IJITCE

ISSN 2347- 3657

International Journal of Information Technology & Computer Engineering

Email : ijitce.editor@gmail.com or editor@ijitce.com

International Journal of Information Technology & Computer Engineering

ISSN 2347–3657
Volume 9,Issue 1,Jan 2021

Research Methodology on Code Clone Detection with Refactoring Using Textual and Metrics Analysis in Software

Ms. BANDARU MAKARA JYOTHI

Abstract

For big industrial software systems, the process of snippeting code and making small, non-functional changes is a big issue. Code clones, which are copies of existing code, are the result. Cloned code makes maintenance more of a pain, which is the primary effect. It is common practice to utilize code clone detection to identify instances of reused code in various applications. Because they add complexity to the system, evolution clones are seen as detrimental in software maintenance. The development of clone detection has led to better outcomes while also simplifying the system for easier maintenance.

You can tell if clone detection is primarily concerned with line-by-line detection or tokenization-based detection by looking at the current status of code cloning. This method adds complexity to the system and slows down the process of processing the source code to locate clones. The existing clone identification algorithm is not able to identify clones that are not perfect copies but have functional similarities with other code fragments.

Keywords — Code Clone, Clone detection, refactoring, metrics, textual analysis,

I. INTRODUCTION, OVERVIEW, CONCLUSION OF RESEARCH WORK AND FUTURE ENHANCEMENTS

II. For big industrial software systems, the process of snippeting code and making small, non-functional changes is a big issue. Code clones, which are copies of existing code, are the result. Cloned code makes maintenance more of a pain, which is the primary effect. It is common practice to utilize code clone detection to identify instances of reused code in various applications. Because they add complexity to the system, evolution clones are seen as detrimental in software maintenance. The development of clone detection has led to better outcomes while also simplifying the system for easier maintenance.

You can tell if clone detection is primarily concerned with line-by-line detection or tokenization-based detection by looking at the current status of code cloning. This method adds complexity to the system and slows down the process of processing the source code to locate clones. The existing clone identification algorithm is not able to identify clones that are not perfect copies but have functional similarities with other code fragments.

ASSOCIATE PROFESSOR

makarajyothi.mca2014@gmail.com

SRINIVASA INSTITUTE OF MANAGEMENT SCIENCES

ISSN 2347–3657
Volume 9,Issue 1,Jan 2021

III. The suggested study model for clone detection technique demonstrates an easier detection method with efficient outcomes. Combining a textual approach with metric analysis of the provided source code allows this method to find all four kinds of clones in a group of code fragments in Java source code. Clone clusters are created by collecting all the identified clone pairs and saving them in files. Clones that have been found may all be In response to the programmer's request for refactoring.

IV. In order to facilitate clone identification, many semantics have been developed and their values have been used. Combining these measures with textual analysis makes clone detection more easier and yields more reliable results.
The Precision and Recall numbers are used to evaluate the efficiency of the approach. Benchmark tools such as Clone DR and CCFinder, among others, are used to compare the outcomes of the suggested approach. Results from the experiments demonstrate that compared to the prior methods, these ones provide greater Precision and Recall values.

