Viva voce for Student Assessment and Learning

Nandan Parameswaran[a], Ravi Gorthi[b], and Ajit Tiwari[c]
School of Computer Science and Engineering, University of New South Wales, Sydney, Australia[a]
Department of Computer Science, LNM-IIT, Jaipur, India[b]
Babu Banarasi Das University, Lucknow, India[c]
Corresponding Author Email: paramesh@cse.unsw.edu.au

Structured Abstract

BACKGROUND
Viva voce is an assessment technique often used for evaluating the skill and knowledge of a student. In this paper, we have argued that viva voce provides an opportunity not only for assessing a student's knowledge on a specific topic, but also for imparting knowledge wherever the student has been assessed to lack that knowledge in that topic.

PURPOSE
Our goal is to investigate methods for implementing vivo voce process using ontologies and dialog programming techniques.

METHOD
We model the viva voce based assessment as a dialog between the Instructor *Agent* (a software with which a human instructor interacts) and the student who is being assessed. The architecture of our system consists of the following four modules.
Ontologies: There are two ontologies - (a) the main domain ontology; and (b) the supporting deviation ontology.
Dialog Modules: There are two dialog tree modules: (i) the assessment dialog tree module; and (ii) the deviation management dialog tree module.

OUTCOMES
We present an ontology based approach for viva voce driven assessment where the domain ontology is used to guide the assessment and the other ontology, called the Deviation ontology, to help impart knowledge that was found lacking in the student. Strategies for dialog programming for viva voce in the area of Algorithms in field of Computer Science are presented.

CONCLUSIONS
Our approach to use viva voce both for assessment and imparting knowledge to the student who is being assessed uses novel techniques and it is feasible to implement them. However, it is knowledge intensive in nature. The domain ontology constructed should enable student assessment effective and natural. The deviation ontology requires deeper understanding of the solutions to the problems presented in the assessment, and its concepts not only span the problem domain but also the domain of solutions to the problems.

KEYWORDS
Assessment, viva voce, ontology, dialog programming, software agents.

Introduction

Teaching students in the discipline of computer science has always been very challenging particularly in the areas such as programming, algorithms and data structures since in these areas the student is expected to reason about the dynamic behavior of objects from their static descriptions. Assessment is typically done by conducting examinations, quizzes, assignment tasks and viva voces. Assessing a student's skill is also a difficult task due to the limited time devoted for assessment and the associated mental stress the student typically goes through during the assessment process. The results of an assessment may not only be qualitative, but also quantitative so that they can be interpreted unambiguously and used effectively for organizing materials that may be taught in future.

In this paper, we argue that the purpose of the assessment phase should not only be to evaluate the knowledge level of the student but also to impart knowledge where the student has been found to lack that knowledge during the assessment phase. In particular, we propose a scheme of viva voce based assessment technique where we demonstrate how imparting knowledge can be achieved while assessing the student. We present our approach in the section on Methodology where we structure the questions and the related implied knowledge in a list of nested hierarchy of trees and perform the assessment by traversing the trees systematically.

Background

Teaching is increasingly populated by a collection of personalized services where a student has several options to choose the way he wants to learn and get assessed. Assessment is an important component in learning. As teaching techniques go through changes, new strategies are used in assessment techniques. Peer-marked assignments have been used successfully to improve student engagement (Holland, Brain & Mowjoon, 2013). Adaira, Jaegerb & Pua (2012) have reported a computer-aided method of assessing students' attitudes for general safety, work area tidiness and cleanliness, care and good use of hand tools, and accuracy and testing of equipment, and to compare the results with those obtained using oral assessment. Markulis & Strang (2008) present several guidelines for implementing oral examination methodologies.

Effective oral and written communication skills are teachable and learnable. Bowering (2013) reports strategies and assessment tasks that can be used in many technically based engineering units to develop oral and written communication skills. Assessment techniques can be instrumental in driving students' learning and engaging them in a subject (Ramsden, 2007). In addition to measuring the level of understanding in the subject, assessment may also be used to motivate students to acquire new knowledge (Ooi & Buskes, 2011). Easa (2013) has attempted to assess graduate attributes by defining knowledge elements appropriate for the graduate attributes and then embedding them in multiple-choice questions, and the method has been claimed to perform well in certain engineering courses.

