

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/228729388>

Effective and Efficient Plagiarism Detection

Article · January 2003

CITATIONS

37

READS

982

1 author:



Thomas Lancaster
Imperial College London

57 PUBLICATIONS 681 CITATIONS

[SEE PROFILE](#)

Some of the authors of this publication are also working on these related projects:



Contract Cheating Research [View project](#)



South East European Project On Policies For Academic Integrity [View project](#)

Effective and Efficient Plagiarism Detection

A thesis submitted in partial fulfilment of the requirements of South Bank University for the degree of Doctor of Philosophy.

Thomas Lancaster
School of Computing, Information Systems and Mathematics
South Bank University

January 2003

Acknowledgements

With thanks to Fintan Culwin, Xristine Faulkner and Nigel Phillips for their help, support and supervision throughout the preparation of this thesis.

Abstract

Student plagiarism, the process where a student uses the words or ideas of another without acknowledgement and for academic credit, is believed to be increasing. This is of concern since it devalues the awards that academic institutions make and has recently been receiving increased media attention. This thesis presents a process through which similarity within a corpus of documents can be found and verified by a tutor to see if it represents plagiarism. Two key requirements for the process are identified. The first is that it should be effective in that it correctly identifies those documents that are the most similar. The second is that it should be efficient; this means both computationally and in terms of tutor workload.

A number of new ideas are introduced. The literature study reveals that there is no consistency in the terms used to talk about plagiarism and a taxonomy is proposed. It also finds inconsistencies in classifications of detection engines for source code plagiarism. Alternative classifications that do not preclude free text engines are presented. The main shortcoming of existing systems is that although the engines might be effective the systems they support impact too greatly on a tutor's time. Hence they are not efficient.

A four-stage detection process consisting of collection, analysis, verification and investigation is proposed. The greatest need for tool support would be on the labour intensive verification and investigation stages. Here a tutor has to examine two documents that have been flagged during the analysis stage. A visual approach to demonstrate the similarity is recommended. A new graphic known as a similarity visualisation is used that presents pixels whose intensity is generated by the commonality of overlapping word fragments. The visualisations are deployed by an interactive tool named VAST that allows quick verification and investigation of suspect areas of the two submissions.

The similarity visualisation is argued to provide the best representation of similarity between two submissions and an ordering of pairs based on its properties is argued to be effective. Generating visualisations for all possible pairs of a large corpus is considered to not be currently computationally feasible. Instead, this ordering is approximated using less computationally intensive metrics. Using real and synthetic submissions it is argued that the word pairs metric, based upon the proportion of two consecutive words that two submissions have in common, is demonstrated to be the most efficient and effective metric.

Table of Contents

<i>Acknowledgements</i>	<i>i</i>
<i>Abstract</i>	<i>iii</i>
<i>Table of Contents</i>	<i>v</i>
<i>Appendices</i>	<i>ix</i>
<i>Table of Figures</i>	<i>xi</i>
<i>Table of Tables</i>	<i>xiii</i>
<i>Chapter 1:</i>	<i>1</i>
<i>Introduction</i>	<i>1</i>
1.1 - Motivation	<i>1</i>
1.2 - Defining the problem.....	<i>1</i>
1.3 - Aims of the thesis	<i>2</i>
1.4 - Structure of the thesis.....	<i>2</i>
1.5 - Conclusions	<i>4</i>
<i>Chapter 2:</i>	<i>5</i>
<i>A Framework for Plagiarism Detection</i>	<i>5</i>
2.1 - Motivation	<i>5</i>
2.2 - The need for detection.....	<i>7</i>
2.3 - Types of plagiarism	<i>8</i>
2.4 - Methods of plagiarising.....	<i>9</i>
2.5 - The collaboration, collusion and copying continuum.....	<i>10</i>
2.6 - Plagiarism detection policies.....	<i>10</i>
2.7 - Four-stage plagiarism detection process	<i>11</i>
2.8 - The use of metrics in the analysis stage to assess document similarity.....	<i>12</i>
2.9 - Detection engines classified by submission type	<i>13</i>
2.10 - Detection engines classified by features.....	<i>13</i>
2.11 - Classifications of detection engines through number of submissions processed by the metrics used	<i>14</i>
2.12 - Classifications of detection engines through operational complexity of the metrics used	<i>16</i>
2.13 - The detection process using metrics.....	<i>17</i>
2.14 - Developing a research question	<i>19</i>

2.15 - Conclusions	21
Chapter 3:.....	23
Source Code Plagiarism Literature Review.....	23
3.1 - Motivation	23
3.2 - Introduction	24
3.3 - Literature identified	24
3.4 - Defining source code plagiarism	26
3.5 - Literature classifications of detection engines	27
3.6 - Comparison of source code detection engines.....	32
3.7 - Singular metric engines.....	36
3.8 - Singular and paired metric engines	37
3.9 - Paired metric engines	38
3.10 - Paired metric engines with tokenisation.....	38
3.11 - Other detection engines.....	40
3.12 - Other technical detection approaches.....	41
3.13 - Literature comparisons of existing engines.....	42
3.14 - Comparison of Web-based detection services	44
3.15 - Attitudes to source code plagiarism detection.....	45
3.16 - Other source code papers.....	46
3.17 - Conclusions	47
Chapter 4:	49
Free Text Plagiarism Literature Review.....	49
4.1 - Motivation	49
4.2 - Free text plagiarism detection literature	50
4.3 - Comparison of Web-based plagiarism detection services.....	52
4.4 - Linguistical detection methods for plotting similarity	58
4.5 - Other linguistical tools for detection.....	60
4.6 - Copy detection tools.....	61
4.7 - Technical solutions for identifying similarity.....	62
4.8 - Technical solutions for investigating similarity	65
4.9 - Other technical solutions for detecting free-text plagiarism	66
4.10 - Conclusions	69
Chapter 5:	71
Plagiarism Visualisation Literature.....	71

5.1 - Motivation	71
5.2 - Visualisation literature identified.....	72
5.3 - Visualisations for a whole corpus.....	75
5.3 - Visualisations for a pair of submissions	85
5.4 - Visualisations for a single submission.....	89
5.5 - Conclusions	92
<i>Chapter 6:</i>	93
<i>Literature Related to Plagiarism</i>	93
6.1 - Motivation	93
6.2 - The problem of plagiarism.....	94
6.3 - Plagiarism Prevention and Detection.....	97
6.4 - Academic plagiarism and research fraud.....	99
6.5 - Legal Aspects of Plagiarism.....	102
6.6 - Sociological, Cultural and Psychological Aspects of Plagiarism	104
6.7 - Extent of student cheating and plagiarism outside the UK	107
6.8 - Extent of student cheating and plagiarism in the UK	112
6.9 - Authorship Attribution Techniques.....	116
6.10 - Conclusions	122
<i>Chapter 7:</i>	123
<i>Efficient Finding of Similar Pairs During the Detection Stage</i>	123
7.1 - Motivation	123
7.2 - The use of simple metrics.....	123
7.3 - Calculating similarity scores using simple metrics.....	124
7.4 - Comparing the effectiveness of simple metrics on small document collection....	130
7.5 - Introducing plagiarism into a real corpus.....	133
7.6 - Conclusions	136
<i>Chapter 8:</i>	137
<i>Effective Investigation of Similarity</i>	137
8.1 - Motivation	137
8.2 - Identifying areas of similarity within two documents	138
8.3 - The Fragmentary Intercept Matrix	142
8.4 - Similarity Visualisations	143
8.5 - Similarity Intersections	144
8.6 - Visualisations of real student submissions for intra-corporeal plagiarism.....	147

8.7 - Visualisations of real student submissions for extra-corporeal plagiarism	149
8.8 - False hit verification	150
8.9 - Conclusions	153
Chapter 9:.....	155
A Visual Approach to Identify the Most Effective Simple Metric	155
9.1 - Motivation	155
9.2 - Visualisations show whether or not two submissions contain similarity	156
9.3 - Practical problems with this approach.....	157
9.4 - Similarity visualisations can give the best ordering of similarity.....	158
9.5 - Error free metrics of low computational intensity	159
9.6 - Choosing a good visual metric.....	160
9.7 - Correlating the simple metric with the mean of max metric.....	167
9.8 - Noise free synthetic documents	168
9.9 - Noise free borderless synthetic documents.....	174
9.10 - The effects of noise on ordering synthetic documents.....	176
9.11 - A synthetic corpus with randomly generated noise.....	180
9.12 - A synthetic corpus that preserves real structure and noise levels.....	181
9.13 - Real corpora.....	185
9.14 - High similarity corpora.....	186
9.15 - Submitted student corpora	187
9.16 - The words pair metric	193
9.17 - Conclusions	195
Chapter 10:.....	197
A Complete Detection Process.....	197
10.1 - Motivation	197
10.2 - Requirements of a complete detection process	198
10.3 - Student Submission System	199
10.4 - Text Ranker	199
10.5 - Visualisation and Analysis of Similarity Tool	199
10.6 - Tool use in a complete detection process	202
10.7 - Main contributions of thesis	203
10.8 - Further research	204
10.9 - Conclusions	212
References	213

Appendices

<i>Appendix A: Glossary</i> -----	A
<i>Appendix B: A descriptive taxonomy of student plagiarism</i> -----	B
<i>Appendix C: Plagiarism prevention, deterrence and detection</i> -----	C
<i>Appendix D: Plagiarism issues in higher education</i> -----	D
<i>Appendix E: A review of electronic services for plagiarism detection in student submissions</i> -----	E
<i>Appendix F: Source code plagiarism in UK he computing schools, issues, attitudes and tools</i> -----	F
<i>Appendix G: Towards an error-free plagiarism detection process</i> -----	G
<i>Appendix H: Visualising intra-corporeal plagiarism</i> -----	H
<i>Appendix I: Visualisations for high similarity corpus</i> -----	I
<i>Appendix J: Word fragments for simple metrics for three sample documents</i> -----	J

Table of Figures

Figure 2.1 - Collaboration, collusion and copying	10
Figure 2.2 - Four-stage plagiarism detection process	11
Figure 2.3 - Engine using singular metrics	18
Figure 2.4 - Engine using paired metrics	19
Figure 3.2 – Culwin, MacLeod & Lancaster classifications.....	29
Figure 3.4 - Key to Figures 3.1, 3.2 and 3.3	30
Figure 3.5 - Improved classifications of detection engines	31
Figure 5.1 - SHERLOCK list of similar pairs	78
Figure 5.2 - SHERLOCK clustering of similar submissions.....	79
Figure 5.3 - PRAISE used to find clusters of similar submissions	80
Figure 5.4 - DotPlot.....	81
Figure 5.5 - Categorical patterngram	83
Figure 5.6- Composite categorical patterngram	84
Figure 5.7- JPlag used to investigate two student submissions.....	87
Figure 5.8- SHERLOCK comparing two submissions	88
Figure 8.1 - First example of a similarity visualisation	144
Figure 8.2 - A synthetic similarity visualisations with 200 words of similarity	145
Figure 8.3 - More synthetic similarity visualisations	146
Figure 8.4 - Intra-corporeal plagiarism under the word count metric.....	148
Figure 8.5 Intra-corporeal plagiarism under the word sequences metric.....	149
Figure 8.6 - Extra-corporeal plagiarism.....	150
Figure 8.7 - Similarity visualisation for tutor 1.....	151
Figure 8.8 - Similarity visualisation for tutor 2.....	151
Figure 8.9 - Similarity visualisation for tutor 3.....	152
Figure 8.10 - Similarity visualisation for tutor 4.....	152
Figure 9.1 - Two similarity visualisations	156
Figure 9.2 - Simple representative similarity visualisations	157
Figure 9.3 - Calculation of the max metric for two representative similarity visualisations.....	162
Figure 9.4 - Calculation of the mean metric for two representative similarity visualisations.....	162
Figure 9.5 - Calculation of the median metric for two representative similarity visualisations.....	163
Figure 9.6 - Calculation of the mode metric for two representative similarity visualisations.....	163
Figure 9.8 - Calculation of the mean of row max metric for two representative similarity visualisations	164

Figure 9.9 - Calculation of the mean of col max metric for two representative similarity visualisations	165
Figure 9.10 - Calculation of the max of row mean metric for two representative similarity visualisations	165
Figure 9.11 - Calculation of the max of col mean metric for two representative similarity visualisations	166
Figure 9.12 - Calculation of the mean of max metric for two representative similarity visualisations.....	167
Figure 9.13 - A synthetic corpus with known best ordering of pairs	169
Figure 9.15 - Second synthetic corpus with known best ordering of pairs	174
Figure 9.16 - 800 word similarity intersections at different noise levels	179
Figure 9.18: Similarity visualisation for two documents in corpus preserving linguistic properties.....	182
Figure 9.19: Sequenced visualisations for synthetic corpus preserving linguistic properties.....	183
Figure 9.20: Unsequenced visualisations for synthetic corpus preserving linguistic properties.....	184
Figure 10.1 - VAST in use	200

Table of Tables

Table 3.1 - Source code plagiarism literature identified	26
Table 3.2 - Reviewed detection engines.....	33
Table 3.3 - Languages processed by detection engines	35
Table 4.1 - Free text plagiarism detection literature identified	52
Table 4.2 - Detection services for Web-plagiarism	54
Table 4.3 - Literature comparisons of Web-plagiarism detection services.....	55
Table 5.1 - Visualisation literature for plagiarism detection identified	73
Table 5.2 - Detection engines discussed in chapter	74
Table 5.3 - Techniques for visualising similarity in an entire corpus	77
Table 5.4 - Techniques for visualising similarity in pairs of submissions.....	86
Table 5.5 - Techniques for visualising similarity in single submissions	90
Table 6.1 - Problem of plagiarism literature identified.....	97
Table 6.2 - Plagiarism prevention literature identified	99
Table 6.3 - Academic plagiarism literature identified	101
Table 6.4 - Legal plagiarism issues literature identified.....	103
Table 6.5 - Plagiarism cases discussed	104
Table 6.6 - Plagiarism papers identified from cultural viewpoints	105
Table 6.7 - Plagiarism papers about writing methods.....	106
Table 6.8 - Other papers describing views of plagiarism	107
Table 6.9 - Summary of admitted results in non-UK student cheating literature	108
Table 6.10 - Summary of believed results in non-UK student cheating literature.....	109
Table 6.11 - Summary of opinion results in non-UK student cheating literature.....	109
Table 6.12 - Summary of observed results in non-UK student cheating literature	110
Table 6.13 - Summary of admitted results in UK student cheating literature	114
Table 6.14 - Summary of believed results in UK student cheating literature	115
Table 6.15 - Authorship attribution literature identified.....	117
Table 6.16 - Main authorship attribution techniques identified.....	119
Table 6.17 - Main authorship studies identified.....	121
Table 7.1 - Sample document D1	125
Table 7.2 - Document D1 split into four word fragments.....	125
Table 7.4 - Sample document D1	126
Table 7.5 - Document D2 in four word fragment representation	126
Table 7.6 - Calculating similarity between D1 and D2	127
Table 7.7 - Sample document D3	127

Table 7.8 - Document D3 in four word fragment representation	128
Table 7.9 - Calculating similarity between D1 and D3	128
Table 7.10 - Calculating similarity between D2 and D3	129
Table 7.11 - Similarity scores for pairs of D1, D2 and D3.....	129
Table 7.12 - Similarity scores for pairs of D1, D2 and D3 under simple metrics	131
Table 7.13 - Similarity ranks for pairs of D1, D2 and D3 under simple metrics.....	132
Table 7.14 - Consolidated similarity ranks for pairs of D1, D2 and D3.....	132
Table 7.15 - Plagiarism techniques used	134
Table 7.16 - Simple metric ranks for source/copy pairs containing known plagiarism	135
Table 7.17 - Consolidated metric ranks for source/copy pairs containing known plagiarism	136
Table 8.1 - Two document showing possible areas of similarity.....	138
Table 8.2 - Two documents showing similarity in fragments	138
Table 8.3 - Two documents, showing word proportions fragments have in common	139
Table 8.4 - All possible fragments of two documents.....	140
Table 8.5 - Word count in common for all possible fragment pairings of two documents	140
Table 8.6 - Percentage in common for all possible fragment pairings of two documents	141
Table 9.1 - Possible visual metrics	161
Table 9.2 - Summary of comparative scores for possible visual metrics	166
Table 9.3 - Expected similarity ranks for synthetic corpus.....	169
Table 9.4 - Words metric similarity ranks for synthetic corpus.....	170
Table 9.5 - Characters metric similarity ranks for synthetic corpus	170
Table 9.6 - Sentences metric similarity ranks for synthetic corpus.....	171
Table 9.7 - Mean of max metric similarity ranks for synthetic corpus	172
Table 9.8 - Words metric similarity ranks for second synthetic corpus	175
Table 9.9 - Characters metric similarity ranks for second synthetic corpus.....	175
Table 9.10 - Sentences metric similarity ranks for second synthetic corpus	175
Table 9.11 - Mean of max metric similarity ranks for second synthetic corpus	176
Table 9.12 - Correlations between simple words metrics and mean of max metric for corpora containing known noise levels	178
Table 9.13 - Correlations between simple characters metrics and mean of max metric for corpora containing known noise levels	178
Table 9.14 - Correlations between simple sentences metrics and mean of max metric for corpora containing known noise levels	178
Table 9.15 - Correlations between simple words metrics and mean of max metric for 10% stochastic noise synthetic corpus	181

Table 9.16 - Correlations between simple characters metrics and mean of max metric for 10% stochastic noise synthetic corpus	181
Table 9.17 - Correlations between simple sentences metrics and mean of max metric for 10% stochastic noise synthetic corpus	181
Table 9.18 - Correlations for synthetic corpus containing known English language sequences.....	185
Table 9.19 - Correlations for high similarity corpus between simple and mean of max metrics.....	186
Table 9.20 - Details of submitted student corpora	187
Table 9.21 - Statistics for submitted student corpora.....	188
Table 9.22 - Statistics for submitted student corpora.....	188
Table 9.23 - Submitted student corpora as processed	189
Table 9.24 - Correlations for corpus P1.....	189
Table 9.25 - Correlations for corpus P2.....	190
Table 9.26 - Correlations for corpus P3.....	190
Table 9.27 - Correlations for corpus P4.....	190
Table 9.29 - Correlations for corpus P6.....	191
Table 9.30 - Correlations for corpus P7.....	192
Table 9.31 - Correlations for corpus P8.....	192
Table 9.32 - Most significantly correlated metrics for real corpora.....	193
Table 9.33 - Two short documents.....	194
Table 9.34 - Counts of similar word pairs for documents A and B	194
Table 9.35 - Total counts of common and unique word pairs in A and B	194

Chapter 1: Introduction

Overview:

- The problem of detecting student plagiarism is discussed.
- The structure for the remainder of the thesis is described.

Plagiarism by higher education students has been a thorn in the side of academics for years. This thesis proposes and answers the question - how can plagiarism be detected effectively and efficiently.

This chapter describes the problem, outlines the aims of the thesis and introduces a structure for the rest of this work.

1.1 - Motivation

This chapter provides a concise introduction to plagiarism detection and this thesis as a whole. Detection is becoming necessary to preserve the integrity of academic awards in the marketplace of higher education.

Section 1.2 gives an introductory working definition to what plagiarism is and why its detection is necessary. Section 1.3 sets out the aims for the thesis, namely to identify technical methods to aid tutors in the human led process of detection. Section 1.4 describes the structure of the remaining chapters.

1.2 - Defining the problem

Plagiarism is the act of using the words or ideas of another person without acknowledgement. Plagiarism is a skill learned by most people, from children copying from their friends to complete homework assignments, to those people applying for jobs with an edited version of a friend's CV. It becomes apparent that students at university level might also plagiarise, a practice that is wrong because it means them obtaining academic credit without demonstrating appropriate skills or knowledge. If this is happening regularly it could go so far as devaluing the academic awards given out by an institution. The practice is also offensive to students who might complete work legitimately but are being penalised by the students who plagiarise.

Many tutors would be dismayed if their students were plagiarising, but the time and effort needed to detect such plagiarism can be counterproductive for a busy tutor. What is needed is a process, automated as far as possible, that would show tutors which student submissions such as essays are similar to each other and where they are similar. Such a process would be difficult to completely automate, since a computer cannot decide whether areas of similarity are legitimate, such as shared use of the same cited sources. Since the whole language and process behind plagiarism is ill defined a better description is needed alongside an efficient and effective detection process and that is what this thesis sets out to achieve.

1.3 - Aims of the thesis

There is an ever-growing body of work detailing that students are plagiarising, but work relevant to the UK is in its infancy. The main survey, by Franklyn-Stokes and Newstead, in 1995, found that over half of all students admitted to cheating behaviours that could be considered forms of plagiarism (Franklyn-Stokes & Newstead 1995). All indications today suggest that the figure would be higher, since the Franklyn-Stokes and Newstead study was carried out before widespread use of the Web provided students with easy access to material to 'copy and paste' into their work. Many media articles have stated that such Web plagiarism is a serious problem (Denhart 1999, Sutherland 2000, Cramb 1999).

It is not the purpose of this thesis to identify how students are plagiarising, why they are plagiarising, what steps tutors can take to stop them, or what issues exist in plagiarism detection. There are publications and assorted Web sites on each of these (Gajadhar 1998, Ryan 1998, Austin & Brown 1999). Instead this thesis is concerned with detection from a technical standpoint.

One of the main aims of the thesis is to develop tool support for tutors investigating plagiarism. Since any developed tools are intended to be used on a standard home or office PC the processing requirements should be modest. For this reason all the data processing necessary when researching technical solutions to the plagiarism problem has been carried out on a similar powered machine. For the most part this has been adequate, although this has meant that occasionally only samples of data can be processed where intense calculations are needed, for instance in Chapter 6.

The remainder of the thesis will detail the technical solutions for plagiarism. The limited amount of assistance initially available will be detailed as part of a review of the literature. Methods will be found to identify student submissions that are similar in some way. Finally tool assistance will be developed to help tutors who are investigating similarity for possible plagiarism, possibly the most involved part of the entire process.

1.4 - Structure of the thesis

The Introductory Chapters

Chapter 1 (this chapter) has introduced plagiarism and given some motivation as to why it is considered an important area for research.

Chapter 2 formalises a language for discussing plagiarism and a process for detecting plagiarism. Since these are areas not considered in the existing literature a suitable framework is needed in which to formulate a research question. The question - how can plagiarism be detected in free-text student submissions is formulated and discussed.

The Review Chapters

Chapter 3 reviews the literature on technical methods for finding plagiarism in source code submissions. This is an area that has been researched for years and in which the most important questions have been answered. The methods used do not translate directly into free text detection. The chapter notes inconsistencies in the ways that different detection engines can be classified and proposes alternative classifications.

Chapter 4 reviews the literature on technical methods for finding plagiarism in free text student submissions. This is a new area of research and most of the material presented here is recent. The chapter particularly focuses on literature on the services that aim to find material copied from Web sites and the body of the work that shows how submissions can be compared to assess if they contain similarity.

Chapter 5 reviews the literature for visualising text sources, in particular for showing where or how student submissions are similar. It is argued that appropriate uses of such techniques can reduce the time taken by tutors when investigating possible plagiarism cases.

Chapter 6 overviews the remaining relevant literature. Plenty of material has been published on plagiarism that is not written from a technical perspective and that material is covered here. This includes studies on the extent of and motivations for student cheating, the growing problem of Web plagiarism and linguistical methods for authorship attribution.

The Research Chapters

Chapter 7 describes a set of metrics that can be used to find pairs of similar student submissions. It is argued that those with high measures of similarity are the pairs that should be investigated further by tutors. The different metrics are contrasted based on their effectiveness on corpora containing known plagiarism.

Chapter 8 describes a way of making the plagiarism detection process more efficient and effective, from the point of view of aiding tutors investigating similarity that might or might not represent plagiarism. An innovative graphic known as a similarity intersection is described and a tool that uses the graphics to partly automate the process of finding similarity is presented.

Chapter 9 shows how the process of finding out which pairs are worthy of further investigation can be made more efficient. A suitable metric is identified that orders pairs of submissions based on properties of their associated similarity visualisations but takes only a fraction of the time to compute than producing an entire set of similarity visualisations would.

Chapter 10 summarises the thesis and presents some conclusions and ideas for further work. A whole process is presented that is partly automated based around the techniques described in the thesis and developed tools.

The Appendices

The *Appendices* contain published related papers and other information that would break the flow of the main text.

Appendix A contains of the glossary of the main plagiarism related terms introduced in the thesis.

Appendix B contains the paper *A Descriptive Taxonomy of Student Plagiarism* which exists in pre-publication state but the language and ideas are gaining acceptance in the academic community. This thesis as presented uses the language presented in that taxonomy for purposes of consistency and clarity.

Appendix C contains the paper *Plagiarism Prevention, Deterrence and Detection*, a paper commissioned by the Institute for Learning and Teaching and available as online resource.

Appendix D contains the refereed paper *Plagiarism Issues in Higher Education* commissioned by the Vine journal and presented at a related workshop.

Appendix E contains the paper *A Review of Electronic Services for Plagiarism Detection in Student Submissions* presented at the 2000 LTSN conference.

Appendix F contains the technical report *Source Code Plagiarism in UK HE Computing Schools, Issues, Attitudes and Tools* commissioned by the Joint Information Systems Committee and presented at several workshops.

Appendix G contains the refereed paper *Towards an Error Free Plagiarism Detection Process* presented at the ITiCSE 2001 international conference.

Appendix H contains the refereed paper *Visualising Intra-Corporeal Plagiarism* presented at Information Visualisation 2001 international conference.

Appendix I contains a set of graphical visualisations for the high similarity corpus introduced in Chapter 9. These allow pairs of submissions to be quickly verified to see whether they contain similarity or not.

Appendix J contains a complete processing of simple metrics for three sample documents introduced in Chapter 7.

1.5 - Conclusions

This chapter has briefly introduced the material and problem area that will be covered in the remainder of the thesis. Chapter 2 will provide a fuller introduction introducing a new language that allows plagiarism to be more readily discussed. Chapters 3 to 6 will review the existing plagiarism literature. Chapters 7 to 10 will present new ideas that allow the human led process of detection to be machine aided.

Chapter 2: A Framework for Plagiarism Detection

Overview:

- Terms for discussing plagiarism are proposed.
- Ways of classifying plagiarism detection engines are discussed.
- A process for part-automated plagiarism detection is defined.
- Details of document comparison techniques are given.
- A research question is formulated.

This chapter describes background material that enables plagiarism to be both discussed and defined as a research area. This is necessary since plagiarism detection in free text is considered a new field of research.

The chapter has a number of specific components. These include the introduction of a new language for plagiarism that allows it to be talked about. Methods of classifying plagiarism detection engines are declared. A process for identifying similarity and hence finding potential plagiarism is presented. The process is part human led and part computer assisted. Further details of how the computer-assisted stages of the process may be operated are given.

The introductions are used to motivate a more focused area of research, namely how texts can be compared in an efficient way to find similarity within a corpus or class set and how this can be done whilst minimising extra work for tutors. The ideas developed in this chapter will be followed up in the rest of the thesis.

2.1 - Motivation

Traditionally a thesis contains a small amount of introductory material and a substantive literature review used to motivate a targeted research question. This approach depends on there being suitable literature available to both define a problem area and identify a suitable question.

For plagiarism detection this is not the case. Although the problem of finding plagiarism in program source code has a long history and substantial body of relevant work this is not directly applicable to finding plagiarism in free text. Much relevant work on the free text problem has only become available during the production of this thesis and some has been motivated by the material that will be presented in this chapter.

It is quite unusual to have to define an area before work can be done on it. Hence much of the material in this chapter is an original contribution to the field of plagiarism detection research.

All indications suggest that plagiarism is a substantial problem. It is easy for a time pressurised student to piece together material from a source or a number of sources on the World Wide Web and hand it in as their own work and all indications suggest that students are plagiarising in this way (Denhart 1999, Gajadhar 1998, Ryan 1998).

There are a few services around to detect such plagiarism, mostly existing on a fee basis and a study of these Web-based plagiarism detection services is included in Appendix E. At the time of that report these services were in their infancy and the actual line up has changed but the material has been used to motivate the discussion in this chapter. A similar review that is related to plagiarism in program source code is included in Appendix F.

One of the first problems identified when trying to research plagiarism detection was how little work had been done on it. In particular the lack of a standard language to discuss plagiarism and a standard automatable process to follow when investigating plagiarism were immediate shortcomings.

The first of these was solved by the preparation of a plagiarism taxonomy that is used throughout this thesis. The taxonomy is included in Appendix B. Where relevant terms are introduced for the first time in the main body of text they are presented in italics. A further definition of these terms can also be found in the glossary in Appendix A. The language allows the whole subject of plagiarism to be discussed and is presented in Sections 2.2 to 2.6.

An automatable process has been solved by the introduction of the four-stage plagiarism detection process, which splits off the process of finding that a student has plagiarised from the process of cataloguing evidence and presenting that evidence. The process has been described in several papers including an in-depth description in 'Plagiarism, Prevention, Deterrence and Detection' which also describes the existing tools and research in the area and is included in Appendix C. The paper 'Plagiarism Issues in Higher Education' further describes some of the issues involved at each stage of partly automatable detection. The paper is included in Appendix D. Both papers motivate the description given in Section 2.7 which is much compressed to cover the technical aspects of the process. Sections 2.8 and 2.13 go into more detail about automating part of the process.

The ways in which plagiarism detection engines work differ. Ways that engines can be classified are discussed through the types of submissions they operate on and the features of the engines are discussed in Sections 2.9 and 2.10. Sections 2.11 and 2.12 describe classifications through the metrics that the engines deploy. These will be used in the literature review of source code detection in Chapter 3 to show inconsistencies and limitations of classifications used in the literature.

The new language allows a research area to be identified and defined in Section 2.14. The process of machine assistance of the four-stage process will form the main body of work in this thesis in Chapters 6 and 7.

This chapter is intended as an introduction suitable for modelling plagiarism detection from a technical perspective. Some of the included papers in the appendices cover detection from a more sociological side but material from them is not directly included here to avoid breaking the flow of the text.

2.2 - The need for detection

Plagiarism describes the process of using the words or ideas of another without suitable acknowledgement. This could include describing the same general points as another writer but using different words, without declaring that the other writer had presented those ideas first. It could include simply using the same words as another, even if those words are placed in quotation marks, without naming the source. There are also places where the acknowledgement is debatable, such as a section of text that is correctly referenced, followed by a section of text that is sourced from the same place but is phrased in such a way that it could be original. All of these are examples of bad writing practice that could constitute plagiarism.

Plagiarism is immediately unacceptable because it is akin with stealing. In the case of *student plagiarism*, that is plagiarism carried out by students for *academic credit*, the result can see the stealing of an academic qualification. This is unacceptable because it can bring an *academic institution* into disrepute if a student turns out to be unsuitable for a job offered due to their alleged academic skills. It also means that students who are working hard may see themselves as being unfairly penalised, since others may have the same or better results through cheating.

Plagiarism detection in this context refers to the process of finding out which *student submissions*, those pieces of work submitted by students for academic credit, are plagiarised from which *plagiarism sources*. Possible sources include but are not limited to the World Wide Web, textbooks and other students. The computer-based tool used to assist in finding plagiarism is known as a *plagiarism detection engine* or, interchangeably, *plagiarism detection system*.

One reason that plagiarism has become so much in the public eye is the level of media attention (Cramb 1999, Claire 2000). The growth of the World Wide Web has provided students with easy access to a large amount of material on any given topic. Although this could be legitimately used and cited students may instead find a complete report on a subject they need to write about in one of the many *essay banks* that now exist. Some tutors now frequently report finding that students have submitted work that has been 'copied and pasted' from the Web whilst others, perhaps already under academic pressure, might not notice that there is a problem, or may give their students the benefit of the doubt. That is not to say that traditional plagiarism, such as copying from textbooks, friends or obscure academic papers, has disappeared. Nor has paying someone to produce work on your behalf. It is the growth of Internet assisted cheating that has led to many people believing that plagiarism is now endemic.

As such academic institutions need a robust *plagiarism detection process* where plagiarism is identified and then investigated following a defined procedure. A secondary requirement for finding a suitable process is that it should impose minimal amounts of extra work on *tutors*.

2.3 - Types of plagiarism

Plagiarism can be classified in different ways.

The first possible way is by the type of person who is committing the plagiarism. Student plagiarism is where a student is submitting plagiarised work for academic credit. *Academic plagiarism* is where an academic uses plagiarised materials for professional development, primarily by submitting plagiarised papers to conferences or journals. *Professional plagiarism*, such as copying a report from a competitor, refers to plagiarism within the workplace. Academic plagiarism could be considered as one example of professional plagiarism. This thesis is primarily interested in student plagiarism.

Plagiarism can also be classified according to the source of the plagiarism. These definitions are concerned especially with student plagiarism but could be applicable to elsewhere. Traditionally a student will be asked to complete a piece of work to a given *assignment specification*. The piece of work they hand in is then known as a *student submission*. The set of all student submissions for a given assignment specification is then known as a *corpus* (a corpus is a standard term used for a set of documents in linguistics). *Corpora* (the plural of corpus) can also be made in different ways so long as all the documents inside are linked in some way. A number of corpora could be produced from a single assignment specification if the same specification was presented in multiple years. Then a corpus could be produced for every new batch of students studying the material, or a single corpus containing every submission could be produced. Another possible linkage might be a corpus containing all the work produced by a given individual student. Submissions within such a corpus could be expected to have similar linguistic properties.

Plagiarism within a corpus is known as *intra-corporeal plagiarism*. For a corpus containing the work of a single group of students this would represent the case where one student was copying from another. In such a case the *source submission* and the *copy submission* would not be immediately identifiable, although there may be clues. When the plagiarism source is outside the corpus, such as in a journal, or in a submission from another institution, this is known as *extra-corporeal plagiarism*. One type of extra-corporeal plagiarism worth classifying further is that of *Web plagiarism* where some or all of a submission is sourced from the World Wide Web, a problem that has sprung up over the last few years. Many cases of Web plagiarism are *multiply sourced*, where material has been copied from more than one place. Intra-corporeal plagiarism is more likely *singularly sourced*, committed with a one-to-one correspondance. A set of submissions containing similar material within a corpus is known as a *cluster*.

It is important to differentiate between plagiarism and *similarity* in this context. Similarity refers to two documents, or part of two documents, containing material that has been judged alike in some way. This similarity can only be referred to as plagiarism once it has been examined and verified in some way by a tutor. Otherwise there might be legitimate reasons for the similarity, such as the two students using the same correctly cited materials.

This thesis is primarily concerned with detecting and investigating intra-corporeal plagiarism, although extra-corporeal plagiarism can be identified in a similar way if potential or known sources are added to a corpus of student submissions. It is also possible for extra-corporeal plagiarism to appear as intra-corporeal plagiarism, when two or more students have plagiarised from the same external source.

A final useful classification is through the type of material being examined. *Source code* refers to computer programs written in languages such as C and Java. *Free text* refers to submissions written in a natural language, such as English, covering regular essays and reports. *Source code plagiarism* and *free text plagiarism* refers to plagiarism within the respective type of submissions. These are not the only types of plagiarism possible, for instance students could plagiarise program design diagrams such as UML or musical compositions but these are of little interest in textual corpora. This thesis is concerned with free text plagiarism detection.

2.4 - Methods of plagiarising

It is useful to be able to discuss the ways in which a plagiarism source is altered into a copy that a student may hand in. Different methods of plagiarising might suggest different ways of detecting such plagiarism.

In some cases a student might not change a source at all, for instance if the student believes they are downloading work that suitably meets an assignment specification from an obscure Web site. This is known as a *direct copy*. Anything other than a direct copy is a demonstration of attempted *disguise*. This might include rewriting the start and end of a source in the hope that the common material in the middle of the copy goes unnoticed or re-arranging *fragments*, those split sections, of a source with some new material into a copy.

How fragments of a text from the source are changed into a copy also gives different methods of plagiarising. If the fragment is padded out with some additional material, for instance filler words, sentences or new points, this is known as *expansive plagiarism*. Where the same ideas are presented but in less words, or parts of the source fragment are removed, this is known as *contractive plagiarism*, also known as *summarising*. Presenting the same ideas but in a rewritten format, without suitable acknowledgement, is *paraphrasing*. Where many fragments from one or more sources are presented in a rearranged form, the result is *mosaic plagiarism*, this being a common linguistic definition (DePauw 2002).