CONCLUSIONS

An integral part of every software development life cycle is software maintenance. Making life easier for those working in software maintenance is one way to boost morale in the sector. Customers believe software is more adaptable than other products, thus they anticipate more maintenance needs (i.e., it's only writing few instructions). So, adjustments may be made easily whenever needed. The literature on software engineering, however, claims this to be untrue.
Large software systems are more vulnerable to software cloning. Simply copying and pasting is the most common cause of cloning. Since creating code from scratch takes more time, almost every developer does this activity in an effort to reduce development time. Cloning is a solution that developers may have to consider when they are short on time. These clones are unintentionally created by certain maintenance engineers. While cloning may seem to be a quick and easy way out of a developer's jam, it often results in recorded actions that have a detrimental impact on software quality. It raises the overall system code and the amount of lines of code that need maintenance.Finding and removing clones from software systems is a hotspot for current research in the field of clone detection. Code cloning was described in many ways in the literature that was reviewed for this study. There has been a lot of discussion and comparison of current methodologies and tools. There has to be a solution to this issue and it is critical to find all the clones in the code. You can fix the code copying issues using the current refactoring approaches. The suggested technique makes good use of refactoring. In this work, a light-weight method has been proposed to identify functional clones. This method uses the computation of several metrics in combination with simple textual analysis technique. The usage of metrics with existing exponential rate of comparison overhead of the other methods is reduced to minimum number of comparisons. This is possible by early analysis of potential clones and applying comparisons only on code fragments that are identified as clones in this analysis. Since the string matching/textual comparison is performed over the short listed candidates, a higher amount of recall could be obtained.

The Proposed work is divided into two stages. The first one is selection of potential clones and the second one is comparison of potential clones. The proposed technique detects exact clones on the basis of metric match and then by text match. Potential clones are compared line-by-line to determine whether two potential clones really are clones of each other. The experiments proved that this method can do better than existing systems in finding and classifying the clones in JAVA. The Precision and Recall values that are obtained describe the efficiency of the work proposed. It has been proved that Precision 98% and Recall 96% is achievable in code cloning. In addition it also identifies the functional clones.

Future Enhancements

Though the proposed technique is working

efficiently for Programming languages like JAVA, it can be extended to find clones in multiple languages. When it comes to identify only type I, type II and type III clones this method can identify clones in almost all object oriented programming languages. Research work can be extended not only to find the clones but also to remove the actual clones. Though refactoring process has been used, it can be fully automated and implemented so that no human intervention is required.

The proposed method is experimented on medium sized software applications only. These applications are of 10 to 15 KLOC only. Experiments on large scale systems can be conducted to observe efficiency of the method. The parameters for the efficiency are taken only in the form of precision and recall values. It also can be extended to scalability, portability and robustness etc.

REFERENCES

IEEE Standard 1219, 1998, Relating to Software Maintenance.
[2] As a result of ISO/IEC.Software Engineering—Software Maintenance/ISO/IEC 14764, 1999.
Three, lean. Hello, Arthur. Keeping Up with Software Maintenance in an Evolving Landscape. Press, 1988.
Tip S.W.L. and Tip T. "A software maintenance survey" by Lam was published in the proceedings of the conference. paper presented at the 1994 December Asia-Pacific Software Engineering Conference, pages 70–79. Five, S. C. Chidamber and R. A metric suite for object-oriented design was published in June 1994 by Kemerer in the IEEE Transactions on Software Engineering, volume 25, issue 5, pages 476–493.
[6] "Software configuration management: A clear case for IBM Rational" (IBM Redbooks, 2004), by Jennie Brown and Matti Teinonen
"Clone detection and refactoring" by Robert Tairas was published in 2006 in the Proceedings of the OOPSLA '06 Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications (pp. 780–781). The conference was held in New York, USA.
Code clone detection systems compared and evaluated by Chanchal K. Roy, James R. Cordya, and Rainer Koschkeb [8] Methods and Resources: A Qualitative Perspective", Journal of Scientific Computing, Vol. May 2009, volume 74, issue 7, pages 470–495.
In the 1998 International Conference on Software Maintenance, "Clone Detection Using Abstract Syntax Trees," Ira D. Baxter, Andrew Yahin, Leonardo Moura, Marcelo Sant Anna, and Lorraine Bier presented their work on page 368.
[10] "Analyzing Web Service Similarity Using Contextual Clones" (ACM Journal, 2011), Douglas Martin and James R. Cordy.
Referenced in [11] M. Fowler's 1999 book "Refactoring: improving the design of existing code" published by Addison Wesley.
[12] "Refactoring Workbook" by William C. Wake, published by Pearson Education Inc. in 2004.
In their article "On the Use of Clone Detection for Identifying Crosscutting Concern Code" published in the IEEE Transactions On Software Engineering, Magiel Bruntink, Remco van Engelen, Arie van Deursen, and Tom Tourwe provide more information. page 804–818 of volume 31, issue 10, October year 2005
"The enhanced suffix array and its applications to genome analysis" (In Proc.) by Abuelhoda, Kurtz, and Ohlebusch [14] describes this tool. Volume 2452, pages 449–463, Berlin, 2002, Workshop on Algorithms in Bioinformatics
The paper "Detecting Higher-level Similarity Patterns in Programs" was presented at the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering in Lisbon in 2015. The authors are Hamid Abdul Basit and Stan Jarzabek.
September 2005
“Context- based detection of clone-related bugs” was published in 2007 in New York, USA, by Lingxiao Jiang, Zhendong Su, and Edwin Chiu. It was part of the Proceedings of the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering.
[17] "A Survey on Software Clone Detection Research" (Chanchal Kumar Roy and James