Multiple choice question (MCQ) structure has been shown to improve the performance of the students (Klimovskia & Cricentia 2013, Landrum 1993, Rodriguez 2005) implying that this structure may also suitable for embedding (a weak form of) learning. Thorpe (2013) has evaluated the assessment of certain engineering courses and found "reflections" on assessment techniques result in an improved, more authentic assessment that better addresses industry and professional requirements. While investigating the role of assessment in deep learning, Hanandeha (2013) has argued that assessment items offer opportunity to link theoretical subjects to more practical engineering skills, hence engaging students in deep learning.

Methodology

An assessment activity can be modelled as a process. Figure 1(a) shows a model of the traditional assessment process described using BMPN (White & Miers, 2008) where the instructor specifies a task (T1), student performs it (T2), and the instructor assesses the performance (T3). However, in a complex task such as T2 in Figure 1(a), exceptions often occur. Exceptions in this context refer to situations such as where the student does not understand the specifications correctly, or only knows partial answer that he, under perhaps emotional stress, is not able to present coherently within the available time constraints, etc. Figure 1(b) shows a modified version of the assessment process (proposed in this paper) where: (i) we have captured the exceptional situations as Performance Failure; and (ii) provide additional knowledge to the student and give another opportunity to re-perform the task, which is then assessed by the instructor at T3.

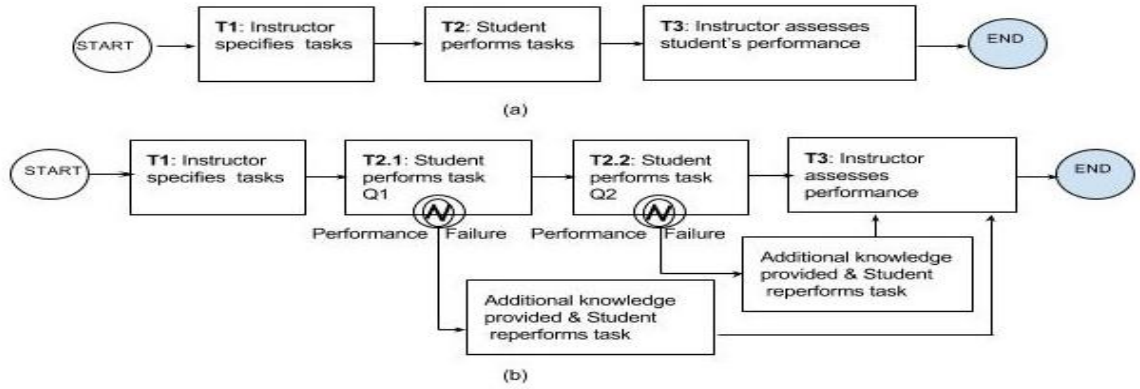

Figure 1. (a) Traditional assessment model; (b) Proposed assessment mode

Viva voce is a form of assessment where the student is assessed in an informal and less stressful setting. In our approach, we use two sub ontologies from the domain in which we want to assess the student: Assessment Ontology (AO) and Deviation Ontology (DO). (See Figure 2.) AO is a set of concepts and relations (from the domain) that we use primarily for assessing the student. When a student is assessed, failures (exceptional situations) occur. In order to recover from the exceptions, we use the concepts from DO. We model the viva voce as a dialog between an Instructor *Agent* (IA) (a software with which a human instructor interacts) and the student who is being assessed. The dialog consists of two threads: Main Dialog (MD) thread, and Deviation Dialog (DD) thread. The primary purpose of the MD thread is to assess the student's knowledge, while the primary purpose of the DD thread is to provide knowledge that was found lacking in the student while he was assessed in the MD thread.