A student might take a source and selectively replace words with their synonyms or words with the same meaning, or expand abbreviations to their full form. This plagiarising technique is known as *thesaurising*. The most general word by which all words could be replaced is known as the *root word*.

2.5 - The collaboration, collusion and copying continuum

For plagiarism detection to be successful it is necessary to differentiate between activities that students may engage in and those that they may not engage in. Otherwise two student who work closely together when preparing their submissions may submit work judged partly similar thus committing intra-corporeal plagiarism.

Figure 2.1 shows the continuum of *collaboration*, *collusion* and *copying*, based on material presented by Culwin and Naylor (Culwin & Naylor 1995). Collaboration is defined as students discussing between themselves how they will go about completing work to an assignment specification and is deemed both acceptable and suitable for encouragement. Collusion is where two students have access to the same material when writing up their submission. Copying is where two students write up a submission together, or one student writes up a submission and passes it onto another and is definitely unacceptable.

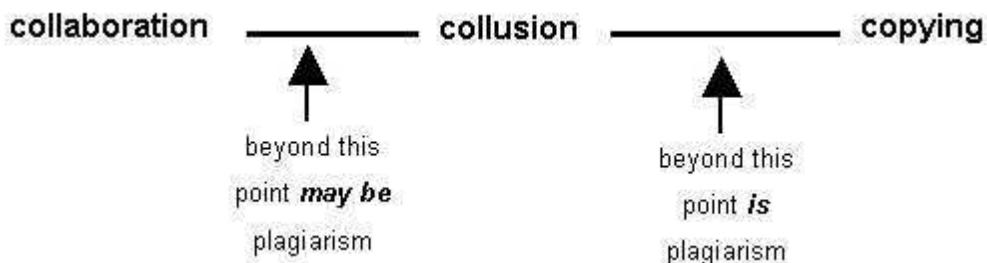


Figure 2.1 - Collaboration, collusion and copying

The arrows on Figure 2.1 demonstrate that whilst collaboration is acceptable and copying is not there is no fine line between the two as to where plagiarism starts. It is a matter open to debate by both tutors and dependent on subject area and the nature of an assignment. All that can be said is that if students do more than collude they are plagiarising. If students do more than collaborate than they may be plagiarising.

This continuum also serves as a warning for automated detection methods that there may be some debatably acceptable collusion leading to small parts of two submissions being similar. This means that there will always be a need for human judgement to determine if copying has or has not occurred. Anything more than small sections of duplicated material can thus be judged as intra-corporeal plagiarism.

2.6 - Plagiarism detection policies

Academic institutions can make use of three main types of plagiarism detection processes. Those institutions that do nothing about plagiarism have an *ostrich policy*. This is a rare situation at the current time since plagiarism is a hot issue in many institutions and few believe that plagiarism is just not happening. The Quality Assurance Authority for Higher Education provide guidelines in the UK that state that institutions should have a valid policy (QAA 2002).

If an institution takes plagiarism seriously when it is found, but does not go out of its way to look for it, then it can be said to have a *reactive policy*. Such a policy would not present many opportunities for automation but could still be tool assisted where plagiarism is believed to have happened and some investigation is necessary.

A *proactive policy* is necessary for institutions that want to not only take plagiarism seriously but also actively seek it out. An automated or part automated plagiarism detection policy requires a proactive policy since it will require electronic versions of submissions to find similarity in and it is not really feasible to scan in typed submissions. *Electronic submission*, the handing in of work in electronic format, such as on disk, through e-mail or through a Web-based engine has an additional advantage of reducing the workload on clerical staff.

2.7 - Four-stage plagiarism detection process

The *four-stage plagiarism detection process* is an example of a proactive plagiarism policy where all student submissions are examined for plagiarism and those that may contain it are investigated further. The process, as shown in Figure 2.2, illustrates a framework within which detection can take place, with stages that can be automated in full or in part. The computer icon indicates a stage that can be automated. The tutor icon indicates where human assistance is necessary. Where a stage has both icons the stage requires both computer aid and human assistance.

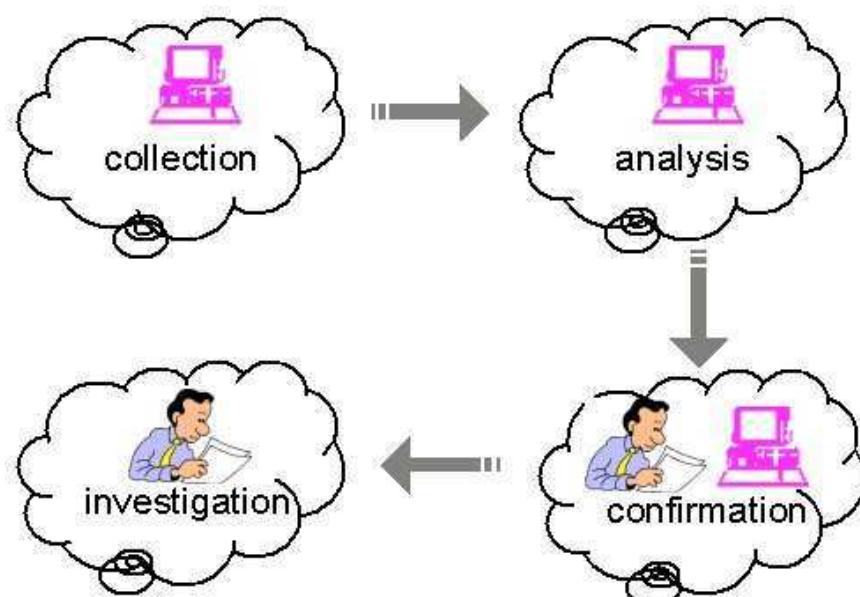


Figure 2.2 - Four-stage plagiarism detection process

The stages of the process are as follows. First is the *collection* stage, which is where students submit their work to the engine, perhaps by using a Web front end. Next is the *analysis* stage. Here the corpus of submissions is run through a computerised *similarity engine* that produces some sort of measure of which submissions are potentially plagiarised. The effectiveness of the detection will depend on the methods used in this stage. An intra-corporeal engine will likely return an ordered list of similar pairs. An extra-corporeal engine may return appropriate Web links. The third stage is the human led stage of *confirmation*. This is necessary to decide if the reported similarity represents plagiarism or not. Although this stage is carried out primarily by hand this research will show that machine assistance is both possible and desirable. Any

similarity deemed plagiarism is considered further in the *investigation* stage where a penalty may be finally given.

The four-stage process separates the finding of similarity, which here is done in a proactive automated way, from any decisions about the impact of this similarity. There are two important things to consider when automating the process. The first is the effectiveness of the algorithms used to initially find similar submissions. The second is the time taken during the human led confirmation and investigation stages and how this could be reduced, but never eliminated, with appropriate machine assistance.

2.8 - The use of metrics in the analysis stage to assess document similarity

This thesis is concerned with the successful detection of intra-corporeal plagiarism. This means that in the analysis stage of the four-stage process it is necessary to find out which student submissions are similar to which other student submissions. This will be used to inform a tutor which pairs they have to check for possible plagiarism.

The result of the analysis stage is that a list of pairs of student submissions is produced, from most to least similarity, under chosen criteria. The list is known as a *similarity rank list* and the position of each pair in that list is known as its *similarity rank*. This means that every submission in a corpus will have to be compared with every other submission.

One operational way of then checking for intra-corporeal plagiarism in the confirmation stage would be for a tutor to check the first pair in a similarity rank list for plagiarism. They would then work their way down successive ranks in sequence until the pairs no longer show evidence of excessive similarity or the tutor's available time resources are exhausted.

The criteria used to rank pairs of submissions are known as *metrics*. The results from metrics can be merged in some way to give a *similarity score* for each pair of submissions. The similarity score can be normalised to give a numeric value between 0 and 100, with 100 indicating complete similarity and 0 indicating complete uniqueness. Higher scores would then represent a pair of submissions with more similarity than those with a lower score although the scores themselves may not be directly translatable into percentages of similarity. Section 2.13 gives more details about the process through which a similarity score is assigned to two submissions.

The similarity scores produced give no directly obvious cut off point, above which plagiarism has occurred and below which it has not. Instead it is more useful to think of the similarity rank list can then be produced by ordering the pairs based on their similarity scores.

Clearly there may be problems if the metrics used do not produce a similarity rank list that is well ordered. Pairs that do contain intra-corporeal plagiarism may not be examined or pairs that do not contain similarity may be ranked too high.

A pair containing similarity that is not ranked in the top portion of the list (that is the portion that a tutor would manually check) is known as a *missed pair*. This represents a problem with the metrics used since this means that two students may be plagiarising without being detected.

A pair that does not contain any notable similarity but is ranked in the top portion of the list is known as a *false hit*. This is problematic since it means that tutors would waste their time investigating two submissions that have been unsuitably ranked. If false hits are only occasional they are not as big a problem as missed pairs since they could be considered as a type of noise but if they happen regularly they could swamp the similarity rank list. These false hits may also serve to reduce a tutor's faith in using the engine.

2.9 - Detection engines classified by submission type

Perhaps the most important way to differentiate between plagiarism detection engines is through the type of submissions they operate on. Intuitively *source code plagiarism detection engines* find similarity in student source code submissions and *free text plagiarism detection engines* find similarity in student submissions written in free text.

Although this thesis is concerned with plagiarism detection in text, plagiarism is also possible in other artefacts and hence detection should be possible too. The possibilities here are limitless, so these are only examples. Students can plagiarise diagrams, for instance UML design notation for software. Here although the content or meaning of two diagrams may be the same the layout could be different. Another possibility is music plagiarism where a tune or part of a tune has been copied. It is also possible to plagiarise dance sequences.

2.10 - Detection engines classified by features

Some more general classifications are useful. It is useful to know whether engines are *Web-based* or *local*. It is also useful to know whether engines are available for academic use. Those that are available to anyone either directly or through registration will be known as *public*. Some engines are available by *special arrangement* usually by supplying submissions to be checked directly to the engine provider. Others are available only in *private* to their local institution. Some of the engines have been since superseded, so they can also be classified as *past* or *current*.

Source code plagiarism detection engines can be further classified by the programming languages that they operate on. For instance an engine might only operate on submissions in an imperative language such as C, an object oriented language such as Java, a functional language such as Haskell, or a logical language such as Prolog. Or the engine might operate on a combination of any or all of these. Further some engines will only operate on source code that is parseable, hence not checking submissions that will not parse. Others will operate on non-parsable source code submissions, perhaps because structure is not checked. An alternative approach to non-parsable submissions might involve adding to the submissions automatically to make them parseable although no engines using such techniques are known.

2.11 - Classifications of detection engines through number of submissions processed by the metrics used

One possible method of classifying different metrics is through the number of submissions they need to process in order to detect similarity. For each type of metric possible examples are given. Some of the examples for source code are pulled from the metrics commonly used in the literature. The examples for free text are intended to be comparable to those for source code and are intended to be illustrative rather than suggestions that the metric might be appropriate for plagiarism detection.

Some of the examples mention *tokenisation*. This is a process of word or identifier replacement that is intended to remove the effects of disguise in plagiarised submissions. For source code this might involve reducing certain keywords, for example both ‘for’ and ‘while’ loops to a common token and all identifiers to a different common token. For free text submissions words could be represented by their word group, e.g. noun, verb, or by their root word, the most generic possible replacement word. The tokenisation techniques used are dependent on the engine.

Singular Metrics

A *singular metric* is one that can be applied to a submission to generate a simple number representing some property of the submission. The examples given below are intended to start with ones that are language generic and end with those that are specific to the programming language used:

- The mean number of characters per line.
- The number of lines that contain comments.
- The proportion of ‘while’ loops to ‘for’ loops.
- The mean number of lines per subprogram.
- The number of unique operators in the submission.
- The number of compilation errors when the code is compiled.
- The run time with a chosen set of standard inputs.

Examples of singular metrics for free text include:

- The mean number of words per sentence.
- The number of nouns used in the free text submission.
- The mean word number of words between occurrences of ‘the’.
- The reading age of the text.
- The proportion of uses of ‘their’ compared with ‘there’.
- The proportion of words that are not found in a dictionary.
- The time taken for a computerised voice to ‘read’ the text aloud.

Paired Metrics

A *paired metric* is defined on two submissions, to give a direct measure of similarity. This should give more information than could be gleaned by simply computing singular metrics for two submissions and comparing them. Examples of paired metrics for source code include:

- The number of keywords that occur in both source code submissions.
- The length of the longest tokenised substring common to both.
- The number of functions in the first submission that can be paired with a function with the same number of lines in the second.

Examples of paired metrics for free text include:

- The number of capitalised words that occur in both free text submissions.
- The length of the longest substring common to both.
- The number of paragraphs in the first submission that can be paired with a paragraph with the same number of sentences in the second.

Research to find suitable metrics in this thesis will be concentrated around paired metrics. Some further definitions will be useful later. Those paired metrics that can be applied to two documents without needing any intermediate processing will be known as a *simple metric*, partly because the amount of computer processing needed to identify similarity is low. Since this metric is being applied to the entirety of two submissions it can be said to be being applied at *gross granularity*. If a metric were being applied to only a single fragment of two submissions it would be being applied at a *fragmentary granularity*. A set of metrics known as *visual metrics* will later be introduced that compare two submissions at fragmentary granularity and then combine the results to give an overall similarity score, but will not be discussed further here.

Multi-Dimensional Metrics

These metric descriptions are intended to be inclusive. It is possible to define singular metrics on single submissions and paired metrics on pairs of submissions. Although there are no examples in the literature there must therefore be metrics that can be applied on three submissions, called, say, three-dimensional metrics and metrics that can be applied on four submissions, say, four-dimensional metrics. Singular metrics can be thought of as a one-dimensional metric and paired metrics as a two-dimensional metric. In general, for some integer n , an n -dimensional metric is one applied on a group of n submissions at a time. Such *multi-dimensional metrics* could be useful for clustering groups of plagiarised submissions involving cohorts larger than two students.

Corpal Metrics

A corpal metric is a multi-dimensional metric, the number of dimensions of which is equal to the number of submissions in the corpus. This means that the metric is a measure of some property of the entire corpus.

An example of a corpal metric for source code submissions would be the proportion of submissions using the keyword ‘while’.

An example for free text would be the proportion of submissions that contain the word ‘hence’.

One use of corpal metrics might be to compare submissions from one group with another, perhaps to consider which group produces the ‘best’ source code under some chosen criteria. A second could be to identify keywords with a high usage rate allowing the corpus to be re-examined using those keywords in paired metrics.

2.12 - Classifications of detection engines through operational complexity of the metrics used

A further method for classifying metrics is through the complexity of the calculations needed to process them.

Superficial Metrics

A *superficial metric* is a measure of similarity that can be gauged simply by looking at a student submission or a number of student submissions. No knowledge of the structure of a programming language or the linguistic features of natural language is necessary.

Examples of superficial metrics include:

- The count of the reserved keyword ‘while’ in a source code submission.
- The number of runs of five words common to two free text submissions.

Structural Metrics

A *structural metric* is a measure of similarity that requires knowledge of the structure of a student submission or a number of student submissions. For source code submissions this might involve a parse of the submissions. For free text submissions this could involve reducing words to their linguistical root word form.

Examples of structural metrics include:

- The number of operational paths through a program for a source code submission.
- The length of the longest continuous run of identical words for two free text submissions.

The borderline between where a superficial metric stops and a structural metric starts is a fuzzy one. For instance if a submission is tokenised and a superficial metric applied could the whole process instead be thought of as just a structural metric since tokenisation is a structure dependent process. Hence in some cases these definitions are open to individual interpretation.

2.13 - The detection process using metrics

Calculating Similarity using Singular Metrics

A singular metric for every submission in the corpus can be computed. The pairs of submissions with the closest values may contain undue similarity. These are the pairs that tutors may choose to investigate for plagiarism.

In practice a submission does not have just one singular metric associated with it. Instead it is evaluated under a number of different metrics. The vector of metric values representing a submission is known as its *fingerprint*. Such metrics might be unrelated, or may be related. One such related example for free text submissions would be a fingerprint containing the number of sentences of word length one, two, three, etc. This technique which would mean that every sentence in the submissions affected the fingerprint by influencing one of the comparative values. Those pairs of submissions with substantially similar fingerprints are hence the most similar.

Similar pairs of submissions are usually defined as those whose pairs of fingerprints are positioned close together when the values of their associated metrics are plotted in n-dimensional space. The distance between two vectors of metrics can be calculated using an n-valued version of Pythagorus. Hence those pairs with a distance between them closest to zero are the ones requiring further investigation.

Submissions represented by a fingerprint can be compared under more complex schemes. One reason for this is to minimise the effect of metrics that might introduce disparate weightings into the closeness calculation. There are two main ways of enhancing the closeness calculation that can be used individually or in conjunction with one another.

The first is *normalisation*. Results for individual metrics can be scaled to between two chosen values, say 0 and 100, so that each is distributed with the same underlying characteristics.

The second is *weighting*. Individual metrics can be weighted, i.e. by multiplying them by a chosen value, in order for that metric to count more or less towards the overall closeness calculation. Possible weighting strategies include pragmatic weighting, where weightings are chosen by practical investigations and observations; or reasoned, where weights are calculated in some manner.

Figure 2.3 operationalises the scheme for calculating similarity using singular metrics for a fingerprint of N metrics. The tokenisation stage is an optional pre-processing stage in the metric calculations, where those submissions to be compared are reduced to a simpler or alternative form before metrics are applied using the tokenisation techniques described earlier.

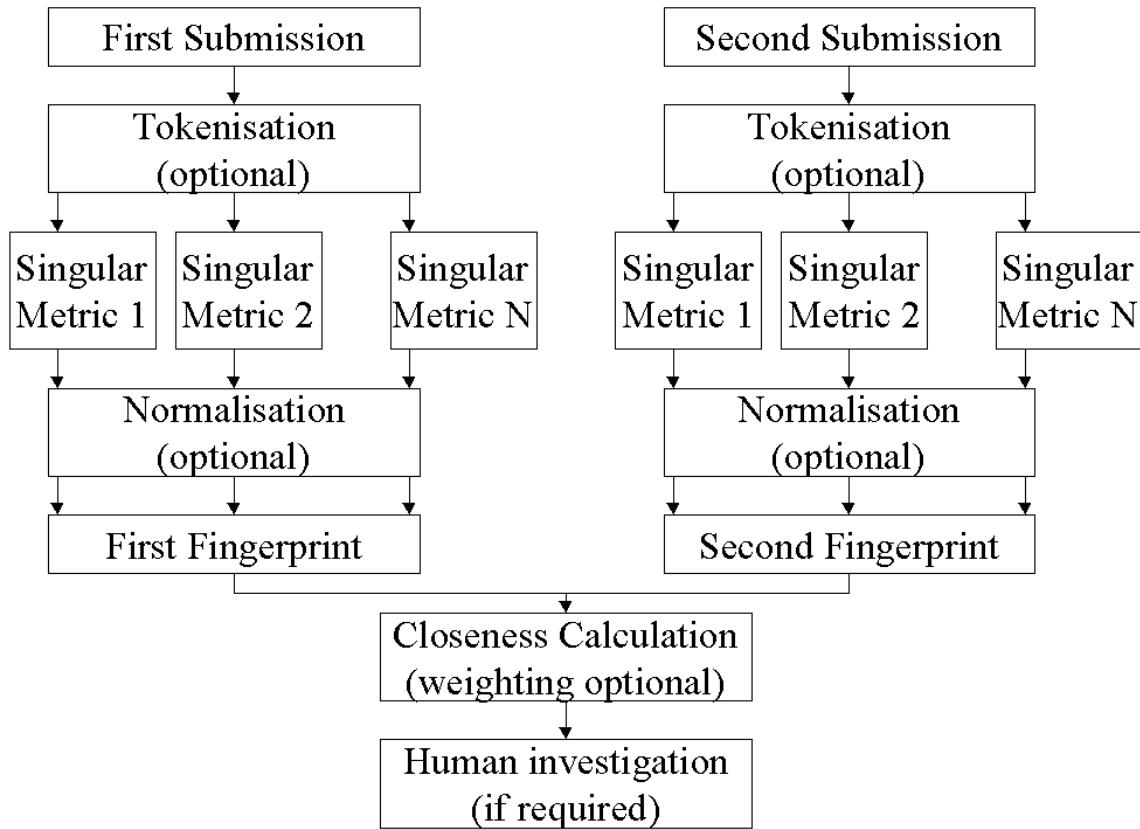


Figure 2.3 - Engine using singular metrics

Fingerprints for individual students can additionally be tracked over time. Any sudden change in the fingerprint generated from a new submission might give tutors reason to worry that the submissions were not all written by the same student. Further fingerprints can be produced for sections of a submission. Any sections which have markedly different fingerprints might have been sourced from somewhere else.

Calculating Similarity Using Paired Metrics

By its very nature a paired metric gives a direct measure of the similarity between two submissions.

The results of multiple paired metrics can be put together to give a paired fingerprint for two student submissions. As with singular metrics, normalisation and weighting of the individual metrics is possible.

Figure 2.4 operationalises the scheme for calculating similarity using paired metrics for a paired fingerprint of N metrics. Here the closeness calculation is not comparing two fingerprints, but instead extracting one fingerprint down to a single numeric value.

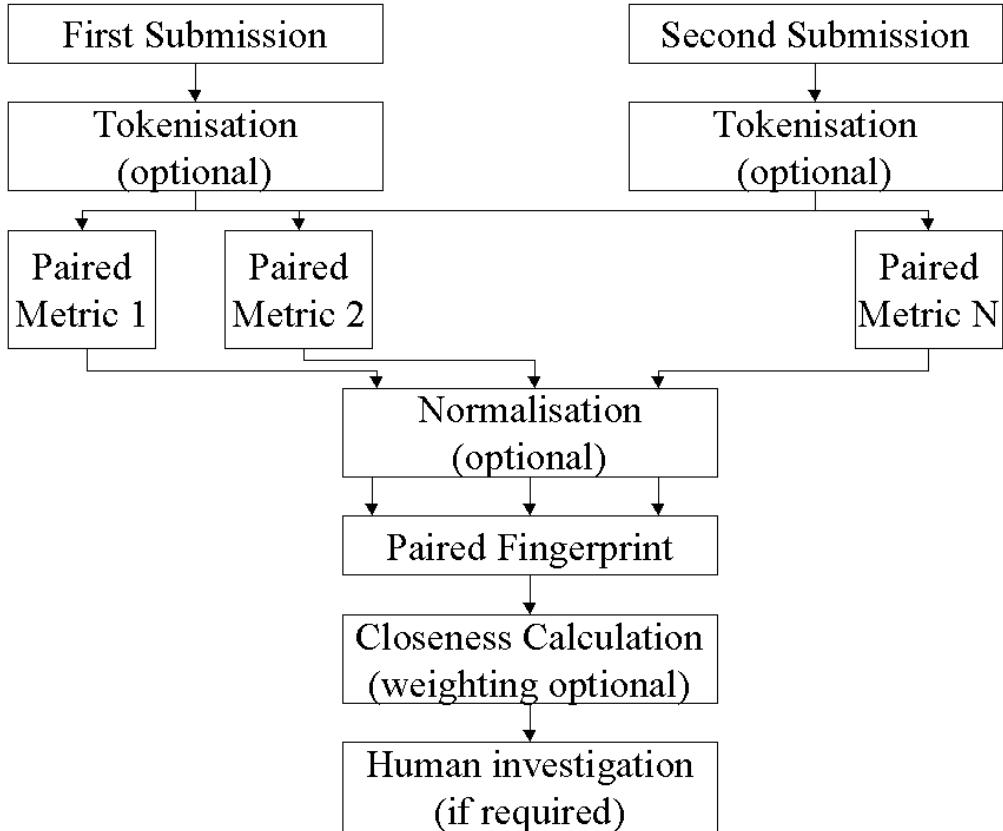


Figure 2.4 - Engine using paired metrics

Tokenisation is often necessary as a prerequisite for applying paired structural metrics. For example source code plagiarism is often detected by searching pairs of submissions for common tokenised strings. They are believed to aid in finding disguised plagiarism, such as re-arrangement of code fragments or changed variable names.

2.14 - Developing a research question

Since the field of free text plagiarism detection is open there are any number of different areas on which a research question can be formulated. However a thesis needs to be tightly focused. This thesis focuses on how technology can aid in plagiarism detection, in particular during the four-stage process. This is intended mainly as a deterrent, since non-technological steps can be taken to minimise the likelihood of student cheating. Two parts of the four-stage process have been identified where work is urgently needed to find plagiarism in free text.

The first is in the algorithms used in the analysis stage. It is necessary to find a simple metric, or perhaps a number of simple metrics that rank similar pairs of submissions in a suitable order. In particular this ordering should avoid false hits and missed pairs. The work will identify a metric that meets these requirements whilst also presenting other desirable qualities, such as minimal computer processing times.

The second observation is that the confirmation and investigation stages are human led and can be demanding of resources to investigate fully. Any tool support that could be provided to tutors here would be of particular benefit. One weakness of being presented by a list of ordered pairs is that this does not give any indication about why the pairs have been considered similar. A tutor presented with two long submissions may find it hard to identify the similar sections in all but the most blatant cases, particularly if re-arrangement has positioned these fragments in different sections of the reports. Hence a new method of showing which areas of two submissions are similar and how similar they are at those points will be developed, using visual methods. The machine assistance, presented in the form of a prototype tool, can then be used to reduce, but never eliminate entirely, the time taken for potential plagiarism to be investigated.

The two questions go hand in hand since they are both linked and so there is a certain amount of cross over in the chapters of the thesis that deal with them. The results from showing why submissions are similar will be used to feed back down the sequence and identify which submissions are similar since it makes sense for both stages to use comparable calculation methods.

The main research question is thus:

**How can intra-corporeal plagiarism be identified *effectively* and
efficiently in free text student submissions?**

The terms effectively and efficiently need defining within this context.

An *effective plagiarism detection engine* is one that produces a sensible similarity rank list, where the pairs at the top of the list that will be manually examined contain similarity and those further down the list do not. This means that the ordering should avoid both false hits and missed pairs.

An *efficient plagiarism detection engine* is one that adds as little as possible to a tutor's workload. Primarily this means during the human-led stages of the four-stage process, namely verification and investigation. This can also include the computer processing time of the analysis stage and indeed any requirements when setting up the collection engine.

Clearly there are going to be trade offs between effective and efficient detection, for instance the most effective metric might not be the quickest to compute. Hence this work will try to find a balance between the two, in particular reducing the time spent by tutors exploring plagiarism cases.

2.15 - Conclusions

Plagiarism detection is a new and developing field of research, but is not one that has received a lot of formal attention. Therefore unlike many fields there is not a great deal of background research to call on.

It has been important to develop a well-defined language with which to talk about plagiarism. This allows a research area to concentrate on in this thesis to be clearly defined. The language also enables plagiarism to be discussed in an academic setting, important when investigating students for potential breaches of academic institution's student misconduct regulations. This chapter has presented the parts of such a language relevant when considering technical and automated means of plagiarism detection.

The whole process by which plagiarism is detected has also been poorly defined. In order to decide which parts of the process are most worthy of assistance it has been necessary to model it in a well-defined way. The four-stage plagiarism detection process clearly separates the process of finding and verifying plagiarism from that of deciding what, if any, penalties should be awarded. Details about how part of it can be computerised and how detection can go ahead operationally has also been given. Looking at the process has also shown that the areas that would most benefit from tool assistance are the human led stages.

The chapter categorises the different types of metrics that can be used in engines for detecting plagiarism. These different types of metrics will be used for comparative purposes in the source code literature review in Chapter 3. A number of ways in which different types of detection engines can be classified, by including the metrics they deploy, have also been presented.

The question of how intra-corporeal plagiarism can be detected effectively and efficiently, with low amounts of human intervention and minimising wrongly ranked pairs of submissions are considered in Chapters 7 to 9.

Chapter 3: Source Code Plagiarism Literature Review

Overview:

- The source code plagiarism literature is discussed.
- Limitations are found in the literature classification methods for detection engines.
- More inclusive classifications are proposed.

Although this thesis is concerned with technical methods for finding plagiarism in free text student submissions that is an area where there is very little literature. By contrast finding plagiarism in source code submissions is well understood and indeed there has been little new research done on it in recent years.

This chapter discusses the available literature on source code plagiarism detection. The methods used are not directly translatable to free text although at some points comparisons are made.

The literature is shown to contain inconsistencies on the ways that different types of source code detection engines can be grouped. Some comments and definitions are not suitable for grouping free text engines. An alternate set of classifications is proposed that is more inclusive. The new classifications are then used to discuss the source code plagiarism detection literature.

3.1 - Motivation

This chapter discusses the literature on source code plagiarism detection. There is little technical work historically on finding plagiarism in free text and hence source code detection is the closest alternative. The chapter assumes the reader has a working knowledge of computer programming and program source code.

The literature traditionally classifies source code plagiarism detection engines into two types, *attribute counting systems* and *structure metric systems*. It is suggested that it is more useful to classify these engines on the metrics they employ to find similarity, as defined in Chapter 2. This approach allows inconsistencies in the current classifications to be explored and more sensible classifications proposed. This also removes a weakness of current classifications that are only intended to describe source code detection engines.

This chapter will discuss the source code plagiarism detection literature. Chapters 4 to 6 will further discuss the plagiarism literature, including automated methods of free text plagiarism detection, visual methods for investigating similarity and non-technical areas that fall outside these chapters' remits.

3.2 - Introduction

Students of computing have traditionally written programs as part of their courses. Program source code is an example of constrained text as it is stylistically constrained to be in parsable. Detection engines date back to at least 1977 (Ottenstein 1997, Halstead 1977) and so known source code plagiarism can be expected to predate that, to at least the early 1970s. It has been software development tutors who have had both the skills and resources to build engines to find plagiarism that stay one step ahead of the students thus preserving academic integrity and ensuring that academic awards hold their values.

Plagiarism in free text submissions, such as student essays, has also been a long-standing problem. Software to find such plagiarism has been dependent on large amounts of affordable computer processing power, something that has only become feasible in recent years. Some of the ideas presented in this section may be applicable to plagiarism detection for free text submissions. Others are dependent on the lexical properties that make up source code submissions and so will not be applicable.

The terminology used throughout this section is consistent with that already introduced in Chapter 2. The original terminology used in the literature is not preserved.

3.3 - Literature identified

Table 3.1 shows, in alphabetical order, a selection of references relevant to source code plagiarism detection. This is not intended to be an exhaustive list as it focused primarily on recent publications and omits the earliest detection engines that have long been superseded. The brief introductions are intended to give a flavour for each piece.

Authors	Date	Content
Boywer & Hall	1999	Describes the MOSS online engine for plagiarism detection.
Carter	1999	Investigates how students collaborate on programming assignments outside formal teaching time.
Clough	2000a	Reviews some of the tools and technologies available for plagiarism detection in both source code and free-text.
Culwin, MacLeod & Lancaster	2001	Identifies the attitudes towards source code plagiarism and its detection in UK higher education. Evaluates Web-based engines that could be used for detection.
Culwin & Naylor	1995	Describes the TEAMHANDIN combined source code submission, plagiarism detection and plagiarism reporting engine.
Cunningham & Mikoyan	1993	Discusses a novel idea for detection that involves Case Based Reasoning and neural networks.
DetectaCopias	2002	Web page for a Spanish language Java plagiarism detection engine.

Ducasse, Rieger & Demeyer	1999	Describes a tool called Duploc that shows similarity in source code with an interactive interface, intended to allow duplication to be minimised.
Granville	2002	Student project that compares a single attribute counting metric with a single structure metric.
Gray, Sallis & MacDonnel	1998	Describes a metric tracking engine for identifying authors of source code.
Hamblen, Parker and Wachtel	1998	Describes a software laboratory where source code is automatically checked for plagiarism.
Harris	1994	Discusses technologies available to prevent students from plagiarising.
Halfman	1994	Presents a visualisation method for showing similarity in source code.
Irving	2000	Slides describing an outbreak of source code plagiarism in Glasgow.
Irving, MacDonald, McGookin & Prentice	2002	User manual for version 2.0 of the Big Brother plagiarism detection engine used internally at Glasgow University.
Jones	2001	Investigates detection using counts of program properties.
Jones	2001a	Again investigates detection using counts of program properties.
Joy & Luck	1999	Describes the SHERLOCK tool and its clustering visualisation method for source code.
Parker & Hamblen	1998	Gives techniques and metrics that can be used for identifying plagiarism.
Prechelt, Malpohl & Phillipson	2000	Describes attempts to validate the JPlag Web based plagiarism detection engine.
Prechelt, Malpohl & Phillipson	2001	A journal submission describing attempts to validate JPlag.
Ribler	1997	PhD thesis describing new visualisation methods with case studies. The study on plagiarism detection using the visualisations is of interest.
Ribler & Abrams	2000	Presents two visualisations that show what a piece of source code has in common with the rest of a corpus.
Rieger & Ducasse	1998	Describes Duploc tool for visualising similarity in source code.
Sallis, Aakjeer & MacDonnal	1996	Describes how metrics can be extracted by source code to identify a programmer's stylistic properties.
Sanders	1998	Reports on a trial of the MOSS Web-based plagiarism detection service.
Saxon	2000	Tests a number of plagiarism detection methods including a novel idea based around the compressibility of source code.
Stephens	2000	Describes a partial plagiarism detection process using attribute counts.

Verco & Wise	1996	Tests plagiarism detection engines against the author's YAP3.
Verco & Wise	1996a	Conference paper that compares YAP3 with other detection engines.
Watson	2001	Discusses how compression with the WinZip software can identify texts written in foreign languages
Williams	1999	Discusses how collaborative programming is viewed differently in industry and academia.
Wise	1996	Describes the technical implementation of the YAP3 detection engine.

Table 3.1 - Source code plagiarism literature identified

3.4 - Defining source code plagiarism

Parker and Hamblen define source code plagiarism as 'a program which has been produced from another program with a small number of routine changes' (Parker & Hamblen 1989). Joy and Luck define plagiarism more simply as 'copying or modifying the work of others' (Joy and Luck 1999).

Culwin and Naylor go further and define a three-point scale of collaboration, collusion and copying (Culwin & Naylor 1995). Collaboration, where students share ideas between themselves, is encouraged. Collusion, where students show designs or code to one another is discouraged. Copying, which is completely prohibited, is where a student allows a physical or electronic copy of their work to be taken by another student. This would hence lead to plagiarism. It is interesting to note that the Culwin and Naylor scheme puts the onus of guilt on students allowing their work to be copied, rather than those who are actually committing the plagiarism. Joy and Luck note that the borderline between cooperation to share knowledge and plagiarism is poorly defined, which adds onus to the Culwin and Naylor view (Joy and Luck 1999).

Joy and Luck define plagiarism further by listing the types of changes students make in plagiarising (Joy and Luck 1999). These are split into two groups. Lexical changes do not involve any knowledge of the programming language and include simplistic changes like altering comments or line spacing. Structural changes do require knowledge of a programming language. Examples include changing procedure calls to function calls or changing the order of statements without affecting program meaning. Verco and Wise seem to believe in a similar idea, stating that plagiarists should be broken down into two groups, those who are novice programmers and those who are experienced programmers (Verco & Wise 1996, Verco & Wise 1996a). Although they do not define the difference between the two this would seem to agree somewhat with the distinction between lexical and structural changes noted by Joy and Luck.

Some papers (Parker & Hamblen 1998 etc) do not attempt to define plagiarism themselves but instead reference a program plagiarism spectrum, originally devised by Faidhi and Robinson. This defines the types of changes students are likely to make when committing source code plagiarism. The spectrum given, in order of increasing sophistication, is:

- L0 – No changes
- L1 – Comments changed
- L2 – Identifiers changed
- L3 – Variable positions changed
- L4 – Procedure combination changed
- L5 – Program statements changed
- L6 – Control logic changed

Although this is not made explicit in any of the literature surveyed this seems to agree with Joy and Luck's notion of lexical and structural changes. L1 and L2 would certainly classify as lexical changes. L4, L5 and L6 would classify as structural changes. It is unclear which group L3 would be classified under so the exact border between the two groups could be said to be fuzzy, in a similar way to the borderline between collusion and copying.