R Cordy, Computer and Information Science, Vol. No. 541, 115 pages, 2007 "Identifying Redundancy in Source Code Using Fingerprints" by J. Howard Johnson was published in the Proceedings of the 1993 Conference of the Centre for Advanced Studies (CASCON'93). On pages 171–183, in October 1993, in Toronto, Canada.
[19] "CP-Miner: Finding Copy-Paste and Related Bugs in Large-Scale Software Code" in IEEE Transactions on Software Engineering, Vol., written by Zhenmin Li, Shan Lu, Suvda Myagmar, and Yuanyuan Zhou. Number 32, Issue 3, March 2006, pages 176–192.
[20] Simon Thompson and Christopher Brown. The article "Clone Detection and Elimination for Haskell" was published in the ACM journal in 2010. "Investigating the maintenance implications of the replication of code" was published in the Proceedings of the 13th International Conference on Software Maintenance (ICSM'97) in Bari, Italy in September 1997 by Elizabeth Burd and Malcolm Munro. Matthias Rieger [22]. A doctoral thesis titled "Effective Clone Detection without Language Barriers" was completed in June 2005 at the University of Bern in Switzerland.
"Aspect Mining using Clone Class Metrics" by Magiel Bruntink was published in 2004 in the Proceedings of the 1st Workshop on Aspect Reverse Engineering. The article "Function Clone Detection in Web Applications: A Semi automated Approach" was published in the Journal of Web Engineering and was written by Fabio Calefato, Filippo Lanubile, and Teresa Mallardo. number one, pages 003-021, 2004.
[25] "The Software Similarity Problem in Mal-ware Analysis" by Andrew Walenstein and Arun Lakhotia was published in July 2006 in the Proceedings of the Dagstuhl Seminar 06301: Duplication, Redundancy, and Similarity in Software. The paper is on page 10.
For example, in November 2001, in Florence, Italy, at the 17th IEEE International Conference on Software Maintenance (ICSM'01), Giuliano Antoniol, Gerardo Casazza, Massimiliano Di Penta, and Ettore Merlo presented a paper titled "Modeling Clones Evolution through Time Series" (pp. 273–280).
27. W-K. B. Li, Chen, and R. "Code Compaction of Matching Single-Entry Multiple-Exit Regions" (Gupta's work), in STAC's tenth annual conference proceedings