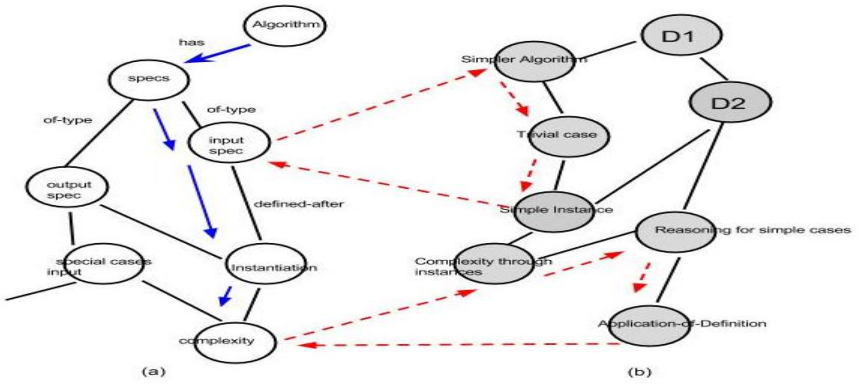

Figure 2. (a) Assessment Ontology (AO); (b) Deviation Ontology (DO).

In the MD thread, IA selects a concept C_i from AO and performs the assessment by initiating a dialog. The performance of the student is assessed during the dialog and if the performance is considered satisfactory, IA selects another concept C_j from AO, and the process is repeated. However, if the performance of the student in C_i is not satisfactory, an exception is said to have occurred. This can typically happen when the student lacks adequate knowledge in the concept C_i. In order to handle this situation, IA initiates a DD thread. In this thread, IA conducts a dialog using the concepts from DO where the purpose of the dialog is to provide knowledge that the student was found to be lacking. At the end of the thread, IA returns to the MD thread and performs a reassessment of the student in the concept C_i. Once IA finishes with assessing the student in the concept C_i, IA selects another concept C_j and repeats the process until it decides to terminate the assessment and compute the final score.

In Figure 2(a), for example, the first concept selected for assessment is ALGORITHM. Assuming that the performance of the student in this concept was satisfactory, the instructor agent IA moves on to the concept SPECS and then to INPUT-SPEC where the student's performance is assessed as unsatisfactory and thus an exception is said have occurred. To handle the exception, a deviation from assessment is made to Figure 2(b) where knowledge about INPUT-SPEC is provided to the student using the concepts SIMPLE-ALGORITHM, TRIVIAL-CASE and SIMPLE-INSTANCE. (This is done by the deviation dialog thread DD.) Assuming that the student has performed satisfactorily in SIMPLE-INSTANCE, the assessment resumes from INPUT-SPEC (now with the remaining questions). A similar exception is also handled at the concept COMPLEXITY where the problem of lack of adequate knowledge in COMPLEXITY is solved using COMPLEXITY-THROUGH-INSTANCES, REASONING-WITH-SIMPLE-CASES and APPLICATION-OF-DEFINITION.

Conceptual similarity between Assessment Ontology and Deviation Ontology

When an exception occurs during the assessment in a concept C_i in AO, it is necessary to select a sequence of concepts $<D_1, D_2,...,D_n>$ from DO such that the similarity measure between C_i and D_j should not be less than a predefined threshold value, for any j. This will make sure in practice that the deviation concepts D_j are semantically closer to C_i (at an acceptable level) which is a necessary condition for providing knowledge about Ci.

Viva Voce Programming

The questions that are used in the viva voce are organized as trees as shown in Figure 3. Each node in Figure 3(a) consists of a set of questions that test the understanding of a concept from the assessment ontology AO. Thus, the node labeled C1 from AO will have questions primarily related to C1. In this case, C1 is the first concept that the agent wants to assess the student in. The assessment begins by initiating the MD thread. If the assessment ends normally, an evaluation u1 of the performance of the student in C1 is now produced by the agent, and the agent then moves on C2 and the process is repeated. On the other hand, if an exception was encountered during the assessment of C1, the agent initiates the DD thread and the control moves to the tree (b) in Figure 3 and u=0. The DD thread begins (just as MD thread did Figure 3(a)) a dialog using the questions associated with the node D1 where D1 denotes a concept from DO. The requirement on the concept D1 is that the questions in D1 (along with similar questions in the other concepts D2, etc. in the tree) provide the knowledge that the student was found to be lacking while being assessed in C1. At the end of the dialog at D1, an assessment v1 is produced, the agent moves on to D2. If an exception is encountered at D1, the agent initiates another DD thread and the control moves to the tree (c) in Figure 3 (tree not shown).