Source code plagiarism can be defined in a number of different ways and this study has found both simplistic and involved definitions. Some papers make no attempt to define plagiarism (Harris 1994, Irving 2000, Saxon 2000 etc), presumably assuming that their readership agree with their understanding. This does beg the question as to why students are expected to be able to understand what plagiarism is, if academics will not or cannot define it explicitly. Culwin and Naylor state that a clear definition of plagiarism given to students is essential in order to operate plagiarism detection measures (Culwin & Naylor 1995) but this literature would make it questionable whether all tutors are aware of this.

3.5 - Literature classifications of detection engines

Chapter 2 suggested a number of ways in which plagiarism detection engines could be classified, primarily based on the types of metrics they deployed. The existing source code plagiarism detection literature also presents classifications. It splits engines into two groups, attribute counting systems and structure metric systems; but the ways these classifications are applied are inconsistent. They are also limited as there are detection engines that do not fit into either classification. These classifications were established before the development of detection engines for free text submissions and do not allow for the different characteristics of such engines.

Verco and Wise report what they state as the common classifications, attribute counting systems and structure metric systems (Verco & Wise 1996, Verco & Wise 1996a). Attribute counting systems are presented as those that measure some property of an individual system. This is referred to as an *attribute count*. Examples given include the number of operand occurrences in a program or a count of the number of possible different paths through a program. This is a system that uses only singular metrics, as defined in Chapter 2.

Structure metric systems are presented by Verco and Wise as any engines that look for similarity in a structural representation of two pieces of source code in addition to using the attribute counting techniques above. This can be considered to be an engine that uses both paired metrics on tokenised versions of source code submissions, as defined in Chapter 2, but also uses singular metrics.

Verco and Wise acknowledge that recent engines, including their own YAP3, use only what they call *structure metrics*. The structure metrics are tokenised paired metrics, as opposed to attribute counts. The grouping that would include YAP3 is not given a unique name, only classified as a structure metric system. This is an immediate limitation of the classifications Verco and Wise presented since their YAP3 cannot be differentiated from an engine using structure techniques but not attribute counts. Perhaps the only way that this classification is useful is from a chronological viewpoint. Attribute counting systems were the earliest plagiarism detection engines, followed by structure metric systems, which, with the odd exception for comparative purposes, e.g. by Saxon (Saxon 2000), have been replaced by engines using only structure metrics. More recently Jones fails to differentiate between the two groups, stating that his attribute counting system works along the same principles as the YAP3 structure metric system (Jones 2001, Jones 2001). This shows there is confusion amongst even developers of detection engines.

There are possible approaches to detection engines that cannot be classified in the Verco and Wise manner. One example is Saxon's detection engine that finds similarity based on the compressibility of source code (Saxon 2000). There are no structure metrics or attribute counts used, so this cannot be classified under the Verco and Wise scale. It could however be stated to use paired metrics since calculations are made on two submissions at a time.

Culwin, MacLeod and Lancaster provide definitions vaguely in line with those of Verco and Wise (Culwin et al 2001). Their definition of attribute counting systems states that these use only superficial metrics, their definition in line with the one given earlier in the section. Structure metric systems are said to only use tokenisation techniques and the tokens searched for long common substrings. The belief is that these use only structure metrics. Again there are engines that cannot be classified on this scale. The main difference to the Verco and Wise view would be which engines fall under the structure metric categorisation. For instance, an engine using both attribute counts and structure metrics would be classified by Verco and Wise as a structure metric system, whilst it would not fit into the classifications given by Culwin, MacLeod and Lancaster.

One possible way of dealing with this problem is simply to make these classifications consistent and introduce new classes to cover the engines that cannot yet be classified. structure metric systems should be those engines that use structure metrics exclusively and do not use attribute counts. Those engines that use both attribute counts and structure metrics should be referred to as *hybrid metric systems*. A catchall *other systems* is also necessary for anything that cannot be classified by the above. The definition for attribute counting systems, as using only attribute counts, is the only one that does not need changing.

Additionally, to keep the classifications consistent, those engines that use attribute counting metrics, even if tokenisation is required, will be classified as attribute counting systems. In a similar manner, engines that use structure metrics, even in an untokenised form, will be classified as structure metric systems. These changes make the classifications usable also for free text engines, where tokenisation is unusual.

Figures 3.1 to 3.3 show the original inconsistent classifications and the improvements. Figure 3.1 shows the Verco and Wise classifications. Here only one quarter of the possible types of engines have names identified. Figure 3.2 shows the Culwin, MacLeod and Lancaster classifications. Again only one quarter of engine types are classified but note here that the structure metric system is positioned differently. Figure 3.3 shows the inclusive classifications where every possible type of engine has a name identified and there are no empty boxes. A key to the shading used is given in Figure 3.4.

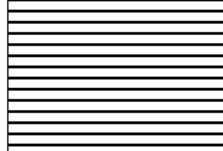
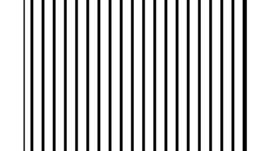
	Attribute Counting Metrics	Structure Metrics	Attribute Counting and Structure Metrics	Other Metrics
Non-Tokenised				
Tokenised				

Figure 3.1 – Verco & Wise classifications

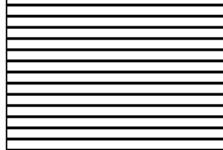
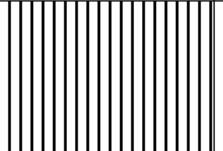
	Attribute Counting Metrics	Structure Metrics	Attribute Counting and Structure Metrics	Other Metrics
Non-Tokenised				
Tokenised				

Figure 3.2 – Culwin, MacLeod & Lancaster classifications

	Attribute Counting Metrics	Structure Metrics	Attribute Counting and Structure Metrics	Other Metrics
Non-Tokenised				
Tokenised				

Figure 3.3 – More inclusive classifications

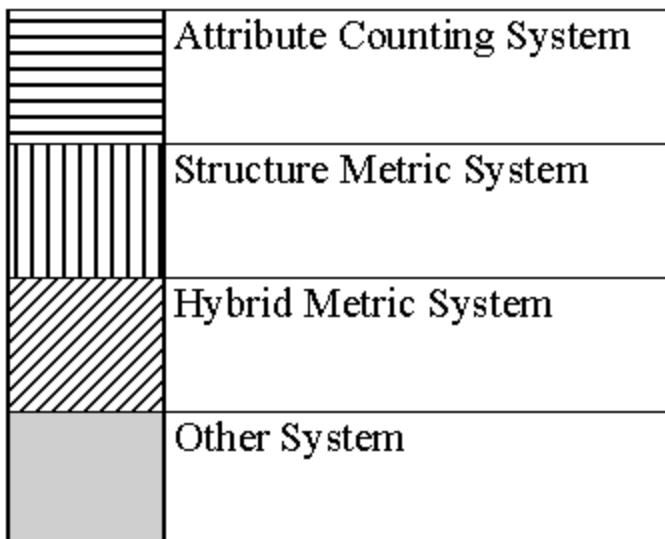


Figure 3.4 - Key to Figures 3.1, 3.2 and 3.3

Figure 3.5 expresses the relationship between the main features of the different types of detection engines, under the improved more inclusive classifications. The plagiarism detection engines are split into four different classifications, with the 'other systems' branch there to include any future or current engines that do not use traditional metrics. The metrics used by the three main classifications are also shown with the number of arrows at the right of a classification showing the number of different types of metrics used and their names. The engines will use one or more of each of these indicated metric types.

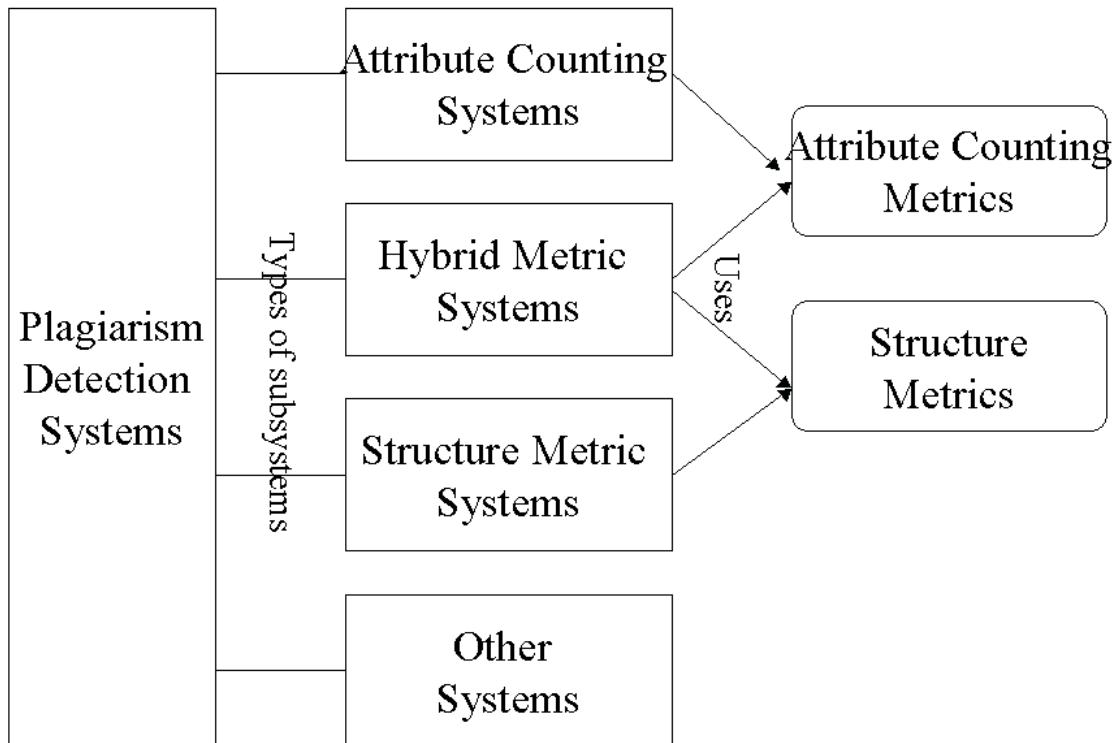


Figure 3.5 - Improved classifications of detection engines

These classifications are still not ideal. Figure 3.3 shows four different types of engines, represented again by the different shadings. These engine classifications are loose and do not provide an easy way to differentiate between, for instance, an engine where two submissions are searched for similar strings as is and one where they are tokenised prior to being searched. The terminology is also geared strongly towards source code engines. For instance although tokenisation can be defined for free text engines it is really a process intended for source code. These improved classifications will be acknowledged but are not considered the best way to classify new engines, since more information is required than can be contained within four classifications.

Due to these shortcomings it makes more sense to classify engines using the metrics introduced in Chapter 2. Primarily this means classifying engines by the dimension of the metrics they use (singular, paired, multi-dimensional or some combination), the complexity of the metrics (superficial, structural or some combination) and whether they use tokenisation techniques. There is no exact correlation possible between the new set of metrics and the traditional classifications. Figure 3.6 shows an approximation, using the same shading as before.

	Singular Metrics	Paired Metrics	Singular and Paired Metrics	Other Metrics
Superficial Metrics	Horizontal lines	Vertical lines	Diagonal lines	Grey
Structural Metrics	Horizontal lines	Vertical lines	Diagonal lines	Grey

Figure 3.6 - Approximation of traditional classifications using new metrics

3.6 - Comparison of source code detection engines

Table 3.2 shows the most recent detection engines identified in the literature and how they fit into the different groups of classifications. The hosting states whether the engine must be deployed on a local machine or can be accessed over the Web. The availability covers whether the system is available automatically for academics, on a limited basis, or only in the owning institution. The table also shows whether the system is believed to be current or since superseded. These terms were defined in Chapter 2.

Name of Engine	Literature	Class of Engine	Hosting	Available	Current
Big Brother	Irving 2000, Irving et al 2002	Paired Metric, Structural, Tokenised	Local	Limited public	Yes
Cogger	Cunningham & Mikoyan 1993	Paired Metric, Structural	Local	Private	No
DetectaCopias	DetectaCopias 2002	Paired Metric	Local	Public	Yes
Jones	Jones 2001, Jones 2001a, Stephens 2000	Singular Metric, Superficial, Tokenised	Local	Private	Yes
JPlag	Prechelt et al 2000, Prechelt et al 2001	Paired Metric, Structural, Tokenised	Web-based	Public	Yes
MOSS	Bowyer & Hall 1998	Paired Metric, Structural, Tokenised	Web-based	Public	Yes
Saxon	Saxon 2000	Other	N/A	Private	No
SHERLOCK	Joy & Luck 1999	Paired Metric, Structural, Tokenisation	Local	Public (in summer 2002)	Yes (in summer 2002)
SIM	SIM 1989	Paired Metric, Tokenised	Local	Public	Yes
TEAMHANDIN	Culwin & Naylor 1995	Singular Metric, Superficial	Local	Private	No
YAP3	Verco & Wise 1996, Verco & Wise 1996a, Wise 1996	Paired Metric, Structural, Tokenised	Local	Public	No

Table 3.2 - Reviewed detection engines

The literature column only details the main literature published on each engine; there are many cross-references. Saxon's project (Saxon 2000) does not name the new tool so it is referred to as Saxon here. It has also been available for research purposes only. Likewise Jones' engine is not named and so is referred here as Jones (Jones 2001, Jones 2001a). The work by Jones appears to be a rework of a paper from one of his students (Stephens 2000) with more detail of the metrics used and additional speculation.

The documentation for DetectaCopias is limited to a single Web page which is in Spanish (DetectaCopias 2002). The translation through Altavista Babel Fish is not very clear but there do not appear to be any technical details about how it operates on the Web page. The engine is included here for two reasons, first that it is a recent engine and so its existence should be noted and second because it is a rare example of a detection tool written in a language other than English.

The technical details of how MOSS operates are not known. Culwin, MacLeod and Lancaster believe it to be tokenisation followed by matching substrings (Culwin et al 2001) which is why it is described as above.

Table 3.3 shows, for each detection engine, the programming languages that the detection engines operate on and whether they require submissions to be parsable, based on the descriptions given in the literature. Some of the detection engines will not process submissions that are not parsable. This means they are not syntactically valid so that, for instance, opening braces have a matching closing brace or appropriate lines end in a semicolon in Java. Instead they are excluding from the corpus being considered, in which case plagiarism may be missed. Where such information is not given but is implied in some manner an informed judgement is made, where no implications are possible the parsable status is listed as unknown.

Name of Engine	Languages	Must be parsable?
Big Brother	Java Ada 95	No
Cogger	C	Unknown
DetectaCopias	Java	Unknown
Jones	COBOL	No
JPlag	Java Scheme C C++	Yes
MOSS	C C++ Java Pascal Ada ML Lisp Scheme	No
Saxon	Java	No
SHERLOCK	Pascal Modula-2 Miranda	No
SIM	C Java Pascal Modula-2 Lisp Miranda	Unknown
TEAMHANDIN	Cobol Ada Pascal C C++	No
YAP3	C Lisp	Unknown

Table 3.3 - Languages processed by detection engines

The download Web site for SIM states that the engine works on both the programming languages listed and free text, although no evidence about exactly how this is done is given (SIM 1989). The Big Brother user manual states that it operates on free text (Irving et al 2002). Wise has alluded to using YAP3 on plain text but nothing on this has been published (Wise 2000, private communication). In theory there should be little stopping the source code detection engines being able to process free text but the algorithms used would hence not be optimal.

Although JPlag is known to reject submissions that cannot be parsed and MOSS is known to accept them the literature is very vague. Irving states that Big Brother does not require submissions to be parsable (Irving 2002, private communication). The other decisions stated have been made based on a consideration of the types of metrics used and whether they should require a full parse.

3.7 - Singular metric engines

For the most part singular metric engines have been superseded by the paired metric engines that are more commonly used today. Verco and Wise have shown that attribute counting systems, which primarily use singular metric techniques are inefficient compared to other metrics available (Verco & Wise 1996, Verco & Wise 1996a).

The earliest detection engines were developed to further the ideas of Halstead and his *software science metrics* (Halstead 1977). Halstead suggested four specific measures that could be derived from a student source code submission in order to assess similarity:

- n1 – the number of unique operators used
- n2 – the number of unique operands used
- N1 – the number of operator occurrences
- N2 – the number of operand occurrences

Ottenstein used these measures in the one of the earliest, if not the earliest automated plagiarism detection engine (Ottenstein 1977). Such counts were especially suited for detection since they required little computational effort. Pairs of programs that share similar metrics for n1, n2, N1 and N2 represent significant similarity that may constitute plagiarism.

Later engines have expanded and changed this list in order to find the best overall metrics for plagiarism detection, which may vary from programming language to programming language. Many of the early engines operated on FORTRAN and Pascal. Robinson and Soffa first contributed to this with their ITPAD engine, measuring such program features as counts of data structure usage and the number of block headers (Robinson & Soffa 1980). A comprehensive review of different metrics that have been subsequently investigated can be found in Parker and Hamblen's work (Parker & Hamblen 1989). These tend to primarily involve program layout and program identifiers, demonstrating a point where structural metrics have taken over from more superficial metrics used previously.

A more recent singular metric engine is a module of Culwin and Naylor's TEAMHANDIN engine (Culwin & Naylor 1995), which consists of three linked modules for source code submission, plagiarism detection and plagiarism reporting. The plagiarism detection module is of interest here. The metrics used count, for each submission, the number of reserved words and the number of identifiers that exist on a pre-declared list, giving a number of superficial metrics. Submissions with close values for both metrics are flagged as potential plagiarism. TEAMHANDIN has one additional benefit over the attribute counting engines described earlier and that is an ability to cluster source code submissions. Hence the user can be presented with

groups of similar submission as opposed to pairs of just two submissions when investigating similarity.

Culwin and Naylor's TEAMHANDIN engine weighted tokens pragmatically according to the frequency of their occurrence with a corpus, with more those more frequently occurring rated higher (Culwin 2001, private communication). It can be argued that a reverse approach, with those tokens that occur only in a few submissions should be weighted higher, because they represent more unusual similarity.

The authors stated that they had used this engine for over five years and it had been effective in finding plagiarism. Additionally they validate it by submitting known plagiarism, which is all flagged by the engine. Although singular metric engines have been largely superseded this demonstrates that they can be perfectly adequate for finding simple cases of plagiarism.

There has been one recent attempt to use attribute counts for plagiarism detection and that is in work by Jones (Jones 2001, Jones 2001a). The papers are rewritten and extended versions of work by Stephens (Stephens 2000). The same ideas are discussed in all of these, in particular the two Jones conference papers are near word for word identical. Jones papers also contain some serious flaws in differentiating between different types of detection systems, details skipped over in the Stephen's paper. Jones states that his approach is the same as the YAP approach, when YAP uses paired metrics and Jones clearly uses singular metrics.

Jones creates three different fingerprints for each source code submission. A similarity score is produced for each pair of submissions under each fingerprint with some normalisation to within a known range. The first fingerprint is a vector of physical aspects of the file, namely the number of characters, words and lines. This is said to be easily fooled. The second fingerprint contains three Halstead metrics, namely the total number of tokens, the number of unique tokens and a combination of the two. This is shown to be more effective, but can be fooled by adding new unused procedures or functions. The third fingerprint contains both the physical and Halstead fingerprints and is said to give fairly random results.

The results support that the Halstead metrics can give reasonable results, but add nothing new to results from many years before. Further acknowledgements in the papers are very sketchy, with no discussion of the dominant structure metric systems and some notable confusion about the different types of systems.

3.8 - Singular and paired metric engines

Engines using a combination of both singular and paired metrics developed from the ideas of Donaldson, Lancaster and Sposato, who produced what is probably the earliest such engine (Donaldson et al 1981). No known engines of this type are available at present.

3.9 - Paired metric engines

Only Joy and Luck's SHERLOCK engine has been identified as a paired metric engine that primarily avoids tokenisation (Joy and Luck 1999). The stated benefit of SHERLOCK is that it is language independent; this means that there is no need to reprogram SHERLOCK to find plagiarism in new programming languages. However some of the metrics listed do seem to contradict this, as removing comments, for instance, requires knowledge of the syntax of the language.

Five metrics are used to compare each pair of submissions. These involve finding the length of the longest common substring in the four combinations of source code considered: with and without white space and comments, as well as in a completely tokenised version of the code. Joy and Luck state that the language independent approach finds much of the plagiarism, but does not always find as much as pure tokenisation engines.

It is difficult to critique the engine since although the metrics used by SHERLOCK are shown to find much of the plagiarism there is little indication in the paper about how it compares with other engines. The comment that the engine is language independent does seem to be impossible considering that some knowledge of syntax is needed. Luck has stated that the intention is to make SHERLOCK available for download as open source software (Luck 2002, private communication).

3.10 - Paired metric engines with tokenisation

The paired metric engines with tokenisation tend to follow a similar pattern, differing only in the exact mechanisms used to extract fragments of source code into tokens. Clough describes some of them as part of his general plagiarism literature review (Clough 2000a). The review comes across a little as a work in progress, since there are a few papers, including those on the tools developed in the course of this thesis, that are not cited.

One of the more advanced engines is Michael Wise's YAP3 (Wise 1996). This is the third version of an engine developed from the likes of Plague (Whale 1990) using structure matching algorithm to detect shuffled areas of code. Wise states that there are two main improvements to the algorithm: sorting functions into the order they were first called and mapping synonyms into a common form. This is combined with a fast comparison algorithm and partial parsing of the source code files, which merely extracts words from a lexicon, giving operational time efficiency benefits. This means that programs that will not compile can still be checked for plagiarism.

Irving's Big Brother engine is an example of an engine developed for use by a single institution, in this case Glasgow, although Irving will make it available on a limited basis for external tutors who require assistance (Irving 2000). Big Brother proved to be particularly effective in finding plagiarism in an outbreak during the 1998 to 1999 session, where nearly 15% of a 400 student cohort were found to have submitted work with notable similarity. Big Brother is a standard tokenisation engine that works on programs with comments and formatting removed. It can find both the longest common substring and longest common subsequence as the longest program of the

pair's length (Irving et al 2002). Of the two Irving states that the latter is the most successful plagiarism differential. Additionally the program can find the combined length of common substrings over a given size, which is said to be more accurate when the copying is fragmented, but operationally much slower. Generally the methodology seems fairly standard and hence should be accurate although the engine does not contain a reporting mechanism as to why two submissions have been judged similar.

The Cogger engine is described as being novel since it does not require exhaustive analysis of all pairs of submissions in a corpus (Cunningham & Mikoyan 1993). Instead the engine computes a function call tree, one type of tokenisation, for each piece of source code and clusters these using a neural network and Case Based Reasoning (CBR) techniques. This could also be considered as a multi-dimensional metric engine due to the methods of comparison. The paper concludes by stating that the results given are variable, since applying CBR to the plagiarism problem successfully needs more work, particularly in generalised cases, which are often missed. Hence this approach can be considered more of a curiosity than one likely to be developed into a real engine, since there doesn't seem to have been any further work done on Cogger since 1993.

The Software and Text Similarity Tester (SIM) is mentioned briefly here since it is available for free use by a Web download (SIM 1989). SIM is reported to be used with more than adequate results at Vrije University, Amsterdam. It is said to work by finding the longest common strings in tokenised student submissions.

Two final engines, MOSS and JPlag, operate over the Web and allow legitimate tutors open access. The engines previously described in this section are available only for download, or sometimes seem to exist only for research purposes, whereas MOSS and JPlag do provide services which are known to be used by computing departments around the UK, Europe and the rest of the world. Hence both MOSS and JPlag will be discussed in more depth.

Perhaps the better known of two to computing academics is Aiken's Measure of Software Similarity (MOSS) (MOSS 1994). No technical details of MOSS are available, only an independent critique describing one approach for using the engine and integrating it into a Computing course with an appropriate policy (Bowyer & Hall 1999). More detail about the engine is given in the local university press, which stated that 10% of students on one unit had been suspended after their work was checked by MOSS (Sanders 1998). The article stated that MOSS was particularly useful for classes on compilers.

Culwin, MacLeod and Lancaster describe MOSS from an operational perspective (Culwin et al 2001). Tutors submit source code using a submission script, usually on Unix platforms. Results of the plagiarism analysis are available on a Web site on the MOSS server for two weeks. A list of pairs ordered by the number of tokens matched is presented to the user, who can then select a pair for further investigation. Two pieces of source code are shown side by side with areas that MOSS believes similar highlighted in the same colour. A tutor can make a decision as to whether this constitutes plagiarism. The main additional feature noted is that the MOSS results are self-adjusting and sections of code common to a majority of submissions are assumed

to be supplied code and hence excluded from the calculations. This is reported to be counter-intuitive, since two identical submissions may be reported as less than 100% similar.

The alternative Web-based tool, JPlag, offers similar functionality to MOSS (JPlag 2001). It is available for public use over the Web or by command line where additional parameterisation is possible. Prechelt, Malpohl and Phlippsen describe JPlag from a technical perspective (Prechelt et al 2000, Prechelt et al 2001). The engine computes tokenisation metrics for code using an optimised version of the YAP3 algorithms. After processing, results are made available on an overview Web page, containing a histogram of submissions at ten levels of similarity. In practical usage only those submissions at the highest levels of similarity will be checked further. Users can view pairs on a side by side comparison, a technique developed by JPlag and used in the same way, with permission, by MOSS.

Prechelt, Malpohl and Phlippsen evaluated JPlag by submitting four corpora of Java source code to it (Prechelt et al 2000, Prechelt et al 2001). Each corpus contained a maximum of 59 submissions, some corpora had plagiarism manually added. The authors define precision (P) and recall (R), where precision is a measure of false hits and recall is a measure of missed pairs. The performance of the engine is rated by calculating $P+3R$ and trying to minimise this value, rating missed pairs as far more crucial than false hits. The papers show that parameters exist for each corpus that rank all the plagiarism highly, although these are not the same for every corpus. The parameters include such things as the minimum tokenised string match length and the choice of tokens to be used. This does suggest that there is not one best solution for finding plagiarism and that knowledge of the content of submissions is necessary to choose appropriate parameters.

The papers on JPlag list a number of plagiarism methods and note how effective the engine is for finding plagiarisms that represent each of these (Prechelt et al 2000, Prechelt et al 2001). More complex source code plagiarism techniques, such as exception handling, modifying data structures and adding redundant methods are shown to be likely to fool the engine. It is open to debate how far a student needs to modify a program before they have demonstrated sufficient skill to meet the learning outcomes for a programming assignment, so this may not be a cause for concern.

3.11 - Other detection engines

Only one other evaluated technique for detection is known. Saxon's project on source code plagiarism detection seems to have produced a rare original detection method (Saxon 2000). This is, in essence, a paired metric technique, although the metric does not classify as superficial or structural. The method stems from finding the entropy of a piece of source code, a numerical measure of the compressibility of the submission. This is done using standard gzip functions from the Java libraries with the size of the compressed file measured. Defining $\text{com}(A)$ as the compressibility of a file A and $X++Y$ as the concatenation of files X and Y (or Y appended after X), a similarity score is taken as $((\text{com}(A++A) + \text{com}(B++B)) / (\text{com}(A++B)*2))$. The idea is that if there is redundant information in B that already occurs in A the pair of files will compress greatly. A large benefit of the technique is that it is language independent, since no properties of programming languages are needed, although the compression

calculations are computationally intensive. Saxon finds the technique to be highly effective in comparisons against standard structure metric systems.

Limited initial results from Campbell have been less successful (Campbell 2002, private communication from South Bank University, London). Campbell compares the ranking list generated by using the metric on submitted student Java source code pairs with the list ordering returned by the MOSS detection service. Campbell finds that few documents occur in the upper portion of both lists, the area that would likely be checked by tutors for potential plagiarism. His best results use tokenised then compressed versions of the files. The results however are stated to be only initial results and depend on the reliable of the MOSS ordering, so Saxon's technique may still be suitable with modifications.

Saxon's technique would also seem to be ideally suited for applying to free-text student submissions. Only one such trial is known and this has only been done in a very limited fashion (Medori et al 2002). Watson describes how a similar technique can be used to identify articles written in foreign languages (Watson 2001). A file in an unknown language can be compressed appended to texts written in known language and it should be possible to identify the language based on the compressed file sizes. The article suggests using the WinZip software commonly used on PCs.

3.12 - Other technical detection approaches

Although not strictly a completely automated engine some other approaches have been suggested in the literature. Harris advocates a different approach (Harris 1994). He states that using automated detection introduces too many problems, suggesting instead that plagiarism should be found by examining source code history files and quizzing students on their knowledge of their programs. This approach would seem to be too labour intensive for already busy tutors.

Jones suggests using a version of the Glatt plagiarism detection process (Jones 2001, Jones 2001a). The Glatt process for free text student submissions involves removing words and asking students to replace them. The argument is that if a student wrote the submission they should be able to place a majority of these words. Jones suggests, but does not evaluate, removing random lines from the code, or introducing known errors into the program that students would then have to correct. This is intended to ensure that students wrote the source code submission themselves and has some programming ability.

A second approach suggested by Jones, but again not investigated, involves examination of 'second-order products' (Jones 2001, Jones 2001a). These are products that can be generated from source code submissions. Jones first suggests generating a log of error messages at compilation time. A number of tokens could be determined based on the error messages and singular metrics generated for each log. Those metrics with high similarity scores based on attribute counts generated here could then be investigated for plagiarism. The success of this approach rests on Jones' assumption that plagiarising student submissions would generate the same error messages, which is suspect. Additionally a sensible student plagiarist would re-use work that compiled and ran successfully, with no error messages.

Jones second idea involves generating logs when running the programs with known inputs. These logs could then be tokenised and closeness calculations carried out as before. This approach seems like a lot of work as a fresh set of tokenisation strategies needs generating for every new assignment specification. The results generated are likely to be in English which presents a research problem as to what tokenisation strategies would be appropriate. Further it does not take much programming skill to change the lines of code that print values to the screen. One solution to this might be to consider only the values of variables that change during execution. This would give a better indication of whether programs were executing in the same manner. But this might also mean that many independently developed programs giving the same, correct, output would be ranked here as significantly similar. Jones needs to be more convincing that his second-order products are feasible indicators of source code plagiarism.

Using visualisation methods is a completely alternative detection technique to calculating metrics to determine how similar submissions are. The visualisation techniques involve generating a graphical representation of one or more submissions and allow a tutor to make a judgement on where similarity lies. Since such techniques will be shown to be largely applicable to both source code and free text they will be presented together in Chapter 5. The visualisation techniques from the source code literature include a clustering technique included in SHERLOCK (Joy & Luck 1999). A pair of linked techniques that plot submissions on a grid are DotPlot (Helfman 1994) and a related tool DUPLOC (Rieger & Ducasse 1998, Ducasse, Rieger & Demeyer 1999). Two graphics for presenting an entire corpus are also known, called patterngrams (Ribler 1997, Ribler & Abrams 2000). Techniques such as the comparative Web pages approach used by MOSS and JPlag can also be considered visual.

3.13 - Literature comparisons of existing engines

Most comparisons of existing engines are linked with trying to show that the new engine produced by the authors exceeds the capabilities of all the other tested engines and as such the results may be biased. No entirely independent reviews of the techniques have been identified, which seems to be a shortcoming. Saxon notes that it is difficult to compare detection engines in the same way since there are no standard source code corpora available (Saxon 2000). Producing standard corpora would not be an easy undertaking since there are many different programming languages in use and also some engines will not process code that does not compile.

A study by Verco and Wise provides a good example of this. They compared what they described as attribute counting engines and structure metric engines (Verco & Wise 1996, Verco & Wise 1996a). These definitions have been redefined for consistency and completeness in Section 3.5, the definitions now more vaguely in line with singular and paired metric engines respectively. Both papers cover much the same ground, with the latter being an extended version of the former examining more engines. Apart from their own YAP3 the engines tested were recreated based on their descriptions in the literature, since the original engines were not available for comparison. The attribute counting systems were prepared by tuning them on three corpora of student submissions. Tests were done on five corpora of Pascal

submissions containing some known plagiarism and with code that could not be parsed removed, since some engines cannot process this.

The authors differentiated between the type of plagiarism that would likely be carried out by inexperienced programmers and that that would likely be carried out by experienced programmers. The results for novice programmers had Plague finding most plagiarism, followed by YAP3 and Sim. For experienced programmers YAP3 came first, followed by Plague and Sim. No one engine found all the plagiarism. In both cases all the structure metric systems tested outranked all of the attribute counting systems tested, supporting the authors' expected view. The authors argue that although Plague finds slightly more plagiarism than YAP3 the latter is better since source code processed by YAP3 does not require full parsing, which would seem to be a valid viewpoint, since eliminating unparsable code is a major shortcoming of some engines.

Granville provided a comparison of a single attribute counting metric with a single structure metric but his results were suspect (Granville 2002). The attribute counting metric chosen was McCabe's Cyclic Complexity, intended to identify the complexity of source code, but one that has not often been used in attribute counting systems, particularly since such systems usually rely on a fingerprint of many metrics for accuracy. As such the singular untokenised metric was found to give false hits nearly 50% of the time, an undesirable result. The structure metric tested, here a tokenised paired metric, was said to perform slightly better. This would be the expected result, but the comparison has to be considered invalid since the singular metric tested negated most known research.

Saxon provides the nearest thing to an independent test as part of his project (Saxon 2000), which also involves examining a new compression based comparison technique. The comparisons were done on corpora produced by first year Cambridge students. Saxon did not examine any existing engines *per se*, instead recreating techniques found in the literature.

Saxon tested four paired metrics. The first compared the raw submissions and found the longest contiguous run of identical lines within them. The second repeated this with comments and spacing of lines ignored. The third found the longest run of identical comments. The fourth found the longest run of identical tokenised lines, using basic tokenisation. The exact measurements allowed lines to be arranged in certain situations, for instance where methods had been re-arranged within a class file, if this would produce larger values. Saxon also computed simple attribute counting metrics, including the number of classes, functions and if statements. The results for each distribution were normalised to equal means and standard deviations. Saxon estimated the effectiveness of each technique by measuring what he called the distribution's 'peakiness'.

Saxon found that tokenising and finding long substrings was the least reliable technique, although he speculated that this could due to the simplistic tokenisation he used. This is in contrast to the results of Verco and Wise and the generally accepted view that sees tokenisation as part of the best methods for finding undue similarity.

Saxon found that the best metric overall involved simply standardising formatting, removing comments and finding the longest contiguous subsequence of identical lines without tokenisation. The singular metrics rated near the bottom of the list and so Saxon dismissed them as being useless for finding anything other than blatant plagiarism cases. JPlag, used as a control case, was found to be a worst performer than most of the metrics, missing much known plagiarised material simply by being unable to parse it. Saxon found that his new compression technique bettered the results of all of the metrics for identifying similarity. It would be interesting to see how it compares against some of more developed and fine tuned detection engines.

3.14 - Comparison of Web-based detection services

Apart from institutions that have their own localised engine, the best solutions for institutions wanting to activate a pro-active anti-plagiarism policy would seem to be to use one of the two available Web-based detection engines, MOSS and JPlag. Culwin, MacLeod and Lancaster provided a review of the different services from a technical perspective and tested them by submitting an identical corpus of source code submissions to each (Culwin et al 2001). The tests were restricted to source code submissions that could be parsed by both engines.

They are unable to rate one detection engine above the other, finding beneficial features to each. They do recommend that institutions seriously consider using a service such as MOSS or JPlag.

The authors find that there is little correlation between the ordered similarity results supplied by JPlag and those supplied by MOSS. Only the results of the MOSS percentage ranked metric compared with JPlag's only metric, also representing percentage, showed any significant correlation. There are cases of pairs that rank highly under one engine but do not rank anywhere in the limited list of top matches returned by the other. This means that if one engine is used exclusively there may be missed pairs in the rankings obtained if students can figure out how to plagiarise and avoid the automated detection.

