Source Code Plagiarism Detection 'SCPDet': A Review

Tapan P. Gondaliya
Research Scholar
School of Computer Science
RK University
Rajkot, Gujarat, India

Hiren D. Joshi (PhD)
Associate Professor
School of Computer Science
Dr. Babasaheb Ambedkar Open University
Ahmadabad, Gujarat, India

Hardik Joshi
Assistant Professor
Department of Computer Science
Gujarat University
Ahmadabad, Gujarat, India

ABSTRACT
Internet has stored large amount of data, information [30] or source code. In this large amount of data or source code it is very difficult and time consuming task to find out the similarity or plagiarism in the source code, research publications in academic.[1] In this paper here we describe the some of the techniques and algorithms for how to find out the plagiarisms in source code. So in large organization or academic institute can easily find out the plagiarism in source code and research publications using this technique. We also differentiate all the techniques of plagiarism for find out how can one technique is differing then the other as well.

General Terms
Source Code Plagiarism Detection in Student Assignments

Keywords
Plagiarism, Source code, Source code reuse, Plagiarism Detection System

1. INTRODUCTION
Digital documents can easily copy from one place to another. [2] Plagiarism is work of others is reproduced without acknowledging the source, this is known as plagiarism. [7] Work that can be plagiarized in many formats includes words, computer program, computer software, graphics or drawing, electronic material and more.[7] Plagiarism is one of the growing global problems experienced by the publishers, researches and educational institutions which are generally defined to be the literary theft. Probably in the most frequent cases appear in academic institutions where students copy material from books, journals, on the Internet, their peers without citing references.

In Computer Science field there are most probably this kind of the case occurred. There are lots of students submit their project work or the programming assignment that is copied form some another students or they change only the name of the variables or the change the some of the methods name and values so it's very difficult and time consuming to find out the manually plagiarism in source code. In next phase we define the different tools and its description.

2. Plagiarism Detection Tools
There are lots of plagiarism detection tools available for the finding the plagiarism in source code. [1] But here we mentioned or described some of the important tools is as under

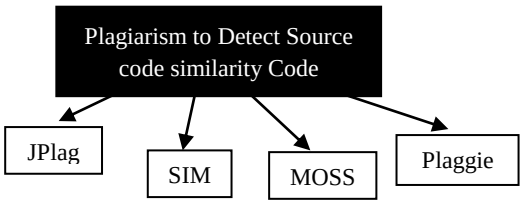

Fig 1: Different plagiarism detection tools

2.1 JPlag
JPlag is a kind of plagiarism technique that finds the similarities between multiple sets of source code in the program. [5] JPlag does not aware with the byte or text but it's aware with the program syntax, program structure. And JPlag support natural language text and different kind of programming languages like a java, c, c++, c# and as well. [5]

JPlag is generally work in the two phase in this two phase in first phase programs to be compared are parsed and converted into token strings. [6] In second phase token strings are compared in two different pairs for formative the similarity of each pair. In this comparison, JPlag attempts to cover one token stream with substrings ("tiles") taken from the other as well as possible. [6] Here we give the demonstration image of JPlag simulator for we know how can work this technique.

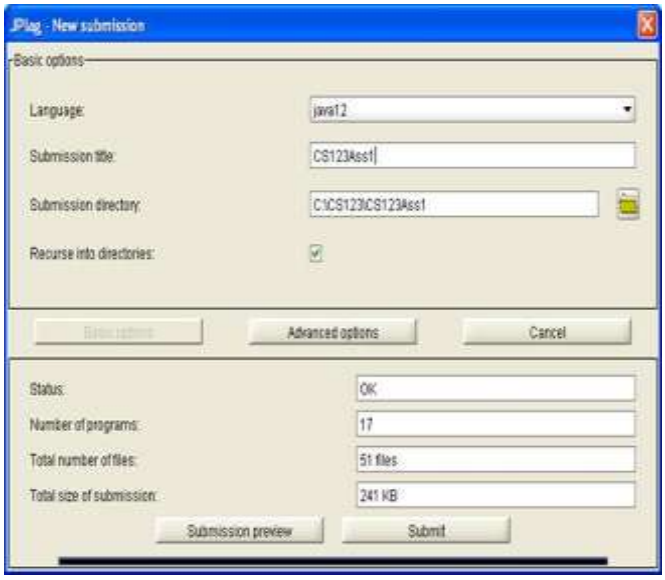

Fig 2: New Assignment in JPlag Tools [7]