Conference (SAS'03), June 2003, San Diego, California, USA, pp. 401-417. The paper "An Evaluation of Clone Detection Techniques for Identifying Crosscutting Concerns" was presented at the 20th IEEE International Conference on Software Maintenance in 2004 in Washington DC, USA. The authors of the paper are Magiel Bruntink, Arie van Deursen, Tom Tourwe, and Remco van Engele.
[29] "Using Clone Detection to Manage a Product Line" (pp. 1-3, 1998), proceedings of the International Conference on Clone detection using abstract syntax trees, authored by Ira D. Baxter and Dale Churchett.
[30] "MeCC: Memory Comparison-based Clone Detector" presented by Heejung Kimy, Yungbum Jungy, Sunghun Kimx, and Kwangkeun Yi at the 33rd International Conference on Software Engineering held in Waikiki, Honolulu, Hawaii from May 21st to the 28th, 2011.
Citation: "Clone detection in automotive model-based development" by Florian Deissenboeck, Benjamin Hummel, Elmar Jurgens, Bernhard Schatz, Stefan Wagner, Jean-François Girard, and Stefan Teucher, published in Proceedings of the 30th international conference on Software engineering (pp. 613-622), New York, [31]. 2007 in New York City, USA "Visualization of clone detection results" by Robert Tairas, Jeff Gray, and Ira Baxter was published in the 2006 OOPSLA Workshop on Eclipse Technology Exchange Proceedings, edited by ACM. New York, USA, 2006, pp. 50–54.
An article titled "Towards Flexible Code Clone Detection, Management, and Refactoring in IDE" was presented at the Fifth International Workshop on Software

Clones by Minhaz F. Zibran and Chanchal K. Roy.
May 23, 2011, Waikiki, Hawaii, USA in reference [34] M. A. Lau, D., L. Bergman, and Kim, K. "An Ethnographic Study of Copy and Paste Programming Practices in OOPL" by Notkin, accepted for publication in the International Symposium on Empirical Software Engineering proceedings. held in August 2004 (ISESE '04), pages 83–92 [/35] M. S. Ducasse, G. Rieger, and B. In the proceedings of the European Conference on Object-Oriented Programming (ECOOP '99), June 1999, Golomingi presents "Tool Support for Refactoring Duplicated OO Code" (pp. 177-178).
[36] "Clone Detection Using Abstract Syntax Suffix Trees," In Proc., Raimar Falke, Pierre Frenzel, and Rainer Koschke, 2007. published in October 2006 by the 13th Working Conference on Reverse Engineering in Benevento, Italy, pages 253–262.
[37] "Research On the effectiveness of clone detection by string matching," Journal of Software Maintenance and Evolution: Research and Practice, Vol., Stephane Ducasse, Oscar Nierstrasz, and Matthias Rieger, in press. 18 (January 2006), pages 37–58.
in "Managing Duplicated Code with Linked Editing" (Michael Toomim, Andrew Begel and Susan L. Graham, 2004), pages 173–180, published in September 2004 in Rome, Italy, in the Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'04).
"KClone: A Proposed Approach to Fast Precise Code Clone Detection," in Proceedings of the Third International Workshop on Detection of Software Clones (IWSC 2009), pp. 12-16, 2009, by YueJia, David Binkley, Mark Harman, Jens Krinke, and Makoto Matsushita ("39").
[40] R.D. With contributions from R. Brooks, P. Y. Govindaraju, M. Pirretti, N. Vijaykrishnan, and M. "Clone Detection in Sensor Networks with Ad Hoc and Grid Topologies," International Journal of Distributed Sensor Networks, Vol., published by Kandemir. vol. 5, pages 209–223, 2009.
"SHINOBI: A Tool for Automatic Code Clone Detection in the IDE," written by Shinji Kawaguchi, Takanobu Yamashinay, Hidetake Uwanoz, Kyhohei Fushida, Yasutaka Kamei, Masataka Nagura, and Hajimu Iida and published in the proceedings of the 16th Working Conference on Reverse Engineering in October 2009, pages 313–314.
"Complete and Accurate Clone Detection in Graph-based Models," in 2009's Proceedings of the 31st International Conference on Software Engineering, Washington, DC, by Nam H. Pham, HoanAnh Nguyen, Tung Thanh Nguyen, Jafar M. Al-Kofahi, and Tien N. Nguyen.
This is Kodhai. Okay, Kanmani. Hello, Kamatchi. A., VidyaSaranya; Radhika.R. "Detection of Type-1 and Type-2 Code Clones Using Textual Analysis and Metrics," in Proc. of the 2010 International Conference on Recent Trends in