The main advantages of MOSS found over JPlag are that MOSS will process files that JPlag cannot parse, finding plagiarism that JPlag might otherwise miss. MOSS uses three metrics to compute its results. This gives more information to a user than JPlag's one.

The main advantages of JPlag found over MOSS are that JPlag will accept submissions over a regular Web browser (Netscape only) or command line, whereas MOSS requires command line submission only. A quirk of the MOSS implementation means that identical submissions may sometimes be reported at less than 100% similarity; this is not a problem with JPlag.

The results are noted to be unduly capricious, since the choice of which pairs are going to be identified depends entirely on which detection engine is being used. It is desirable for detection engines to be non-capricious since otherwise identified students can be argued to be a matter of chance. Culwin, Lancaster and MacLeod state that it is beyond their resources to investigate why this is and suggest this may suit a master's level project (Culwin et al 2001). A solution might be to have a third engine that takes results from both JPlag and MOSS and moderates between the two, so that no pair marked highly by both engines escapes investigation.

3.15 - Attitudes to source code plagiarism detection

The only known reference to attitudes of tutors to source code plagiarism is the result of a survey by Culwin, MacLeod and Lancaster (Culwin et al 2001). There are no known surveys about attitudes of students to source code plagiarism, which perhaps represents something of a shortcoming.

Culwin, MacLeod and Lancaster's survey, which had a return rate of 50% out of 110 UK HE Computing schools questioned, found that source code plagiarism was frequent on initial programming courses, involving anything up to 20% of students on the unit (Culwin et al 2001). The frequency of plagiarism occurrences would go down on later units. This may be because the plagiarists were no longer studying programming, or because they realised that the work submitted had to be their own. A later question gives a mean of 5% of all students committing plagiarism, although this ranged from approximately 0% to 13% across all the institutions. In the same report Alex Aiken, the developer of MOSS, states that approximately 10% of submissions processed by MOSS are plagiarised, so source code plagiarism in the UK is probably much more extensive than tutors would estimate.

Only 36 of the 55 institutions said that they tried to detect plagiarism and only 19 of these used any form of detection engine. The rest relied on eyes alone. 11 of the remaining 17 institutions said that they had never considered using such an engine, implying that the remaining 6 had considered and rejected the idea of any form of plagiarism detection. A later question states that only four would completely rule out the idea of using a detection engine, which implies that the other two would reconsider the situation. Most tutors stated that plagiarism detection was required to validate assessments, however the results above suggest that few of these tutors were carrying out such validation themselves. Four fifths of institutions had a policy on source code plagiarism, but very few of these required all work to be automatically checked. It would be interesting to discover how many of these policies were identical to the regular plagiarism policy for free-text.

Tutors believed that students plagiarised source code submissions since the students were too disorganised or because programming was too difficult for them to learn. Most respondents stated that plagiarism was a 'routine headache'. Some tutors said that they were forced to ignore plagiarism, since the efforts required to take students through formal procedures were too big a drain on their time.

The tutors were asked which programming languages a detection engine would need to support. Java, C++ and C were the most common and these are all commonly supported. Visual Basic and Prolog both scored highly but little support for these is known. This suggests that there is a gap in the market for plagiarism detection engines that needs to be filled.

3.16 - Other source code papers

There are a few papers that have some relevance to source code plagiarism detection but do not directly fit within any of the more general categories. Each is discussed in brief here.

Sallis, Aakjeer and MacDonnal describe metrics that can be derived from submitted student source code and used to draw up a profile of the programmer (Sallis et al 1996). In a later paper the same authors describe how the metrics can be used to track the location of source code (Gray et al 1998). This is intended as a way to identify the author of rogue source code, such as viruses. In both papers they note that, if a series of submissions are collected from a certain student, a new piece of code with different stylistic metrics may be plagiarised. The choice of which metrics to use for this is not given.

Carter describes the differences between collaboration and plagiarism when students are writing object oriented source code (Carter 1999). Based on a survey of fifteen students in a UK higher education institute she notes that students can be categorised, each category showing differences in their abilities to work independently. She finds that students who merely share ideas can help each other to become more engrossed in the material. Tutors are said to underestimate how far students work together on work that is intended to be completed individually. Loners were seen to usually complete work without incident, but would deliberately plagiarise when stuck. Collaborators were seen to share work across tutorial groups and produce template solutions for one another to work from. The author argues that plagiarism guidelines need to be less rigid to take actual student cheating behaviour into account. A larger study could also influence how source code assignment specifications were set, so as to minimise the potential for plagiarism.

Hamblen, Parker and Wachtel describe their Unix laboratory set up, where all student submission are automatically checked for plagiarism as a matter of course (Hamblen et al 1998). The engine used is standard, stripping comments and blank space from files then finding the percentage character similarity within pairs, using diff, grep and wc. Similarity levels greater than 45% are examined further. The algorithm is noted to take two to three hours to check a corpus of fifty submissions.

Williams notes how collaborative programming, also known as extreme programming, which is used in industry, is seen as a problem likely to lead to plagiarism within academia (Williams 1999). She investigated getting students to work in pairs, with the proviso that they didn't split the work into two chunks but instead worked on it together, finding that 84% of students preferred it, being less distracted whilst working. The only disadvantage was noted that students had to physically get together to work, something that could be difficult at a commuter university. The technique was believed to reduce the chances for plagiarism for experienced pair programmers.

3.17 - Conclusions

This chapter has reviewed the main pieces of literature about source code plagiarism and its associated detection. Much of it is not directly relevant to free text detection, although some of the techniques have variations that can be used on free text. Examples of these are included in Chapter 4. Additionally the graphical methods briefly introduced will, on the whole, be shown to be applicable to both source code and free text detection. They will be compared as part of Chapter 5, with other review material related to plagiarism presented in Chapter 6.

The main original contribution in this chapter has been the proposal of new types of categorisations for plagiarism detection engines. It is suggested that these classifications, based mainly on the types of metrics used by the engines, improve on the standard ones used in the literature, produced when detection was still only in its infancy.

Chapter 4: Free Text Plagiarism Literature Review

Overview:

- Web-based detection services are compared.
- Current technical methods for detecting plagiarism are compared.
- Techniques are discussed that fit into the four-stage detection process.

Plagiarism detection in free text student submissions is a relatively recent area of research. During the production of this thesis there has been a sudden growth in the number of ideas proposed and tried. This chapter reviews the existing detection literature, focussing on solutions to the problem of Web plagiarism. There has also been recent work done on intra-corporeal plagiarism detection, finding which students might have plagiarised from which other students, a process analogous to although not directly related to the way that source code plagiarism is detected.

4.1 - Motivation

Chapter 3 reviewed the literature on plagiarism detection in source code submissions. This chapter reviews the literature for detection in free text. This includes detection of intra-corporeal plagiarism, a problem similar to that which would have been tackled by those writing source code detection engines, but where the existing techniques are found not to be directly relevant. Intra-corporeal free text plagiarism detection will be seen to be one of the main themes of the remainder of the thesis.

This chapter also includes a discussion about Web plagiarism detection. A growing number of Internet hosted services exist that will allow student submissions to be sent to them and will search to see if any of the content matches already published resources (pages) on the Web. This is usually done by searching for relevant strings through Internet search engines, although the details vary from detection engine to detection engine and are often a company secret. There have been a number of reviews of these services and these are discussed.

Other technical methods of finding plagiarism that have been suggested are also discussed. These range from a system for students to check each other's work, to trialled metrics for finding similarity.

Chapters 5 and 6 will extend on this chapter with a study of other literature relevant to plagiarism detection. Chapter 5 looks at methods by which plagiarism can be visualised. Techniques for this can be seen to be relevant to both free text and source code detection. There are also many articles that describe why Web plagiarism is a problem, how assignment specifications can be modified to restrict plagiarism opportunities or those that describe why students cheat. These related areas, not directly relevant to technical detection or visualisation, will be considered in Chapter 6.

4.2 - Free text plagiarism detection literature

Much of the most relevant research on finding plagiarism in free text has only come about during the production of this thesis. As such the research and techniques that can be reported are limited and the methods used differ widely, with their being little evidence available about how well they work and which are best. The literature on free text detection is substantially limited when compared to that on source code detection.

It is worth noting that there are many sites and articles on the Internet that cover plagiarism. Many institutions around the world and the departments within them have one. The content on the sites is fairly standard. Instructions for students about what plagiarism is and how to cite properly may be provided. Details of electronic tools, mainly the Web-based plagiarism detection services are given. Links to other sites and articles of interest are common. They might also contain advice for tutors on how to assess students whilst reducing the chance of cheating. These sites have been deliberately excluded for the most part since they are both of no technical interest and cover no original ground. In some cases it could be argued that the shared similarity on them is rather suspect.

Table 4.1 shows, in alphabetical order, a list of the identified sources for technical solutions to free text plagiarism detection, along with a short description for each of them. A few of these sources include material relevant for both source code and free text detection and so are included in this chapter, although they have already been discussed in Chapter 3.

Authors	Date	Content
Allen	2002	Trials techniques with which paraphrasing across two documents can be identified.
Altin	2001	Student project providing an interface to investigate pairs of student submissions for similarity through a visual tool.
Baker	1993	Describes techniques for searching for similar strings in text files.
Baron	2001	Opinion piece on the Intelligent Essay Assessor for automated marking which has a small plagiarism detection module.
Bloomfield	2002	Web page with links to detection software. Notes a concern about programmers writing software for both paper mills and plagiarism detection.
Braumielle & Gainer	2001	Paper describing the differences in levels of plagiarism found when students were and were not warned that their submissions would be electronically checked.
Bull et al	2001	JISC commissioned review on free text detection, comprising a technical review, user trial and a survey of tutors.
Chester	2001a	JISC study. Describes experiences of UK institutions trialing the iParadigms detection service.
Christie	1999	Describes an automated engine for marking essays that could be adapted for plagiarism detection.

Clough	2000a	A literature review of tools and techniques for finding plagiarism in student submissions. Contains some comments on the free text literature.
Clough	2000b	Discusses a framework within which reuse of texts can be assessed and quantified.
Combrink-Kuiters et al	1999	Describes an engine used to compare Law student submission for similarity.
CopyFind	2002	Web page describing and providing a download for CopyFind intra-corporeal plagiarism detection software.
Culwin & Lancaster	2000	Compares Web-based plagiarism detection services.
Decoo	2002a	Book describing academic misconduct split into a number of stages. Contains a description of Cerberus, a specially developed intra-corporeal plagiarism detection engine designed to show similarity in two submissions.
Dee	2001	Some comments about electronic support for the four stages of the Culwin and Lancaster detection process. Uses the plagiarism taxonomy for a case study of the Leeds University physical system.
Denhart	1999	Online article comparing Web-based plagiarism detection services. There have been more in-depth later comparisons.
Denning	1995	Describes how SCAM copy detection engine used to verify academic plagiarism.
Faber	1999	Evaluates plagiarism.org Web-based plagiarism detection services with free and purchased papers.
Garcia-Molina et al	1996	Describes dSCAM distributed extension to SCAM that can be used for copy protection across multiple sites.
Hearst	2000	Describes automated essay marking systems, some of which have plagiarism detection modules.
Halfman	1994	Presents a visualisation method for showing similarity in documents.
Hersee	2001	Student project describing how plagiarism may be identified by plotting two different linguistical properties of documents and seeing where the plots differ greatly.
Hoad & Zobel	2002	Investigates how documents can be identified that have been derived from another.
Irving et al	2002	User manual for Big Brother source code detection engine which states that the engine can operate on free text.
Kontaxis	2001	Student project from SBU plotting changing linguistical properties throughout documents. Such changes might represent plagiarism.
Lathrop & Foss	2000	Book covering aspects of plagiarism prevention and detection, much like an extended version of the articles available.
Lyon et al	2001	Presents a method for comparing two documents by comparing their fingerprints.
Medori et al	2002	Evaluates possible methods for finding plagiarism in laboratory practical reports, including compression techniques.

Monostori et al	2000	Describes how plagiarism may be detected using spare processing power over the Internet, although no implementation is given.
Oxford	2002	Compares some plagiarism detection tools and services on four sample essays.
Rahman	2002	Student project from SBU improving the tool described in Kontaxis 2000.
Ribler & Abrams	2000	Presents two visualisations that show commonality in documents. Primarily used with source code but may be applicable to free text.
Sanderson	1997	Describes an investigation to quantify the amount of similarity in a document collection.
Satterwhite & Goerin	2000	Compares Web-based plagiarism detection services with a number of different sources.
Saxon	2000	Compares plagiarism detection methods in source code. Included here because of a novel detection method based on compressibility that may translate to free text.
ScienceDaily	2002	Short study comparing differences in plagiarising when students had and had not been warned that their work would be electronically checked.
Shivakumar & Garcia-Molina	1995	Describes the SCAM engine that registers documents for copy protection purposes.
Shivakumar & Garcia-Molina	1998	Describes a clustering method that allows a database of Web documents to be produced, that may subsequently be checked for plagiarism.
Si et al	1997	Describes the CHECK engine that registers LATEX formatted documents so that subsequent reuse of them can be identified.
Voumard	1994	Allows personalised assignment specifications to be created and marked in numeric subjects to make plagiarism unlikely.

Table 4.1 - Free text plagiarism detection literature identified

4.3 - Comparison of Web-based plagiarism detection services

Since students are known to be copying material from the World Wide Web it seems intuitive that it should be possible to find the source of the work that they have copied. One method that tutors might use is to type a sequence of words from a submission into a search engine and see if it returns any hits. By choosing a suitable sequence of words, perhaps containing an uncommon phrase or misspellings, a tutor can reduce the number of false hits. In this context false hits are pages containing the phrase but in some other context. The remaining pages should identify the source of any plagiarism.

There are issues with this approach. It is not easy for a tutor to choose which words are appropriate to run through a search engine. Much time may be wasted searching for inappropriate phrases. Further the whole process of finding fragmented plagiarism is laborious. There are known problems with using search engines in that the results

returned differ over time, that is the results on one day may differ to the results another day, perhaps due to the server being busy, or due to different advertisers influencing the results. Hence the search engine approach for finding plagiarism cannot be entirely recommended.

A number of detection services have appeared that aim to automate this process. Series of words from submitted documents are run through search engines and the results pieced back together to give a report that can be read by a tutor. Most such services are Web-based, that is they are accessed remotely through the Internet. There is a downloadable tool called EVE that performs a similar job from a local computer. Processing is done locally, although an Internet connection is needed to access the search engines and so this is also included here.

It is difficult to give a complete and definitive list of detection services, since these have changed over the last few years, since the late 1990s when the earliest services first came into operation. Additionally the names of some services have changed, for instance the self-reported market leader now goes under the company name iParadigms, Inc., not the Web-site name plagiarism.org and to add confusion in some reports is referred to by the name of the student site, turnitin.com. To avoid confusion the current name for each service will be used.

Few technical details of the services of how the services search for similarity are known. Some state that they have an additional localised database that is searched and/or access to other reference works that students may plagiarise from, such as thesauruses. The services also vary in cost, ranging from limited free services, to those that are charged per submission, or through site license. The operational costs are also not always known.

Table 4.2 shows the detection services, past and present, that have been identified in the literature. The table, sorted into alphabetical order, gives the current or most recent name of the services, along with any previous or alternative names. The availability of the services lists how they can be accessed and if they are believed to be current. The cost lists the price of using the services where it is known. For some services additional notes are also given.

Name of service (previous names)	Availability	Cost	Notes
Copycatch.com	Web-based No longer exists	Free	Accepts both Word and text format. Not to be confused with CopyCatch text concordance tool
EduTie	Web-based	Unknown (negotiable)	Commercial version of Plagiserve. Not included in any reviews.
EVE 2 (EVE)	Download Operated locally	\$30 registration fee	New version said to be in development
FindSame	Web-based No longer exists	Commercial	Few details known - existed briefly in trial form.
HowOriginal.com	Web-based Believed discontinued	Free	Maximum 1k file size per document.
iParadigms, Inc. (plagiarism.org turnitin.com)	Web-based	\$1 per paper or \$4000 per annum site licence.	Self-professed market leader
Paperbin (Integriguard)	Web-based Believed discontinued	\$20 per paper, per year	Commercial version of HowOriginal.com
Plagiserve	Web-based	Free	Service recently made available, similar to others.

Table 4.2 - Detection services for Web-plagiarism

The companies concerned have not published technical details on their services, apart from occasional and sometimes dubious claims on their Web sites. There have, however, been independent attempts to verify the different services. Table 4.3 describes the tests that have been done, in approximately chronological order. For each article the services tested are listed (using the most recent names to avoid confusion), along with the corpus used to test the engines and the main findings of the report.

Authors	Date	Engines compared	Corpus	Main findings
Denhart	1999	iParadigms EVE 2	A single text file - formed by collating material from different Web sources.	EVE recommended over iParadigms.
Faber	1999	iParadigms, Inc	A number of free papers from essay banks. 2 ready prepared papers purchased from commercial	iParadigms found all the free papers, but none of the purchased papers.

			paper mills. 2 papers purchased from commercial papers mills and ghost written to order.	
Culwin & Lancaster	2000	iParadigms EVE 2 HowOriginal.com Copycatch.com	A single text file - formed by collating material from different Web sources and self written material.	Copycatch.com recommended - most usable, found most similarity and free. Issues with other engines.
Satterwhite & Goerin	2000	EVE 2 HowOriginal.com Paperbin iParadigms	A number of free papers from essay banks. 4 ready prepared papers purchased from commercial papers mills. A number of self compiled essays.	EVE recommended for thoroughness, iParadigms for presentation, but overall authors advise against using detection services.
Bull et al	2001	iParadigms FindSame EVE2	Eleven papers each tested 116 times over three months.	iParadigms recommended, although writers favour CopyCatch linguistic text concordance tool. FindSame only tested in demo form.
Chester	2001a	iParadigms	Five institutions, up to five subjects in each invited to submit work.	Main strength reported as large database of sources. Weak technically, with speed of results and login process.
Braumoeller & Gaines	2001	EVE2	Mocked up essay. Corpus of 180 Political Science essays	EVE2 found to be credible.
Oxford	2002	iParadigms EVE2 Plagiserve	Four papers by same author.	iParadigms most successful, followed by Plagiserve then EVE.
Medori et al	2002	iParadigms	Corpus of first year biomedical student submissions with some plagiarism added.	iParadigms found to be partly successful but unsuitable for mostly intra-corporeal plagiarism.

Table 4.3 - Literature comparisons of Web-plagiarism detection services

Denhart, in probably the earliest test, recommends EVE over iParadigms (Denhart 1999). The main reasons given are that EVE found more of the plagiarised material and is much cheaper to run than the iParadigms solution. However EVE's reporting method, a text file of Web sites, is criticised, since it requires more manual checking compared with iParadigms' solution of a Web page of hyperlinks. Denhart warns that iParadigms is very quick to say that a document contains plagiarism, when it should only say similarity, since some of the document submitted was legitimately cited. He warns that tutors must carefully check the results from either engine. This is one reason why such a stage is explicitly included in the four-stage detection process. Although only one document was used, hence giving a small sample size, this was produced from a number of different sources and hence giving the appearance of a larger sample. Braumoeller & Gaines found that EVE2 detected most known plagiarism (Braumoeller & Gaines 2001, ScienceDaily 2002).

Faber found that iParadigms successfully identified papers obtained free from essay banks, but not where they had been changed (Faber 1999). The service failed to find papers written to order, a not unsurprising result since these could not be in a database. iParadigms also failed to find ready prepared papers that had been purchased, suggesting that the internal database is limited. Medori, Atwell, Gent and Souter also found iParadigms relatively unsuitable for a more specialised corpus (Medori et al 2002). This was because the engine only flagged highly suspicious cases, missing known plagiarism in what was mainly short papers with students copying from each other. In this case the checks for Web plagiarism were said to be unnecessary. Also since the submissions were short some of the false hits were noted to give alarmingly high percentages of similarity where similar unrelated wording could be found on the Web. Although the writers do not differentiate between similarity and plagiarism this would seem to be another suitable warning for tutors to carefully check any reported similarity from these engines.

Culwin and Lancaster found that there were issues with all the engines they tested (Culwin & Lancaster 2000). All the engines found some of the plagiarised material they submitted, in the style of a self-produced student submission, but none found all of it. The results from iParadigms and Copycatch.com both came presented in a similar hyperlinked format, whilst Paperbin returned an e-mail of hyperlinks that needed to be cut and pasted into a browser and EVE returned a text file of Web site urls. Copycatch.com came out favoured as the only engine to identify most of the similarity, the only engine to accept files in a format other than plain text and the only free solution. Unfortunately the service is no longer available.

Satterwhite and Goerin found that EVE 2 was both very thorough and very slow, since it runs on the local machine (Satterwhite & Goerin 2000). The number of papers found out of the eight submitted was not given. The services HowOriginal.com, PaperBin and iParadigms were all noted to be unsatisfactory, finding few of the sources. iParadigms was said to be slow but selected as the best of the bunch due to its presentation of the results. PaperBin was said to be unsuitable as it only checked part of the documents and HowOriginal.com only allowed very short texts to be checked. No service found an article copied from Encyclopedia Britannia, suggesting that the site had not been crawled. Overall Satterwhite and Goerin stated that their own partial checks using phrases in Internet search engines were more successful and recommended the cheaper solution of careful assignment specification preparation.

The tests by Bull, Collins, Coughlin and Sharp appear to be the most thorough, although there are some notable exceptions with detection services that are not examined (Bull et al 2001). They recommend iParadigms out of the services tested. This should mean that the service is soon available across the UK courtesy of the Joint Information Systems Committee. One point that the writers note is that the results returned by all the services were inconsistent depending on when papers were submitted. So they may well be missing plagiarism. JISC's own small scale pilot of iParadigms found it to be poorly designed from a usability viewpoint, with the results slow to be returned and technical support taking up to eight days (Chester 2001a). The service is being offered nationwide by JISC regardless.

Other interesting findings related to the paper mill services, which were generally slated. A paper written to order for Faber at \$354 was immediately made available on the company Web site at half the price (Faber 1999). This would make it possible for other students on the same course could purchase the paper. Another paper failed to arrive in time for the deadline. The quality of the papers was generally said to be low. This would suggest that students using paper mills are doing so at their own risk. Satterwhite and Goerin purchased ready written papers from pay essay sites at costs of up to \$90 and found them to be of a low standard (Satterwhite & Goerin 2000). The authors recommend taking and editing a free paper to be a better strategy to plagiarise, rather than risking using something supplied by the pay sites.

The general impression from the reports is that the services exist, that they find some, but not all plagiarism, but the costs are substantial and all the services require manual checks. The results in the literature also do not reflect local experience, where the Plagiserve service comes recommended, which appears to be very much like a free version of iParadigms, albeit one dependent on tutor's submitting work that they deem suspicious. This indicates an immediate shortcoming in the published research since this is only briefly mentioned and again indicates how quickly the plagiarism literature becomes out of date. EduTie, which is a commercial solution based on Plagiserve, believed to be more along the lines of iParadigms complete solution, is not tested anywhere.

There are also many places where the engines available are not recommended over a simple search engine human led approach. For instance Dee recommends just using Google within the four-stage detection process (Dee 2001). The then existing engine FindSame was also suggested.

Bloomfield does note concerns with both Plagiserve and EduTie (Bloomfield 2002), namely that both share servers with sites responsible for selling ready written papers to students and that the same programmers provided the software for both sites. This could mean that the programmers have provided a 'backdoor' in Plagiserve that would mean papers purchased from them would not be detected. However Bloomfield cites personal correspondance with Lytvyn and Shevchenko, the programmers, that states that this is not the case and instead blames the limited funds for running servers in the Ukraine. Bloomfield suggests that other academic institutions might instead offer to partner Plagiserve to eliminate this problem. However the whole situation serves as a warning about there are companies that make money out of student cheating.

One other problem is that the sites all seem to have serious usability issues. Although the hyperlinked interfaces used for navigating the submissions are said to be appropriate there is little to compare them with and better navigation tools may be possible. There are also usability issues from a submission point of view. The services generally require students to upload their own work in plain text format. This means that the work submitted cannot be the same as that handed in at the student's own institution. At the very least formatting changes are necessary and the student may electronically submit a different piece of work. Further there are no facilities for batch upload, so submissions are made on an individual basis. Generally the existing services require a full usability investigation, which has not been done to date.

4.4 - Linguistical detection methods for plotting similarity

Linguistics is a whole area of science concerned with identifying different language features in both speech and writing. As such this encompasses many areas but can include plagiarism detection. All the plagiarism detection methods discussed here could be argued to use linguistic methods but this section will discuss only those that are looking for some characteristics of writing to identify possible plagiarism.

Two different methods have been identified, based on measuring properties in an individual document and finding those areas where the style changes. Those areas could then be assessed as being suspicious. Both investigations exist in the form of student projects. Hersee uses a cumulative sum method comparing two metrics on a graph, with plagiarism judged to be in those areas where do not align (Hersee 2001). Kontaxis and Rahman write about the FreeStyler tool in two separate projects, building on to preparation by Culwin and Lancaster (Kontaxis 2001, Rahman 2002). Here a single metric is plotted on a graph and those areas with significant jumps are the ones that are suspect and may represent plagiarism.

Both reports uses similar approaches, but with slight differences. The central idea is that writing style can be plotted as it changes throughout a document. Styles at different points in the document can be examined by looking at the appropriate point on the associated graph.

The investigation by Hersee works on a sentence level. A graph is plotted showing deviations from the average sentence length (Hersee 2001). A second graph is plotted on the same axis showing what Hersee refers to as a habit again plotted as deviations from the habit average. These include such things as the number of two and three letter words in the sentence or the number of words starting with a vowel in the sentence. These have been identified from existing stylometrics literature, the areas of linguistics concerned with identifying authorship. There is a small problem in that Hersee treats these habits as being mutually exclusive, summing them, when a word such as 'our' could clearly appear in both groupings listed above. It is not the exact deviation from the average that is plotted in both cases, the value at the previous sentence is also taken into account. The graphs are scaled to try to fit them together as closely as possible. Those areas where they differ are then said to be written with a different style that Hersee equates to plagiarism.

The second approach works at the word level (Kontaxis 2001, Rahman 2002). A document is divided into overlapping fragments of a chosen size and the value of a metric calculated at those points. Examples given include the average number of words per sentence and the calculated readability age, such as that given by the Fleisch or Fog metric. The metrics are plotted on a graph. Then those areas that are stylistically different, such as through copying from the Web should show different characteristics, deviating from what should otherwise be a straight line. The size of the fragment will reflect the graph produced. A small fragment would likely show more deviation in the graph. A larger fragment would presumably give a more consistently flat graph but shorter areas of stylistic change would be hidden away.

Although Kontaxis notes the possible use of the software for plagiarism detection no evaluation is done (Kontaxis 2001). Instead the tool is discussed more to be of use as a general tool for writers to identify stylistic differences of their own. Rahman develops the tool further but again stops short of evaluating it for the detection purpose (Rahman 2002). Observations suggest that the tool could show areas of interest, but is rather assuming that students write with a consistent style, a hypothesis that may well prove to be ungrounded.

Hersee does provide an analysis of his tool plotting a number of possible habits on documents where plagiarism has been added at known locations (Hersee 2001). A numerical measure of how the two graphs differ is produced for each, although it is not clear that all the graphs are scaled in the same way, which would influence the results. The measure is said to indicate how plagiarised the documents are, although here Hersee is assuming that all the stylistic inconsistencies represent plagiarism, when there may be legitimate reasons. Hersee's results are inconclusive, stating that for every plagiarised sentence the method delivers another two sentences are falsely identified. It is speculated that this may mean that the underlying hypothesis, that students write with a consistent style, is false. This would seem to be a suitable question for future investigation. It may also mean that plagiarising short parts of a document do not sufficiently influence the habits examined.

The general conclusion seems to be that linguistical methods for finding plagiarism in a single document, particularly one as short as a single student submission, are not working. Even if they were there is a further problem that has not been stated. Areas would be identified that may be plagiarised, but there would be no evidence as to where they'd been plagiarised from or what the source of the plagiarism was. Without this evidence, or suitable research that shows that these results can be accepted without a source, there is little evidence to take forward to future plagiarism penalisation proceedings.

4.5 - Other linguistical tools for detection

The literature lists a number of other tools that really stand on their own.

The first tool, as seen at plagiarism.com, could perhaps better be considered as an approach. Plagiarism.com argue that you can replace every fifth word in a submission with a standard sized blank and ask the alleged student writer to fill in the blanks. If the student wrote the submission they should be able to do this accurately. Plagiarism.com sell software to aid with this test known as the Glatt test. Satterwhite and Goerin looked at this word replacement process as part of their analysis of Web services for plagiarism detection. The process is said to be intrusive and results unreliable. Presumably this is because it is a non-passive method requiring cooperation from the student (Satterwhite & Goerin 2000). There is little evidence as to whether the hypothesis, that authors can successfully identify and complete their own writing, is true. This may be further complicated when submissions have been written in a last minute rush or late at night. This area would seem to be suitable for future investigation.

The other pieces of software are versions of standard textual concordance tools, CopyCatch and WORDCheck. These are intended for intra-corporeal plagiarism detection. Although there are no publications by those behind the tools there is one comparative critique available, as part of Bull, Collins, Coughlin and Sharp's technical review of plagiarism detection software (Bull et al 2001).

CopyCatch lists pairs of submissions and calculates the level of similarity within them. Any over a user-selected threshold can be examined further through a listing of similar phrases in both documents. Bull, Collins, Coughlin and Sharp give this a maximum five star rating (Bull et al 2001). They praise the instant results and the ability to process Word documents, but note that the software is expensive and the stability of the vendor suspect. They don't comment on the fact that a linguistical tool would seem to not be completely suited to detection, depending on tutors understanding the complexities of such a tool and what linguistical checks are appropriate. A suitable tool would better hide much of this complexity from the user. Medori, Atwell, Gent and Souter also rate CopyCatch highly, submitting a corpus with known plagiarism to a more recent version of the software than tested elsewhere and noting that all the similarity was found (Medori et al 2002). A final test found CopyCatch to be efficient at processing texts but noted that the percentages of similarity returned were not always accurate and that essays on a similar theme but from different viewpoints could also return high percentages (Oxford 2002). This would mean that close checks of any results are necessary.

Clough cites personal correspondance with the producers of CopyCatch (Clough 2000a). This states that 40% of shared words in submissions is normal and 70% is suspicious being perhaps the most interesting statistic to come out of the software.

A similar tool WORDCheck has been mentioned in the literature, but any plagiarism detection features here seem purely co-incidental as this is more intended for linguistical investigation. Satterwhite and Goerin states that WORDCheck is unsuitable for Web plagiarism detection as the source has to be found independently (Satterwhite and Goerin 2000). Bull, Collins, Coughlin and Sharp give this a low rating of one star out of five stating that the performance of the software was poor (Bull et al 2001).

Overall there is little evidence in favour of using the linguistical tools for plagiarism detection although CopyCatch has been received favourably in UK HE, suggesting that some academics find the intra-corporeal plagiarism checking useful. The plagiarism.com approach would seem to have possible uses if a tutor believes that a student has not written a submission or part of a submission and a source cannot be identified. More evidence is needed that this is a suitable approach to use before this can be recommended further.

4.6 - Copy detection tools

Copy detection tools are those that keep a record of documents, either in full or part and then check other documents against this database to see if they have been plagiarised. Although they are intended for commercial use in preserving company intellectual property some observers have suggested that they may be suitable for extra-corporeal plagiarism detection, provided a suitable database can be established. Clough includes them as part of his general plagiarism detection literature review, identifying three such tools, COPS, SCAM and CHECK (Clough 2000a).

Si, Leong and Lau discuss CHECK at more length in the literature (Si et al 1997). The engine comprises document registration, document comparison and document parsing, although it works only on documents formatted in LATEX, which has huge usability issues. LATEX is used so that specially formatted keywords can be included in the submissions and unnecessary comparisons avoided. Other keywords are extracted with the aid of a lexicon and user supplied parameters. The document is then represented as a set of keywords with associated weightings. New documents entering the engine are reduced to keywords and compared with previous submissions to decide if they share subject matter. Any that do are further compared at sentence granularity. The engine sounds like it might have plagiarism detection applications but at present it is not far beyond the idea stage.

The Stamford Copy Analysis Mechanism (SCAM), developed by Shivakumar and Garcia-Molina, is also discussed in more detail (Shivakumar & Garcia-Molina 1995). It creates an indexed database of the documents submitted to it to enable partial copies to be identified. Denning describes an incident of academic plagiarism, where one academic stole technical papers from a university FTP site and submitted them (Denning 1995). When the plagiarism was discovered SCAM was used to successfully verify it.

A Shivakumar and Garcia-Molina extension of SCAM aims to solve a more advanced version of the copy detection problem (Garcia-Molina et al 1996, Shivakumar & Garcia-Molina 1998). It allows the millions of documents existing on the Web to be clustered. This has applications for both plagiarism detection and reducing Web

search time if similar documents are factored out of searches. The tool works in two stages, chunking documents in order to find ones that are similar, then comparing these further using a probabilistic-counting based approach. However the computing power required to do this is huge; the 24 million documents compared took up 150Gb of disk space. The most interesting part of the research is the figure that 36% of Web documents are replicated on the Web at least once.

A related process for Web plagiarism detection has been described by Monostori, Zaskavsky and Schmidt although it has not yet been implemented (Monostori et al 2000). The first stage of the distributed process involves identifying sources that submissions may have been plagiarised from using a locally held database using a standard algorithm. The second and final stage involves a detailed comparison between all the submissions and the identified sources using a tool called Cluster and spare Internet processing power. The process is said to be being developed into an engine called MatchDetectReveal.

Sanderson uses an approach by breaking up documents into chunks, although the exact metric used is not specified (Sanderson 1997). This is said to be successful in finding documents that have been accidentally duplicated with minor (e.g. spelling) changes in a bibliographical database. Hoad and Gobel specifically extend this information retrieval approach for finding changed versions of documents (Hoad & Gobel 2002). They also trial copy detection techniques. This was tested on a corpus of around 80,000 documents from Linux documentation, known to contain much duplication, as well as on a smaller corpus with known introduced similarity. They find that a highly mathematical approach based on counts of features of fragmented documents is successful. The technique briefly trialled on student submissions and said to find some, but not all, intra-corporeal plagiarism.

Generally, although such engines appear to be applicable for plagiarism detection there are issues. Creating a database for extra-corporeal detection might be appropriate, especially where documents are not already available through Web search engines, but it seems like a duplication of effort. They are also only a partial solution, since informing a tutor both how and why submissions are considered similar is a necessary requirement.

4.7 - Technical solutions for identifying similarity

Chapter 3 described engines for identifying similarity in program source code. Although the engines were really designed for a tokenisation approach three engines were noted to be usable on free text, namely SIM, Big Brother and YAP3. The only operational details given are for Big Brother, which, for source code, is a fairly standard paired metric system working on tokenised submissions (Irving et al 2002). Free text submissions are tokenised by hashing all their words to give a numeric token. Then the engine finds the longest common substring or subsequence. In this case the hashing seems of little use apart from as a hack to make the submissions appear tokenised and possibly also to reduce their size. This approach could be expected to find some similarity but not to cope with much more than superficial changes.

In all three cases there is little evidence about how effective the engines are and this functionality seems more of a side effect of the engines main use. Since the engines have been discussed in detail in Chapter 3 they will not be described further here but their existence has been noted.

Some further solutions for finding and identifying intra-corporeal plagiarism are known, all of which have been introduced during the production of this thesis. As with source code detection engines it is also believed that some institutions may have their own private localised solutions. These can be considered solutions to the analysis stage of the four-stage process.

Combrink-Kuiters, De Mulder, Elffers and Van Nortwijk developed a fraud detection module for the Conceptual Document Analysis System (CODAS) an automated grading system for Law submissions (Combrink-Kuiters et al 1999). Two submissions are broken down into a list of individual words and a similarity score computed based on a weighted comparison of those lists. Bloomfield's CopyFind engine finds long continuous substrings in pairs from a corpus of student submissions (CopyFind 2002). Where these are identified pages of HTML are generated to mark the similarity for further investigation for potential intra-corporeal plagiarism. Lyon, Malcolm and Dickerson also have a localised solution which is unnamed (Lyon et al 2001).