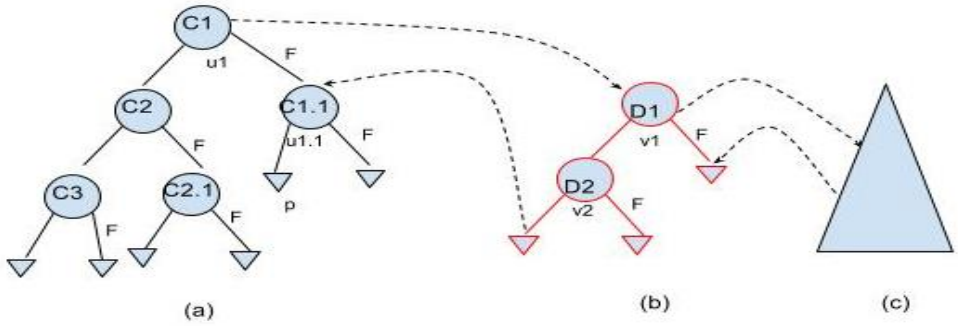

Figure 3. Dialog structure. (a) Main Dialog thread for concepts from Assessment Ontology; (b) Deviation Dialog thread for knowledge provision using Deviation Ontology(first level); (c) Deviation Dialog thread for knowledge provision using Deviation Ontology (second level). Triangles denote empty nodes. Score p = f(u1,u1.1, g1(v1,v2)).

Finally, when the dialog at D2 is finished, control is returned to node C1.1, and the MD thread continues the assessment starting from the concept C1.1.

Computing Score

The performance of a student in an assessment can range from *all-correct* to *all-fail.* In *all-correct*, the student performs satisfactorily at every concept along the leftmost path of the assessment dialog tree (Figure 3(a)), leading to a score of 100%. In the *all-fail* case, the student fails at every concept in the assessment dialog tree and in every deviation dialog tree, leading to a score of 0%. The score in all other cases may be calculated incrementally at each node as we navigate through it. The overall assessment will be a function of the performance at every node. For example, assuming that the assessment at C1.1 ended successfully (left branch), let u1.1 be the score at this node. The score u1.1 is used along with the scores at other nodes to produce a final overall score as: p = f(g1(list of scores in tree (a)), g2(list of scores in tree (b)), g3(list of scores in tree (c), ...) for some functions f, g1, g2, etc. For example, in Figure 3, assuming that tree (c) is null, the score returned from tree (c) is 0, and the final score at node C1.1 in tree (a) will be given by p = f([u1.1], g([v1,v2])). The score for the entire viva voce is given by the score at the root node C1 of the tree in Figure 3(a).

Strategies for Dialog Tree construction

The tree structure for MD is organized so that each path of the tree to its leaf node forms an assessment session consisting of a strategically acceptable sequence of concepts from the root to the leaf. The concepts along a path may be organized from most abstract to most concrete (top down organization of concepts) or concrete to abstract (bottom up organization of concepts). While the construction of the assessment ontology AO is fairly straightforward, the construction of the deviation ontology DO will be a challenging task. Following are some of the concepts we have identified for DO in the field of Algorithms in Computer Science.

SIMPLIFY-CONCEPT: In this, when the result of the assessment (of the student's knowledge) in a given concept C_i from AO is unsatisfactory, we simplify the concept C_i to yield D_i and assess student about D_i with the purpose of providing an opportunity to the student to understand D_i. For example, if C_i is *Recursion*, then D_i can be PROCEDURE-INVOCATION where a procedure P1 invokes a different procedure P2.

TRIVIAL-CASE: A trivial case of a concept C_i may be obtained by considering instances that form the extreme values of the concept. For example, a trivial case of SORTED-SEQUENCE may be a sequence consisting of no elements at all, or just one element in the sequence.

ABSTRACTION-FROM-INSTANCES: A given concept C_i sometimes may be too abstract that the student finds it difficult to reason with, and in such cases, we use instances of C_i to help in reasoning with the abstract concept C_i.

KEY-OPERATIONS: When the definition of a concept C_i involves several operations that the student finds hard to cope with, we simplify the assessment to test the student with those

operations that are central to the concept C_i. For example, the definition of the concept of STACK involves operations such as PUSH, POP, OVERFLOW, UNDERFLOW, IS_EMPTY, GET_TOP, etc. of which PUSH and POP may be considered as the key operations central to the notion of STACK. Though an understanding of these concepts only leads to a partial understanding of STACK, it can still help in understanding a more elaborate definition of STACK.