Washington, DC: Information, Telecommunication, and Computing, 2010, pp. 241-243.
[44] "Finding Software License Violations Through Binary Code Clone Detection," edited by Armijn Hemel, Karl TrygveKalleberg, Rob Vermaas, and EelcoDolstrac, was presented at the 8th working conference on Mining software repositories in May 2011 in New York, NY.
[45] "Clone Detection using Textual and Metric Analysis to figure out all Types of Clones," in Proceedings of the International Joint Journal Conference on Engineering and Technology (IJJCET 2010), submitted by Kodhai, Perumal, and Kanmani. 2010 (pp. 99–103).
[46] "Function Clone Detection in WebApplications: A Semiautomated Approach" by F. Calefato, F. Lanubile, and T. Mallardo was published in the Journal of Web Engineering in 2004 and can be found in Volume 3, Issue 1, pages 3-21.
[47] Kamiya, Kusumoto, and Inoue's paper titled "CCFinder: A MultiLinguistic Token-Based Code Clone Detection System for Large Scale Source Code" was published in the IEEE Transactions on Information Theory. Software Eng., Vol. Volume 28,

Issue 7, pages 654-670, July 2002.
In the Proceedings of the Second Working Conference on Reverse Engineering (WCRE'95), B. Baker wrote an article titled "On Finding Duplication and Near-Duplication in Large SoftwareSystems." published in July 1995 in Toronto, Ontario, Canada, pages 86–95.
"Detection of Plagiarism in University Projects Using Metrics based Spectral Similarity," presented at the 2007 Dagstuhl Seminar on Software Redundancy, Duplication, and Similarity, is cited as [49].
Presented at the 15th International Conference on Software Maintenance (ICSM'99), pages 109–118, in September 1999, "A Language Independent Approach for Detecting Duplicated Code" was written by S. Ducasse, M. Rieger, and S. Demeyer.
The paper "Archeology of Code Duplication: Recovering Duplication Chains From Small Duplication Fragments" was presented at the 7th Inter-national Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC'05) in Timisoara, Romania in September 2005 by Radu Marinescu and Richard Wettel.
The paper "Identification of high-level concept clones in source code" was presented by Andrian Marcus and Jonathan I. Maletic in November 2001 at the 16th IEEE International Conference on Automated Software Engineering (ASE'01). The paper is located on pages 107–114 and was held in San Diego, California, USA.
"Finding Clones with Dup: Analysis of an Experiment," published in the IEEE Transactions on Software Engineering, volume 53, was written by B. Baker. Volume 33, Issue 9, pages 608-621, 2007.
55. J.R. K. Cordy and C.E. June 2011, Kingston, Canada: Roy, "The NiCad Clone Detector," 19th International Conference on Program Comprehension.
[55] "Bauhaus: A Tool Suite for Program Analysis and Reverse Engineering" by Aoun Raza, Gunther Vogel, and Erhard Plaodereder was published in the proceedings of the 11th Ada-Europe International Conference on Reliable Software Technologies (LNCS 4006). June 2006, Porto, Portugal, pages 71–82.
[56] Z. Yang, "Determining Where Two Programs Differ Syntactically," in Software Practice and Experience, Vol. 20, 21, 7, Page numbers 739–755, July 1991.
[57] Wahler, Gustav, Gudenberg, and Seipel, V. D. In the Proceedings of the 4th IEEE International Workshop Source Code Analysis and Manipulation (SCAM), "Clone Detection in Source Code by Frequent Itemset Techniques" (pp.128–135), held in September 2004 in Chicago, IL, USA, Fischer presents his work.
The paper "Clone Detection via Structural Abstraction" was presented at the 14th Conference on Reverse Engineering (WCRE'07) in October 2007 in Vancouver, BC, Canada, and was co-authored by Williams Evans and Christopher Fraser.
59. "Phoenix-Based Clone Detection Using Suffix Trees" by Robert Tairas and Jeff Gray was published in March 2006 in Melbourne, Florida, USA, in the Proceedings of the 44th annual Southeast regional conference (ACM-SE'06), on pages 679-684.
In the Proceedings of the 12th International Conference on Software Maintenance, J. Mayrand, C. Leblanc, and E. Merlo conducted an experiment titled "Experiment on the Automatic Detection of Function Clones in a Software System Using Metrics."