The CODAS fraud detection module breaks submissions down into a list of every word used, ignoring ordering and frequency (Combrink-Kuiters et al 1999). Every word that exists at least once in the corpus is given an associated weight, which is the complement of the probability that if a random submission were chosen it would contain that word.

Four measures are then produced for every possible pair of submissions:

- P1 – the sum of all weights associated with words common to both submissions.
- P2 – the sum of all weights associated with words absent from both submissions.
- P3 – the sum of all weights associated with words existing in one or both submissions.
- P4 – the sum of all weights associated with words absent from one or both submissions.

A similarity score for each pair of submissions is then defined as:

$$S1 = (P1 + P2) / (P3+P4)$$

When the corpus is large or the submissions are small P2 and P4 are said to introduce bias into the similarity score and so a new version is defined as:

$$S2 = P1 / P4$$

Testing the results is done by choosing a pair from the corpus. The initial similarity rank of the pair is measured. Material is then slowly added from one of the pair to the other, representing increasing plagiarism, and the rank recomputed. This gives a continued increase in rank under S1 and S2.

A smaller test has a duplicate submission added to a corpus and words replaced with synonyms throughout. The source/copy pair ranked at position 1 under S2 and position 17 under S1. This suggests that the results are generally good, although S1 may not be a reliable metric.

Few technical details are known about CopyFind, other than it searches all possible pairs of submissions for long common substrings (CopyFind 2002). A few parameters can be manually set such as a minimum number of words that must be matched for a phrase to be marked and a minimum number of total words in phrases that must be common before HTML pages are generated. The developer Bloomfield recommends a figure one third of the document word count to avoid false hits. When a pair is judged similar CopyFind generates two HTML pages where those parts judged similar are underlined in each. This to aid a tutor with investigation. Although the approach is similar to that of the Web-based plagiarism detection services the underlined fragments are not links so some manual searching is still required.

Medori, Atwell, Gent and Souter test CopyFind as a tool for detection plagiarism in biomedical reports (Medori 2002). They praise the immediate results but find the text-based interface difficult to use. They also note that the engine misses some of the known similarity. Bloomfield himself alludes to this (CopyFind 2002) since the engine only searches for exact matches, so it could be fooled by plagiarism through refragmentation or thesaurising techniques.

Lyon, Malcolm and Dickerson's tool is said to collect work in textual or HTML format and split it into three word sequences, known as feature metrics (Lyon et al 2001). There is little evidence as to why this metric was chosen although it is shown to find similarity in three corpora, TV news reports, the Federalist papers and actual student submissions. Submissions can be compared by working out the proportion of sequences in common in pairs of feature metrics. This seems to be a fairly standard solution although the lack of analysis here on student plagiarism is limiting.

One further system that has been identified is the Intelligent Essay Assessor (Baron 2001, Hearst 2000). This is a system for automatically marking essays based on a number of metrics, which is said to have a plagiarism detection module. No details on how the engine works are given, or on whether it is successful, but presumably it is an attribute counting system. Many people would suggest that completely automated essay marking is unobtainable and so any detection engine based around it is probably unsuitable.

Most of these detection engines have shortcomings in that they are not a complete solution to detecting intra-corporeal plagiarism. Although similarity can be identified there is no evidence given as to why two submissions have been judged to be similar. Hence it cannot be considered a complete solution. CopyFind is the most complete, but the algorithm used and the interface both need some improvement.

4.8 - Technical solutions for investigating similarity

The CopyFind engine described could be useful during the verification and investigation stages of the four-stage process, although it depends on the software judging that two submissions are similar enough to generate HTML pages for. One other engine is known that is intended to aid with investigating potential similarity. Cerberus, was developed as part of the Crisis on Campus book and consists of a technical description with partial source code (Decoo 2002a). Cerberus works by finding common text clusters and may be useful during the verification and investigation stages of the process, although this is not made clear.

Cerberus uses string-matching algorithms on RTF and text documents. Users select the comparison mode. Window's built-in 'Find' functionality is used when exact matches are selected. This is stated to be fast but can miss similarity.

Engine developer Colpaert however recommends his own algorithm used for approximate matching. This algorithm is said to be much slower than the exact matching algorithm but more accurate. Approximate matching is calculated as a function of the number of two character sequences two fragments have in common. Colpaert demonstrates, by making small changes to sentences to represent scanning or typing errors that this is suitable, however the results are not overly convincing as the technique is not demonstrated against any major changes. Colpaert also states that longer chains of characters may be useful since even unrelated paragraphs can give a commonality measure of over 50% but this is argued to be merely a scaling measure and not to influence the results. Colpaert states that 60% of similar pairs of characters should be considered similar, which does not seem greatly above the 50% quoted for unrelated paragraphs.

Users also select the type of delimitation required. Submissions can be searched at the sentence level in which case long series of sentences that match exactly or approximately, depending on the comparison mode chosen, are highlighted. Similarly at word and character level sequences of matching of approximately matching words and characters are shown. There are additional options available, such as a fixed search length that can be set, or a minimum threshold for approximate matching.

Results are presented on screen in two windows with sections of texts judged similar highlighted in red. There is no evidence given about how users can tell which red section from the first document refers to which red section of the second document, or if only one section can be highlighted at a time, which would be a shortcoming. There is also little verification about how successful the approach is, or whether the engine is used in practice.

One problem with the book presentation is that Colpaert does not recognise that Cerberus is only a partial solution for plagiarism detection. Cerberus depends on a tutor being suspicious enough of two submissions to run them through the engine. They then also have to select appropriate values for the parameters to get any useful results out and the presentation still looks limited, with little evidence about how successful the algorithms used are. It is not established how a tutor would decide which pairs to investigate in the first place.

Perhaps a point of interest are that this tool, along with CODAS presented in the previous section, could be possible solutions to different stages of the four-stage process, although none of the authors make this explicit. Hence CODAS could be used to find pairs that have been judged similar and Cerberus then used to show why the pairs are similar. Although this is a drawn out process it is an improvement over other options. However although the CODAS approach looks suitable to find similarity in many cases the evidence on Cerberus' use is limited and so that tool may not be appropriate. Also CopyFind could be judged to be a more complete solution to intra-corporeal plagiarism, although in this case the identification algorithm used is suspect. This does again reinforce the point that appropriate tool support during the verification and investigation stages of the process would be particularly useful to reduce the time spent by tutors investigating possible plagiarism.

4.9 - Other technical solutions for detecting free-text plagiarism

A number of other more unusual methods have been suggested for finding plagiarism in free text student submissions.

Medori, Atwell, Gent and Souter carried out some very specialised tests, based on finding possible plagiarism in laboratory practical reports (Medori et al 2002). This problem is said to differ from the standard plagiarism detection problem due to increased constraints in the way that submissions have to be presented. These include a standard format of 'introduction, method, results, discussion' that the submissions are expected to abide by, vocabulary that is commonplace in scientific work but generally unusual and the common appearance of questions from the assignment specification in the submission. Hence current engines are argued to be of only limited suitability, although the argument is suspect since few are tested and many ignored.

The most interesting results from the paper are related to the potential use of text compression techniques (Medori et al 2002). This does have shortcomings since Saxon's project where the idea was first tested is not acknowledged (Saxon 2001). A number of different variations are tried and the results presented roughly, most inappropriate due to assumptions that all submissions will be of an equal length. The most interesting test is the one equivalent to Saxon's source code work, where two plagiarised submissions, when concatenated together, should compress to around the same size as either of the submissions individually. The results find that this technique works successfully where the amount of similarity is large, but is of little use where the similarity is only moderate. Hence the technique is not recommended. The corpus used to test the compression technique is limited and the submissions unusual so, contrary to the report, it would seem there is still more work to be done here. In particular the author's do not suggest anywhere in the paper that filtering out the questions from the submissions would be a suitable solution to the partial false hits. Also comparing just equivalent sections of the reports, such as the 'introduction' sections alone would also look to be a promising refinement.

The same group also test some possible metrics where they state they have an original idea that two similar submissions should share a lot of the same sequences of words or characters (Medori et al 2002). Again this approach shows a lack of awareness of current detection solutions since this is a common approach, the other similar approaches again going uncited. The difference here is that once these sequences are generated the tests only keep the 10, 20 or 30 most common sequences, which are then used to order similar pairs. They only test pairs of consecutive characters and pairs of consecutive words. The pairs of characters metric tested is immediately shown to be unsuitable as there is little left to differentiate between submissions. The words metric approach is shown to only give usable results when the top 50 word pairs are considered and then to only detect plagiarism with minimal changes, suggesting that the approach is unsuitable. However the authors recommend investigating this further using Markov models.

Possible solutions that might make this approach more acceptable are again similar to the compression metric attempted. Removing supplied text, such as material in the questions would be one solution. It would also be expected that certain sequences of characters, such as those in common words like 'the' or subject specific words would immediately make using only the most common sequence unusable. A possible counter to this would involve filtering out sequences that appear in a majority of submissions, so that they do not influence the results. Another solution might be instead to consider rare sequences that appear in two or more submissions, as these are more likely to be plagiarism indicators. There are no indications in the paper that these approaches have been considered, again indicating flaws.

A completely alternative approach is suggested by Davies (Davies 2000). This is a human-led process using his online assessment engine. The engine is partly used to administer online tests but also includes an essay report system. Here essays are allocated for peer marking and students are expected to mark areas of other essays that include Web sources and investigate areas of suspicion. Any of the eight students marking each paper who miss plagiarism that their peers find are penalised. Each submission is given an overall mark that is the mean of the eight marks that the students decide on. However the plagiarism detection part of this is somewhat dubious as to its effectiveness since it depends on the integrity of the students. Franklyn-Stokes and Newstead found that 65% of UK students would be overgenerous in peer marking suggesting that this is not a suitable assessment method to use (Franklyn-Stokes and Newstead 1995).

Two other possible solutions are limited to individual subject areas. Voumard describes how unique engineering problems can be generated and marked using an online system (Voumard 1994). Since each is numeric nature they can be automatically checked and feedback given. However the approach is limited to numeric disciplines.

Christie describes the Schema, Extract, Analyse and Research engine (SEAR) (Christie 1999). This is an automated essay marking engine under development that marks both style and content. Christie suggests that the engine may have plagiarism detection implications. Although how this might work isn't stressed this would presumably through treating the style and content marks as attribute counts and hence looking for pairs where both of these are closely related. It is an interesting idea, since these are possible metrics like any other, but this might be more dependent on the success of the main engine, stated to be no worse than two human markers.

Allen trials an approach for finding paraphrasing in student submissions (Allen 2002). This is an extension of the METER project for finding where material supplied by the Press Association has been reused (Clough 2000b). Clough uses linguistical techniques to identify nineteen different types of paraphrases that are common in newspaper reports. Allen extends this to work on the sentence level of essays. The main difference is that for intra-corporeal plagiarism there are multiple possible sources, but for newspaper reuse there is only one source considered. Broadly Allen's technique, used in the REPEAT tool, aims to find sentences containing shared words in the same order. These can then be checked using linguistical algorithms and flagged as potential paraphrases. This is a reduced version of the problem of comparing whole essays. Clearly this could be a partial solution to plagiarism detection, but looks like it would struggle with sentences where the order of terms have been re-arranged. The initial results suggest that this would be more suitable as a partial solution to the verification and investigation stages of the four-stage process than the analysis stage, although this is not obvious from Allen's work.

Generally, the techniques tested and the currently available engines evaluated demonstrate that there is no best way, or even a method that can be declared to be a good way, for detecting plagiarism in free text. Much is speculation and even slightly developed engines are only recently becoming available, notably during the production of this thesis. There also seems to be a lack of knowledge of other work being done in the area in many cases. Certainly some metrics might be appropriate and some very simple approaches, such as the idea based simply on word counts, used by CODAS, can be suitable. The techniques tried also demonstrate that there is not one catch-all solution for plagiarism detection. Different subjects might require different detection techniques, in particular essays could be different to short answer questions and both numeric and diagrammatic notations require consideration.

Chapter 5 will consider some of the visualisation methods for source code detection (Helfman 1994, Ribler & Abrams 2000) and whether they are applicable to free text. Rare techniques developed specifically for intra-corporeal plagiarism in free text will also be considered (Altin 2001).

Overall it seems that there are still two main and most generic research questions left, finding a suitable metric to identify similar pairs and a way to compare those pairs when they have been found. This would be ideally suited to a linked engine, fitting in with second, third and fourth stages of the detection process. This really ties in as a companion study to the research question developed in Chapter 2.

4.10 - Conclusions

This chapter has reviewed the main literature about technical methods for detecting free text plagiarism. Chapters 5 and 6 will go further discussing the literature in a number of other related areas.

The chapter appears to be fragmented. That is it has been very difficult to compare and contrast different detection methods with a semblance of order. Many approaches listed are still in the planning stages and the fact it has been necessary to report ideas demonstrates how necessary further research on plagiarism detection is.

The main benefit of the chapter is that it motivates the discussion to come. There have been seen to be problems with all the existing techniques for finding plagiarism, many of which are recent and have not been properly assessed. In particular any human related issues to finding plagiarism, in particular minimising tutor time spent on the problem have been severely missing.

Chapter 7 will describe how such time spent can be reduced based on a ranking of similarity in pairs of submissions. Chapter 8 shows how investigation of similarity can be improved based on a new graphical representation of plagiarism called a similarity visualisation. Chapter 9 will build on this to show how the machine time can be reduced. The reduction to overall time gives an efficient detection process. Showing that similarity is unlikely to be missed gives an effective detection process. Developing an efficient and effective detection process, within the confines of the four-stage process, is the main aim of this thesis.

Chapter 5: Plagiarism Visualisation Literature

Overview:

- The visualisation literature for text similarity is discussed.
- Visual methods for single submissions, pairs of submissions and whole corpora are compared.
- Methods for source code and free text are contrasted.

Chapters 3 and 4 have identified literature on automated detection for both source code and free text plagiarism detection. Some tools contain methods for visualising where submissions are similar. This is a useful aid in the four-stage plagiarism detection process since visual methods can be used to reduce the time spent by tutors on plagiarism related issues.

This chapter describes the different visualisations available, categorising them based on the number of submissions they operate on. It also shows that methods intended for source code are usually applicable to free text and vice versa.

5.1 - Motivation

The literature presented so far has been focused on areas of detecting plagiarism, using technical methods where possible. The established area of source code plagiarism detection was presented in Chapter 3. Although this area is interesting little research into this problem is still ongoing. Chapter 4 presented the new and similar area of research into detection in free text. This is mainly focused around using the Web search engines to find plagiarism but some progress has recently been made on finding metrics for identifying similarity. Little empirical work has been done on finding such metrics.

This chapter builds on the literature that has been presented so far, showing how visual methods can aid in the detection process. The methods and tools come from the literature on both free text and source code plagiarism but are here presented as one whole since most are applicable to either. Further chapters will argue that visual methods are useful when verifying and investigating similarity.

Visual methods are split into three main categories. The first is visualisations that operate on a single submission at a time, such as through finding linguistical properties that are out of place. The second looks at visualisations that compare two submissions. These usually show where or how two submissions are similar. The third set of visualisations operates on an entire corpus. One reason for this is to show how a single submissions compares with all the other submissions in a single image.

5.2 - Visualisation literature identified

One way in which a user can interact with data is through a mere presentation of numbers. In the case of plagiarism detection this might be a list of ranked pairs, or a list of submissions that are believed to be a cluster. Few people would dispute that this is not an ideal way of looking at the data, since it is not always easy to process large amounts of numerical data. Instead *visualisation* techniques can be used to make the data easier to interpret.

The primary work on information visualisation is probably Spence's book (Spence 2001). It is testament to the wide-ranging views on what is involved that Spence does not give his own definition, instead citing definitions that say that "visualisation is an activity in which a human being engages in" and adding that "it goes on in the mind". No further definition how this involves a computer is given. Instead it is perhaps sufficient to say, within the context of this chapter, that the visualisations described involves presenting data in an appropriate form, primarily to allow a human to process the ideas it represents quicker than could be done based solely on numeric or textual measures. That is to say that the visualisations reduce the human processing time in some ways, perhaps by presenting data more efficiently, or perhaps by presenting it over a smaller space. However this is not intended as anything more than an illustrative description and is not intended to be a watertight description.

Most of the techniques that have been identified for visualising similarity, and hence possibly finding plagiarism, exist in the program source code literature. Many of these visual methods look like they would be applicable for free text submissions too. This is in contrast to the general methods used for source code detection which were shown in Chapter 3 to be inappropriate for free text detection. Table 5.1 shows the literature identified in both the free text and source code detection literature where visualisation techniques have been used.

Authors	Date	Content
Altin	2001	Describes PRAISE tool which allows pairs of submissions to be selected for investigation using a visual interface.
Boywer & Hall	1998	Presents details of the MOSS online service for source code plagiarism detection.
Clough	2000a	Literature review which discusses some plagiarism detection visualisation methods.
Ducasse, Rieger & Demeyer	1999	Describes Duploc tool, a clickable version of DotPlot.
Helfman	1994	Describes DotPlot 2D visualisation technique for similarity.
Hersee	2001	Two measures of a student submission are plotted to possibly indicate plagiarism.
Joy & Luck	1999	Visualises source code plagiarism using a clustering method, part of the SHERLOCK tool.
Kontaxis	2001	Describes a tool for visualising stylistic properties of student submissions.
Prechelt, Malpohl & Phillippsen	2000	Attempts to validate the JPlag Web-based detection service for intra-corporeal plagiarism in source code.
Prechelt, Malpohl & Phillippsen	2001	Journal submission detailing attempts to validate JPlag.
Rahman	2002	Student project improving the tool described in Kontaxis 2000.
Ribler	1997	PhD thesis describing visualisation methods including a plagiarism detection case study.
Ribler & Abrams	2000	Shows two new visualisations that shows how a single source code submission is similar to an entire corpus.
Rieger & Ducasse	1998	Describes the Duploc similarity visualisation tool.

Table 5.1 - Visualisation literature for plagiarism detection identified

The literature describes a number of different tools that use visualisation techniques. Table 5.2 lists the different tools and their main references. These can be cross-referenced with the sources listed in Table 5.1. If the literature has been previously described a backward reference is given. The table shows the types of visualisations available, whether the visualisation is for a whole corpus, a pair of submissions, or a single submission. One advantage of visualising a whole corpus is that it might allow clusters of similar submissions, that is sets of more than two submissions sharing similar material, to be identified.

Table 5.2 does not include the Web services for free text plagiarism detection, such as iParadigms. They were discussed in detail in Chapter 4. Also Hersee's detection engine, which was not named in the original text, has been referred to as Hersee here for ease of reference.

Name of Engine	Literature	Operates on	Described in	Visualisations available on
DotPlot	Helfman 1994	Source code	This chapter	Pairs Whole corpus
Duploc	Rieger & Ducasse 1998, Ducasse, Rieger & Demeyer 1999	Source code	This chapter	Pairs
FreeStyler	Kontaxis 2001, Rahman 2002	Free text	Chapter 4	Singles
Hersee	Hersee 2001	Free text	Chapter 4	Singles
JPlag	Prechelt et al 2000, Prechelt et al 2001	Source code	Chapter 3	Pairs
MOSS	Bowyer & Hall 1998	Source code	Chapter 3	Pairs
PRAISE	Altin 2001	Free text	This chapter	Whole corpus
Same	Ribler 1997, Ribler & Abrams 2000	Source code	This chapter	Whole corpus
SHERLOCK	Joy & Luck 1999	Source code	Chapter 3	Pairs Whole corpus

Table 5.2 - Detection engines discussed in chapter

The three different sets of submissions on which the visualisations are available on will be discussed in Sections 5.3 to 5.5. Section 5.3 will describe the identified techniques for visualising a whole corpus. Section 5.4 will describe ways of visualising two submissions at a time. Section 5.5 will show ways of visualising within a single submission.

5.3 - Visualisations for a whole corpus

A number of different techniques have been identified that allow similarity in a whole corpus to be identified. One use of this might be during the analysis stage of the four-stage process so that those suspicious pairs can be identified. Another use might be to identify clusters of similar submissions.

Table 5.3 shows the techniques that have been identified. Each is given a representative name. In some cases no obvious name for the technique has been given in the literature. Here a representative name has been chosen for consistency of later use. A brief description of each technique is given, along with examples of tools where the technique has been used for source code detection and free text detection, where applicable. These examples can be cross-referenced with the examples given in Table 5.2. For each technique some brief initial opinions about its usefulness are also given.

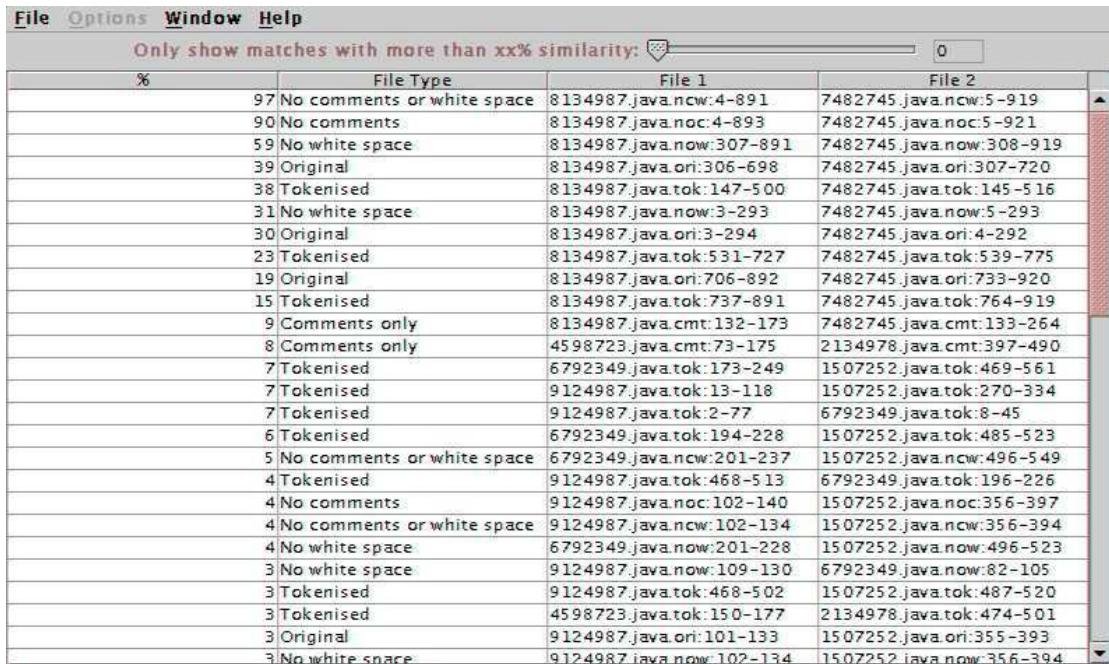
The first technique is not strictly visual but is included as a comparative measure. A standard list of ordered pairs is the default from many engines and compares with the believed efficient tutor processing method, where the tutor starts at the top of the list and works down it, checking pairs that are believed to be of interest.

Name of technique	Description of technique	Examples of source code uses	Examples of free text uses	Comments
Ordered list of pairs	Table of pairs listed in order from most to least similarity. A tutor can investigate pairs from the top to bottom of the list. Table might contain additional information such as the percentage of similarity, or details why the pair was chosen.	SHERLOCK (first view option)	Most intra-corporeal detection engines.	The most common listing method. There is an obvious and effective processing method for tutors. However large amounts of numeric information are presented. Also technique is not useful for showing clusters, where groups of students may have collaborated.
SHERLOCK view	Each submission represented as a dot. Those similar submissions are joined by lines. More submissions joined by shorter lines.	SHERLOCK (second view option)	Shows clusters.	SHERLOCK's view generated through a neural network. Shows clusters. Minimum similarity level shown is selectable.
PRAISE view	Submissions plotted around the arc of a circle. Submissions judged similar joined by lines. Those most similar are joined by thicker lines. Submissions ranked highest appear near the start of the circle circumference.	PRAISE	View generated from ordered list of pairs. Shows clusters.	Minimum similarity level is selectable. Maximum number of submissions to show is selectable.

DotPlot	A corpus of student submissions are concatenated together to give a single long file. Words in it are plotted against one another in a 2D grid, a dot indicating similarity, a blank otherwise. Intense areas with lots of dots represent similarity.	DotPlot	Computationally expensive. Technique requires trained teachers.
Categorical Patterngram	A plot of text from a chosen base file and the number of files that text occurs in.	Same	Text that occurs in a small number of files greater than one is argued most likely plagiarised.
Composite Categorical Patterngram	A plot of text from a chosen base file showing which other files that text occurs in.	Same	Along with categorical view allows potential sources to be identified.

Table 5.3 - Techniques for visualising similarity in an entire corpus

SHERLOCK can be used as an appropriate demonstration of the first two techniques (Joy & Luck 1999). Figure 5.1 shows the SHERLOCK list view, where pairs of submissions are presented sorted in order of decreasing similarity¹. Every pair of submission can be compared in different ways and only the one with the highest similarity percentage is shown here. This is a fairly common technique throughout source code and free text detection, although there are problems here. SHERLOCK is using different metrics for different pairs of submissions and hence the system is not comparing like with like. This does beg the question exactly what percentage similarity means.



The screenshot shows a software interface titled 'SHERLOCK' with a menu bar including File, Options, Window, and Help. A search bar at the top says 'Only show matches with more than xx% similarity:' followed by a numeric input field set to 0. Below this is a table with four columns: %, File Type, File 1, and File 2. The table lists numerous pairs of submissions, each with a percentage value (e.g., 97, 90, 59, 39, 38, 31, 30, 23, 19, 15, 9, 8, 7, 7, 6, 5, 4, 4, 4, 3, 3, 3, 3, 3) and a file type (e.g., 'No comments or white space', 'Comments only', 'Tokenised', 'Original'). Each row also includes the file names and line numbers for both 'File 1' and 'File 2'.

%	File Type	File 1	File 2
97	No comments or white space	8134987.java.ncw:4-891	7482745.java.ncw:5-919
90	No comments	8134987.java.noc:4-893	7482745.java.noc:5-921
59	No white space	8134987.java.now:307-891	7482745.java.now:308-919
39	Original	8134987.java_ori:306-698	7482745.java_ori:307-720
38	Tokenised	8134987.java.tok:147-500	7482745.java.tok:145-516
31	No white space	8134987.java.now:3-293	7482745.java.now:5-293
30	Original	8134987.java_ori:3-294	7482745.java_ori:4-292
23	Tokenised	8134987.java.tok:531-727	7482745.java.tok:539-775
19	Original	8134987.java_ori:706-892	7482745.java_ori:733-920
15	Tokenised	8134987.java.tok:737-891	7482745.java.tok:764-919
9	Comments only	8134987.java.cmt:132-173	7482745.java.cmt:133-264
8	Comments only	4598723.java.cmt:73-175	2134978.java.cmt:397-490
7	Tokenised	6792349.java.tok:173-249	1507252.java.tok:469-561
7	Tokenised	9124987.java.tok:13-118	1507252.java.tok:270-334
7	Tokenised	9124987.java.tok:2-77	6792349.java.tok:8-45
6	Tokenised	6792349.java.tok:194-228	1507252.java.tok:485-523
5	No comments or white space	6792349.java.ncw:201-237	1507252.java.ncw:496-549
4	Tokenised	9124987.java.tok:468-513	6792349.java.tok:196-226
4	No comments	9124987.java.noc:102-140	1507252.java.noc:356-397
4	No comments or white space	9124987.java.ncw:102-134	1507252.java.ncw:356-394
4	No white space	6792349.java.now:201-228	1507252.java.now:496-523
3	No white space	9124987.java.now:109-130	6792349.java.now:82-105
3	Tokenised	9124987.java.tok:468-502	1507252.java.tok:487-520
3	Tokenised	4598723.java.tok:150-177	2134978.java.tok:474-501
3	Original	9124987.java_ori:101-133	1507252.java_ori:355-393
3	No white space	9124987.java.now:102-134	1507252.java.now:356-394

Figure 5.1 - SHERLOCK list of similar pairs

The alternative SHERLOCK view uses a neural network to organise and cluster a corpus of student submission. Pairs of submissions that are deemed more likely to be plagiarised are joined with shorter lines and positioned closer together within the visual representation. This provides a representation of the similarity in an entire corpus, since every document appears in the graphic at some point. Figure 5.2 shows this aspect of SHERLOCK in use.

¹ Figures 5.1 and 5.2 have been supplied by Weiliang Zhang on behalf of Mike Joy, the developer of SHERLOCK and are used with permission. They are from a new pre-release version of SHERLOCK and hence some aspects differ slightly from the literature description.

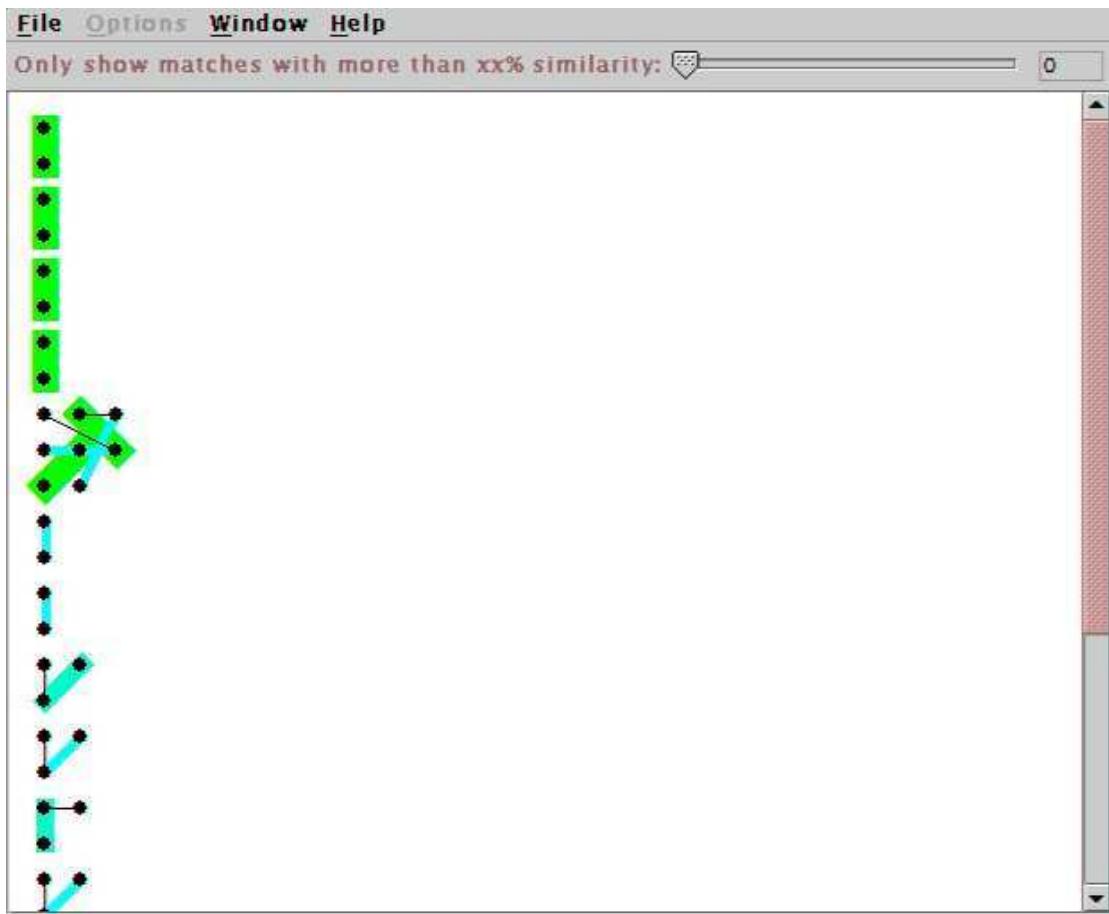


Figure 5.2 - SHERLOCK clustering of similar submissions

There would seem to be little evidence about how successful this view of SHERLOCK is. The simple list approach is more intuitive but cannot show clustering. However the technique is applicable to both source code and free text submissions. It would seem to be appropriate for an independent usability investigation.

The Plotted Ring of Analysed Information for Similarity Exploration (PRAISE) tool provides an alternative method for selecting suspicious sets of documents. Altin describes PRAISE from a functional perspective as part of his student project (Altin 2001). Figure 5.3 shows PRAISE in use².

² Figure 5.3 is used with permission of Jonas Altin and is taken from his project report(Altin 2001).

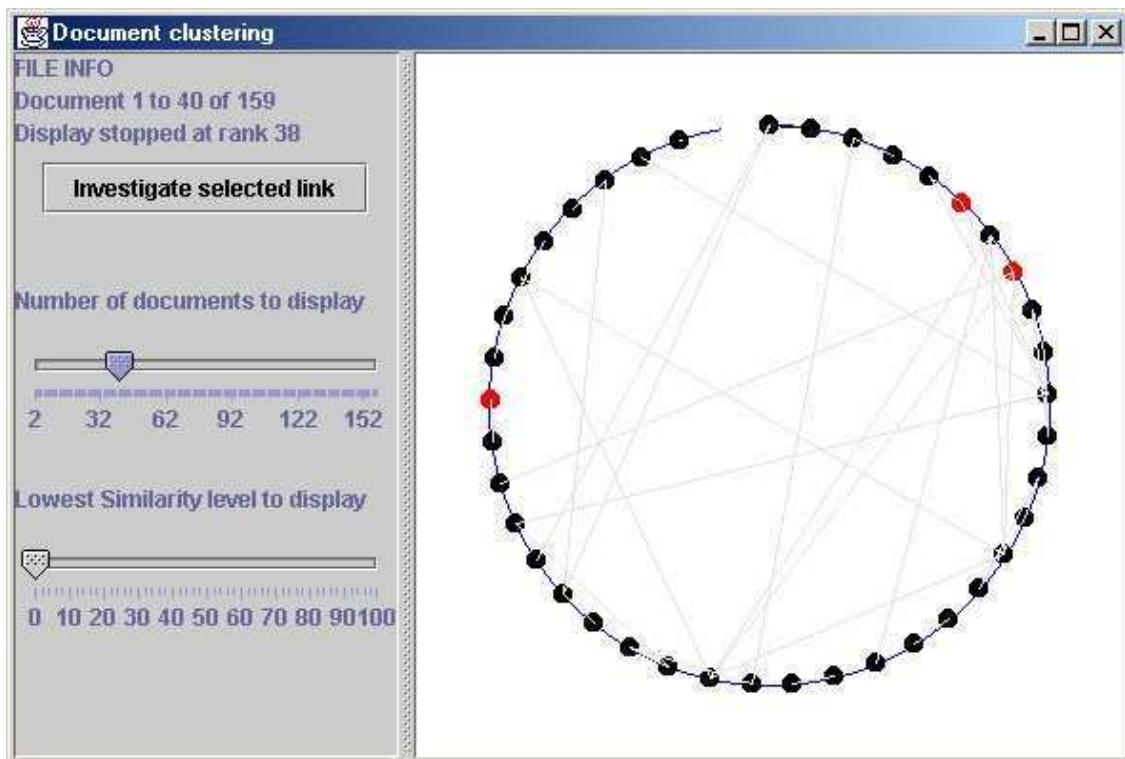


Figure 5.3 - PRAISE used to find clusters of similar submissions

The view shows a set of documents plotted, equally spaced, around the circumference of a circle. A slider allows the number of documents to be displayed to be selected. These are taken by starting at the top of a similarity rank list and working down until the correct number of documents has been displayed. Documents that are judged to be similar are then joined by straight lines. The judgement of similarity is based on the similarity score of the two documents, in the range 0 to 100. The minimum level of similarity to show is selectable by the user with the second slider. Since the first submissions in the ranked list are plotted starting at the twelve o'clock position the first two dots on the circumference are the ones judged by the system to be the most similar.

A PRAISE user can select a pair of submissions to investigate further by selecting the appropriate documents represented as dots on the circle circumference. PRAISE has been successfully linked with VAST, the tool which will be described in Chapter 10. There has been little evaluation done on the PRAISE approach. Altin reports that tutors using the system for the first time found that 46% of the pairs they selected worthy of further investigation, which seems like a high and worthwhile proportion. There are noted to be a few problems with the colour-coding, layout and with selecting pairs where things could be improved. The big question left unanswered is whether the view helps with clustering and whether a tutor is likely to miss pairs based on the visualisation method, something that requires further investigation.