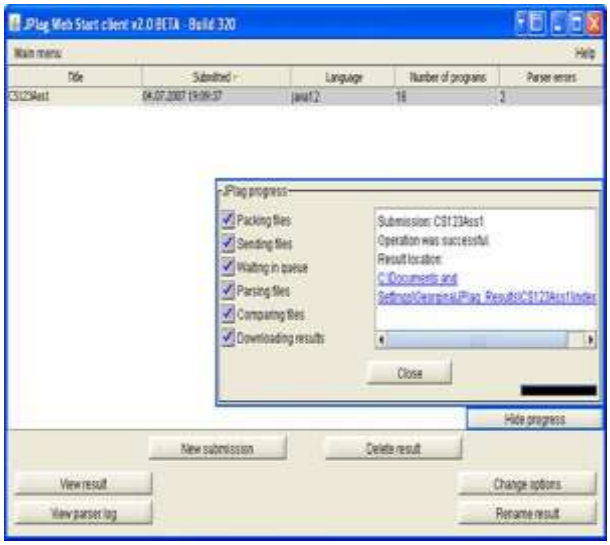

Fig 3: Comparison between the files in JPlag [7]

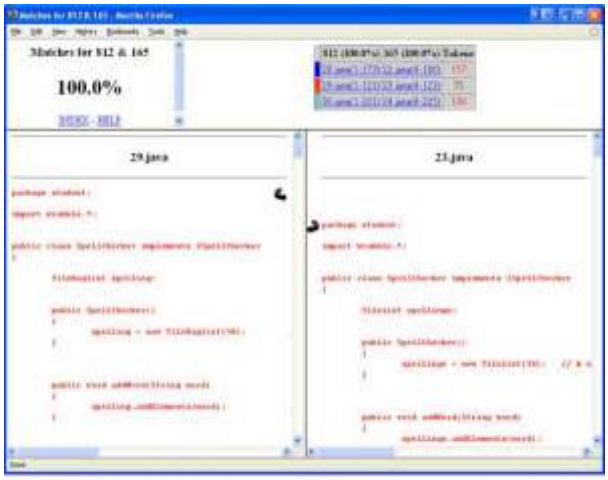

Fig 4: Results of compared program in JPlag Tools [7]

2.2 SIM

SIM is a kind of the Software tool that was developed in 1989 and created by the Dick Grune. [10] SIM is a very important tool for finds the similarity in different language source code like a Pascal, c, java and more. [8] SIM is basically work on the tokenization. Process of similarity found in SIM is first they tokenize the source code and then SIM create a forward reference table and that can be used for detect the best matches between new files, and the text and after they compared both of the things [8]. In this technique programs are first parsed via the flex lexical analyzer for produce a sequence of integers or we can say tokens. The tokens for symbols, keywords, and comments, at the same time the tokens for identifiers are assigned dynamically and saved in a shared symbol table and whitespace is deleted from them.[8] SIM detects similarities between two different source code by using their correctness, fashion, and uniqueness as well. SIM is also used for DNA string matching. [9]

Here we give the demonstration image of SIM Tools for we know more about the SIM tools and its process.

Fig 5: Assignments Submission in SIM Tools [11]

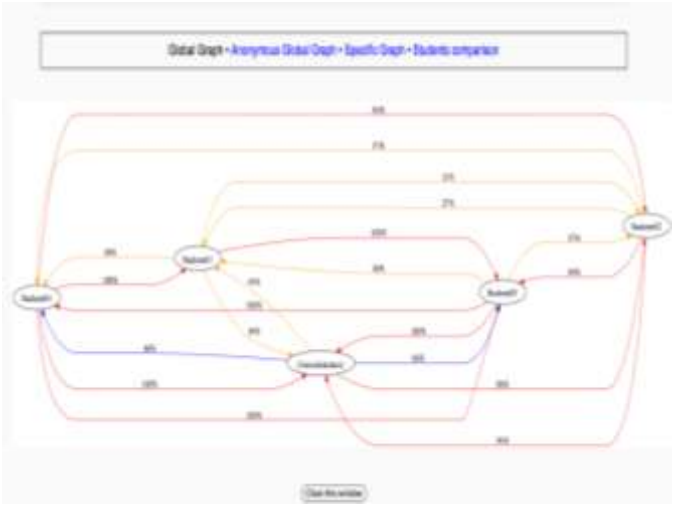

Fig 6: Comparison of different assignments of students in SIM Tools [12]

