IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

AUTOMATED ASSESSMENT AND GRADING OF A STRUCTURED C++ PROGRAM USING GENERALISED INTER - LINGUA AND EXPERT SPECIFICATION

[1]Prof. Maxwell Christian, [2]Prof. Bhushan Trivedi, PhD.

[1]Asst. Professor, Faculty of Computer Technology, GLS University, Ahmedabad, India
[2]Dean, Faculty of Computer Technology, GLS University, Ahmedabad, India

Abstract: The automated approach of assessment and grading of a structured C++ program on basis of generalised inter - lingua and expert specification achieves a faster and consistent evaluation of the program on basis of language features used, grades provided and time consumed for the completion of evaluation process. The areas of the program in terms of components where the grades can be enhanced form the achieved grades can also be suggested by such automated process of assessment. Adapting to the novel approach of implementing components of a program also adds to the knowledge base of expert specification for future cycles of assessment and grading. *The work presented here is published for patent at Patent Office Branch, Mumbai, India with the reference number E-12/215/2021/MUM and application number 202121000796*

Index Terms - Automatic evaluation, automatic grading, structural evaluation, inter - lingual representation, component-based evaluation, contextual binding

INTRODUCTION

The process of automatic assessment and grading of a structured C++ program does not fall into the simpler domain due to vast number of affecting factors like development approach, wide variety of language features, the implementation order of different components and the relevance in terms of contextual use of each component. The generalised representation of the program in terms of inter - lingua [1] along with the expert specification assists to address this problem in the simplest manner achieving the assessment as well as proper grading of each component.

MOTIVATION

The structured C++ program targeted for automated assessment can be evaluated on basis of numerous parameters with multiple grading options. Each affecting parameter can have multiple grading options. The automated process saves the time of assessing the same program on multiple developed versions by different students. There also stands a need to not only assess the completeness of a program and grade it, on component basis, but also to provide suggestive measures to the student which can aid the student to achieve better grades and also gradually improvising the student's programming capability. Also, the process of automated assessment should have a feature of adaptiveness which can enhance the knowledge of assessment when encountered with novel approach of implementations of structured programs.

CHALLENGES IN AUTOMATED ASSESSMENT

Formalising the process of automated assessment on basis of inter - lingua and the expert specification is encountered by many hurdles which are mentioned as listed below:

- The generalised inter - lingua needs to be parsed correctly for assessment process
- A single standard process to assess a specific language feature-based implementation is desired
- The most applicable grade from a grade range for a particular language feature use in the component is most crucially desired
- Providing suggestions to students only for the components where grade enhancement is possible
- Adapting the automated process to incorporate novel implementation approach for component of structured program

DEVELOPED SYSTEM

The automated assessment process of evaluating and grading a structured C++ program on basis of the expert specification successfully achieves the following listed:

- Evaluating the existence of components: checking the presence of each component desired for implementation from the student's version of the program. For example, presence of all possible constructors, member functions and operators, etc.
- Selecting an applicable grade from possible grade range for the developed version of component: the model selects an applicable grade from list of possible grades on basis of applicability i.e., for example int data type for count may incur grade of 2 while const int for the same achieves grade of 3
- Formulating the suggestive measure for grade enhancement for each applicable component of the program: In case the student's implemented version of component does not achieve the maximum possible grade, the student is suggested of the possible enhancement e.g., instead of int using const int for count will incur higher grades
- Adapting to the novel implementation approach of the component: for example, in case a student has implemented a desired function using static approach which confirmed by evaluator is added to the specification and evaluation dictionary

The above-mentioned process for automated assessment of a structured C++ program using the inter - lingua and expert specification is depicted in figure 1.

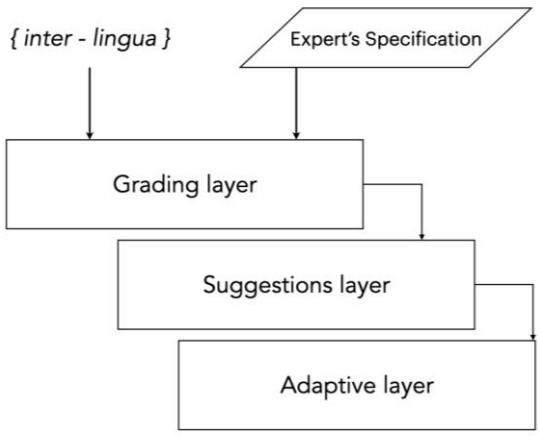

Figure 1. Automated assessment process using inter - lingua and expert specification

The automated process of assessment and grading specified in figure 1 has been successfully tested for various components pertaining to symbolic and value-based constants, functions with and without container-based scope and container based structural declarations. The system depicted in figure 1 also achieves applying the most appropriate grade from possible grade range depending the language-based feature applicable options. The system from figure 1 also achieves the mechanism of providing suggestions at all such components where grade enhancement is possible. Adapting to novel approach of implementing components of a program is also the feature achieved by the system depicted in the figure 1.

ADVANTAGES

The developed phase of automated assessment and grading process based on inter - lingua provides advantages of faster assessment process as compared to manual grading process for the same versions of structured C++ programs. The developed process also has an advantage of consistent assessment and grading of the same program in multiple cycles over a period of time and even after a gap of time. The automated process also demonstrates the advantage of feedback mechanism for the students for their version of program in regards of all such components where grade enhancement is possible. The most sought advantage demonstrated by the automated process is the addition to the knowledge base as and when encountered by a novel approach of implementation of components from the targeted program to automated assessment and grading.