DotPlot is an alternative tool that can be used to compare student submissions that may be plagiarised, although it is more usually used to investigate texts for other linguistic properties (Helfman 1994). Figure 5.4 shows a DotPlot of 20,000 lines of C code. The more intensely pixelated areas show greater self-similarity³.

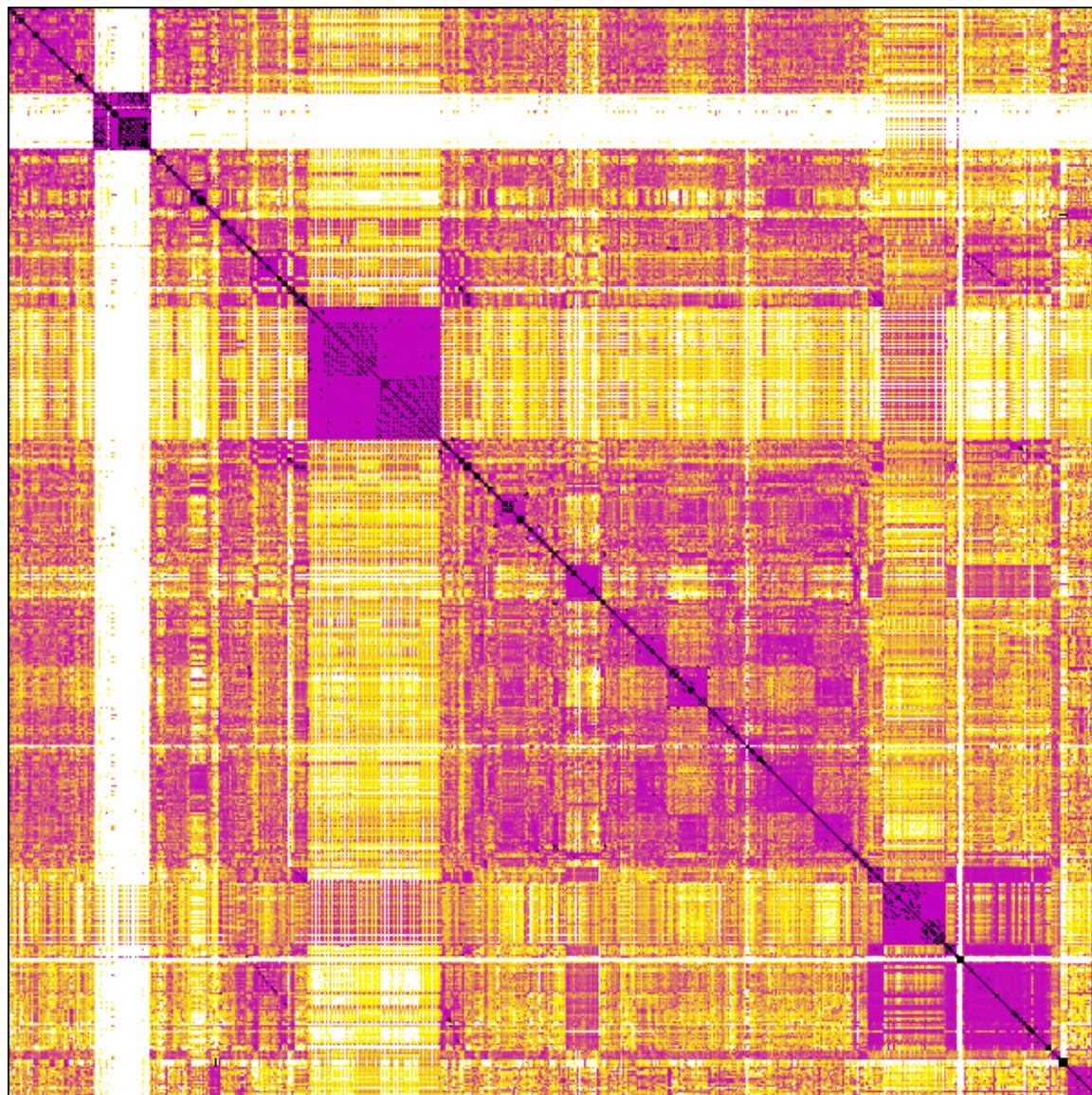


Figure 5.4 - DotPlot

The technique used by DotPlot is to provide a representation of an entire corpus at a time. Every source code submission in a corpus is concatenated together. The order of concatenation is calculated using a built-in algorithm. Helfman states that this is intended to group submissions to have the maximum visual effect on the associated graphic, although quite what this is considered to mean is not clear. A representation of the entire concatenated text is plotted against itself, giving similarity down the axis denoted by the line. Where a line of code matches a dot is plotted on the appropriate pixel denoted by the appropriate position in both directions. For instance a line mid way through the concatenated text that matches the first line in the concatenated text

³ Figure 5.4 is used with permission of Jonathan Helfman. A link to his Web site <http://www.research.att.com/~jon/dotplot/ama.html> is included as requested.

would cause a dot to be plotted in the centre of the uppermost row of pixels and symmetrically in the centre of the leftmost column of pixels.

DotPlot is intended to aid information retrieval. Helfman states that plagiarism detection might be a suitable use but no further evidence about the success of the approach is given. Observation suggests that it is not immediately obvious how the plot relates to the student submissions. Submissions could easily be disguised, say by changing variable names or spacing in every line, and this would throw the engine. Perhaps tokenisation methods could be used to avoid this. Similarly, although the engine may be translate to free text, perhaps by considering sentences instead of lines of source code, any reasonably sized corpus would produce an enormous DotPlot and it is not obvious how a tutor would decide which parts of it to investigate.

Ribler and Abrams introduce two new types of graphics that can be used for finding similarity in source code (Ribler & Abrams 2000), although they give little evidence as to how well they work in practical terms. Further details of the process are given in Ribler's PhD thesis, which supports the results with extensive tables of results (Ribler 1997). The associated engine is demonstrated to be useful in finding similarity in a small corpus of student submissions that have been deliberately plagiarised.

The aim of these graphics is to discover how similar one chosen submission is to all the remaining submissions in a corpus, which Ribler and Abrams state is a computation of linear time complexity, claimed to be an improvement over existing algorithms which operate in quadratic time. This however is very misleading, since their new algorithm would still require a set of comparisons for every pair of submissions, which gives an operational time complexity which is quadratic. Adding a single new submission to a corpus requires it to be compared with every existing submission. For n -submissions the number of comparisons is proportional to n -squared.

The two types of graphics, here known as patterngrams, are implemented in the prototype tool named Same. The first type of patterngram, the categorical patterngram shows how many files a piece of text from the base file occurs in. A piece of source code, called the base file, is chosen for visualisation against the rest of the corpus. Characters from the base file, in sequence, are listed across the bottom of the screen, unless they occur in other files, in which case they are listed the number of lines up equal to the number of files they occur in. Sequences are only shown when they are above a minimum defined length in characters. Ribler and Abrams state that sequences occurring in many files are likely to be shared use of language that is purely coincidental. Sequences that occur in just one or two files are said to be likely indicators of plagiarism. A categorical patterngram is shown in Figure 5.5⁴. It shows similarity to the file labelled 0.

⁴ Figures 5.5 and 5.6 have been included with permission of Ribler and are taken as from his thesis (Ribler 1997).

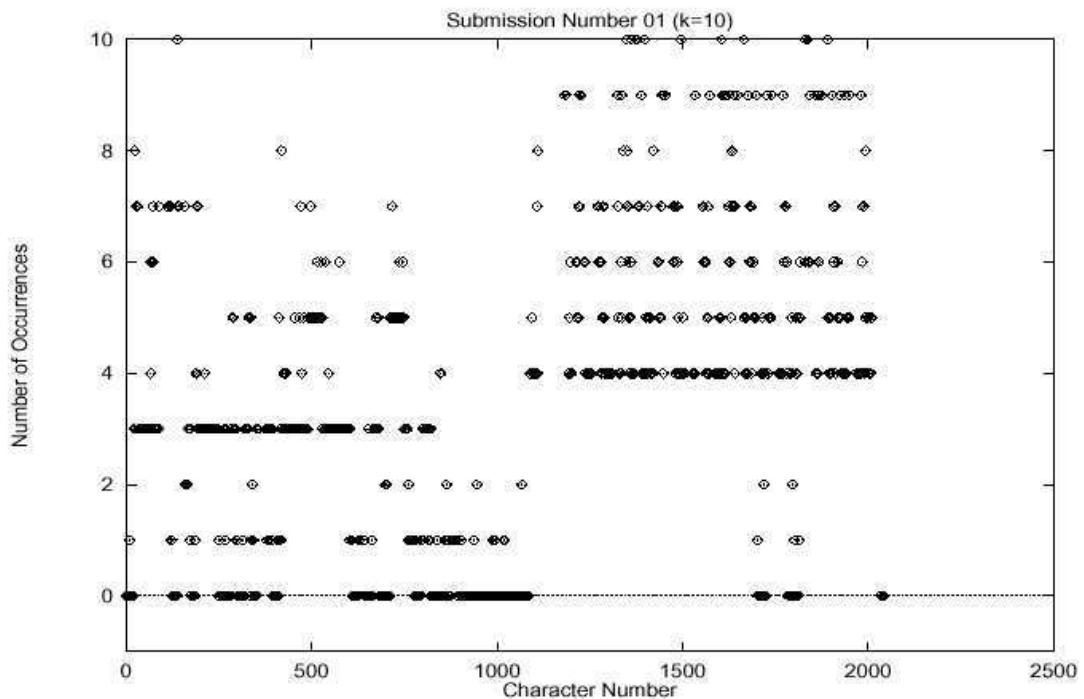


Figure 5.5 - Categorical patterngram

The second type of patterngram, the composite categorical patterngram identifies the files that are common to the base file. Here every file in the corpus is plotted on subsequent screen lines, with the base file shown wherever it occurs in the corpus. For every file a point is plotted at a given horizontal co-ordinate if the sequence of characters of a defined length starting at that position in the base file occurs anywhere within the file in question. This gives an indication of which parts of the base file are being reused, but, of course, does not give any indication about where within the other individual files the sequence is being reused in the rest of the corpus. Since the base file will always match with itself it will become a horizontal line across the scene. Source code files are tokenised before being processed to find changes of variable names. Figure 5.6 shows a composite categorical patterngram. It again shows similarity to file 0.

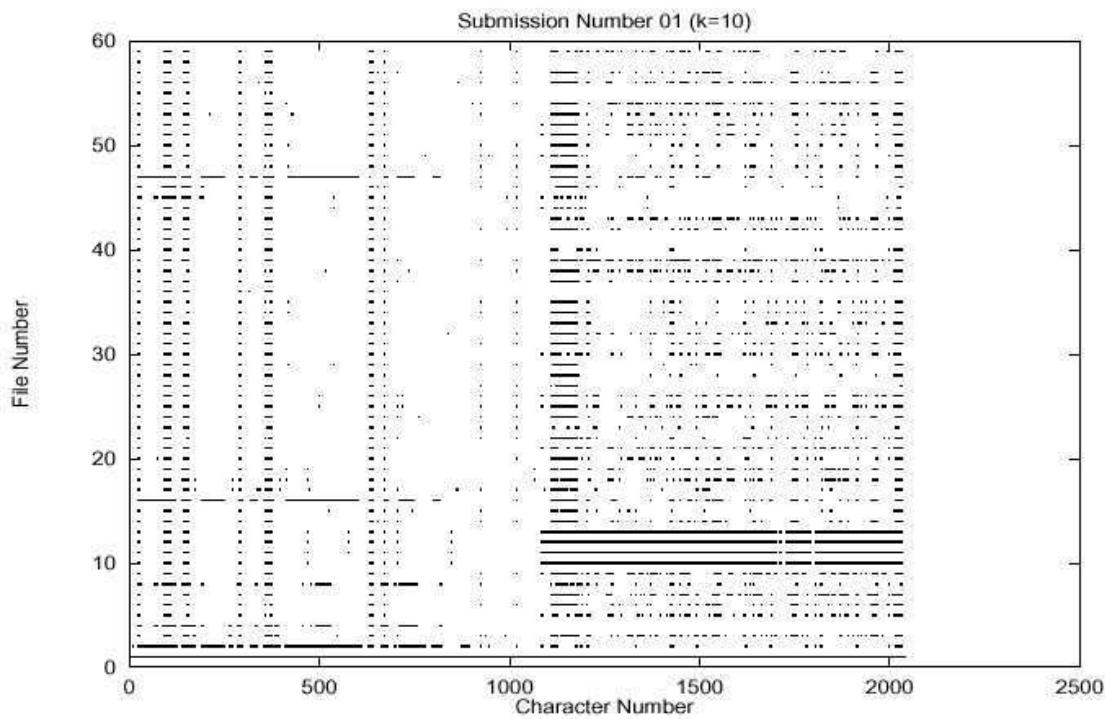


Figure 5.6- Composite categorical patterngam

Although no speculation is given there seems to be no reason why this technique would not translate into free text plagiarism detection. However the patterngams would already seem to be of limited use. Choosing one file to use as an initial base file is not an easy task. One solution might be to develop more software to determine which submission to use as the base file. This would also require further research and insights as the publications give no advice as to how a suitable base file should be chosen. The composite categorical patterngam could be useful for clustering since it allows many programs to be compared at the same time, providing a base file could be established. An independent evaluation of the software is needed to see if it is of any use for plagiarism detection in corpora that consist of many submissions as are becoming common at many institutions.

In general some of the techniques here appear to have potential for free text plagiarism detection, but none appear to be ideal. Apart from its lack of ability to show clusters of similarity the simple approach presenting an ordered list of pairs appears to be the single most appropriate technique. This technique has the advantage that the way for a tutor to investigate similarity is intuitively obvious, simply start at the top of the list and examine each pair until the similarity is judged minor or available time expires. The other techniques require a usability examination and comparison to see if they are appropriate.

One other solution might be to have a system that allows multiple views of the same corpus. That is it might default to the ordered list of pairs view, but with options to switch to a PRAISE-like view and/or a SHERLOCK-like view, whilst keeping the same documents selected during the view change where possible. This would allow users the choice of their preferred viewing method, but such a system would be more involved to engineer.

5.3 - Visualisations for a pair of submissions

The verification and investigation stages of the four-stage process are the two most crucial and the two most time consuming of the whole process. This is because they are the least automatable and require human intervention and assistance. It is a relatively straightforward process for a machine to rank pairs of submissions that it believes are similar, whether these are provided as a list, or in one of the whole corpus visualisations suggested above. It is a more involved process for a tutor to deduce why the machine believes they are similar and in what places. It is here that visual assistance can be useful.

Table 5.4 shows the different visualisation techniques have been identified for this purpose. The layout is similar to that of Table 5.3, again naming and describing the technique, along with a list of the tools from Table 5.2. The print and look approach, listed first, is again not strictly visual, but is included for comparative purposes.

Name of technique	Description of technique	Examples of source code uses	Examples of source text uses	Examples of free	Comments
Print and look approach.	Two submissions deemed similar are printed and examined by a tutor with no similarity indications.	Any with no visual assistance.	Any with no visual assistance.		Deemed unsuitable and time consuming.
Comparative hyperlinks	Submissions are presented in two windows with similar sections hyperlinked.	MOSS JPlag			The most common visualisation technique.
SHERLOCK text mark up	Submissions are presented in two or more windows with textual annotation.	SHERLOCK			Purely textual is not easy to navigate. But submissions can be presented in more than form at once, for instance in original and tokenised forms.
Two text DotPlot	Two texts are presented against one another using the DotPlot approach.	DotPlot DUPLOC			This technique not actually suggested in DotPlot but seems appropriate. DUPLOC is a refined version made into a complete tool, although not really intended for plagiarism investigation.

Table 5.4 - Techniques for visualising similarity in pairs of submissions

The comparative hyperlink approach, originated with the JPlag detection engine for source code plagiarism and is also notably used by MOSS. It is probably the most common comparison approach. Figure 5.7 shows JPlag in action comparing two source code submissions.⁵

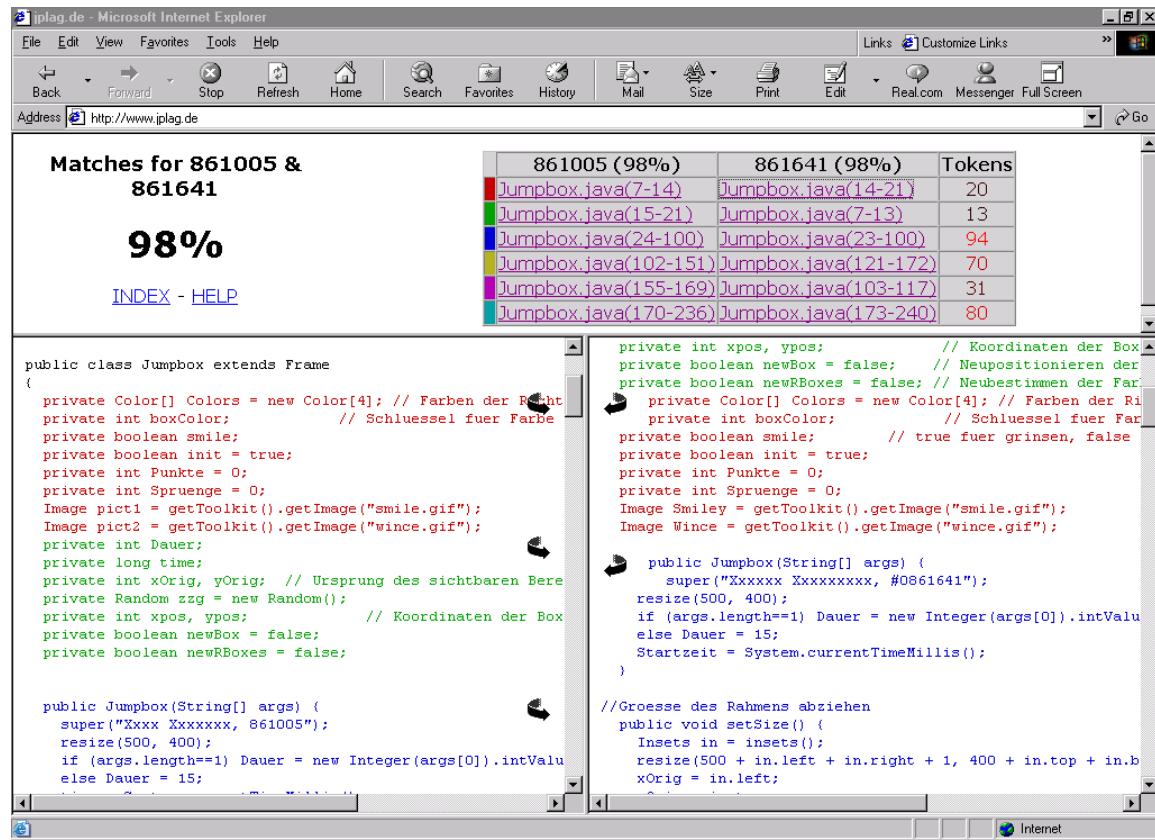


Figure 5.7- JPlag used to investigate two student submissions

The actual view used varies a little across the tools, but all work along the same principle. In the case of JPlag two windows are presented side by side showing the two submissions in different frames. Areas of the submissions judged similar are presented in the same colour. These links might be hyperlinks, where clicking on one would navigate the other frame to the corresponding area. In the case of JPlag it is necessary to click the links in the top status window instead.

The hyperlink approach looks like it should be successful but there is little to compare it with. One usability concern is that it is very difficult to get a representation of the entirety of both submissions at the same time, to establish what proportion of them the similarity represents. This can be vital in deciding how severe the similarity is and how severely the problem should be pursued. The approach would seem to be applicable to free text submissions, although fragmentation may here cause comparison problems.

⁵ Figure 5.7 is taken from the JPlag Web site, at www.iplag.de.

Figure 5.8 shows one further aspect of SHERLOCK. This is a view of two submissions that have been judged similar⁶.

```

File Options Window Help
Find... Ctrl+F
Find again F3
Save original sections to ASCII file...
Print...
Close

Submitted file, 7482745.java.noc
// This class is used to give information about the car
public class Car extends Thread
*****BEGIN SUSPICIOUS SECTION*****
{
    Road road;
    TrafficLight light;

    int number;
    int position = 0;
    int lane;
    int carDir;
    String Dir;

    Junction junctionTop;
    Junction junctionBottom;

No comments, 7482745.java.noc
*****BEGIN SUSPICIOUS SECTION*****
public class Car extends Thread
{
    Road road;
    TrafficLight light;

    int number;
    int position = 0;
    int lane;
    int carDir;
    String Dir;

    Junction junctionTop;
    Junction junctionBottom;

    Road [] allRoad;
    #line 20
    #line 21
    public Car(int lane, int carDir,
               int carNo, Road [] roads, TrafficLight light, Junction junctionTop, Junction junctionBottom)
    {
        allRoad = roads;
        if (lane == 0)
            this.road = roads[0];
    }
}

Submitted file, 8134987.java.noc
// This class is used to give information about the car
public class Car extends Thread
*****BEGIN SUSPICIOUS SECTION*****
{
    Road road;
    TrafficLight light;

    int number;
    int position = 0;
    int lane;
    int carDir;
    String Dir;

    Junction junctionTop;
    Junction junctionBottom;

    Road [] allRoad;
    public Car(int lane, int carDir,
               int carNo, Road [] roads, TrafficLight light, Junction junctionTop, Junction junctionBottom)
    {
        allRoad = roads;
        if (lane == 0)
            this.road = roads[0];
    }
}

```

Figure 5.8- SHERLOCK comparing two submissions

The SHERLOCK view shows each submission presented twice, once with comments included and once with comments removed. Suspicious sections have been marked with comments, although it is unclear if this is done automatically or has been manually added (the software is not currently available for trial). Generally such marking would seem to be useful and applicable to free text intra-corporeal detection, although multiple versions of the same files in these cases would not be appropriate.

An alternative technique, as listed in Table 5.4, is the two text DotPlot comparison. Although this is not mentioned in the DotPlot literature this approach might be appropriate. Instead of plotting the same submission, or set of submissions, against itself or themselves one suspect submission could be plotted against another suspect submission. This would give a more direct and less computationally intensive view of why the two submissions could be judged similar.

⁶ Figure 5.8 has been supplied by Weiliang Zhang on behalf of Mike Joy, the developer of SHERLOCK and is used with permission. It is from a new pre-release version of SHERLOCK and hence some aspects differ slightly from the literature description.

This is more similar to the approach suggested by the DUPLOC tool. DUPLOC has been described in the literature in a couple of places (Rieger & Ducasse 1998, Ducasse, Rieger & Demeyer 1999). The interface contains a visualisation similar to a DotPlot. Here dots are plotted where lines of code are identical with formatting, comments and common keywords removed. Attempts are also made to clean up the graphic, single dots are removed and dots are added where a partial run of identity is discovered.

The main advantage over DotPlot is that the visualisation is interactive, with the represented lines of code presented in two windows, allowing easier checking of similarity.

Overall DUPLOC seems like a minor improvement over DotPlot due to the improved interface, although still suffers from similar problems with choosing the files to identify. DUPLOC is intended to find commonality in libraries of source code, so that unnecessary duplication can be removed, but has obvious plagiarism detection uses, although the authors do not discuss these. The authors note that they plan fuzzy matching. They also allow the results to be viewed in a text form, suitable for printing, something that could be useful when compiling reports about suspicious cases of plagiarism. There would seem to be no reason why this approach could not be used for free text detection.

The approaches all seem like they might be suitable for detecting plagiarism, with the comparative hyperlink approach looking the most appropriate. But there have been no usability studies done on these. The DUPLOC tool, although not directly intended for plagiarism detection, has an interactive interface, allowing areas represented by the visualisation to be immediately selected using the mouse. But the actual visualisation used looks limited. An alternative approach might be to combine a more suitable graphical visualisation with a suitable linked interface. Such an approach will be pioneered in later chapters.

5.4 - Visualisations for a single submission

It seems inappropriate to leave visualisation techniques without considering how a single submission could be visualised for possible plagiarism. However on the whole these techniques have a problem, if they identify something suspicious there is nothing for it to be compared with. Deciding which submissions to evaluate in this way is also problematic. The only fair way would seem to be to evaluate every submission in a given corpus, which can involve a lot of work for a tutor unless some ordering of single submissions can be generated.

Table 5.5 shows the different visualisation techniques have been identified for this purpose. The layout is as before.

Name of technique	Description of technique	Examples of source code uses	Examples of free text uses	Comments
Look when marking approach	Tutors look for suspicious text when marking			Not strictly visual but the most basic approach.
Cusum technique	Graphs of two properties of a submission are plotted. The areas where they differ may represent plagiarism.		Hersee	Hasn't proved successful in practise.
Freestyle	A stylistic property of a document is plotted throughout. The areas where it changes may represent plagiarism.	Freestyle		More evaluations needed.
Hyperlinked Web page	The submission is plotted as a Web page with hyperlinks to sites where the same words have been found.	iParadigms Most Web plagiarism detection services.		Common technique for Web plagiarism detection.

Table 5.5 - Techniques for visualising similarity in single submissions

The hyperlinked Web page approach is not a single approach in the strictest sense, since the Web page is linked to any number of other pages, each represented a different possible source. But it operates on only a single submission at once. The source submissions linked to can be found outside the direct corpus. Hence it is included here.

Table 5.5 includes two linguistical techniques, namely the cusum technique used by Hersee and the linguistic plot technique used by Freestyler. These were discussed in Chapter 4. Both looked initially promising techniques, but the student writing style has not appeared to be sophisticated enough for areas in a single submission to stand out as suspicious. Both however are appropriate for further study and might prove useful, particularly on longer submissions, such as student projects, or on later work, where the student's writing style has become more consistent. Both these are intended solely for free text submissions and do not look like they would be appropriate for source code.

The hyperlinked Web page is now a very common technique for finding plagiarism in a single submission used by all the known Web plagiarism detection services. This is a variation on the comparative hyperlink approach used by engines such as MOSS. Generally it seems appropriate, but there is no guarantee that it is the most effective approach, since there have been no usability studies done. An alternative approach might be to identify all the possible sources for a given submission, join them together, then plot them using one of the techniques for pairs of submissions previously identified. This might allow similarity to be more quickly assessed without needing to follow multiple hyperlinks.

There is still a lot of work to be done on successful visualisation on similarity both in source code and in free text. Many of the techniques that will work on source code will also work on free text, although the reverse is not always true, since there is no real equivalent to Web plagiarism for source code. There do not yet appear to be stores of programming solutions on the Web in a similar way to there are stores of sample essays.

A better approach would perhaps consist of two stages. The first stage would establish, in a visual pattern, which submissions are similar to which other submissions. The second stage would allow two selected submissions to be presented alongside one another in such a way that makes similarity obvious. The Joy and Luck approach is part way there, presenting a visual representation of a whole corpus, but only presents selected submissions side by side in a window, providing little indication as to why their SHERLOCK engine believes them similar. The MOSS and JPlag technique could be used, where sections of the programs believed similar are hyperlinked in identical colours, to give a more usable two-stage engine. There have been no usability studies for this technique so it cannot yet be wholeheartedly recommended.

A second and more appropriate graphic for comparing two submissions might also make the engine easier to use. This is something that will be explored, particularly for intra-corporeal plagiarism, in Chapter 8.

5.5 - Conclusions

This chapter has described an area relevant to plagiarism detection, work on the visualisation of similarity. Investigation of similarity is important, because just because a machine ranks two submissions highly does not mean that they are plagiarised. The similarity may be acceptable, for instance in the case of shared and correctly cited quotations. Therefore there will always be a need for human examination of similarity.

Simply looking at the printed submissions alone is not time effective, particularly for tutors who are already time constrained. The methods of visualisations suggest some ways in which this may be improved. Although they have not been independently evaluation for usability issues all the indications suggest that these methods offer time savings for tutors who are trained in the associated tools use. But the extent and value of those time savings are questionable.

Chapters 8 to 10 will see the development of a new tool, named VAST, which shows both where two submissions are similar and how similar they are at those points, whilst providing a user interface that allows such similarity to be quickly investigated.

Chapter 6: Literature Related to Plagiarism

Overview:

- Plagiarism prevention and detection literature is listed.
- The literature on student is studied.
- Related areas such as academic plagiarism are discussed.

There are many reports on plagiarism detection that do not fit comfortably within the free text, source code or visualisation literature. This is primarily because they are relevant to plagiarism but unrelated to automated detection. This chapter gives a broad overview on the different reports, articles and books that have been published, classifying them in areas ranging from why plagiarism is an increasing problem to how it is related to copyright.

The chapter also reviews some of the main pieces of literature on student cheating, focusing on the main study carried out in the UK to date by Franklyn-Stokes and Newstead. Since this study was carried out before widespread use of the Web it is suggested that cheating might need to be re-evaluated in the current academic climate.

6.1 - Motivation

The literature presented so far has been focused around areas of detecting plagiarism, using technical methods where possible. The established area of source code plagiarism detection was presented in Chapter 3. Although this area is interesting little research into this problem is still ongoing. Chapter 4 presented the new and similar area of research into detection in free text. This is mainly focused around using the Web search engines to find plagiarism but some progress has recently been made on finding metrics for identifying similarity. Little empirical work has been done on this. Chapter 5 presented visual methods relevant to either or both.

This chapter builds on the literature that has been presented so far, picking up on areas relevant to plagiarism that are interesting or useful and necessary from a completeness viewpoint. Much of this literature is presented from a sociological viewpoint or as advice to concerned tutors. Other related areas have been identified and the main pieces of literature in them is presented in a tabular form, along with an associated discussion. Although this is unlikely to directly motivate future direction for work related to this thesis it can be argued that this might inform and improve the design of tools for tutors to deploy.

Since there is enough material available for some of these sections to fill a whole chapter alone, the publications are presented in much less depth and with much less analysis than in previous chapters. Equally the search for relevant literature has been less thorough as the material is much less relevant. This chapter is included primarily to suggest other areas where plagiarism research is ongoing and to give some sense of completeness to this review.

Some of the areas looked at include the literature on student cheating detailing how and why students cheat, plagiarism by academics, related authorship attribution studies, legal aspects of plagiarism, social aspects and Web related plagiarism.

6.2 - The problem of plagiarism

Much has been made about the problem of students copying material from the Web and handing it in for academic credit. Such Web plagiarism has been the subject of widespread discussion, with many Web sites devoted to describing the problem and possible solutions, together with articles in journals and presentations at conferences. Media all around the world has picked up on the issue. Articles are now not uncommon in British newspapers and any search of the Web will reveal thousands of sites devoted to plagiarism, ranging from prevention articles, personal experiences, discussion forums, detection services and different departments sites warning students not to plagiarise.

As such, any exhaustive treatment of Web and other plagiarism sources is impossible. This is not as great a hindrance as it might appear. Many of the Web sources are notable, amongst other reasons, because they largely contain the same advice and many of the same anecdotes. Therefore this section discusses just a sample of the different sources available. Reasons for inclusion vary, but the focus is very much on the most referenced sources with the most original treatments.

Table 6.1 summarises the more substantial pieces of literature on plagiarism that have been identified, presented in alphabetical order. It focuses primarily on more established articles from both academic sources and from the British media coverage. The short descriptions include interesting or unusual points to come out of the literature.

Authors	Date	Content
Associated Press	2002	Article describing how high school tutor Christine Hamilton was forced to resign when governing board refused to support her zeroing student marks for Web plagiarism. Suggests this is why most plagiarism goes unsupported.
Atkinson	2002	Suggests that there should be a more considered approach towards plagiarism.
Austin & Brown	1999	Raises concerns about plagiarism and suggests steps tutors should take to reduce the possibilities for it.
Baty	1999	Discusses the resignation of an external examiner when plagiarism was judged a 'minor' offence.
Baty	2002c	Describes how the Revise.it Web site forum is commonly used for plagiarism and paper trading, including offers on there from alleged tutors to write papers in return for a fee.
Boyer et al	2000	Compilation of results from an online discussion where the participants believe that plagiarism is becoming an increasing problem.
Carroll	2001	Article that discussed plagiarism within a UK higher education context. States that a national conference on plagiarism was overbooked by 50%, suggesting large concern. Also gives a 34% level of pre-warned students getting work flagged by an electronic detection service.

Clare	2000	Newspaper article detailing the problems faced by academic institutions with students downloading work from the Web.
Cramb	1999	Newspaper article describing an outbreak of electronic aided plagiarism in Edinburgh University.
Currie	2000	Article discussing Bradford students selling their work to essay banks.
Curtis	2002	Discusses Elizabeth Hall Associates, a UK service that will write submissions for students to order, including MBA theses at £50 per hour.
Decoo	2002b	Asserts that plagiarism and cheating is increasingly commonplace and that tutors must retake control.
Gajadhar	1998	One of the earliest Web plagiarism papers describing a view that the Web is 'free' information and suggesting possible solutions. Advocates institutions having a clear and discussed anti-plagiarism strategy.
Goot	2002	Discusses the availability of the Google answers service, where users bid money for researchers to answer their questions using Web sources. This is said to produce many requests for answers to homework questions and related essays.
Griggs	2002	Article that describes how plagiarism and cheating are both common and acceptable in the University of Utah, with students even using ebay to sell papers they've written to other students.
Grossman	2002	Describes how students think little of copying work from the Web and don't believe that they will be caught, particularly at GCSE level.
Harnden	2002	Discusses how the resignation of Christine Pelton over student plagiarism penalties that were not upheld has divided a US community (also in Associated Press 2002).
Heyward	1998	Long lasting Web site providing questions for discussion and suggestions for tutors who may have a plagiarism problem.
Hilton	2002	Plagiarism is briefly identified as an issue during a survey of electronic submission in UK HE and FE. Suggests that even with electronic submission making documents available most detection is manual. The OU is said to be the only institution to routinely use electronic detection, although this is said to use Google, which would be a manual process not a commercial package as inferred by the report.
Insights	1998	Online summary of papers detailing that student cheating is likely.
Johnston	2001	Reports on the availability of a new JISC funded guide on good practice in plagiarism detection. Warns that electronic tools alone are not sufficient.

Kleiner & Lord	1999	Describes how student cheating is on the increase, claiming one in eight students will cheat on any given test. Discusses plagiarism with the suggestion that assessed work must be completed in class and that cheating penalties must be made more severe.
Lathrop & Foss	2000	Substantial book detailing plagiarism solutions, providing worksheets and links to Web sites intended as a start for tutors interested in detection. Mainly intended for US high school level but also useful for UK higher education.
Lingua Franca	2000	Transcript of a radio conversation with John Sutherland who states that there is a current cheating epidemic.
Major	2002	Expose a Web-based UK ghostwriting service by former academic Elizabeth Hall, promising confidentiality to students using it.
Mallon	2001	Book describing what plagiarism is and detailing a number of case studies, mainly from a publishing perspective. 2nd Edition contains an afterword describing the ease of plagiarising from the Web.
Maslen	2002	UK article that cites that at least 1 in 12 Australian student submissions are plagiarised from the Web in the largest study of its type to date.
Miles & Burleigh	2001	Details how an A-Level Maths paper went on sale before the examination and may have appeared on the Internet.
Okkelmo	2001	Short article talking to students which finds a particular problem of people paying others to complete academic work for them.
Pitts Jr.	2002	Describes the plight of tutor Christine Pelton who was forced to resign from her job after an academic board refused to withhold her zeroing of student's marks for plagiarising (also in Associated Press 2002).
Ryan	1998	Early Web plagiarism paper based on the author receiving a suspicious paper and tracing much of the content to Web sites through search engines. A thorough check revealed one in six of the author's students had committed Web plagiarism. Discusses features of student submissions that tutors should look out for that might suggest they have been plagiarised.
Stoerger	2002	Annotated page of links to online articles and sites relevant to plagiarism.
Strom	2002	Describes how the author found that many of his students plagiarise through sharing answers through Instant Messenger software.
Sutherland	2000	Newspaper article describing difficulties faced by academic institutions with students downloading work from the Web.
Telegraph	2002	Short article that states that plagiarism by historians is setting a bad example for students.
Tysome	2002	Details concern when an MSc student who found plagiarism in a previous thesis had her complaint dismissed by the University of Wales.

