Journal of Computer Science 4 (5): 415-420, 2008
ISSN 1549-3636

The Design of ADAT: A Tool for Assessing Automata-Based Assignments

Zarina Shukur and Nurul Fatihah Mohamed
Department of Computer Science, Fakulti Teknologi dan Sains Maklumat,
Universiti Kebangsaan Malaysia, 43600 Bangi, Malaysia

Abstract: This study describes the design of an automatic assessment system for assessing an automata-based assignment. Automata concept is taught in several undergraduate computing courses such as Theory of Computation, Automata and Formal Languages and Compilers. We take two elements into consideration when assessing the student's answers; static element and dynamic element. The static element involves the number of states (initial and final as well) and the number of transitions. Whilst the dynamic aspect involves executing the automata against several test data. In this work, we rely heavily on the JFLAP for drawing and executing the automata.

Key words: Automatic assessment, marking tool, automata diagram assessment

INTRODUCTION

The aim of this study is to describe the design of a tool for automatic assessment of an automata-based assignments. The development of tools for automatic assessment has generated considerable interest over the past years. One of the earliest that we know is the tool that asses the assignment for numerical analysis subject in 1960-s[6]. From that time onwards, many assessment tools had been developed. However, the computing subjects are the focus of the researchers at that time. Van Verth[20] stated in his thesis that until the mid of 70-s, the focus of the assessment is on the program correctness. However, the focus of assessment became optimal between 70-ies and early 80-ies[12]. We believed that the birth of software engineering discipline in 1963 gave impact to this issue. The automatic assessment systems such as[3,5,6,8,11,13,15,16,20,21] used several software quality factors in their design.

Apart from that, many automatic assessment tools for various kinds of subjects such as essays-based assignment[2] (Foxley and Lou, 1994): Diagram-based assignment[17,19]: (Ali et al. 2007); as well as the new design of tools for computing program assignments[4,7,9,10,18] have been developed.

Automatic assessment tools help educators by providing a consistent, accurate and efficient marking process. These tools can also assist students in improving their assignment, providing sufficient and fast feedback. From the managerial perspective, these tools can aid in handling a large number of students assignments.

In this study, we propose a tool, which we call Automata Diagram Assessment Tool (ADAT) to assess an automata-based assignments. Automata-based assignments usually involve automata diagrams. Although some assessment systems to assess diagrams[19] already exist from the literature that we have done, there is no system to specifically assess an automata-based assignment.

AUTOMATIC MARKING CRITERIA

The prospective users of making systems are so large and therefore marking systems for any subjects ideally should be highly needed. However, like many other software from research product, some automatic marking system are being use in house actively, while some others are just left as a research product. Therefore, in order to have these marking tools accepted and used by the targeted user, it should fulfill certain criteria such as standards, guidelines, or benchmark.

In pedagogic perspective, any automatic marking system should comply with its criteria. Higgins and Bligh[7] have analyzed whether computer based assessment (CBA) meets pedagogic criteria for measuring the quality of assessment suggested by Brown et al. (1996). Table 1, which is taken directly from[7] shows the analysis.

Based on the analysis, they concluded that CBA meets 7 out of 10 criteria, therefore CBA can be said to have concrete pedagogic benefits.

From the modeling aspect, [9]try to formalize automatic marking system in order to have a depth

Corresponding Author: Zarina Shukur, Department of Computer Science, Fakulti Teknologi dan Sains Maklumat, Universiti Kebangsaan Malaysia, 43600 Bangi, Malaysia

Table 1: Application of assessment criteria to CBA[7]

Criterion	Application to CBA
Valid	Will measure specified coursework aspects assuming good initial assessment design
Reliable	The same assessment process will run for each submission: Consistency is absolute
Fair	Design-dependent: CBA has no inherent advantages
Equitable	The same assessment process will run for each submission: Discrimination is non-existent
Formative	CBA provides a good opportunity to run assessment frequently throughout the learning process and to provide multiple submission with full feedback each time
Timely	CBA provides a good opportunity to run assessment frequently throughout the learning process
Incremental	Design-dependent: CBA has no inherent advantages
Redeemable	CBA is suited to allowing multiple submissions should the designer wish this
Demanding	Design-dependent: CBA has no inherent advantages
Efficient	Considerable time and other resource saving to be made: Originally a motivation for CBA's development

understanding of the systems. They considered a set of document as the main object of any automatic marking systems. The document can be either a correct answer script, incorrect answer script, or the marking result. To make it clearer, we rewrite their formalism by using Z notation and focus only one marking system. Let (document) be the set of any documents, the following is the Z specification for the automatic marking system:

AMSystem

$$\begin{aligned}
&\text{Answers}: \mathbb{P}\,\text{DOCUMENT}\\
&\text{Correct Answers}: \mathbb{P}\,\text{DOCUMENT}\\
&\text{Incorrect Answers}: \mathbb{P}\,\text{DOCUMENT}\\
&\text{Outputs}: \mathbb{P}\,\text{DOCUMENT}\\
&\text{Marking}: \text{DOCUMENT} \rightarrow \mathbb{P}_1\,\text{DOCUMENT}\\
&\hline
&\text{Correct Answers} \subseteq \text{Answers}\\
&\text{Incorrect Answers} \subseteq \text{Answers}\\
&\text{Correct Answers} \cup \text{Incorrect Answers} = \text{Answers}\\
&\text{Correct Answers} \cap \text{Incorrect Answers} = \{\}\\
&\text{Outputs} \cap \text{Answers} = \{\}\\
&\text{Dom marking} = \text{Correct Answers}\\
&\forall a: \text{DOCUMNET} \bullet a \in \text{dom marking} \Rightarrow \text{marking}\,(a) \subseteq \text{outputs}
\end{aligned}$$

The specification describes there are two type of answers; correct answer and incorrect answer. An answer script cannot be correct or incorrect at the same time. The other document is output document. Only correct answer can have output documents. In[9] formalism, it seems that they did not give any output to the incorrect answers. Also they did not further mentioned about the specification of correct answer. Shukur *et al*.[16] stated that in general, any automatic marking systems involve two types of input; the students answers and the model answer. The model answer can be either in the form of possible answer, or in the form of answer specifications. The improvement of this formalism is quite interesting but will not be further discussed in this study.

The other important thing is the standard interface and infrastructure. Infrastructure is for the management aspect of the course such as the collection of answer scripts, distribution of questions etc. CourseMaster (or previously known as Ceilidh) is one of the example. As well as the interface should embed the standard icon and functions as proposed in the discipline of user interface design.

By having a standard interface and infrastructure, we hope that the developed marking tool can be just plugged and played on it, for any subject. And much better if for one particular subject, more than one marking tool can be installed so that the user can have a choice of marking tool to use.

AUTOMATA-BASED ASSIGNMENT

An automaton is a simplified, formalized model of a computation. Figure 1 shows an example of automata diagram. The basic automata only consist of states (represented by circles) and transitions (represented by arrowed lines). It may be used to compute the membership function for a language, as well as it can also model other kinds of things such as a state-based system. In undergraduate curriculum, automaton is normally used to compute the membership function for a language.

Automata-based assignments usually involve the drawing of automata diagram. Therefore, one way to check the correctness of an automata diagram is by 'running' it against several strings of the respective described language and several strings that should be rejected by it. It can be done by hand, but as expected, it will inherit all the problems created by manual checking. Thanks to[14], as we can execute automata diagram using JFALP. However, for one particular problem, we can have many forms of automata diagrams. So, which is the best? Hence, we enhance the marking approach of automata diagram by embedding quality factors. By employing two of the software quality factors i.e maintainability and readability, we define the following factor to be included in assessing an automata diagram.

The correct input strings: A correct diagram should accept all correct input strings.

The incorrect input strings: A correct diagram should reject all incorrect input strings.

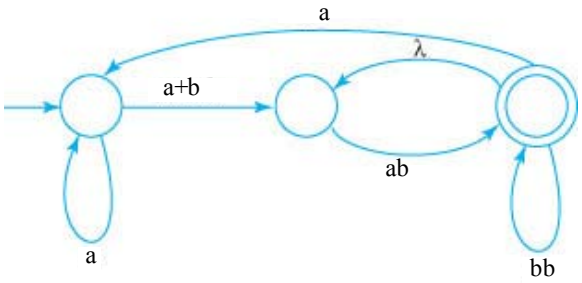

Fig. 1: An automata diagram

The number of states: A good diagram should have at most the same number of states with the model answer.

