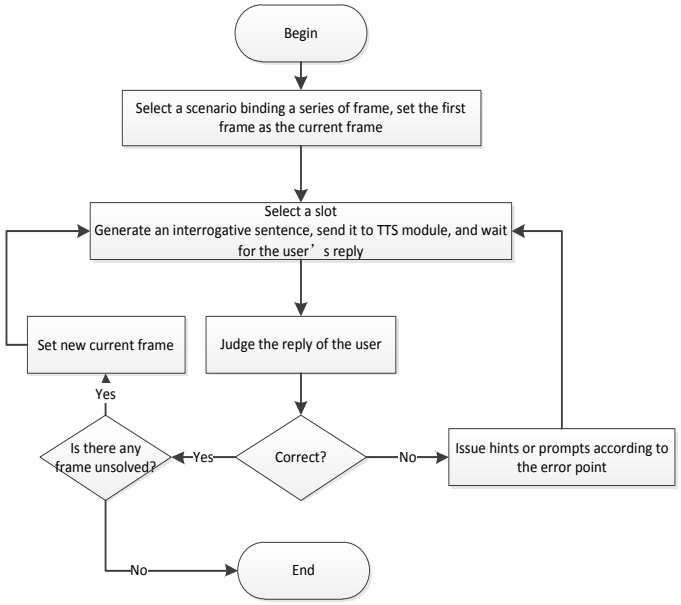

Fig. 2. The proposed dialogue management method

The user can be assigned an overall score for a frame according to the number of errors before he/she reaches the right solution. The fact that after each error solution the system has to issue a prompt or hint makes it possible to substitute the number of errors with the combination of number of hints(Nh) and the number of prompts(Np). Our measuring function is as follows:

$$\text{Score} = \begin{cases} 0 & \text{unable to reach a right solution} \\ \dfrac{K}{C_1 N_p + C_2 N_h + K} & \text{reaches a right solution} \end{cases} \tag{2}$$

Where C_1,C_2,K are constants.

Score(N_p,N_h) is a monotone decreasing function about N_p and N_h. it follows the intuition: the more the user receives hints or promts, the less he gets the score.

Figure 2 shows the workflow of the dialogue management module.

Conclusions

In this sample paper, we have presented series methods to deal with the building of ITS for training of operational rule/regulation. A ITS for damage controlling training is implemented

following the above method, and highlights the potential of these methods to implement ITS of operational regulation training domain.

References

[1] Litman D J, Silliman S. ITSPOKE: An intelligent tutoring spoken dialogue system. Proceeding of HLT-NAACL 2004 Conference, pp.5-8, 2004.

[2] Pedro J. Munoz-Merino,Manuel Fernandez Molina, Mario Munoz-Organero, Carlos Delgado Kloos. An adaptive and innovative question-driven competition-based intelligent tutoring system for learning. Expert Systems with Applications ,Vol. 39, No. 8, pp. 6932-6948, 2012.

[3] Pon-Barry, Heather, et al. "Contextualizing reflective dialogue in a spoken conversational tutor." Educational Technology & Society Vol.8,No.4, pp.42-51, 2005.

[4] Pon-Barry, H., Clark, B., Bratt, E., Schultz, K., & Peters, S. Evaluating the effectiveness of SCoT: a Spoken Conversational Tutor. In Mostow, J. & Tedesco, P. (Eds.), ITS Workshop on Dialog-based Intelligent Tutoring Systems, pp. 23-32, 2004.

[5] David Arnau, Miguel Arevalillo-Herráez, Luis Puig, José Antonio González-Calero. Fundamentals of the design and the operation of an intelligent tutoring system for the learning of the arithmetical and algebraic way of solving word problems. Computers & Education, Vol. 63, pp. 119–130,2013.

[6] Zipitria, I., Elorriaga, J. A., Arruarte, A., & de Ilarraza, A. D. From human to automatic summary evaluation. In Intelligent Tutoring Systems. Springer Berlin Heidelberg, pp. 432-442, January,2004.

[7] Yulan He a, Siu Cheung Hui, Tho Thanh Quan. Automatic summary assessment for intelligent tutoring systems. Computers & Education, Vol. 53, pp. 890-899, 2009.

[8] Bram E. Vaessen, Frans J. Prins, Johan Jeuring. University students' achievement goals and help-seeking strategies in an intelligent tutoring system. Computers & Education, Vol. 72, pp. 196–208, 2014.

[9] Feng Yuqiang,Wu Fei and Huang Tiyun. Research on Automatic Model Selection Based on Natural Language Understanding (in Chinease). Journal of The China Society For Scientific and Technical Information, Vol.21, No. 1, pp. 17-20,2002.

[10] ZHU Feng ,CAO Cungen. A Survey of Narrative Generation Approaches(in Chinease). Journal of Chinese Information Processing,Vol. 27, No. 03, pp. 33-40, 2013.

[11]Raquel Hervasa, Virginia Franciscoa, Pablo Gervasb. Assessing the influence of personal preferences on the choice of vocabulary for natural language generation. Information Processing and Management,Vol. 49, No. 4, pp.817-832,2013.

[12] Foo S, Li H. Chinese word segmentation and its effect on information retrieval. Information processing & management, Vol.40, No.1, pp. 161-190, 2004.

[13] Oliver Lemon. Learning what to say and how to say it: Joint optimization of spoken dialogue management and natural language generation. Computer Speech and Language, Vol. 25, No. 2, pp.210–221,2011.

[14] Wilks, Yorick, et al. "Some background on dialogue management and conversational speech for dialogue systems." Computer Speech & Language, Vol. 25, No.2, pp. 128-139, 2010.