White	1999	Updated version of a pre-Web article stating that, even then, institutions had a problem with old assignments being resubmitted and students using paper mills. Suggests students need to be taught how to cite.
Woodward & Lomax	2001	How a private tutor leaked an A-Level Maths paper, which ended up sold across the country for £400 (also in Miles & Burleigh 2001).

Table 6.1 - Problem of plagiarism literature identified

Generally there is a high level of concern within the UK and abroad that students are plagiarising. The Lathrop and Foss book comes highly recommended from many different sources (Lathrop and Foss 2000). This provides the most complete existing guide to what plagiarism is, why it is believed to be a problem and how it can best be prevented. However it is no longer completely up to date since the detection services available have been changing and growing at a fast rate.

6.3 - Plagiarism Prevention and Detection

Many Web sites on plagiarism offer advice to tutors on how to stop students plagiarising. The advice given is usually consistent from site to site and includes such things as changing assignment specifications regularly, asking students for early drafts of their submissions and searching the Web to see what is available on a particular topic before setting an assignment specification on it.

Table 6.2 shows the main and most interesting sources that have been identified on plagiarism prevention and detection.

Authors	Date	Content
Austin & Brown	1999	Raises concerns about plagiarism and suggests steps tutors should take to reduce the possibilities for it.
Boone	1999	Describes the author's Web-based submission system. States that plagiarism is less common when using this system, although evidence is only anecdotal.
Boyer et al	2000	Compilation of results from an online discussion containing several suggested prevention methods.
Bridgeman et al	2000	Describes PILOT, a tool that could be a partial cheating solution since it generates different online questions based on algorithm design for different students.
Carroll & Appleton	2001	Most extensive UK study of how plagiarism can be prevented, the result of a JISC sponsored survey and evaluation.
Chester	2001b	Document summarising the different strands of the JISC sponsored plagiarism detection project and the results from the associated workshops. The result has seen JISC provide a UK branded version of iParadigms detection solution.
Evans	2001	UK specific plagiarism paper taking a standard form providing links, discussing plagiarism pointers and possible solutions. Suggests that plagiarism must also be considered as a breach of institutional acceptable use policy.

Gajadhar	1998	One of the earliest Web plagiarism papers describing a view that the Web is 'free' information and suggesting possible solutions. Advocates institutions having a clear and discussed anti-plagiarism strategy.
Green	1999	Suggests that penalties for cheating and plagiarising should be standarised across the UK and made public.
Guiliano	2000	Suggests tutors could concentrate on plagiarism prevention rather than detection and gives some prevention measures, including the use of honour codes.
Harris	1999	Suggests ways tutors can step assignment specifications to stop plagiarism opportunities.
Heyward	1998	Long lasting Web site providing questions for discussion and suggestions for tutors who may have a plagiarism problem.
Howard	1999	States that most assignment specifications encourage 'patchwriting', copying from sources such as the Web making only minor changes. Suggests changes to specifications.
Jenkins	1995	Argues that open book assessments are a more suitable assessment method for computing degree programmes, which could have implications for reducing plagiarism opportunities.
Johnston	2001	Reports on the availability of a new JISC funded guide on good practice in plagiarism detection. Warns that electronic tools alone are not sufficient.
Lathrop & Foss	2000	Substantial book detailing plagiarism solutions, providing worksheets and links to Web sites intended as a start for tutors interested in detection. Mainly intended for US high school level but also useful for UK higher education.
Martin	1992a	Pre-Web article suggesting that plagiarism was already causing concern then and suggesting some changes to assessment procedures to reduce it.
McCabe	2002	Online sourced interview with academic integrity researcher who suggests that honour codes might be an appropriate deterrent for potential plagiarisers.
Moon	1999	Article describing how cheating is a growing problem and places a lot of stress on academics.
Pearson	2002	Online seminar on plagiarism prevention and detection, taking the form primarily of Web links to other useful pages.
Rose	1999	Proposes eight new types of citations, such as an iterative citation for a citation of a citation, designed to help students who struggle to cite successfully. Seems overcomplicated.
Seymour	2002	Short article describing a service to find names of songs based on known parts of tunes. Said could be use to counter music plagiarism.
Stefani & Carroll	2001	LTSN published guide to prevention plagiarism in the UK. Suggests ways of explaining plagiarism to students and changes to the assessment process.

Utley	2002	UK newspaper article about the provision of iParadigms within the UK in a JISC supported project.
Various	2000	Forum based discussion, mainly UK based, providing some ideas on plagiarism detection. Cultural differences are discussed as is peer marking with students expected to find others that are cheating. More standard ideas suggest alternative assessment methods and more student vivas, although a lack of academic resources may prevent this.
White	1999	Updated version of a pre-Web article stating that, even then, institutions had a problem with old assignments being resubmitted and students using paper mills. Suggests students need to be taught how to cite.
Wools	1999	Publicises the availability of tools such as CopyCatch for intra-corporeal plagiarism. States that, at the time, tools for Web plagiarism not widely available.
Writing Tutorial Services	2001	One of a number of sites helping to teach students to cite correctly and avoid plagiarism.

Table 6.2 - Plagiarism prevention literature identified

It is impossible to list and cite the sources of many of the different prevention methods. To a greater or lesser extent this is collective wisdom. Generally the advice seems to be to make students aware about plagiarism and to make them aware that you know how to find and detect plagiarism. This can be coupled with assignment specifications where ready written solutions are not easily available, either on the Web, or from previous students on the course.

6.4 - Academic plagiarism and research fraud

Academics do not always set students a good example about how to behave. The literature contains many examples of academic plagiarism, that is plagiarism by academics for professional gain, as described in Chapter 2. In most cases this involves academics submitting research to journals or conferences that is not their own. Academic plagiarism could also encompass academics using teaching materials from other institutions without acknowledgement. Additionally fraud within research is discussed here, since this is an area relevant to academics, and in some cases taking research data and altering it to find different conclusions can be considered a variant on plagiarism.

Table 6.3 shows the main and interesting pieces of literature that have been identified. It is not intended to be exhaustive, mainly representative.

Authors	Date	Content
Baty	2002a	Discusses the investigation of plagiarism early in the academic career of the Monash v-c.
Baty	2002b	Reveals that the Monash v-c was forced to resign due to academic plagiarism earlier in his career.
Bodnar	2002	Discusses the result of a landmark Canadian lawsuit where Boudreau successfully sued former tutor Lin for using his paper without acknowledgement.
Boseley	2002	Reveals how many medical papers are actually ghostwritten, scientists put their names to papers written by drugs companies in return for funding.
Comfort	1999	Review of a book detailing how fraudulent data in a Baltimore experiment caused a scientific dispute.
Denning	1995	Describes an incident where technical reports were plagiarised from a university FTP site and submitted to journals and conferences. Some reached publication without being discovered. Similarity verified using SCAM copy detection software, discussed in Chapter 4. Denning proposes libraries of academic work and that all submissions are checked against these.
Farrar	2002	Discusses a report that research fraud by academics is growing and not being treated seriously.
Guardian	2002	Discusses the resignation of David Robinson, v-c of Monash, who was found to have plagiarised sections of a 1976 book.
Joo-Hee	2001	Article from the Korea Herald describing how three professors were being investigated for academic plagiarism.
Kock	1999	Kock describes how one of his published papers was submitted by another academic, with minor changes, to another journal. Kock claims he could not pursue it due to the expense involved with US legal system. Kock states such plagiarism inevitable due to academic research commitments and that it will become more common.
LaFollette	1992	Book detailing cases of research fraud in science publications. Gives a number of examples. Also considers how fraud is discovered from a procedural viewpoint, showing how it can damage the career of a whistle blower and take a long time to resolve cases.
Lee	1995	Describes apparent plagiarism of a 1995 letter from Charles Babbage. The original English letter was lost and a French translation found and translated back to English. Herman Berg rediscovered the original English and claimed that an article covering the finding, but not mentioning him by name, constituted plagiarism. The whole thing sparked a debate running from 1980 onwards especially when it was discovered a second copy of the letter had existed all along. This would seem to be academic misconduct but not plagiarism.

Martin	1992b	Discusses how the way science is financed puts pressure on academics to present fraudulent data and some solutions.
Martin	1997	Empathises the problem of academics putting the names on research papers written by their students, or sometimes removing the student's name entirely.
Samuel	2002	Describes how the publisher of more than 70 groundbreaking Physics papers over 13 years was found to have fabricated his results. Suggests shortcomings in the peer review process.
Samuelson	1994	Discusses self plagiarism in academia, where portions of previous works are re-used in later submissions, often necessarily as background. This is said to cause problems when some publishers require copyright to be issued to them. Prosecutions have been made in film and software development, but seem to have not been made in academic publication. Samuelson states that self plagiarism is common, but how common is not known and that reviewers must be chosen who are subject area experts. Samuelson suggests the application of a 30% rule as the maximum appropriate amount of reuse.
THES	2002	Report that Monash University governing council are sticking by their v-c, who was found to have plagiarised early in his academic career.
Wright Thompson	2002	Article describing how several US staff reporting academic and student plagiarism cases became known as whistleblowers, receiving threats or being forced from their jobs. This suggests why few academics come forward when they suspect plagiarism.

Table 6.3 - Academic plagiarism literature identified

The proposals about how to avoid academic plagiarism are of interest. Denning suggests establishing libraries of academic work (Denning 1995). Future submissions to journals and conferences could then be checked against these libraries to see if they constitute plagiarism. This would seem to be a generally sensible idea, similar to the way in which detection services exist to detect student plagiarism. There is certainly scope for debate about how such a service would be funded, how the library could be ensured to be complete and how submissions could be verified against it. There is also the question about whose responsibility it would be to check the submissions, those of reviewers or those of journal editors or conference organisers.

Samuelson suggests that 30% shared similarity is acceptable for self plagiarism (Samuelson 1994). Any developed engines would need to take this amount of similarity into account. The situation however is not this clear cut. Such similarity would still have to avoid being word for word. It could also be expected that similar background material would be near the beginning of academic submissions. It is also debatable whether 30% means 30% of word count or 30% of page count, the latter would include diagrams, something that current detection services are unable to check. Hence a more detailed examination of what constitutes self plagiarism is necessary to determine what appropriate prevention methods are. It would also be of interest to see if the 30% rule holds and if this 30% is primarily at the beginning of

submissions, something that could presumably be accomplished with current technical tools. There is a lot of scope for further work on technical investigation of self plagiarism in academia.

6.5 - Legal Aspects of Plagiarism

Although this thesis is concerned with plagiarism detection from a technical perspective there are legal issues to be considered when implementing a detection process and policies. A number of Web sites, articles and papers also discuss copyright and its implications. It seems impossible to leave a discussion about plagiarism without mentioning some of these.

Table 6.4 lists the main pieces of literature that have been identified and provides a short description of each.

Authors	Date	Content
Anderson	1998	An annotated bibliography presented in chronological order of 623 sources from 1907 to 1995. A few sources are relevant to plagiarism, most are more relevant to copyright or intellectual property.
Bowers	1997	Book describing a case of literary plagiarism where a poet, publishing mainly in newsletters, found out some of his poems had been reprinted under someone else's name in other publications with minor changes. Describes how Bowers tracked the plagiarist down using private investigators in a costly exercise with little in return.
Clankie	1999	Describes how misuse of US trademarks have made product names, such as xerox, hoover or even aspirin into generic terms and how the use of these generic terms in fiction has had legal implications.
Foster	2002	Discusses the copyright issues of sending student's work to a database such as those used in the Web-based plagiarism detection services. Suggests that explicit permission is needed from students.
Fox	2002	Copy protection on CDs is said to be easily defeated with PC software updates, despite the music industries best efforts.
Halbert	1999	Traces changes in intellectual property thoughts through the feminist movement, comparing copyright to men's ownership to women.
Larochelle	1999	Suggests an alternative view of books from a philosophical perspective as a sum of style, form of content, which would allow functional definitions of plagiarism and copyright.
LeClerq	1999	Finds huge differences in definitions and penalties for plagiarism in 152 accredited US law schools responding out of 177 surveyed.

Lindey	1957	Extensive early book study of plagiarism in different types of media, detailing plagiarism in the likes of films, books, music and plays. Also contains chapters discussing plagiarism from psychological, legal and ethical viewpoints.
Livingston-Webber	1999	Describes how material from the US fanzine industry, now a predecessor to many sites in the Web, commonly used copyrighted material, such as using characters from TV series in fan fiction. This was seen to support an anti-copyright movement.
MacKenzie	1991	Report on how the EC asserted copyright protection to software source code.
Mann	1999	Discusses international copyright laws based on a case where a French magazine stole quotes from a US magazine, a case complicated by translation. Alternative views on both strengthening and weakening copyright are presented.
Paull	1928	Historical study of literary crimes including many case studies. Contains a section on plagiarism, although many other areas described can be considered related.
Randall	1999	Outlines how plagiarism has recently changed from a social problem to a legal problem. Until recently imitation in literature was said to be common and respected.
Ringle	2002	Describes a Science museum taking large sections of explanatory text from an uncited published source.
Runk	2002	Article describing the problems with clergy downloading sermons from the Web and presenting them to their congregations without acknowledgement.
Standler	2000	Discusses legal issues and plagiarism cases relevant to US higher education.
Stearn	1999	Discusses the role of US copyright law in protecting scholarly work. Stearn notes that US authors have few rights in plagiarism cases since they are not usually the copyright holders.

Table 6.4 - Legal plagiarism issues literature identified

Generally the papers identified present many tangents but are not directly relevant. Of more interest are the literature discussions of plagiarism cases. Table 6.5 shows some of the main cases described where plagiarism reached the law courts.

Name of case	Reported in	Description of case
Napolitano vs Princeton University	Standler 2000	Napolitano committed extra-corporeal plagiarism copying much of a 12 page paper from a book and had her bachelor degree withheld by a year. A law court judge upheld the penalty since it was within Princeton's legal code.
Lamberis vs Northwestern University	Standler 2000	47 pages of Lamberis' 93 page thesis were found to have come from two uncited sources. Legal appeal upheld Northwestern's ban on Standler practising law for six months.
General cases	Standler 2000	Standler notes cases where PhDs, medical licences and degrees revoked after they were awarded. The courts of appeal upheld all the university decisions.

Table 6.5 - Plagiarism cases discussed

Where plagiarism cases could not be taken forward internally Standler states that the main problem is where university regulations do not clearly define it (Standler 2002). LeClerq's survey supports this suggesting that many Law schools are ill prepared for appeals by students (LeClerq 1999). This suggests that a more consistent approach to plagiarism policies is necessary.

Standler suggests that a clear line is drawn between a failure to meet ideas, representing an accidental and minor case of plagiarism compared with deliberate and more substantial plagiarism (Standler 2000). He also recommends a pro-active anti-plagiarism policy and plagiarism hearings that place as little extra work on tutors as possible. These views seem reasonable and support the need for a consistent taxonomy and definitions, as proposed in Chapter 2.

6.6 - Sociological, Cultural and Psychological Aspects of Plagiarism

Within the plagiarism literature there are many more generic papers that do not fall directly within a detection remit but which can still provide useful background reading. This section presents some of the papers that have been identified dealing mainly with cultural, psychological and sociological viewpoints.

Table 6.6 outlines the literature that has been identified from a cultural viewpoint. The table is listed with authors sorted into alphabetical order.

Authors	Date	Content
Buranen	1999	Surveys International students in the US finding that they do understand about plagiarism. The cultural views that international students are plagiarists is said to be an oversimplification by a Chinese colleague that would have been true 30 years ago, but not today.
Dryden	1999	Shows that academics in Japan do not find plagiarism a problem. They do not have a word for it, but find exam cheating problematic. The practice is believed to show respect for ancestors. US textbooks plagiarised in Japanese are said to be commonplace.
LaFleur	1999	Describes how traditional histories of China were all plagiarised from previous histories of China, with occasional new comments. The comments are not cited when these two are plagiarised. This gives a communal view of intellectual property in China. The paper also describes efforts by academics to find the original source of the comments of their histories.
Lesko	1996	Descriptive plagiarism survey of tutors of postgraduate students at Edinburgh University about native and non native speakers.
Mallon	2001	2nd edition of a book detailing plagiarism cases across a literary and academic spectrum.
Roy	1999	Roy surveys faculty members by telephone finding that they believe students plagiarise mainly due to a lack of understanding and cultural differences.
Swearingen	1999	Discusses how an international view of shared intellectual property could be considered a historical view where it is essential to find a meaning but not to create it yourself. Swearingen suggests that sharing knowledge may be more important in higher education. However this would require massive cultural changes.

Table 6.6 - Plagiarism papers identified from cultural viewpoints

The cultural literature finds that views and visions of plagiarism differ widely from person to person and country to country, even when largely similar surveys are done. The best suggestion would perhaps be that students and academics outside the countries using a westernised education system should be left alone to operate as they do now. But students and academics entering the westernised system need to be taught to adapt and academics need to be aware that they have this requirement. Perhaps education before entering the westernised system is necessary, just as language instruction in English, say, is necessary.

There are also a number of publications on student writing methods and techniques. These are presented in Table 6.7.

Authors	Date	Content
Baruch	2002	Describes how many autobiographies and novels, especially of celebrities, are fraudulent, since they are ghostwritten with little acknowledgement.
Haviland & Mullin	1999	Compares the student and academic writing processes, finding that academics are encouraged to collaborate on papers, but students are expected to not collaborate on submissions. Suggests that one of these views is inappropriate.
Leight	1999	Suggests that the difficulties institutions have in defining plagiarism also occur in writing textbooks. Around 70 writing textbooks of the 1980s and 1990s fail to give a constant and clear definition.
Paull	1928	Book about literary ethics that contains many historical examples of how work has been copied and plagiarised from.
Shamoon & Burns	1999	US writing centres are said to breed plagiarism as work improves too much during a visit. Shamoon and Burns review the writing centre literature.
Simmons	1999	Discusses how plagiarism was common in the US at the end of the 1800s when US colleges expected students to write a short paper every day and a long paper every two weeks. Students would instead split their workload. Cases from the literature are cited.
Spigelman	1999	Surveys a student writing group. Finds that they believe using a verbal suggestion is fine, but using an idea or reading from a peer's paper, requires credit. Discusses how these views compare with general usage.
Swan	1994	Describes the unintentional plagiarism of deaf-blind writer Helen Keller where words became attached to her mind and were reused in future writing. Such subconscious plagiarism should be seen as a special case but is seen no differently to any other in the courts.
Wilson	1999	Struggles to define the differences between collaboration and plagiarism based on a survey of US plagiarism policies. Only 22% of 95 representative US institutions surveyed in 1992 and 1993 mentioned collaboration in their policies and were generally found to be too generic to be of use.
Zelbroski	1999	Presents a sociohistoric view of plagiarism viewed as a negation of authorship, blaming writing schools for not correctly teaching what is required.

Table 6.7 - Plagiarism papers about writing methods

The writing literature finds a largely common view that writing centres provide opportunities for students to plagiarise. Writing centres will remain a necessity in cultures like the US where most students are encouraged into higher education. Instead it would seem to be necessary for academics to be more prepared to use automated detection and for students to be pre-warned both in class and in textbooks.

Table 6.8 shows the remaining papers that have been identified relevant to a more sociological viewpoint.

Martin	1994	Discusses the view that student plagiarism is being treated too harshly when similar techniques, such as ghostwriting, are common in regular life.
Roy	1999	Surveys faculty members finding that they nearly all believe that plagiarism is a problem, but not with their students. This suggests that many of them must have misplaced trust. Also find faculty have two views of approximately equal numbers. One group views plagiarism as stealing, the other as misrepresentation.

Table 6.8 - Other papers describing views of plagiarism

It is impractical to look at the literature on plagiarism viewpoints in more detail in this thesis but there is certainly scope for an investigation to compare different literature on plagiarism from around the world and the changing views on it. One thing that is obvious is that more education, for both students and tutors, is necessary.

6.7 - Extent of student cheating and plagiarism outside the UK

Although it is not directly relevant to technically detecting student plagiarism it is helpful to know how big a problem student cheating is and the methods student use to cheat. This may be appropriate to inform the requirements for a new tool.

The vast majority of the literature available discusses the North American situation. In particular Lathrop and Foss' book contains a chapter summarising many of the main US surveys over the past few years (Lathrop & Foss 2002). Although these have analogies with the UK situation they are not directly relevant and hence only a few major results from the literature are discussed here. In some cases the results listed come from citations to other sources. In these cases the literature read and the source cited are both listed.

Tables 6.9 to 6.12 summarise the main non-UK surveys identified, sorted into chronological order within each table. One survey is designated pre-Web, this means that it was carried out before widespread use of the World Wide Web. In these cases the cheating percentages listed could be expected to be below present levels. The table also gives a methodology for the survey, where it is given and the size of the survey, where known. The findings are split in four ways, each contained in a different table. Table 6.9 shows admitted results, where students are discussing the cheating that they have done. In these cases they might be lying to either downplay or exaggerate their misdemeanors. Table 6.10 shows believed results, where students are talking about what they think their peers do. Table 6.11 shows the one case where opinion results are given. These results are personal views. Table 6.12 shows another case of observed results. These are found from the work being submitted to a Web-based plagiarism detection engine. The percentage of plagiarism found here might be an underestimate if the engine missed similarity.

Authors	Date	Location	Type of survey	Size of survey	Main findings
Burgess	2002	New Zealand	Not stated	381 students	80% admitted cheating. 63% admitted serious cheating.
Folkers & Campbell	1999	US	General population (parents results only)	Unknown	9% of parents write their child's college entrance exam.
Folkers & Campbell (cited poll)	1999	US	High school students	Unknown	80% have cheated in the past. 95% of students who have cheated have not been caught.
LeClerq	1999	US	Law students	Unknown	60% have plagiarised at least once.
Lamb (cites Patrick Scanlon)	2000	US	Questionnaire	700 students at 9 universities	25% have committed Web plagiarism. 75% believe their peers plagiarise.
Lathrop & Foss (cites Who's Who Among American High School Students survey)	2000	US	Samples of top 5% of US high school students	Unknown	80% had cheated. 67% had copied homework. 13% had plagiarised essays.
Sheard et al	2002	Australia	Questionnaire	137 first year IT students at Monash universities	24% had plagiarised from text books. 33% had copied from a friend. 10% had stolen work from another student. 3% had stolen and copied work from a tutor's pigeonhole. 4% hired someone to write assignment's for them. (Similar results from 150 students at Swinburne University, Australia).

Table 6.9 - Summary of admitted results in non-UK student cheating literature

Authors	Date	Location	Type of survey	Size of survey	Main findings
Morgan Foster	1992	US pre-Web	Ask how and why students cheat	149 students doing human relations unit	81% believe their peers look at each other's exam scripts. 59% believe their peers use crib sheets. 55% believe sharing homework is common. 47% believe peers reuse work in different subjects. 35% believe peers have others do work for them 31% peers plagiarise
Folkers Campbell	1999	US	General population (adults results only)	1000	72% of high school students cheat often. 64% of college students cheat often.

Authors	Date	Location	Type of survey	Size of survey	Main findings
Sheard et al	2002	Australia	Questionnaire	137 first year IT students Monash universities	53% believed their peers copied material from submissions. 36% believed their peers copied from friend's textbooks. 36% believed their peers copied from the Internet. 75% believed their peers collaborated on assignments meant to be completed individually. (Similar results from 150 students at Swinburne University, Australia).

Table 6.10 - Summary of believed results in non-UK student cheating literature

Authors	Date	Location	Type of survey	Size of survey	Main findings
Folkers Campbell	1999	US	General population survey (students results only)	Unknown	90% believe cheaters are punished too leniently. 18% would turn in a student they caught cheating. 5% would cheat themselves if they caught another student cheating.

Table 6.11 - Summary of opinion results in non-UK student cheating literature

Authors	Date	Location	Type of survey	Size of survey	Main findings
Buckell (reports on survey Steve O'Connor)	2002	Australia	Work submitted iParadigms	7151 essays from 17 subjects at 4 universities.	1 in 12 took over 25% or more of their material from the Web. All similarity confirmed by researchers to be plagiarism.
Braumoeller & Gaines	2001	USA	Work submitted to EVE2.	180 essays on same subject.	Around 1 in 8 submissions verified as casual or blatant plagiarism. A warning given to only half the students did not alter this proportion.
CNN	2001	US University of Virginia	Intra-corporeal check on submitted work.	2000 students over 5 years.	60 students in latest year copied from previous years.
Lamb (cites Robert Spear)	2000	US	Submits work to Web-based plagiarism detection engine	1 class	10% plagiarists.
Maslen (reports on survey Steve O'Connor)	2002	Australia	Work submitted iParadigms	7151 essays from 17 subjects at 4 universities.	As Buckell 2002 (same survey)
ScienceDaily	2002	USA	Work submitted to EVE2	180 essays on same subject	As Braumoeller & Gaines 2001 (same survey).

Table 6.12 - Summary of observed results in non-UK student cheating literature

The results shown in Table 6.9, although limited, suggest that cheating occurs at both high school and university level and is commonplace. In particular even the top 5% of students plagiarise, although to a lesser extent than their peers. LeClercq notes a worrying trend, where law students seem to be the worst offenders. 60% of them admitted plagiarising at least once, when these are the people expected to protect intellectual property (LeClercq 1999). The University of Auckland is said to have a novel way of dealing with plagiarists (Burgess 2002). There students are said to be fined up to \$300 (New Zealand dollars) every time they are caught.

Morgan and Foster, as shown in Table 6.10, recommend that academic institutions clearly define cheating and associated punishments to the students as well as set assignment specifications that minimise the potential for students to cheat (Morgan & Foster 1992). The results show that both pre and post Web student cheating and plagiarism is seen as common.

Folkers and Campbell's results, seen in Table 6.11, collectively suggest that other people cheating is seen as acceptable by only a minority of US students, but those people themselves are likely to cheat (Folkers and Campbell 1999).

Lamb, as seen in Table 6.12, notes that whilst committees have the power to award an 'XF' mark to plagiarising students (Lamb 2000). Here the 'X' denotes the academic dishonesty and the 'F' a failing grade for the course. However few will do so and most students are able to claim mitigating circumstances to avoid punishment. 10% seems to be the best indication of how common plagiarism is in the US, although this may be a lower boundary of the actual number. The most substantial survey, by O'Connor and reported throughout the press finds an estimate of 1 in 12 substantially plagiarised essays (Buckell 2002, Maslen 2002). This is slightly less than Lamb although O'Connor does state that he believes his figure to be conservative.

The most recent survey, from Australia, has particularly worrying results (Sheard et al 2002). Here a substantial minority have admitted to stealing work from other students or from tutors or to paying people to write submissions for them. The latter is an area that automated plagiarism detection would probably not detect. All the students surveyed were IT students, which might imply a greater computer and Web literacy than in general.

A US report claims that the apparent growth in student cheating might be related to a growth in cheating in the adult population (Loftus & Smith 1999). Job seekers were known to be stealing CVs from the five million (at the time) available online and altering them to suit their purposes. Three quarters of people surveyed admitted to changing or enhancing details on their CVs, a process that could cause problems at a later date. Such plagiarised CVs could perhaps be detected using techniques akin the ones used to find undue similarities in student submissions.

6.8 - Extent of student cheating and plagiarism in the UK

The most substantial survey on plagiarism and cheating in the UK by Franklyn-Stokes & Newstead aimed to fill the gap in the cheating literature that would allow them to see how students in the UK cheat (Franklyn-Stokes & Newstead 1995). Most, if not all, other cheating literature at that point was US specific. They reported on two separate studies, both conducted before widespread use of the Web. An enhanced study the following year supported the same results (Newstead et al 1996). Lesko also carried out a small survey of postgraduate tutors in Edinburgh to establish whether non-native English speaking students are more likely to plagiarise than their native English speaking counterparts (Lesko 1996).

Table 6.13 shows the admitted results found during the studies. These are formatted in the same way as the tables for US cheating. Table 6.14 shows the believed results by students and staff.

Authors	Date	Location	Type of survey	Size of survey	Main findings
Franklyn-Stokes & Newstead	1995 pre-Web	UK	Students at an old and new university by anonymous questionnaire, by questionnaires.	128	<p>66% of students had paraphrased - reasons given were laziness and peer pressure.</p> <p>65% had agreed to give over generous peer marking - to increase their own mark and to help a friend.</p> <p>65% had altered data - to increase their own mark.</p> <p>63% had copied another student's work with their knowledge</p> <p>9% had copied another student's work without their knowledge</p> <p>9% had submitted work from (pre-Web) essay banks</p> <p>66% had paraphrased</p> <p>25% had collaborated without acknowledgement</p> <p>reasons given for all above</p> <ul style="list-style-type: none"> - to increase their own mark and due to laziness. <p>54% had copied without acknowledgement - because everyone else did the same thing.</p> <p>Students who didn't cheat said it was because it was dishonest or unnecessary.</p>

Newstead, Franklyn- Stokes & Armstead	1996	UK pre- Web	Students (extended/companion questionnaire study above)	943	Cheating more common in men than women. Cheating more common with less able students. Cheating more common with younger students. Cheating more common in science/engineering subjects than art subjects.
					54% had paraphrased without acknowledgement. 48% had invented data. 46% had allowed their work to be copied. 44% had fabricated references. 42% had copied material themselves.

Table 6.13 - Summary of admitted results in UK student cheating literature

Authors	Date	Location	Type of survey	Size of survey	Main findings
Franklyn-Stokes & Newstead	1995 pre-Web	UK	Staff and students at an old and new university - nominal scale questionnaires	132	paraphrasing without references most common cheating activity. copying without references second most common. inventing data third most common. impersonation most serious cheating activity. using exam cribs second most serious. gaining advance information about exam content third most serious.
Franklyn-Stokes & Newstead	1995 pre-Web	UK	Students only at an old and new university - nominal scale questionnaires	112	paraphrasing without references most common cheating activity. copying without references second most common. inventing data third most common.
Franklyn-Stokes & Newstead	1995 pre-Web	UK	Staff only at an old and new university - nominal scale questionnaires	20	allowing coursework to be copied most common cheating activity. paraphrasing second most common. fabricating data third most common.
Lesko	1996	UK	Postgraduate staff at Edinburgh University - by departments questionnaires and interviews	21 staff from 14 departments	All results are descriptive, not quantitative. Finding is that postgraduate non native speakers of English are no more likely to plagiarise than native speakers, but may be spotted easier.

Table 6.14 - Summary of believed results in UK student cheating literature

The most relevant result from the surveys described in Table 6.13 is that around three-quarters of students admitted to activities that could be considered plagiarism. Since the surveys were conducted before the Web was widely used the present results may be higher.

Table 6.14 suggests a difference in opinion between staff and students about how common different cheating activities are. The results from the two studies were said to largely correlate with one another, with the overall view at the time that plagiarism is much less prevalent than cheating in examinations. Franklyn-Stokes and Newstead blamed the cheating on the continued modularisation of courses.

The studies, the most definitive carried out in the UK, were conducted pre-Web. There is little evidence whether the results would hold today. It would therefore be worth re-evaluating student cheating in the present UK academic climate as this might help inform how best to produce tools capable of resisting the problem.

6.9 - Authorship Attribution Techniques

Authorship attribution is a process by which texts can be analysed and the likelihood of them being written by a chosen author deduced. There are many ways in which authorship attribution can be done. Graphologists might analyse handwritten documents looking for characteristic pen strokes. Historians might work on the principle of a document being written in a certain location and at a certain time where the author was known to be resident. These ideas are unlikely to be of any use for automated detection since they are entirely human driven and rely on source documents that are not computer ready.

An alternative approach, based around the idea of stylistics, involves looking at documents for some common word pattern, comparing these with known word patterns from a set of authors and deducing a likelihood each author wrote the questioned document. Given corpora of representative documents from at least two authors the corpora can be used to find writing traits that differentiate between the two authors, for instance one author might overuse the word 'hence' or one author might write in sentences of average length much greater than the other. The suspect document can then be analysed to find which of the author's traits it is closer to. The closer it is to this trait the more likely it is that the given author wrote the document.

There are some obvious and immediate difficulties with such an approach for plagiarism detection. The first is that collecting a corpus of documents written by a selected student is impossible for units early on a course, when detection would be most useful. When a corpus is collected, perhaps by the time of third year units, there is no guarantee that they were all written by the student. The techniques can only give a likelihood that a document was written by a selected author. If the real author of a document is not amongst those that there are corpora for that author cannot be pinpointed and another author within the corpus will be pinpointed, which could lead to a false hit.

Problems also exist when trying to automate the techniques. Although some machine assistance exists it is considered a human-led job to find a trait or set of traits which best differentiates between all possible authors. Once such traits are identified computers can be used to compute authorship likelihoods. There may be ways of computer assisting the search for traits but it will still need trained tutors. When this is coupled with the other problems already identified it looks unlikely that authorship attribution techniques will be appropriate. As such this study is included for information and illustrations about why the techniques are inappropriate. It is also centred around the major historic works, since there is neither space nor motivation to go into any more thorough study within the confines of plagiarism detection.

Table 6.15 lists and describes the main papers that have been identified. It is sorted chronologically. The most relevant will be further compared in later tables. Much of the literature included here represents the traditional authorship studies, so some later work has not been considered, although generally the techniques used have not changed greatly. Many of the papers come out of Doležel and Bailey's collective text (Doležel & Bailey 1969).

Authors	Date	Content
Greg	1927	Proposes a new calculus to express how texts are different to one another that might be relevant. Does not appear to have been used since.
Bailey and Doležel	1968	A then thorough bibliography of stylistics, in both English and other languages, with best techniques noted to be language relevant. The section 'Stylistics and the Computer' is of primary interest.
Antosch	1969	Suggests ratio comparisons that could be used.
Bennett	1969	Discusses use of Yule's measure in stylistics.
Bush	1969	Discusses use of sentence distributions in stylistics.
Carroll	1969	Proposes a multi-dimensional matrix of metrics for use in stylistics.
Doležel	1969	Proposes a method for writing down a probabilistic word structure of a text.
Kroeber	1969	Analyses background tone of different classical novels.
Muller	1969	Discusses use of Waring-Herdan formula in stylistics.
Williams	1969	Considers sentence length as a stylistic measure.
Morton	1978	Book detailing how authorship techniques can be used to prevent fraud, comparing underlying statistical distributions amongst other more techniques.
Mosteller & Wallace	1984	Book containing extensive trial of techniques to discover who wrote which of the Federalist Papers.
Singh & Tweedle	1995	Revisits question of who wrote the Federalist Papers using a neural network aided approach.
Diederich et al	2000	Suggests an authorship attribution technique using support vectors, said to be successful over 60% of the time.

Table 6.15 - Authorship attribution literature identified

The areas of most interest to come out of Table 6.15 are the different comparisons of techniques that exist and the different case studies that they've been tried on. Generally each technique provides a measure or a vector of measures for a document of disputed authorship. These can be compared with measures of known authorship using closeness calculations. Table 6.16 describes the main techniques that have been identified.

Name of technique	Reported in	Details of technique
Active to qualitative ration	Antosch 1969	Measures ratio of words that imply action to words that express properties. Found to be largely insufficient and difficult to apply.
Average sentence length	Bailey 1969 Mosteller & Wallace 1984	Early stylistic method found to be not useful.
Average word length	Bailey 1969	Early stylistic method found to be not useful.
Collocations	Mosteller & Wallace 1984	Consideration of the words following a chosen word, e.g. 'and'. Considered still useful.
Discriminating word pairs	Mosteller & Wallace 1984	Two different words used by two different writers to mean the same thing. Found to be useful on the Federalist papers described in Table 6.17 but labour intensive. Could be considered as a version of marker words.
Discriminating word phrases	Mosteller & Wallace 1984	Suggested but not tested. Could be labour intensive to find.
Distribution of sentence length.	Buch 1969 Williams 1969	Measures number of sentences of length one word, two words, three words etc. Found to only be useful as supporting evidence.
Marker words	Mosteller & Wallace 1984	Words that one writer uses often compared to another writer in samples of 1000 words, say. Found to be useful on the Federalist papers, but very labour intensive to find the words suitably discriminating words, although this could now be computer aided.
Proportion of adjectives used	Mosteller & Wallace 1984	Not found to be useful on the Federalist papers.
Proportion of nouns used	Mosteller & Wallace 1984	Not found to be useful on the Federalist papers.
Proportionate pairs of words	Mosteller & Wallace 1984 Morton 1978	The proportion of time occurrences two related words are used, e.g. 'a' and 'and'. Simple to collect and analyse.