The number of transitions: A good diagram should have at most the same number of transitions with the model answer.

The naming of the states: A good diagram should be easy to understand. Therefore, the naming of the states should be meaningful

THE DESIGN OF AUTOMATA DIAGRAM ASSESSMENT TOOL

ADAT receives two type of input, answer scripts written by students and marking specifications provided by the educator. Marking specification consists of model answer, test weight and set of test data. Two main process of ADAT are the dynamic assessment and static assessment. Dynamic assessment involves testing the answer script against several test data. While static assessment aims to compare the number of states and transitions from answer script with the model answer. In addition, the name of the states will be checked by using language database. Figure 2 illustrates the overall process.

Marking scheme: In order to calculate the marks, six test weights are needed. Four test weights are for the four static elements and the other two are for the two dynamic elements. At a moment we exclude the naming factor of the states. In order to aid the educators, we set the default value 5 for each static element and 20 for each dynamic element. Based on this default value, we have 60 as the total marks. Due to the importance of dynamic elements compared to static, we therefore set the default value of test weight as it is. However, the value can be easily reset.

The static elements refer to the statistic of the elements in the diagram that are; the number of states, initial states, final states and transitions. By assuming

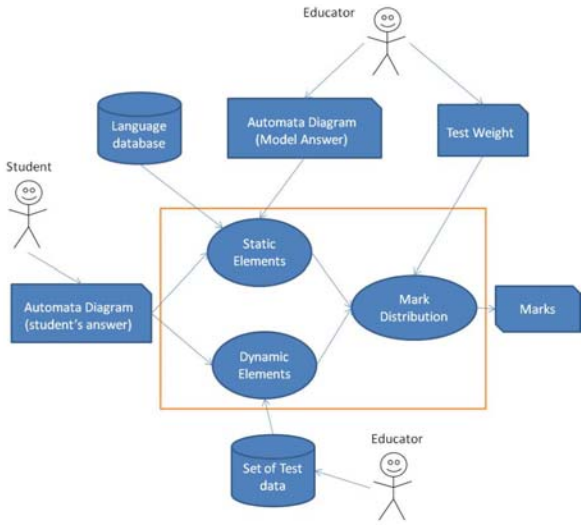

Fig. 2: ADAT conceptual model

that the model answer is the best answer, therefore, any student's answer that contains more than the number of elements of model answer will get less mark. Hence, we propose a simple marking scheme as follows:

If (Number of States-Model)≥(Number of States-Student) then
    MS-1 = Weight for States
Else
    MS-1 = ((Number of States-Model)/(Number of States-Student))*Weight for States

If (Number of Initial States-Model)≥(Number of Initial States-Student) then
    MS-2 = Weight for Initial States
Else
    MS-2 = ((Number of Initial States-Model)/ (Number of Initial States- Student))*Weight for Initial States

If (Number of Final States-Model)≥(Number of Final States-Student) then
    MS-3 = Weight for Final States
Else
    MS-3 = ((Number of Final States-Model)/ (Number of Final States-Student))*Weight Final States

If (Number of Transition-Model)≥(Number of Transition-Student) then
    MS-4 = Weight for Transition
Else

MS-4 = ((Number of Transition-Model)/ (Number of Transition- Student))*Weight Transition

Therefore, total static mark obtained is MS-1+MS-2+MS-3+MS-4.

The dynamic elements focus on the acceptance and rejection of set of strings. For the marking purposes, two set of strings are prepared; the accepted strings and the rejected strings. If the student's automata diagram accepts all strings in the set of accepted string, then he/she will obtain a full mark for this criteria. So as the rejected string. If their answer rejects all the strings in the set of rejected strings, he/she will get a full mark. The following is the calculation scheme:

Accepted strings: MD-1 = (Number of accepted strings)/(Total number of accepted strings)*weight for accepted strings

Rejected strings: MD-2 = (Number of rejected strings)/(Total number of rejected strings)*weight for rejected strings

Therefore, total dynamic mark obtained is MD-1+MD-2.