Fig 7: Comparison between two source codes in SIM Tools [13]

2.3 MOSS

MOSS stands for Measure of software similarity. [8] It is a one kind of the plagiarism detection system created by Alex Aiken and UC Berkeley. [3] Moss tool was basically founded in 1994 and It is the system that finds out the source code

similarity at programming assignment in different programming language like a C, C++, JAVA, and Pascal. [3]

MOSS is a more sophisticated tool then the others. Winnowing technique is used for locate matching sequences between two files in MOSS. In this technique file is basically divided into k-grams which are contiguous substrings of length k. [4] MOSS is available free to use in academics and it is accessible as an online service and also support the UNIX operating system as well as the windows too.[14] Moss is an automatic system for determining the similarity of source code. Moss support many programming languages it may around the 23. [8]

2.4 Plaggie

Plaggie is a kind of engine that find out the source code similarity from programming languages. [8] Plaggie is generally a stand-alone source code plagiarism detection engine. [15] plaggie functionality and graphical user interface are very similar to the JPlag but plaggie is a open source and installation of that engine is in local system as well.[15] plaggie was developed by Ahtiainen et al in 2002. Plaggie only check programs that are written in Java that means support only one language. The basic algorithm used for comparing source code in plaggie is tokenization and after that Greedy String Tiling and do not used the optimization. [8] Above we discussed the four important source code plagiarism detection tools its functionality and show that graphical user interface of that tools and now here we compare that four tools with its different characteristics.

Table 1. Compression of four source code detection tools with its characteristics, function and technique

Tools	JPlag	SIM	MOSS	Plaggie
Open Source Tools/Paid	NO	YES	NO	YES
Local/online tool	Web	Local	Web	Local
Code Submit/File	Submit Code	Submit File	Submit Code	Submit Code
Lang. Support	6	5	23	1
Expandability	No	Yes	No	No
Founded in Year	1996	1989	1994	2002
Founded By	Guido Malpohl	Dick Grune	Aiken et al	Ahtiainen et al
Technique	Greedy String Tiling & Optimization & Tokenization	Flax lexical analyzer	Winnowing technique	Greedy String Tiling & Tokenization

3. LITERATURE REVIEW

In this phase we will describes the some of the research paper that is regarding to the source code plagiarism detection as well as after that description we will also compare that all the research papers or the algorithms in the tabular formats as well.

Towards the Detection of Cross-Language Source Code Reuse, in this article authors Enrique Flores, Alberto Barron-Cede˜no, Paolo Rosso, Lidia Moreno describes how to detect Cross-Language Reuse between Source Codes and main aim behind this research paper of authors is to detect the reuse in source code. In This Research authors mainly used the Character N Gram Comparison algorithm and this algorithm is applied in Code, Comments, Reserved word for comparison point of view. And Language they used is C++, JAVA, and Python. And in the end they found that impact of comments, variable names, and reserved words of the deferent programming languages has been investigated. The best results are obtained when comments are ignored. This suggests that the comments can be safely discarded when aiming to determine the cross-language similarity between two programs. [14]

DeSoCoRe: is acronym for Detecting Source Code Re-Use. It detects source code reuse in many programming languages. Behind this research Paper researcher main goal is to provide a new technology that Detect Source Code Re-Use. Means using this tools authors provide functionality for source code reviewers in order to help them to decide whether the source code has been reused or not. DeSoCoRe in this tool compare two source codes at the level of functions and method even when written in different programming. Method used behind this research is Natural Language Processing (NLP). Software Plagiarism Detection A Graph-based Approach in this article authors Dong-Kyu Chae, Jiwoon Ha, Sang-Wook Kim, BooJoong Kang, Elu Gyu Lm main goal was to creates a software plagiarism detection system that compare between A-CFGs(Control Flow Graphs) by representing each A-CFG as a single score vector through RWR. Results show that there proposed system outperforms existing methods in terms of both accuracy and credibility in a reasonable computation time. [15]