Assessment Policies

Typically a navigation begins at the root of the MD tree and visits nodes in the DD trees starting from the first tree. With the organizational structure given as a tree in Figure 3, it is neither feasible nor necessary to navigate through all nodes during an assessment. Sometimes looking at the current performance of the student, it may be possible to predict a node in a tree from where assessment can safely continue. This will make assessment more efficient and avoid unnecessary work for the student. Thus, it is useful to define a set of policy rules that constrain the navigation of nodes depending upon the background of the students who are assessed. We propose the following policy rules.

Assessment-only policy: This policy does not entertain failure at any node (concept) in the assessment ontology AO, and it permits the navigation of the nodes only along the leftmost path of the MD tree. This corresponds to the traditional assessment technique.

Deviation-always policy: This policy permits deviation for every concept in the MD tree as well as in the DD tree. This policy makes all options in the system available to the student, but results in excessive cost, and corresponds to the most “patient” strategy of assessment.

Repeat policy: In this, a selected set of concepts (both in MD and in DD threads) are visited more than once, and the assessment test is repeated. This is sometimes necessary since repetition is known to strengthen familiarity resulting in improved understanding and performance.

Select-on-the-fly-deviation policy: The performance at every node in all trees are assessed and a running score of the assessment is computed and used for estimating a sequence of nodes that will be visited in the near future. This policy thus monitors the performance of the student and accordingly selects concepts to test that may be appropriate for the performance level of the student. It thus can provide efficiency in the assessment process.

Learn-to-navigate policy: In this, the system uses the past assessment history including the sequence of nodes visited in each tree and the performance at each node to learn the next most probable tree and a node in it, and visits that node to continue the assessment.

Architecture of the Assessment System

Figure 4 shows the architecture of the propo**s**ed viva voce based assessment system.

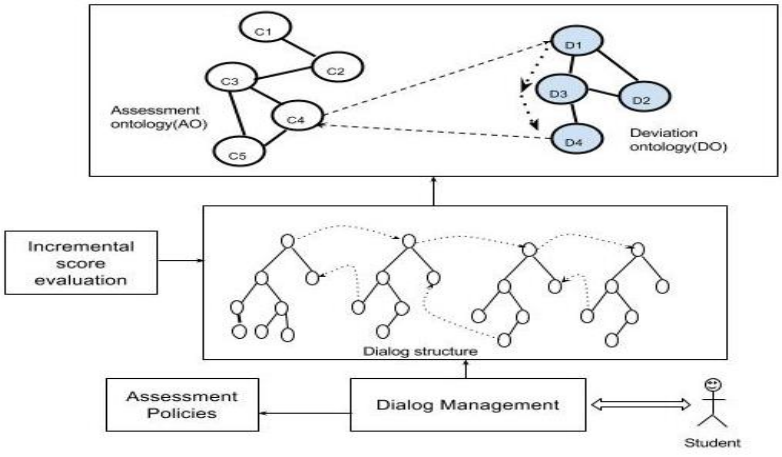

Figure 4. A viva voce based assessment system

The dialog trees are designed based on the concepts from AO and DO. Before the assessment begins, the system interactively configures the policies and the score evaluation function. The number of deviation trees that may be used in an assessment will depend on the student background knowledge and the assessment policies.

Sample Session

Our chosen domain is Algorithms where we have focused on search and sorting algorithms. Specifically, the search algorithms included sequential search, and the binary search; and the sorting algorithms included: selection sort, insertion sort, merge sort, quick sort, shell sort, and merge sort. The assessment ontology AO is a collection of concepts from this domain and presently it consists of 55 concepts (some of which are listed below). The deviation ontology DO is more interesting and presently it contains 38 concepts (some of which were discussed above).