CONCLUSION

The automated approach of assessment and grading a structured C++ program clearly demonstrates faster output of the assessment process. It is also clearly evident that the automated process of assessment demonstrates consistent grading during different cycles of evaluation repeatedly and after variable time gap. The demonstrated automated process is also capable of providing suggestive measures at all applicable components of the program where grades can be enhanced. The most desired feature provided by the automated process is the adaptation to novel implementation approach and addition of the same to the knowledge base of the expert specification of the targeted program for automated assessment

REFERENCES

1. Doshi, J.C.; Christian, M.; Trivedi, B.H. (2014), Effect of Conceptual Cue Based (CCB) Practical Exam Evaluation of Learning and Evaluation Approaches: A Case for Use in Process-Based Pedagogy, Technology for Education (T4E), 2014 IEEE Sixth International Conference on Technology for Education, pp 90 - 94
2. Forsythe, G. E., Wirth, N. (1965). Automatic Grading Programs. Commun ACM, vol. 8, pp. 275- 278.
3. Higgins, C. A., Gray, G., Symeonidis, P., Tsintsifas, A. (2005). Automated Assessment and Experiences of Teaching Programming. Journal on Educational Resources in Computing (JERIC), vol. 5, pp. 5.
4. Douce, C., Livingstone, D., Orwell, J. (2005). Automatic Test-based Assessment of Programming: A Review. Journal on Educational Resources in Computing (JERIC), vol. 5, pp. 4.
5. Ihantola, P., Ahoniemi, T.,Karavirta, V., Seppälä, O. (2010). Review of Recent Systems for Automatic Asessment of Programming Assignments. Proceedings of the 10th Koli Calling International Conference on Computing Education Research, pp. 86-93.
6. Romli, R., Sulaiman, S., Zamli, K. Z. (2010). Automatic Programming Assessment and Test Data Generation a Review on its Approaches. Information Technology (ITSim), 2010 International Symposium, pp. 1186-1192.
7. Caiza, J. C.; Del Alamo, J. M. Programming Assignments Automatic Grading: Review of Tools and Implementations. Inted 2013 Proceedings, 2013, 5691-5700
8. Rodríguez-del-Pino, J. C., Rubio-Royo, E., Hernández-Figueroa, Z. J. (2012). A Virtual Programming Lab for Moodle with Automatic Assessment and Anti-plagiarism Features.
9. Queirós, R. A. P., Leal, J. P. (2012). PETCHA: A Programming Exercises Teaching Assistant. Proceedings of the 17th ACM Annual Conference on Innovation and Technology in Computer Science Education, pp. 192-197.
10. Spacco, J., Hovemeyer, D., Pugh, W., Emad, F., Hollingsworth, J. K., Padua-Perez, N. (2006). Experiences with Marmoset: Designing and Using an Advanced Submission and Testing System for Programming Courses. ACM SIGCSE Bulletin, vol. 38, pp. 13-17
11. Jelemenská, K. C˘ic˘ák, (2012). Improved Assignments Management in MOODLE Environment. INTED2012 Proceedings, pp. 1809-1817.
12. Jackson, D., & Usher, M. (1997). Grading student programs using ASSYST. Proceedings of the Twenty- Eighth SIGCSE Technical Symposium on Computer Science Education, USA, 335-339.

13. Schorsch, T. (1995). CAP: An automatic self-assessment tool to check Pascal programs for syntax, logic and style errors. Proceedings of the 26th SIGCSE technical symposium on Computer science education, USA, 168-172.
14. Emma Enstro ¨m, Gunnar Kreitz, Fredrik Niemela, Pehr Soderman, and Viggo Kann. 2011. Five Years with Kattis – Using an Automated Assess- ment System in Teaching. In Frontiers in Education Conference (FIE), 2011. Institute of Electrical and Electronics Engineers, Piscataway, NJ, T3J–1.
15. Mike Joy, Nathan Griffiths, and Russell Boyatt. 2005. The BOSS on- line submission and assessment system. ACM Journal on Educational Resources in Computing 5, 3, Article 2 (Sept. 2005), 28 pages. DOI:http://dx.doi.org/10.1145/1163405.1163407
16. Lauri Malmi, Ari Korhonen, and Riku Saikkonen. 2002. Experiences in automatic assessment on mass courses and issues for designing virtual courses. ACM SIGCSE Bulletin 34, 3 (2002), 55–59.
17. Jacques Philippe Sauve ´, Oso ´rio Lopes Abath Neto, and Walfredo Cirne. 2006. EasyAccept: a tool to easily create, run and drive development with automated acceptance tests. In Proceedings of the 2006 international workshop on Automation of software test (AST '06). ACM, New York, NY, USA, 111–117. DOI:http://dx.doi.org/10.1145/1138929.1138951
18. Brad Vander Zanden, David Anderson, Curtis Taylor, Will Davis, and Michael W. Berry. 2012. CodeAssessor: An Interactive, Web-Based Tool for Introductory Programming. The Journal of Computing Sciences in Colleges 28, 2 (2012), 73 – 80.
19. Yuen Tak. Yu, Chung Keung Poon, and Marian Choy. 2006. Experiences with PASS: Developing and Using a Programming Assignment Assess- ment System. In Quality Software, 2006. QSIC 2006. Sixth International Conference on. Institute of Electrical and Electronics Engineers, Piscat- away, NJ, 360–368. DOI:http://dx.doi.org/10.1109/QSIC.2006.28
20. A.T. Chamillard and J.K. Joiner, "Using lab practica to evaluate programming ability". In Proceedings of the 32nd ACM SIGCSE Technical Symposium on Computer Science Education (SIGCSE 2001), ACM Press, pages 159-163, 2001.
21. Andrew Sutton and Bjarne Stroustrup, "Design of Concept Libraries for C++", Texas A&M University Department of Computer Science and Engineering {asutton, bs}@cse.tamu.edu
22. Stepanov, A., McJones, P.: Elements of Programming. Addison Wesley, Boston, Massachusetts (2009)