Code Clone Detection Experience at Microsoft in this article the researchers of Microsoft Yingnong Dang, Song Ge, Ray Huang and Dongmei Zhang was well explained or share his experienced and they also Describes that they used three typical usage scenarios of clone detection in the software development process that we have collected at Microsoft. Like a Fix Bug Once, Foot Prints Reduction, Clone Quantity Monitoring. Main objective of the researcher behind this research was to Building a XIAO, a code clone detection tool and its usage at Microsoft. XIAO is a One Kind of the tool which is basically used for detect the source code that was a copied from other by the user. Authors also said that XAIO following keys requirements Near-miss Code Clone Detection, Scalability, Usability, Easy Deployment Based on these requirements, authors developed XIAO, a code clone detection tool. [17]

PDE4Java: Plagiarism Detection Engine For Java Source Code: A Clustering Approach in this research artical authors Ameera Jadalla & Ashraf Elnagar called PDE4Java Was Created a model that is basically used for plagiarism detection and that model namely main purpose of that model is to detect the source code plagiarism for Java. Method used behind this research that is Data Mining, Clustering, N-Gram, Tokenization. At the end authors found that results of this research is Performance of the system pair wise similarity measurement shown promising results compared to JPlag tools finding. To further evaluate the finding of the system, it was compared with the reports of domain experts TA/grader. In some tests, the system reported more clusters than the

domain expert. Manual verification of the extra clusters confirmed the system output to be true positives. [18]

Plagiarism in Programming Assignments in this research paper the authors Mike Joy and Michael Luck was well explained to how the detect the plagiarism is programming language assignment and that was very useful things for the staff as well as the management point of view as well. Method used behind that research is THE WARWICK APPROACH: SHERLOCK and Different technique used for detecting the plagiarism is Calculate and compare attribute counts & Compare programs according to their structure [19] Plagiarism detection using software tools: a study in a Computer Science Degree in this article authors A. Bugarín, M. Carreira, M. Lama, X.M. Pardo well described that what are the software used for plagiarism and how they detect the copy in source code programming. In This Study author used mainly two software tools that is used for detecting source code plagiarism and that tool are JPlag and Turnitin. [20]

Shared Information and Program Plagiarism Detection, in this article author said that main thing is information theory and in computer science is how to measure similarity or the amount of shared information between two sequences. Authors also given the answer of that question and prove that in universal to creating a proposed a metric, based on Kolmogorov complexity. Researchers implement this metric in measuring the amount of shared information between two computer programs, to enable plagiarism detection. Authors also designed and applied in a practical system SID (Software Integrity Diagnosis system) that approximates this metric by a heuristic compression algorithm. In result of SID have clear advantages over other plagiarism detection systems. [21]

A Source Code Similarity System for Plagiarism Detection in this article researchers Zoran Djuric, Dragan Gasevic in starting said that Source code plagiarism is an easy to do task, but very difficult to detect without proper tool Support and also add that Various source code similarity detection systems have been developed to help detect source code plagiarism. Researcher of this paper also designed and developed the one kind of source code similarity system for plagiarism detection. To demonstrate that the proposed system has the desired effectiveness, researcher performed a well known conformism test. Authors said that the system showed good results in detecting source code similarity when various lexical or structural modifications are applied to plagiarized source code. The results were tested on a representative test set of Java source code files. The performance of the SCSDS similarity measurement has shown promising results as compared to the JPlag performance. [22]

A Study of the Uniqueness of Source Code, In This Research paper authors Mark Gabel and Zhendong Su Is defined a one kind of the uniqueness model that main aim is to find out the uniqueness of a unit of source code in the entire body of the software. Author also said that they defined the uniqueness of a unit of source code with respect to the entire body of written software, which they approx with a collection of 420 million lines of source code. Our high-level methodology consists of examining a collection of six thousand software projects and measuring the degree to which each project can be 'assembled' solely from portions of this corpus, thus providing a precise measure of 'uniqueness' that we call syntactic redundancy. [23]

Detecting source code reuse across programming languages, in this research paper auther Enrique Flores, Alberto Barr´on-Ceden˜o, Paolo Rosso and Lidia Moreno build a two main model for the purpose of detecting the Reuse source code in three differnt language C++, Java, Python. Author proposed two models based on character n-grams in order to tackle the problem of cross-programming language reuse of source code at document and fragment levels. In the second model, fragments of source codes are compared with the aim of detecting only those fragments in the source code that resemble more real cases of reuse. [24]