The Domain of algorithms

Algorithm is a description of a method for solving a well specified problem using automatic means such as a computer, and often it consists of a sequence of steps that the executor must faithfully execute within a reasonable time. Algorithms have interesting properties: concise description, unique interpretation, each step well defined requiring only (reasonably) finite resources to execute; amenable to formalization; implementability using a computing or mechanical device; analyzability for its computational complexity; and often short descriptions. These characteristics provide good reason for automating assessment. Traditional techniques for assessment in this domain include tasks such as: modify a given algorithm to include some new cases; prove certain property of a given algorithm; implement the algorithm in a programming language and study its performance on a given data, etc. They are traditionally done in the examination hall (written exam), or in a computer lab with limited access to computer resources and time.

Assessment Ontology

Some of the important concepts from the domain of Algorithms include: SPECS - a formal description of input/output relation; INPUT-SPEC - a set of valid instances; OUTPUT-SPEC – a set of instances corresponding to the input instances; ALGORITHM – a sequence of steps where each step is executable with finite resources; COMPLEXITY - a measure of time and space requirement of an algorithm; SPECIAL CASES – an extreme case of input; INPUT-ABSTRACTION - an abstract input specification; INSTANTIATION - instances such as examples. BASICS - knowledge considered basic to the domain; and OVERALL-FUNCTIONALITY - overall function of the algorithm.

Following is a hand simulated session of assessment where the instructor agent IA assesses a student's knowledge in the algorithm called SEQUENTIAL-SEARCH. The algorithm, given a list of items a[ ] searches for the item v and reports whether v is present in a[ ] or not, and if present, where in a [ ] the item v is present. The algorithm is coded in the C programming language and is shown below.

```
/* The SEQUENTIAL-SEARCH algorithm.   */

 int seqSearch(int a[], int v, int low, int high)
 { int k;
    for (k = low; k <= high; k++)
      if (v ==  a[k]) return k;
    return -1;
  }
```

Assessment regarding this algorithm involved the following major concepts (which were part of the ontology AO): INPUT, OUTPUT, SPECS, ALGORITHM, COMPLEXITY, and SPECIAL CASES. Under each one of these categories there were several sub concepts of AO and DO. In particular, there were 29 assessment concepts in AO and 38 concepts in the deviation ontology DO. There were 192 nodes in the trees (both the assessment dialog tree and the deviation dialog trees included), and there were 100 leaves. This means there are 100 ways a student can be assessed depending upon his knowledge level.

A student can be assessed in this algorithm regarding several concepts defined in AO. If we restrict the assessment to the computational aspects of the algorithm only, the concepts included in the assessment ontology AO were: OVERALL-STRUCTURE, KEY-CONCEPT, TEST-PARTIAL-UNDERSTANDING, and RECOGNITION-IN-ALTERED-CASE. The concepts in the deviational ontology DO included: OVERALL-STRUCTURE, OVERALL-STRUCTURE-SIMPLIFIED, ABSTRACT-FROM-INSTANCE, INFER-RELATED-ABSTRACTION, TEST-KEY-OPERATION-IN-ALGORITHM and EXPLOIT-OPPORTUNITY. (The trees have not been shown due to lack of space.)

The worst case response occurred when the student failed to answer any of the following questions asked in this order.

1. [AO:OVERALL-STRUCUTRE] Q: How many iterations would you require for this algorithm?
2. [DO:OVERALL-STRUCTURE-SIMPLIFIED] Q: Let there be 10 items in the array. You have to match the given item with each item in the array. How many times would you match?
3. [DO:SIMPLE-INSTANCE] Q: Now, how many iterations?
4. [DO:ABSTRACT-FROM-INSTANCE] Q: If there are n items in the array, how many matches?
5. [DO:INFER-RELATED-ABSTRACTION] Q: How many iterations would you need in the algorithm?
6. [AO:TEST-PARTIAL-UNDERSTANDING] Q: Consider the C code for sequential search. (a)Verify (to yourself) that there is only one loop in this program. (b)A bug has been inserted in this program. Locate the bug.
7. [DO:TEST-KEY-OPERATION-IN-ALGORITHM] Q: Hint - correct the *if* statement. Now, locate the bug.
8. [AO:RECOGNITION-IN-ALTERED-CASE] Q: Consider this program.

```
 int seqSearch(int a[], int v, int l, int r)
 { int i;
    for (i = l; i <= r; i++)
            if (v ==  a[i]) return a[i];
return -1
}
```

   Is this program still correct?
9. [DO:EXPLOIT-OPPORTUNITY] Q: Compare the program at step (2) with the program at step (3). What is the merit in (2)?