Total marks = (Total Static marks obtained+Total Dynamic marks obtained)/(Total Static marks+total Dynamic marks) * 100

Implementation of ADAT: ADAT is implemented by using Java. The students are required to draw the automata by using JFLAP. Then the *.jff file of JFLAP will processed by ADAT in order to obtain the static elements of the diagram. In order to produce the static mark, the static elements of the student's answer will be compared to the model answer, by using the proposed marking scheme. The model answer, test weights and two sets of test strings will be provided by the educator. As for the dynamic marking, we run the automata diagram against two sets of test strings. The strings will be classified into two, a set of accepted string and a set of rejected string. We make use of the JFLAP technology as it can 'execute' the automata diagram. The result of the execution will be used to calculate the dynamic mark.

Figure 3 and 4 shows the test weight input screen and output screen of ADAT, respectively.

Testing of ADAT: To test ADAT, we selected one question from the final examination of a course on

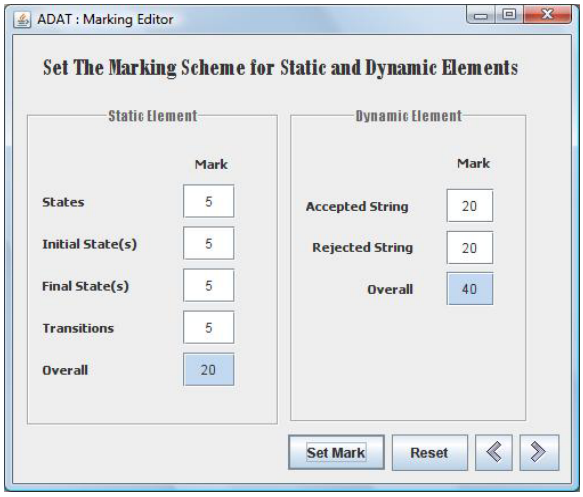

Fig. 3: Test weight input screen

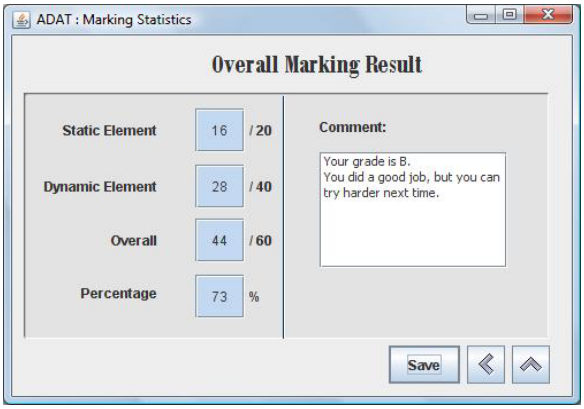

Fig. 4: Marking result screen

Theory of Computation. Out of 46 students' answer scripts we only used 10 of them. The original students' answers are hand-written, hence we redrawn them using JFLAP. The answers are then marked by using ADAT approach. We prepare three marking schemes with different weights as in Table 2. The aim to have three marking schemes is to analyze the effect of different weights. The marking results from ADAT are then compared with the marking result from the human marker. In this case, we only used one human marker.

Table 3 shows the overall result. The table is divided into four main columns which represent the result from human marker, ADAT with the first marking scheme, second marking scheme and third marking scheme, respectively. Each main column is divided into two sub columns that represent the student identity and the result. The list of student has been sorted based on their marks.

Table 2: Marking schemes

	Dynamic scheme			Static scheme		
Weight	Accepted strings	Rejected strings	No. of state	No. of initial state	No. of final state	No. of transition
Marking Scheme 1	10	5	5	2	2	5
Marking Scheme 2	20	10	5	2	2	5
Marking Scheme 3	10	5	0	0	0	0