An Approach to Source-Code Plagiarism Detection and Investigation Using Latent Semantic Analysis, in this Research Paper Authors Georgina Cosma, Mike Joy used the two different tools PlaGate, a novel tool that can be integrated with existing plagiarism detection tools to improve plagiarism detection performance. Researchers also add that tool implements in a new approach for investigating the similarity between source-code files with a view to gathering evidence for proving plagiarism. Graphical evidence is presented that allows for the investigation of source code fragments with regards to their contribution toward evidence for proving plagiarism. This thing is done through the Latent Semantic Analysis. Authors also describe that what is the Latent Semantic Analysis technique and how they used for source code detection. Main aims of the plaGate system are → to detect source-code files missed by current plagiarism detection tools. →To provide visualization of the relative similarity between files. →To provide a facility for investigating similar source- code fragments and indicate the ones that could be used as strong evidence for proving plagiarism. [25]

Lexical Analysis for the Measurement of Conceptual Duplicity between C Program, In this article authors Akhil Gupta, Dr. Sukhvir Singh are main objective behind that work is to find out the duplicity in the c programming language source code using lexical analysis technique as well. Authors Point of view the categories of conceptual duplicity are as here changing data types, changing the order of statement changing the order of block of statement, Identifier Changing the operator sequence, changing the operand sequence, redundant statement, Completing Copy, Changing comments, Replacing control structure with equivalent control structure. [26]

A Machine Learning Based Tool for Source Code Plagiarism Detection In this research article Authors Upul Bandara and Gamini Wijayarathna done a great job and build a well improved model for source code plagiarism detection using the machine learning technique. Author also describes that they used meta-learning algorithm in order to improve the accuracy of the source code plagiarism system. At the end author found that researcher are able to achieve 86.64 percent accuracy by using the same dataset used by Lange and Mancoridis according to the research paper published by Lange and Mancoridis their accuracy was 55 percent. Moreover, authors have shown that this method works with adequate accuracy for small training datasets. [27]

C Code Plagiarism detection System In this research article authors N.Haritha, M.Bhavani, K.Thammi Reddy are maid a one system that detect the source code plagiarism detection in c language. The main advantage of this system is it gives the user with two options like checking file or checking folder and gives a pictorial representation of the result. Author also said that this system is mainly designed for c programs. We can use this system to compare a given program with a set of programs. Local database to find out the similarity .It can also be used to check a given a folder of files, to find out the suspicious and non-suspicious programs. Author describe in

this paper that his system is divide in to three main phase in first phase they used the tokenization because the tokenization helps to detect the plagiarism disguises like changing of names of variables and changing loops. In second phase they used the tokens formed are represented by an alpha numeric character and finger prints created using N Gram Technique. And in the third phase similarity is calculated by using JACCARD'S SIMILARITY coefficient. [28]

In next phase of paper we summarized the above literature review in tabular form with comparative studied of its functionality and its characteristics of different algorithms.

Table 2.Summery and comparative study of the different research papers and its characteristics and functionality