In the above trace which was produced when the student answered all questions wrongly, we used concepts from DO to simplify the problem and offer answers so that at each step the student learns something that was relevant to the current stage of assessment. In particular, the following knowledge segments were incrementally added to the student's understanding during the assessment though his performance score was low: (a) searching a list of 10 numbers uses 10 iterations; (b) searching a list of n numbers uses n iterations; (c) in an iteration, the key operation to perform is the comparison between the given value v and an element a[i] in the list to check if v = a[i]; and (d) sometimes, there are opportunities to produce more information than returning just the answer. A student who answered all questions correctly will have gained no additional knowledge explicitly from the dialog.

Discussion and Conclusion

Viva voce is a flexible and yet powerful technique, and to our knowledge this area has remained unexplored and no automated tools are available that are based on a clear model of the viva voce process. We have proposed a dialog structure that aims to serve two purposes at the same time: (a) assess the student's knowledge in a domain; and (b) provide adequate knowledge to the student using another ontology called the deviation ontology. The assessment technique proposed is more systematic and exhaustive in nature compared to the traditional practice which is based on casual manual techniques. However, our approach is knowledge intensive. The domain ontology constructed should enable student assessment effective and natural. The deviation ontology requires deeper understanding of the solutions to the problems presented in the assessment, and its concepts not only should span the problem domain but also the domain of solutions to the problems. As with any other automated techniques, the knowledgebase in the system must be continually updated to prevent it from getting outdated and ineffective. Since knowledge and skills that need to be assessed vary with each student, the assessment process should match the student's knowledge level and personality. In this context, incorporating learning threads to the existing assessment threads will make the system more effective.

References

Adaira, D., Jaegerb, M., and Pua, J.H., (2012) Assessing Student Attitudes Using a Computer-Aided Approach, Proceedings of the AAEE 2012 CONFERENCE , Melbourne, Australia, 2012.

Bowering, R.(2013) Strategies for developing effective communication skills in engineering students, Proceedings of the 2013 AAEE Conference, Gold Coast, Queensland, Australia, Copyright © Bowering, 2013.

Easa,S.M. (2013) Assessing graduate attributes in large classes without sampling, Proceedings of the 2013 AAEE Conference, Gold Coast, Queensland, Australia.

Hanandeha, A. E. (2013) Encouraging students' deep learning through assessment, Proceedings of the 2013 AAEE Conference, Gold Coast, Queensland, Australia.

Holland, B., Brain, T., and Mowjoon, M.(2013) Running before you can walk: blended learning in collaborative spaces, Proceedings of the 2013 AAEE Conference, Gold Coast, Queensland, Australia, Copyright © Bowering, 2013.

Klimovskia, D., and A. Cricentia. (2013) Does the multiple choice question structure in examinations have an effect on student performance, Proceedings of the 2013 AAEE Conference, Gold Coast, Queensland, Australia

Landrum, R. E., Jeffrey, R., Cashin, A., Kristina, D., & Theis, S. (1993) More evidence in favor of three option multiple choice tests. Educational and Psychological Measurement, 771-778.

Ooi, A , & Buskes, G. . (2011) A survey of strategies for feedback and assessment in engineering subject: Discussions and examples. Proceeding of the AAEE Conference 2011: Western Australia, Australia.

Ramsden, P. (2007) Learning to Teach in Higher Education, London; New York.

Rodriguez, M. (2005) Three options are optimal for multiple-choice items: A meta-analysis of 80 years of research. Educational Measurement: Issues and Practice, 24(2), 3-13.

Thorpe, D. (2013) Reflections on assessment: comparison of assessment processes for postgraduate Engineering management courses, Proceedings of the 2013 AAEE Conference, Gold Coast, Queensland, Australia.

White, S.A., and Miers, D. (2008) BPMN Modeling and Reference Guide: Understanding and Using BPMN, Future Strategies Inc.

Markulis,P.M., and Strang,D.R.(2008) Viva Voce: Oral Exams as a Teaching and Learning Experience, Developments in Business Simulation and Experiential Learning, 35,118-127.