(ICSM'96), at Monterey, California, USA, November 1996, pages 244–253.
In the Proceedings of the Winter 1994 Usenix Technical Conference, Udi Manber discusses "Finding similar files in a large file system."
page 110, January 1994, San Francisco, USA.
In the Proceedings of the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications (OOP- SLA Companion'05), pp. 140-141, San Diego, CA, USA, October 2005, Seunghak Lee and Iryoung Jeong presented SDD: High Performance Code CloneDetection Systemfor Large Scale Source Code.
63 "The Development of a Software Clone Detector" by Neil Davey, Paul Barson, Simon Field, and Ray J. Frank was published in the International Journal of Applied

Software Technology, Volume. 1, pages 219–236 (1996)
(64th) B. "Random Graphs" by BoUobas, published by Cambridge University Press in 2001.
[65] "Code Cloning: The Analysis, Detection and Removal" (0975 - 8887) in the International Journal of Computer Applications by Mohammed Abdul Bari and Dr. Shahanawaj Ahamad April 2011, Volume 20, Issue 7.
66. "Efficient Source Code Plagiarism Identification Based on Greedy String Tilling" (Khurram Zeeshan Haider, Tabassam Nawaz, Sami ud Din, and Ali Javed, 2010). International Journal of Computer Science and Network Security, Volume 10, Issue 12, December 2010.
[67] "Do Code Clones Matter?" in Proceedings of the 31st international conference on Software engineering, by Florian Deissenboeck, Benjamin Hummel, Elmar Jurgens, and Stefan Wagner New York, NY, USA, 2009, pp. 485–495.
[68] "An Empirical Study on Inconsistent Changes to Code Clones at Release Level," Conference on Reverse Engineering, 2009, Bettenburg, Nicola, Weyi Shang, Ibrahim, and W. Adams.
69 "An empirical study on the maintenance of source code clones" (Suresh Thummalapenta, Luigi Cerulo, Lerina Aversano, and Massimiliano Di Penta, 2010), published in the journal Empirical Software Engineering, February, Volume 15, Issue 1, pages 1–34.
[70] "Clones: what is that smell?" by Foyzur Rahman, Christian Bird, and Premkumar Devanbu, published in August by Empirical Software Engineering, Vol. 2012, volume 17, issue 4, pages 503-530.
(71), "Clone evolution: a systematic review" (Journal of Software: Evolution and Process, Volume 25, Issue 3, pp. 261-283), Jeremy R. Patel, Robert Tairas, and Nicholas A. Kraft. March of that year, 2013.
In the Proceedings of the 4th International Workshop on Software Clones, "Model clone detection in practice" (pp. 57-64, 2010), Florian Deissenboeck, Benjamin Hummel, Elmar Juergens, Michael Pfaehler, and Bernhard Schaetz were the authors.
The authors of the 2007 paper "Spatial smoothing and hot spot detection for CGH data using the fused lasso" (Robert Tibshirani and Pei Wang) are cited as [73].
The authors of the 2010 article "Real-time PCR for detection of the O25b- ST131 clone of Escherichia coli and its CTX-M-15-like extended-spectrum lactamases" (HiranDhanjia, Michel Doumitha, Olivier Clermont, Erick Denamur, Russell Hope, David M. Livermore, and Neil Woodford) published in the International Journal of Antimicrobial Agents.