Sr.No	Paper Title with Year	Name of Authors	Method & Technique	Objective and Outcomes
1.	Towards the Detection of Cross-Language Source Code Reuse [14] **Year:-** 2011	Enrique Flores, Alberto Barr´on-Cede˜no, Paolo Rosso, Lidia Moreno	**Method Used :-**Character N-Grams Comparison Model **Method Applied in :** Comments, Code And Reserve words **Language Used:-** C++, JAVA & Python	**Objective:-** To Detect Cross-Language Reuse Between Source Codes **Results:-** In Result they found that impact of comments, variable names, and reserved words of the deferent programming languages has been investigated. The best results are obtained when comments are ignored. This suggests that the comments can be safely discarded when aiming to determine the cross-language similarity between two programs.
2.	DeSoCoRe: Detecting Source Code Re-Use across Programming Languages [15] **Year:-** June 2012	Enrique Flores, Alberto Barr´on-Cede˜no, Paolo Rosso, Lidia Moreno	**Method Used:-** Natural Language Processing (NLP) **Tools: -** DeSoCoRe In This Tools compare two source codes at the level of functions and method even when written in different programming languages.	**Objective: -** Provide a helpful tool for source code reviewers in order to help them to decide whether the source code has been re-used or not. **Results: -** In Result Authors Found That they generates a DeSoCoRe tools and is the first online tool which it can detect source code re-use across languages.
3.	Software Plagiarism Detection A Graph-based Approach [16] **Year:-** Nov 2013	Dong-Kyu Chae, Jiwoon Ha, Sang-Wook Kim, BooJoong Kang, Elu Gyu Lm	**Method Used:-** API- labeled control flow graph (A-CFG)	**Objectives:-** Main aims of that research is to creates a software plagiarism detection system that compare between A-CFGs by representing each A-CFG as a single score vector through RWR. Results show that our proposed system outperforms existing methods in terms of both accuracy and credibility in a reasonable computation time.
4.	Code Clone Detection Experience at Microsoft [17] **Year :-** May 2011	Yingnong Dang, Song Ge, Ray Huang and Dongmei Zhang	**Tools** :- XIAO tools made by Microsoft researchers and share its experience	**Objectives: -** Building a XIAO, a code clone detection tool. and its usage at Microsoft
5.	PDE4Java: Plagiarism Detection Engine For Java Source Code: A Clustering Approach [18] **Year:-** 2007	Ameera Jadalla & Ashraf Elnagar	**Techniques Used:-** Data Mining, Clustering, N-Gram, Tokenization	**Objectives: -** To Create a One Engine that Detect the Source Code Plagiarism for Java **Results:-** Performance of the system pair wise similarity measurement shown promising results compared to Jplag tools finding. To further evaluate the findings of the system, it was compared with the reports of domain experts TA/grader. In some tests, the system reported more clusters than the domain expert. Manual verification of the extra clusters confirmed the system output to be true positives.

6.	Plagiarism in Programming Assignments [19] **Year:-** 1999	Mike Joy, Michael Luck	**Methods Used:-** The Warwick Approach :Sherlock **Technique For Detection:-** Calculate and compare attribute counts & Compare programs according to their structure	**Objectives: -** To developed a package which will allow programming assignments to be submitted on-line, and which includes software to assist in detecting possible instances of plagiarism.
7.	Plagiarism detection using software tools: a study in a Computer Science degree [20]	A. Bugarín, M. Carreira, M. Lama, X.M. Pardo	**Tools Used :-** JPlag and Turnitin	**Objective: -** This Paper Show that how software tools detect the plagiarism. There are mainly two tools used one is JPlag and another is Turnitin.
8.	Shared Information and Program Plagiarism Detection [21] **Year:-** July 2004	Xin Chen, Brent Francia, Ming Li, Brian McKinnon, Amit Seker	**Method Used: - Kolmogorov** Complexity.	**Objective:** Researchers implement Kolmogorov Complexity metric in measuring the amount of shared information between two computer programs, to enable plagiarism detection. Authors also designed and applied in a practical system SID
9.	A Source Code Similarity System for Plagiarism Detection [22]	Zoran Djuric, Dragan Gasevic	**Method :-** Tokenization Process **Languages:-** Java Source Code	**Objective:-** Main aim behind this research is to developed & Designed the source code similarity system for plagiarism detection. **Result:-** The Result also shown that the Source Code Similarity System for Plagiarism Detection is perform well as well as effective then the Jplag Source Code Detection
10.	A Study of the Uniqueness of Source Code [23] **Year:-** Nov 2010	Mark Gabel Zhendong Su	**Method Used :-** Tokenization, Tabulation, Sequencing, lexical analysis **Languages :-** C,C++,Java	**Objective:-** Finding the uniqueness of the Source code in C,C++,Java whole Project
11.	Detecting source code reuse across programming languages [24]	Enrique Flores, Alberto Barr´on-Ceden˜o, Paolo Rosso and Lidia Moreno	**Method Used:-** Character N Gram **Languages:-** Applied In C++, Java, Python	**Objective: -** Author proposed two models based on character n-grams in order to tackle the problem of cross-programming language reuse of source code at document and fragment levels. In the second model, fragments of source codes are compared with the aim of detecting only those fragments in the source code that resemble more real cases of reuse.
12.	An Approach to Source-Code Plagiarism Detection and Investigation Using Latent Semantic Analysis [25]	Georgina Cosma, Mike Joy	**Method Used:-** Latent Semantic Analysis **Tools :-** PlaGate, Novel Tools	**Objective: -** Main goal behind this paper is to PlaGate, a novel tool that can be integrated with existing plagiarism detection tools to improve plagiarism detection performance. And investigate the plagiarism detection using Latent Semantic Analysis Information Retrieval Technique