Table 3: Overall result

Human marker		ADAT with marking scheme 1		ADAT with marking scheme 2		ADAT with marking scheme 3	
Stud 6	100	Stud 6	100.0	Stud 6	100.0	Stud 6	100.0
Stud 7	100	Stud 7	100.0	Stud 7	100.0	Stud 7	100.0
Stud 3	100	Stud 4	87.4	Stud 4	91.7	Stud 3	100.0
Stud 4	80	Stud 3	74.1	Stud 3	83.0	Stud 4	100.0
Stud 1	80	Stud 9	71.6	Stud 1	65.9	Stud 1	66.7
Stud 9	80	Stud 1	65.5	Stud 9	65.3	Stud 9	53.3
Stud 8	40	Stud 8	51.7	Stud 8	50.0	Stud 10	48.9
Stud 10	20	Stu1 0	49.1	Stu1 0	48.9	Stud 8	46.7
Stud 5	0	Stud 5	45.2	Stud 5	43.4	Stud 5	40.0
Stud 2	0	Stud 2	39.6	Stud 2	34.6	Stud 2	24.4

From the table we can see that the human marker and ADAT agree that Stud 5 and Stud 2 obtained the two lowest marks. However the human marker seems quite strict by giving zero marks for the students.

The next two from bottom are Stud 8 and Stud 10 and it is agreed by all the markers. However, for ADAT with marking scheme 3, it gave more marks for Stud 10 compared to Stud 8. This is not a problem because all of the markers (except human) gave a very slightly different mark for both of them. Again, for these two students, human marker is more strict compared to ADAT.

It seems that all markers agreed that the next two students are the same students that are Stud 1 and Stud 9. Human marker gave the same marks for both students. ADAT with marking scheme 1 that is the dynamic and static weight are balance, Stud 9 obtained more marks. With marking scheme 2, that is dynamic weight are higher than static weight, they both obtained nearly similar marks. Whilst for marking scheme 3 that is only dynamic mark is considered, Stud 1 obtained more marks than Stud 9. This means that the automata diagram by Stud 9 is simpler compared to Stud 1. However both diagrams are not totally correct. Human marker classifies the incorrectness by Stud 1 and Stud 9 as the same, without considering the complexity of the drawing. However, by giving more weight on static elements, Stud 9 obtained higher marks than Stud 9, as shown by marking scheme 1. Nevertheless, if we focus on dynamic aspect, Stud 1 will obtained more marks compared to Stud 9, as shown by marking scheme 2 and 3.

The last four students from top are resorted by us so that we can analyzed it easily. The original table are sorted by marks and followed by the student identity. Therefore if they obtained the same marks (as given by ADAT with marking scheme 3) the position should be Stud 3, Stud 4, Stud 6 and Stud 7. However, we rearranged the position as long as the marks are the same. All markers agreed to give Stud 6 and Stud 7 full marks. This means that Stud 6 and Stud 7 is totally correct and the diagram is as good as the model answer. However for Stud 3 and Stud 4, the markers have different evaluation. The human marker gave full marks for Stud 3 and not with Stud 4. However, by considering only dynamic elements, both students obtained full marks. This means ADAT agreed that Stud 3 and Stud 4 have correct answers. Why human marker gave less mark to Stud 4? When we analyzed Stud 4 handwritten answer, we found out that it contained a very small error, like syntax error. However, when we redrawn it to the electronic version by using JFlap, the error is removed as JFlap has certain features that can control us when we draw the automata diagram. Therefore, if any students prepared their diagram by using JFlap, they will not face that problem. As for Stud 3, the answer is correct but the diagram is not as good as model answer. Therefore, Stud 3 could not obtained full marks if static elements are taken into consideration as shown by ADAT with marking scheme 1 and 2.

In summary, based on the testing result, the performance shown by ADAT can be considered as good as a human and more rational then human.

CONCLUSION

The essence of this study is about the design of an automatic marking tool for automata-based assignment, called ADAT. Earlier, we also described about the

environment that might boost the application of marking tools. ADAT has been tested and the initial result looks promising. The ADAT approach is more rational compared to human and more or less similar to the human marker. Our next step is to enhance ADAT to assess as much elements that are related to the theory of computers science subject.