13.	Lexical Analysis For The Measurement of Conceptual Duplicity Between C Program [26] **Year:-** Aug 2013	Akhil Gupta, Dr. Sukhvir Singh	**Methods:-** Lexical Analysis	**Objective :-** Main aim behind this research is to find out the duplicity in the c programs source code using the lexical analysis technique
14.	A Machine Learning Based Tool for Source Code Plagiarism Detection [27] **Year:-** OCT 2011	Upul Bandara, and Gamini Wijayarathna	**Method:-** Machine Learning Technique	**Objective:-** Author is built a new model that basically used the machine learning technique and meta-learning algorithm in order to improve the accuracy of the source code plagiarism system. **Result:-** Result shown that author able to achieve 86.64 percent accuracy by using the same dataset used by Lange and Mancoridis according to the research paper published by Lange and Mancoridis their accuracy was 55 percent. Moreover, authors have shown that this method works with adequate accuracy for small training datasets.
15.	C Code Plagiarism detection System [28]	N.Haritha , M.Bhavani, K.Thammi Reddy	**Tools Used:-** Tokenization, N Gram Technique, JACCARD'S SIMILARITY coefficient	**Objectives:-** Main goal is to build a one system that detects the source code plagiarism detection in c language. Main advantage of the system is it gives the user with two options like checking file or checking folder and gives a pictorial representation of the result **Result :-** According to author the system is working efficiently given a large database and the results are coming clearly and fast. Main thing is detection system is one can easily analyze the results with the visual representation
16.	Efficient Source Code Plagiarism Identification Based on Greedy String Tilling [29]	Khurram Zeeshan Haider, Tabassam Nawaz, Sami ud Din, Ali Javed	**Method Used:-** Greedy String Tilling	**Objective:-** Main aim behind this paper is to find out the efficient result for plagiarism detection in programming source code using Greedy String Tilling algorithm **Result :-** Author found that the proposed technique possesses all features which other tools have and some additional features like showing the false positive clone, inclusion, similarity and coverage. Tokens and their time trend shows it promising fast

4. CONCLUSION

In this paper author first of all describe the real meaning of source code plagiarism after that described the different source code plagiarism detection tools and compared its function, characteristics and technique. In the last phase authors discussed the different research papers and compared in tabular form with its technique, method, characteristics, functionality and its result. More and more contributions work towards achieving superlative efficiency and accuracy in the existing solutions.

5. ACKNOWLEDGMENTS

I would like to thank to Dr. Tushar Deshai, Head of Doctoral Study & Vimal Bhatt Administrator of Doctoral Study at RK University for good technical support for any time. I wish to

express heartiest thanks to my parents and friends Nitin M. Sharma, Kamal Jyoti, Ashish Bhagat, Opinder Kumar, Nidhi for telephonic support regarding to the topic.

6. REFERENCES

[1] Izzat Amsmadi, Ikdam AlHami, Saif Kazakzeh, 2014, "Issue related to detection of source code plagiarism in student assignment", International journal of software engineering and its application, vol-8, no-4

[2] Saul Schleimer, Daniel S. Wilkerson, Alex Aiken, June 2003, "Winnowing: Local Algorithms for Document Fingerprinting", SIGMOD 2003, ACM 1-58113-634-X/03/06

[3] Kevin W. Bowyer, Lawrence O. Hall, "Experience Using "MOSS" to Detect Cheating On Programming Assignments", IEEE Computer Society, pp: 13B3/18-13B3/22vol.3

[4] Christian Collberg, Ginger Myles, Michael Stepp, march 2004,"Cheating Cheating Detectors", Technical Report TR04-05

[5] Jplag tool site URL: http://jplag.ipd.kit.edu

[6] Lutz Prechelt, Guido Malpohl, Michael Phlippsen, March 2000, "JPlag: Finding plagiarisms among a set of programs", Technical Report 2000-1

[7] PLAG tool demonstration image are take from this URL: http://www.ics.heacademy.ac.uk/resources/assessment/plagiarism/demo_jplag.html

[8] Divya Luke, Divya P.S, Sony L Johnson, Sreeprabha S, Elizabeth.B.Varghese, 2014, "Software Plagiarism Detection Techniques: A Comparative Study", International Journal of Computer Science and Information Technologies, Vol. 5 (4), ISSN: 0975-9646