REFERENCES

1. Ali, H.N, Z. Shukur and S. Idris, 2007. Assessment system for UML class diagram using notations extraction. Int. J. Comput. Sci. Network Secur., 7: 181-187.
2. Batten, E., 1994. New computer grading of student prose, using modern concept and software. J. Exp. Educat., 2: 127 142.
3. Benford, S., E. Burke and E. Foxley, 1992. Courseware to support the teaching of programming. In: Proceedings of the Conference on Developments in the Teaching of Computer Science, April, 1992, Canterbury, UK, pp:158-166.
4. Blumenstein, M., S. Green, A. Nguyen and V. Muthukkumarasamy, 2004. An experimental analysis of game: A generic automated marking environment. In: Proceeding of The 9th Annual Conference on Innovation and Technology in Computer Science Education, June 28-30, 2004, Leeds, United Kingdom, pp: 67-71.
5. Brown S., Race P. and Smith B., 500 Tips on Assessment, London: Kogan Page, 1996, ISBN 0749419415.
6. Faidhi, J.A.W., 1986. The complexity analysis of Pascal programs and the application to a university teaching environment. Ph.D. Thesis. University of Brunel.
7. Foubister, S.P., G. Michealson and N. Tomes, 1997. Automatic assessment of elementary Standard ML programs using Ceilidh. J. Comput. Assis. Learn., 3: 99-108.
8. Foxley E., Lou B., A Simple Text Automatic Marking System, Artificial Intelligence and Simulation of Behaviour 94 Conference for: Computational Linguistics for Speech and Handwriting Recognition, Workshop in Leeds University, UK, April 12th, 1994.
9. Higgins, C.A. and B. Bligh, 2006. Formative computer based assessment in diagram based domains. In: Proceeding of The 11th Annual Conference on Innovation and Technology in Computer Science Education, June 26-28, 2006, Bologna, Italy, pp: 98-102.
10. Hung, S., L. Kwok and A. Chung, 1993. New metrics for automated programming assessment. IFIP Trans. A-Comput. Sci. Technol., 40: 233-243.
11. Koike, H., K. Akama and C. Ishikawa, 2007. Toward a software development model for automatic marking software. In: Proceeding of The ACM Special Interest Group on University and College Computing Services 2007, October 7-10, 2007, Orlando, Florida, USA, pp: 190-193.
12. Malmi, L., V. Karavirta and A. Korhonen, 2004. Visual algorithm simulation exercise system with automatic assessment: TRAKLA2. Inform. Educat., 3: 267-288.
13. Marth, U.V., 1994. Kassandra, the automatic grading system. Institute of Advance Computer study, University of Maryland, College Park, Switzerland.
14. Michaelson, G., 1996. Automatic analysis of functional program style. In: Proceeding of the 1996 Australian Software Engineering Conference, Melbourne, Australia, July 14-18, 1996, pp: 38-46.
15. Redish, K.A. and W.F. Smyth, 1987. Evaluating measures of program quality. Comput. J., 30: 228 232.
16. Rodger, S., 2006. Learning automata and formal languages interactively with JFLAP. In: Proceeding of The 11th Annual Conference on Innovation and Technology in Computer Science Education, June 26-28, 2006, Bologna, Italy, pp: 360-360.
17. Saikkonen, R., L. Malmi and A. Korhonen, 2001. Fully automatic assessment of programming exercises. In: Proceeding of The 6th Annual Conference on Innovation and Technology in Computer Science Education, June 25-27, 2001, Canterbury, UK, pp: 133-136.
18. Shukur, Z., E. Burke and E. Foxley, 1999. The automatic assessment of formal specification coursework. J. Comput. Higher Educat. Mass, US, 11: 86-119.
19. Shukur, Z., Y. Away and M.A. Dawari, 2004. Computer-aided marking system for engineering. In: Proceeding of Society for Information Technology and Teacher Education International Conference, March 2004, Atlanta US, pp: 1852-1857.
20. Thomas, P., K. Waugh and N. Smith, 2006. Using patterns in the automatic marking of ER-diagrams. In: Proceeding of The 11th Annual Conference on Innovation and Technology in Computer Science Education, June 26-28, 2006, Bologna, Italy, pp: 83-87.
21. Tsintsifas, A., 2002. Diagram base-computer base assessment. Ph.D. Thesis. Computer Science Department, The University of Nottingham.
22. Van Verth, P.B., 1985. A system for automatically grading program quality. SUNY (Buffalo) Technical Report, 1985.
23. Zin, A.M. and E. Foxley, 1991. Automatic program quality assessment system. In: Proceedings of the IFIP Conference on Software Quality, March 1991, SP University, India.