[9] Edward L. Jones, 2001, "Plagiarism Monitoring and Detection-Towards and Open Discussion" ,Department of Computer and Information Sciences Florida, A&M University

[10] DickGrune website regarding to similarity measure URL: http://www.dickgrune.com/Programs/similarity_tester/

[11] Figure 5 is taken from this URL:https://moodle.org/pluginfile.php/50/local_plugins/plugin_screenshots/513/screen1.png

[12] Figure 6 is taken from this URL:https://moodle.org/pluginfile.php/50/local_plugins/plugin_screenshots/513/screen2.png

[13] Figure7 is taken from this URL: :https://moodle.org/pluginfile.php/50/local_plugins/plugin_screenshots/513/screen3.png

[14] Enrique Flores, Alberto Barr´on-Cede˜no, Paolo Rosso, Lidia Moreno, 2011, "Towards the Detection of Cross-Language Source Code Reuse", Springer-Verlag Berlin Heidelberg 2011

[15] Enrique Flores, Alberto Barr´on-Cede˜no, Paolo Rosso, Lidia Moreno , Jun 2012, "DeSoCoRe: Detecting Source Code Re-Use across Programming Languages" , NAACL-HLT 2012

[16] Dong-Kyu Chae, Jiwoon Ha, Sang-Wook Kim, BooJoong Kang, Elu Gyu Lm, Nov 2013, "Software Plagiarism Detection A Graph-based Approach", ACM 978-1-4503-2263-8/13/10

[17] Yingnong Dang, Song Ge, Ray Huang and Dongmei Zhang, May 2011, "Code Clone Detection Experience at Microsoft", ACM 978-1-4503-0588-4/11/05

[18] Ameera Jadalla & Ashraf Elnagar, 2007, "PDE4Java: Plagiarism Detection Engine For Java Source Code: A Clustering Approach" , iiWAS2007

[19] Mike Joy and Michael Luck, May 1999, "Plagiarism in Programming Assignments", IEEE transactions on education, VOL. 42, NO. 2

[20] A. Bugarín, M. Carreira, M. Lama, X.M. Pardo, "Plagiarism detection using software tools: a study in a Computer Science degree**"**

[21] Xin Chen, Brent Francia, Ming Li, Brian McKinnon, and Amit Seker, Jul2004, "*Shared Information and Program Plagiarism Detection*", IEEE transactions on information theory, vol. 50, no. 7

[22] Zoran Djuric, Dragan Gasevic, "A Source Code Similarity System for Plagiarism Detection"

[23] Mark Gabel Zhendong Su, Nov2010, "A Study of the Uniqueness of Source Code", ACM 978-1-60558-791-2/10/11

[24] Enrique Flores, Alberto Barr´on-Cede˜no, Paolo Rosso and Lidia Moreno, 2011, "Detecting source code reuse across programming languages", FloresEtAl_SEPLN11

[25] Georgina Cosma and Mike Joy, 2012, "An Approach to Source-Code Plagiarism Detection and Investigation Using Latent Semantic Analysis", IEEE transactions on computers, vol. 61, no. x

[26] Akhil Gupta, Dr. Sukhvir Singh, Aug 2013,"Lexical Analysis For The Measurement of Conceptual Duplicity Between C Program", Ijraset, Vol. 1 Issue I, ISSN: 2321-9653

[27] Upul Bandara and Gamini Wijayarathna, Oct 2011, "A Machine Learning Based Tool for Source Code Plagiarism Detection", International Journal of Machine Learning and Computing, Vol. 1, No. 4

[28] N.haritha, M.bhavani, K.Thammi Reddy, July 2011, "c code plagiarism detection system" ,International Journal of Science and Advanced Technology, ISSN No:- 2221-8386, Volume 1, No 5

[29] Khurram Zeeshan Haider, Tabassam Nawaz, Sami ud Din, Ali Javed, Dec2010, "Efficient Source Code Plagiarism Identification Based on Greedy String Tilling*"* , IJCSNS International Journal of Computer Science and Network Security, VOL.10, No.12

[30] Tapan P. Gondaliya, Dr. Hiren D. Joshi, June 2014, "Big Data challenges and Hadoop as one of the solution of big data with its Modules", IJSER, ISSN 2229-5518, Volume 5, Issue 6, June-2014