Relative word frequencies	Bailey 1969	The relative frequencies of two words are measured. Difficult to choose suitable words, which might not always exist.
Total number of one and two letter words	Mosteller & Wallace 1984	Not found to be useful on the Federalist papers.
Verb to adjective ratio	Antosch 1969	Found to be largely insufficient.
Vocabulary richness	Bailey 1969	Measure based on the words that are used.
Waring-Herden formula	Muller 1969	Measures number of words occurring once, twice, thrice and so on.
Word position in sentence	Morton 1978	The distribution of the distance a given word, such as 'the' first occurs in a sentence can be a useful stylistic measure.
Word preceding marker words	Morton 1978	The proportion of different words preceding a chosen marker word can be useful.
Words following marker words	Morton 1978	The proportion of different words following a chosen marker word can be useful.
Yule's measure	Bennett 1969	Measures number of nouns occurring once, twice, thrice, and so on.

Table 6.16 - Main authorship attribution techniques identified

The idea of a suitable metric is that it should be unique to a given author and consistent across all of the author's writings. Suitable metrics will differ from author to author. It is also often the case that several different metrics will need to be combined to give sufficient evidence of the likelihood of an author, making automation very difficult.

Of some interest is how the techniques are applied. Some of the techniques, such as the number of occurrences of a chosen word in a text block, will give a series of numerical values. For this Mosteller and Wallace developed new statistical methods based around Bayes' Theorem (Mosteller & Wallace 1984). This was done to give the odds that Hamilton and Madison had written any of the disputed papers. The probability of a given marker word occurring a certain number of times in a 1000 word block of text is computed by modelling as a Poisson, or more accurately but more computationally difficult, a Negative Binomial. Odds can then be calculated for many words and combined with word dependencies factored into the calculation to give more accurate odds.

Morton advocates a similar process of approximation by a chosen distribution (Morton 1978). Morton suggests that the Poisson, Negative Binomial and Negative Exponential are usually the most suitable distributions. The suitability of the fit must be ensured using hypothesis testing using appropriate Normal or t-tests. Properties of the unknown text can then be measured and hypothesis tested to see if it fits the distribution of any of the possible writers. For these calculations texts need to be split into blocks of a known size. Morton suggests that the reciprocal of the percentage of the most commonly occurring word is appropriate (Morton 1978). For example, if 'and' is the most common word and occurs 10% of the time blocks should be ten words long.

Mosteller and Wallace also try an approach using a linear discriminant function, although this is less successful (Mosteller & Wallace 1984). Here a selection of words are chosen that Hamilton and Madison are each more likely to use. Each is weighted based on their likely occurrence in a text by a known author to try and differentiate between the two. Applying these weights on word counts on an unknown text should then give some indication of a more likely author.

Morton also suggests a more visual approach (Morton 1978) plotting properties of a document in a cumulative sum chart. Those stylistic plots that are misplaced then represent areas of interest to a literary detective. Two similar techniques tried for plagiarism detection were described in Chapter 4.

Generally the techniques reported by Morton appear to be the most commonly used (Morton 1978) although they still require substantial human assistance, making them unsuitable for automated plagiarism detection. Although these can now be computer aided there are still problems with acquiring suitable student corpora. Morton stresses throughout his book that it will be necessary to try many methods and develop new ones to find the author of a text with any certainty.

Table 6.17 describes the main case studies that have been carried out and their results.

Name of study	Reported in	Results of study
Carroll	Carroll 1969	Finds defining sets of metrics on a number of texts. These metrics vary from text to text, suggesting a multi-dimensional metric is needed. Finds that humans cannot differentiate between content and style, so only content based metrics should be used.
Federalist Papers	Mosteller & Wallace 1984 Singh & Tweedie 1995	Best known corpora for authorship attribution. 82 Federalist Papers written 1787-1788 described the proposed US constitution for newspapers. 5 were known to be written by John Jay, 43 by Andrew Hamilton, 14 by James Madison, 3 by Hamilton and Madison, 12 unknown, but known to be Hamilton or Madison. Mosteller and Wallace developed new statistical methods to work out the likely authorship. Singh and Tweedie's repeat neural network and computer aided approach is said to reach 98% accuracy instead of the original 94%.
Bible authorship	Mosteller & Wallace 1984	Only referenced, one of the classic problems of authorship, who actually wrote sections of the Bible.

Table 6.17 - Main authorship studies identified

The most complete authorship study is that of the writers of the Federalist papers of unknown authorship. Mosteller and Wallace were first to apply stylistic methods to this problem (Mosteller & Wallace 1984). They compare a number of simple stylistic measures, such as average sentence length, before finding better discriminants. The study finds two usual discriminants, that Hamilton uses 'while' where Madison uses 'whilst' and Hamilton uses the marker words 'enough' and 'upon' often whilst Hamilton barely uses them. The study concludes that Madison wrote all the disputed papers. The discriminants are successfully checked against those papers of known authorship.

The fact that the Federalist papers study requires a whole book to describe is a reminder of how involved the different techniques for authorship attribution are to apply. It might be the case that if a suitable and extensive body of student work can be found that authorship attribution might be useful in the later stages of a student's academic study. But the techniques cannot be directly applied before then. Hence this section is encouraged more for completeness only and as a flavour for possible future work. Metrics used in this thesis will instead look for things that can be more directly measured from two submissions to give a measure of how similar they are. The techniques used in authorship attribution are far more involved than is necessary for this purpose.

6.10 - Conclusions

This chapter has completed the literature review on plagiarism detection. The chapter considers the main areas in which plagiarism literature has been found that did not fit into the scope of Chapters 2 to 4.

The methods used to find literature for Chapters 2 to 4 has been intended to be exhaustive. This is because the areas of technical detection are important for the rest of the thesis and have revealed inconsistencies and problems with existing approaches. The literature search for this chapter has not been so exhaustive, partly due to the continuing growth of materials on the Web and the continuing media interest in plagiarism. It is intended to only the main and most representative pieces of literature. Any more would be overkill for this purpose.

Where completeness is an issue literature has been identified through thorough searching and following up of available references. Bibliographic catalogues have been used as a starting point, but have been supplemented by searches of the British media through online databases, news sources, Web search engines and Internet book stores. There will always be new material that has not yet filtered through this process and there will always be material so far superseded that it is not considered relevant but this process has found the vast majority of plagiarism literature published in recent years. The fast moving nature of technical detection literature means that much of this has come about during the preparation of this thesis.

Chapters 7 to 9 will be used to present support for the four-stage plagiarism detection process. Chapter 7 will describe how pairs of similar submissions can be found. This is the analysis stage. Chapter 8 will describe a visual technique for presenting how two such submissions are similar. Chapter 9 will describe how both can be used together to give a whole efficient and effective process, areas which the existing literature has showed that research is sorely lacking.

Chapter 7: Efficient Finding of Similar Pairs During the Detection Stage

Overview:

- Simple metrics are discussed
- The metrics are shown to differentiate on known similarity
- Plagiarised material manually added to a real corpus can be found

This chapter introduces techniques for finding similarity that may represent plagiarism during the automated analysis phase of the four-stage plagiarism detection process. This is done through defining a set of metrics, known as simple metrics. Some of the simple metrics are shown to be effective in assessing similarity. Further chapters will argue which of the simple metrics is the most effective.

7.1 - Motivation

One of the two main aims of the thesis is to find an efficient automatable plagiarism detection process. The analysis phase of the four-stage plagiarism detection process is important as this is where pairs of submissions are compared and similar pairs need to be flagged. These can then be investigated for potential intra-corporeal plagiarism.

Section 7.2 introduces the idea of simple metrics, which can be used to compare the pairs of student submissions and rank them in order of similarity. The simple metrics work by breaking down a student submission into its component parts, for instance a list of all the words contained in the submission, then combining a number of straightforward counts based on properties of those words. Section 7.3 describes a process through which those simple metrics can be calculated, giving an example for one simple metric and for three miniature documents, two of which represent intra-corporeal plagiarism and the other of which shares similarity but is not plagiarised. The entire set of simple metrics are compared on these miniature documents in Section 7.4 to further illustrate the process and assess if any appear disparate. The associated table added for completeness here is included in Appendix J.

Some of the material introduced in this chapter has been published in the paper 'Towards an Error-Free Detection Process', which is included in Appendix G. In particular Section 7.5 describes a process also reported there where known plagiarism was introduced into a corpus. The rankings list provided was then verified. This shows that the simple metrics can be both effective and efficient in finding similarity.

7.2 - The use of simple metrics

Chapter 2 described how a plagiarism detection process might be operationalised. Notably it presented a process using paired metrics that can be used to find potential intra-corporeal plagiarism within a corpus. This works through a direct comparison of all possible pairs of submissions within a corpus to generate a similarity score for each pair, nominally scaled to be between 0 and 100, although 0.0 to 1.0 could equally be used. Those pairs with the highest similarity scores and hence the highest similarity ranks are then judged to be the most worthy of human investigation.

Chapter 2 did not elaborate on how these similarity scores would be calculated, apart from introducing a further subclass of metrics known as simple metrics and defined as metrics whose values could be calculated from two submissions without intermediate processing. This definition is a little loose but simple metrics are intended to be those that can be computed efficiently, i.e. a corpus of a standard size, say up to 200 submissions, should take mere seconds to process on a fairly average office machine. Hence anything which requires much more than representing a submission in some intermediate form and simply counting commonality with another equivalently represented submission is likely to be too computationally intensive. Finding an efficient plagiarism detection process, hence with minimal computational time, is one of the aims of this thesis and so simple metrics are to be favoured. This chapter will show that many of the tested simple metrics provide a reasonable ordering of similarity and hence anything more complex is unnecessary.

7.3 - Calculating similarity scores using simple metrics

A number of simple metrics have been chosen, based on dividing submissions down into their simple component parts and then comparing those component parts. The metrics will be based around the ideas of word chains, character chains and sentence chains, each of these being an example of a different types of fragments that can be generated from a submission.

The process for generating fragments is simple. A submission is extracted down to all possible fragments of a given fragment size under the chosen metric, with certain punctuation ignored. So word chains, with a fragment size of three, would contain a fragment of words one to three, a fragment of words two to four, a fragment of words three to five and so on.

Since the number of metrics that can be tested is finite the decision has been made to test word chains from one word to nine words in length, all presented in lower case with punctuation removed. Similarly character chains of one character to nine characters and tested. Again these are presented in lower case with punctuation removed, but here spaces are preserved and counted as characters. For sentences only single sentences, i.e. those of fragment length one, are studied. This is because any series of longer sentences would seem superfluous. Here sentence formatting is preserved, including punctuation such as commas, although they are again reduced to lower case. Further leftover spaces at the start of sentences are removed. This gives a total set of nineteen simple metrics.

Table 7.1 shows a short document, denoted D1. This will be used to demonstrate the procedure. Table 7.2 shows how this would be split under the words 4 metric, taking all fragments of four words in length. Table 7.3 shows these fragments sorted into alphabetical order and with a count of the number of times each fragment occurs, intended to avoid repetition.

The cat sat on the mat. It was a black cat. It was a black mat.

Table 7.1 - Sample document D1

the cat sat on
cat sat on the
sat on the mat
on the mat it
the mat it was
mat it was a
it was a black
was a black cat
a black cat it
black cat it was
cat it was a
it was a black
was a black mat

Table 7.2 - Document D1 split into four word fragments

fragment	frequency
a black cat it	1
black cat it was	1
cat it was a	1
cat sat on the	1
it was a black	2
mat it was a	1
on the mat it	1
sat on the mat	1
the cat sat on	1
the mat it was	1
was a black cat	1
was a black mat	1

Table 7.3 - Document D1 in four word fragment representation

Splitting a second document should produce a comparable arrangement of fragments. Splitting a near similar document should produce some fragments at least that are identical. Splitting an unrelated document should give only a small amount of similar fragments that could be attributed to chance.

A formula is needed to compare these any two documents and hence generate a similarity score, to differentiate between similar and unique pairs of documents, based on their fragments. The formula needs to be influenced by fragments of similarity and uninfluenced by fragments unique to either document.

Here four simple counts are needed, counting commonality and uniqueness in both documents:

c1 - for every fragment common to both documents, this is the sum of the counts associated with those fragments in the first document.

c2 - for every fragment common to both documents, this is the sum of counts associated with fragments in the second document.

u1 - for every fragment in the first document but not in the second document, this is the sum associated with those fragments in the first document.

u2 - for every fragment in the second document but not in the first document, this is the sum associated with those fragments in the second document.

These are then combined to give a similarity score using a fairly standard comparison formula, as given below:

$$\frac{100(c1+c2)}{(c1+c2)+(u1+u2)}$$

The formula is intended to find the proportion of common fragments as a proportion of all words present in the two documents and to scale the result between 0 and 100.

Table 7.4 shows a second document, denoted D2, that has been produced as a plagiarised version of document D1. It contains some attempts at disguise, notably re-ordering, refragmentation and thesaurising. Table 7.5 shows the document broken down into four word fragments and sorted.

The feline sat and purred on the black mat. It was a black cat.

Table 7.4 - Sample document D2

fragment	frequency
and purred on the	1
black mat it was	1
feline sat and purred	1
it was a black	1
mat it was a	1
on the black mat	1
purred on the black	1
sat and purred on	1
the black mat it	1
the feline sat and	1
was a black cat	1

Table 7.5 - Document D2 in four word fragment representation

Table 7.6 demonstrates how a similarity score for documents D1 and D2 could be calculated under the words 4 metric.

fragment	frequency in D1	frequency in D2	contributes to			
			c1	c2	u1	u2
a black cat it	1	0	0	0	1	0
and purred on the	0	1	0	0	0	1
black cat it was	1	0	0	0	1	0
black mat it was	0	1	0	0	0	1
cat it was a	1	0	0	0	1	0
cat sat on the	1	0	0	0	1	0
feline sat and purred	0	1	0	0	0	1
it was a black	2	1	2	1	0	0
mat it was a	1	1	1	1	0	0
on the black mat	0	1	0	0	0	1
on the mat it	1	0	0	0	1	0
purred on the black	0	1	0	0	0	1
sat and purred on	0	1	0	0	0	1
sat on the mat	1	0	0	0	1	0
the black mat it	0	1	0	0	0	1
the cat sat on	1	0	0	0	1	0
the feline sat and	0	1	0	0	0	1
the mat it was	1	0	0	0	1	0
was a black cat	1	1	1	1	0	0
was a black mat	1	0	0	0	1	0
			c1 total	c2 total	u1 total	u2 total
			4	3	9	8

Table 7.6 - Calculating similarity between D1 and D2

This gives a similarity score for documents D1 and D2, under the words 4 metric of:

$$\frac{100(4+3)}{(4+3)+(9+8)} = 29.167$$

There are only three fragments that are common to both documents and so these are the only things that contribute towards a high similarity score. It is hoped that this score would rank the pair of documents near to the top of a similarity rank list so that they would be investigated further by tutors.

One test is to introduce a third document D3, part of the same corpus, which contains a little similarity to D1 and D2, but in which the similarity could be considered merely coincidental. Such a document is shown in Table 7.7.

Looks like raining cats and dogs, purred the feline. It was a black sky outside.

Table 7.7 - Sample document D3

Table 7.8 shows document D3 split into four word fragments.

Table 7.9 shows the calculation of the similarity score for document D1 against D3. Table 7.10 shows the calculation for D2 against D3.

fragment	frequency
a black sky outside	1
and dogs purred the	1
cats and dogs purred	1
dogs purred the feline	1
feline it was a	1
it was a black	1
like raining cats and	1
looks like raining cats	1
purred the feline it	1
raining cats and dogs	1
the feline it was	1
was a black sky	1

Table 7.8 - Document D3 in four word fragment representation

fragment	frequency in D1	frequency in D3	contributes to			
			c1	c2	u1	u2
a black cat it	1	0	0	0	1	0
a black sky outside	0	1	0	0	0	1
and dogs purred the	0	1	0	0	0	1
black cat it was	1	0	0	0	1	0
cat it was a	1	0	0	0	1	0
cat sat on the	1	0	0	0	1	0
cats and dogs purred	0	1	0	0	0	1
dogs purred the	0	1	0	0	0	1
feline it was a	0	1	0	0	0	1
it was a black	2	1	2	1	0	0
like raining cats and	0	1	0	0	0	1
looks like raining	0	1	0	0	0	1
cats						
mat it was a	1	0	0	0	1	0
on the mat it	1	0	0	0	1	0
purred the feline it	0	1	0	0	0	1
raining cats and	0	1	0	0	0	1
dogs						
sat on the mat	1	0	0	0	1	0
the cat sat on	1	0	0	0	1	0
the feline it was	0	1	0	0	0	1
the mat it was	1	0	0	0	1	0
was a black cat	1	0	0	0	1	0
was a black mat	1	0	0	0	1	0
was a black sky	0	1	0	0	0	1
			c1 total	c2 total	u1 total	u2 total
			2	1	11	11

Table 7.9 - Calculating similarity between D1 and D3

fragment		frequency in D2	frequency in D3	c1	c2	u1	u2	contributes to	
a black sky outside		0	1	0	0	0	1		
and dogs purred the		0	1	0	0	0	1		
and purred on the		1	0	0	0	1	0		
black mat it was		1	0	0	0	1	0		
cats and dogs purred		0	1	0	0	0	1		
dogs purred the feline		0	1	0	0	0	1		
feline it was a		0	1	0	0	0	1		
feline sat and purred		1	0	0	0	1	0		
it was a black		1	1	1	1	0	0		
like raining cats and		0	1	0	0	0	1		
looks like raining cats		0	1	0	0	0	1		
mat it was a		1	0	0	0	1	0		
on the black mat		1	0	0	0	1	0		
purred on the black		1	0	0	0	1	0		
purred the feline it		0	1	0	0	0	1		
raining cats and dogs		0	1	0	0	0	1		
sat and purred on		1	0	0	0	1	0		
the black mat it		1	0	0	0	1	0		
the feline it was		0	1	0	0	0	1		
the feline sat and		1	0	0	0	1	0		
was a black cat		1	0	0	0	1	0		
was a black sky		0	1	0	0	0	1		
						c1 total	c2 total	u1 total	u2 total
						1	1	10	11

Table 7.10 - Calculating similarity between D2 and D3

The ordering of the pairs is much more interesting than the actual similarity scores returned, since this reflects much better the process that is believed to be used by a tutor investigating similarity. Table 7.11 shows the similarity scores and similarity ranks formed from the three pairs of documents D1, D2 and D3 under the words 4 simple metric.

Document 1	Document 2	c1	c2	u1	u2	similarity score	similarity rank
D1	D2	4	3	9	8	29.167	1
D1	D3	2	1	11	11	12.000	2
D2	D3	1	1	10	11	8.696	3

Table 7.11 - Similarity scores for pairs of D1, D2 and D3

The results are promising. Documents D1 and D2, the pair containing known plagiarism, is ranked highest under the words 4 metric. The original document D3 is ranked in the bottom portion of the list against both submissions it contains nothing intentionally in common with. Further the similarity score in rank 1 is much higher than that in rank 2 or 3, suggesting that this might be a suitable cut off point for investigating similarity. Hence the simple words 4 metric has been shown to be a reasonable differentiator off similarity on these three documents.

Clearly these are really only illustrative examples, since a real corpus will contain more than three submissions and each will be longer than fifteen or so words long. But assuming the technique is scable the simple metrics look like they could be effective for finding similarity and the metrics have been so chosen to be efficient, two of the aims of the thesis.

7.4 - Comparing the effectiveness of simple metrics on small document collection

Section 7.3 demonstrated how the similarity rank list could be calculated for three small documents under the simple metric words 4. On that corpus the document was demonstrated to be fairly effective, although there is little to compare it with, apart from the known property that documents D1 and D2 were known to be plagiarised.

The remaining identified simple metrics, words 1 to words 9, characters 1 to characters 9 and sentences, denoting fragments of the appropriate size, might also be appropriate.

Appendix J contains shows the different word fragments that would be generated for each of the different documents D1, D2 and D3, for use in each of the identified simple metrics. In the table spaces are represented by underscores (_). The frequency for each fragment in each document is also shown. The table is included in the appendix so as to not break up the flow of the text.

One feature that is immediately obvious from looking at Appendix J is that there is a lot of shared commonality in the character fragments of low size, whilst there is a complete lack of similarity between sentences. This suggests that these might not give effective results.

Table 7.12 shows the similarity scores generated for each pair of documents D1, D2 and D3 under each of the simple metrics.

Metric	Pair of documents		
	D1 D2	D1 D3	D2 D3
characters 1	94.215	88.321	97.101
characters 2	86.207	59.542	75.94
characters 3	77.477	45.6	59.375
characters 4	68.868	39.496	51.22
characters 5	60.396	33.628	43.22
characters 6	55.208	30.841	35.398
characters 7	51.648	29.703	30.556
characters 8	47.674	28.421	25.243
characters 9	41.975	26.966	20.408
sentences	40	0	0
words 1	90	48.387	62.069
words 2	60.714	31.034	29.63
words 3	38.462	22.222	16
words 4	29.167	12	8.696
words 5	18.182	0	0
words 6	10	0	0
words 7	0	0	0
words 8	0	0	0
words 9	0	0	0

Table 7.12 - Similarity scores for pairs of D1, D2 and D3 under simple metrics

Although the similarity scores are interesting the values do not make comparisons immediately obvious. Pair D1 and D2 is expected to be the highest rank pair, ideally with a substantial margin above the other similarity scores to minimise the likelihood of incorrectly ranked pairs. The clearest differentiators are probably word5 and word6 and sentences, giving a positive value for D1 and D2, against a zero score for the other pairs. Some other combinations more than double the similarity score for D1 and D2 compared to the other two pairs. In these cases word7, word8 and word9 do not look like good metrics, since they cannot differentiate between the pairs at all. This is because there are no runs of similarity of seven words or more in the documents, perhaps due to their small size.

A more intuitive way of looking at things is through the rank values. Table 7.13 shows the similarity ranks for the pairs of D1, D2 and D3 under the different metrics.

Metric	Pair of documents		
	D1 D2	D1 D3	D2 D3
characters 1	2	3	1
characters 2	1	3	2
characters 3	1	3	2
characters 4	1	3	2
characters 5	1	3	2
characters 6	1	3	2
characters 7	1	3	2
characters 8	1	2	3
characters 9	1	2	3
sentences	1	2	2
words 1	1	3	2
words 2	1	2	3
words 3	1	2	3
words 4	1	2	3
words 5	1	2	2
words 6	1	2	2
words 7	1	1	1
words 8	1	1	1
words 9	1	1	1

Table 7.13 - Similarity ranks for pairs of D1, D2 and D3 under simple metrics

Again it seems like most of the metrics successfully differentiate between the pair containing intentional plagiarism and the pairs where the similarity is just coincidental. This suggests that the simple metrics should be appropriate on the larger corpora, although no further attempt will be made to analyse them here since the corpora used is clearly very different to a real one, both in terms of size and in complexity. It does suggest that the process used is suitable for further evaluation.

One further consideration is that of preparing a consolidated similarity rank, a single rank based on several of the ranked similarity metrics. This should not increase the processing time complexity but might provide a better overall ordering, or hide potential errant metrics. The consolidated similarity rank has been produced by combining the ranks of the simple metrics for each pair using a non-weighted linear function, then reranking. Table 7.14 shows the consolidated similarity rank for D1, D2 and D3 based on the nineteen simple metrics tested.

	Pair of documents		
	D1 D2	D1 D3	D2 D3
consolidated similarity rank	1	3	2

Table 7.14 - Consolidated similarity ranks for pairs of D1, D2 and D3

In this case the consolidated similarity rank ordering is the same to the modal ordering of the simple metrics, but again successfully ranks the D1 and D2 pair first, which is another good sign.

7.5 - Introducing plagiarism into a real corpus

The limited examples so far have shown that simple metrics are appropriate for finding some similarity in small corpora of documents although their exact uses are not known.

A corpus of 125 student submissions was collected from a computing unit at South Bank University and used for a more thorough test on the simple metrics. Such a corpus gave a total of 7750 pairs, an amount that could not feasibly be checked by hand, again demonstrating the usefulness of an automated method of ranking them. Some initial tests done on the corpus using the consolidated similarity rank formed by combining the 19 simple metrics suggested that the corpus already contained some plagiarism. This included one blatant pair containing extensive intra-corporeal plagiarism and a few other pairs containing some more fragmented pieces of similarity.

The highly ranked pairs were manually verified to show that they did not contain any false hits, that is pairs of submissions ranked highly that contain little similarity. Such initial results were promising, since this meant the simple metrics were finding similarity, even with no attempt to fine tune them. However checking the remaining pairs to show that there were no missed pairs, that is pairs containing similarity that were not high in the consolidated ranking list, was not feasible.

An alternative, less exhaustive, approach was to introduce plagiarised submissions into the corpus and demonstrate that the resulting plagiarised pairs had sufficiently high consolidated similarity ranks for further inspection. This is similar to the methods tutors are believed to use to investigate similarity, starting at the top of the list and working down it to check potentially plagiarised pairs as far as resources allow or it seems to be worth going. As such the plagiarised pairs could be expected to be seen somewhere in the upper portion of the list, which would give some level of assurance that there were no missed pairs.

Four tutors, each with previous experience of how students plagiarise, were given a different source submission and asked to plagiarise in the manner of a student with an impending deadline. This was defined as not writing the assignment from scratch but changing it in such a way that it could fool a human marker. There is no evidence that students change their plagiarising styles when it is going to be checked by a machine. The submissions chosen from the corpus were around the median length of 2234 words, but not verified for content or quality. Tutors were asked to record the methods they used to plagiarise in plain English along with the time taken plagiarising. This has been used to create the table shown in Table 7.15.

	Tutor 1	Tutor 2	Tutor 3	Tutor 4
Changed formatting	√	√	√	√
Changed headings	√	√	√	√
Changed header/footer	√	√	√	√
Rephrased sentences	√	√	√	√
Changed references		√		√
Added filler phrases				√
Search and replace on key words				√
Reordered sections		√		
Removed sections		√	√	
Time taken	30 min	45 min	75 min	90 min

Table 7.15 - Plagiarism techniques used

As shown the time taken varied between 30 minutes and 90 minutes, with the main techniques used being changing the formatting, changing the headings, changing the header and footer and rephrasing sentences. Intuition suggests that changing the formatting might be enough to fool some tutors as the page layout looks different, but should not affect an automated analysis, since the submissions are converted into plain text for processing. The more extensive attempts to plagiarise, for instance by re-arranging the submission, presumably in such a way that it still reads well enough to make sense, appear to have taken the longest. It is interesting that no tutors appear to have simply rewritten the opening section of the essay, since observations suggest that changing an opening section a submission can make it appear more different, although this is something than requires further investigation.

The simple metrics and consolidated metric were generated for the extended corpus, now containing 129 submissions and 8256 pairs. Table 7.16 shows the similarity rank generated for each submission plagiarised by a tutor, against its known source submission, for each of the simple metrics. The ranks represent the placement of the pair within the 8256 pairs possible.

	Tutor 1	Tutor 2	Tutor 3	Tutor 4
characters 1	322	175	3523	4596
characters 2	3	1	2381	4
characters 3	24	1	3	9
characters 4	6	1	3	7
characters 5	4	1	3	6
characters 6	4	1	2	5
characters 7	4	1	2	5
characters 8	4	1	2	5
characters 9	4	1	2	5
sentences	2	1	28	32
words 1	4	1	3	5
words 2	4	1	2	5
words 3	4	1	2	8
words 4	4	1	2	12
words 5	4	1	2	23
words 6	4	1	3	38
words 7	4	1	3	43
words 8	4	1	3	49
words 9	4	1	3	47

Table 7.16 - Simple metric ranks for source/copy pairs containing known plagiarism

In general these results look very promising. In particular the words 1 and words 2, along with characters 6 to characters 9 all rank the plagiarised submission in the top five pairs. In these cases the remaining pair was the previously checked pair of actual submissions known to be plagiarised. This suggests that under these metrics such student plagiarism would almost certainly be found with automated assistance. It is also interesting to note that character chains of between six and ten characters are probably around one or two words in length, suggesting that the correlation is not unexpected.

By contrast most of the other metrics look reasonable. Characters 1 gives completely random looking output, which is probably not surprising, since most submissions will contain most of the possible characters at least once. Characters 2 looks much better for the most part, except that tutor 3's plagiarism would not be found. This is in contrast to the majority of the metrics, where tutor 4 is the most likely to get away with plagiarism. Although tutor 4 does not slip outside the top 50 the tutor falls a long way behind the other three tutors, who for the most part are ranked in the top three. Tutor 4's plagiarism is clearly the most involved based on Table 7.15, having taken 90 minutes and involved adding new phrases and search and replacing key words. These latter changes might have been enough to fool the metrics using the larger chains of words, having successfully broken into them, meaning that this plagiarism may go undetected. This suggests that the shorter word chains or the character metrics may be more appropriate.

Table 7.17 shows the consolidated similarity rank for each of the pairs out of the 8256 pairs possible.

	Tutor 1	Tutor 2	Tutor 3	Tutor 4
Consolidated	3	2	32	15

Table 7.17 - Consolidated metric ranks for source/copy pairs containing known plagiarism

Surprisingly the consolidated metric places tutor 3 furthest down the list at position number 32. This might be far enough down that it would not be detected by a user, although tutor 1, tutor 2 and tutor 4 would almost certainly be investigated, particularly if tool support were available. This suggests that a better consolidated ranking list might be available, perhaps by removing the disparate metrics. One possibility might be by using only the top ten ranks associated with a given submission, hence whatever metrics are disparate for a given pair would not influence the consolidated rank. This might also be effective if different metrics prove to be effective at detecting different types of plagiarism.

Generally the results are promising, showing that the metrics can successfully find known plagiarism when it is added to a corpus, although some metrics are better than others. Chapter 9 will argue that a single simple metric is best, based on criteria introduced in Chapter 8. Chapter 8 will also contain a further look at the four submissions plagiarised by tutors as part of the verification and investigation stages of the four-stage process.

7.6 - Conclusions

This chapter has introduced the idea of simple metrics. These are useful because they can be computed efficiently and the initial tests show that they seem to rank material containing known plagiarism high in a similarity rank list, so that it is suitable for further investigation. The machine process through which pairs of submissions can be ranked comes as part of the analysis stage of the four-stage plagiarism detection process.

Chapter 8 will look into how the detection process can be made more effective, the other main aim of the thesis. This involves improving the verification and analysis stages of the four-stage detection process through the means of visually assisted tool support.

Although the simple metrics used here have been seen to be reasonable there is little evidence as to how good they are, or which, if any, is the best. Chapter 9 will compare the different simple metrics, based around the idea that the approach presented in Chapter 8 is suitable for human use and so a ranking method that feeds into this would be the most suitable, while meeting the efficiency requirement. Although a better consolidated ranking list might be one way to find this, an entirely unrelated approach, perhaps using only a single metric, might be more appropriate.

One question that has come out of Section 7.5 is that although tutors believe they can plagiarise like a student they do not always appear to do so very effectively. This could mean that most student plagiarism is getting detected. Or it could mean that the methods used by tutors were not very realistic. It seems there is a need to know how students plagiarise in order to ensure that the metrics used are appropriate and to see if this can inform the next generation of detection engines.

Chapter 8: Effective Investigation of Similarity

Overview:

- Metrics applied at fragmentary granularity are applied.
- Fragmentary intercept matrices are generated.
- Similarity visualisations and their properties are discussed.

This chapter describes a new visual method through which pairs of documents, primarily student submissions, can be examined. The similarity visualisation, produced from an underlying two dimensional matrix, shows both where the submissions are similar and how similar they are at those points. This is done through a system of colour coding.

The process of producing the matrix is described in detail, using simple metrics applied at fragmentary granularity, that is on overlapping sections from the two documents, to produce numeric similarity scores for each cell of the matrix. The similarity visualisation can then be directly extracted from this matrix based on a numeric correspondance of cell values to different pixel intensities in the visualisation.

The visualisation will later be linked with a similarity exploration tool known as VAST, shown to reduce the time taken to investigate similarity in student submissions.

8.1 - Motivation

It seems fairly intuitive that a list showing which submissions are most similar to which other submissions is useful, but can lead to a long drawn out process of tutor involvement. Tutors have to look at each pair of submissions in turn, identify where they are similar and how similar they are at those points, before deciding if they are worth pursuing. If so, the evidence needs to be documented as cases of litigation are becoming increasingly common within the UK. It is possible that a tutor might easily miss similarity when it exists on different pages within submissions, or when some attempts to change the formatting of submissions have been made.

This chapter describes a new visualisation method that can be used to see if two submissions contain any similarity, where they are similar and how similar they are at those points. The similarity visualisation generated can be linked with other tools to allow quick examination by tutors of suspicious pairs of submissions.

Much of the original work here has been published and sections of the material given here are closely based on that published work. In particular Sections 8.3 to 8.7 are closely related to material published in 'Visualising Intra-Corporeal Plagiarism', which is included in Appendix H. Section 8.8 contains results that were previously discussed in 'Towards an Error-Free Plagiarism Detection Process', included in Appendix G.

8.2 - Identifying areas of similarity within two documents

By its very nature plagiarism detection is an onerous task for tutors involving the manual inspection of large numbers of student submissions. An effective ordering of similarity can help, providing a number of pairs of submissions that need to be checked further based on their similarity score or position in an ordered similarity list. A process for generating such a list was described in Chapter 7. But the manual process of checking those submissions for similar fragments of text is still time consuming and sections of similarity could be easily be missed.

It would be much more beneficial for tutors if they could see exactly where two sections of submissions were similar and how similar they are those points. Some systems have recently come into existence, such as the Web-based plagiarism detection services, that present hyperlinks next to sections of texts that are believed to be similar. But they do not give any indication of how similar they are at such points. This instead is a matter for a tutor to manually assess.

A document can be considered as a number of shorter fragments, each containing an equal number of words. Then an immediate re-application of simple metrics is possible on each of the fragments in two submissions that are believed to contain similarity. The metrics could be said to be applied at *fragmentary granularity* as opposed to the gross granularity at which the regular similarity measurement engines work. Hence such an application could assess which sections of which submissions are most similar to which sections of which other submissions, reducing the number of areas that a tutor has to scan through.

This approach is not yet watertight. Consider the two representative 20 word documents A and B given in Table 8.1. The bolded and underlined regions show possible sections of similarity of two words or more.

Document A	Document B
aa bb cc dd ee ff gg hh ii jj kk ll mm nn oo pp qq rr ss tt.	aa bb cc dd uu gg hh ii vv ww xx xx yy yy aa zz ee rr ss qq.

Table 8.1 - Two document showing possible areas of similarity

The comparison shows a four word run of exact similarity at the beginning of both documents, a three word run of exact similarity inside both documents and a three word run of rearranged similarity at the end of both documents.

Table 8.2 shows documents A and B broken down into five fragments, each of four words. Any fragments or part fragments of two or more words common to both documents are bolded and underlined.

	Document A	Document B
Fragment 1	aa bb cc dd	aa bb cc dd
Fragment 2	ee ff gg hh	uu gg hh ii
Fragment 3	ii jj kk ll	vv ww xx xx
Fragment 4	mm nn oo pp	yy yy aa zz
Fragment 5	qq rr ss tt	ee rr ss qq

Table 8.2 - Two documents showing similarity in fragments

Table 8.3 shows the number (and percentage) of words in common, ignoring the ordering, to fragments from the documents. The non-zero cells are bolded for emphasis.

		Doc. A				
		Frag. 1	Frag. 2	Frag. 3	Frag. 4	Frag. 5
Doc. B	Frag. 1	4 (100%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)
	Frag. 2	0 (0%)	2 (50%)	1 (25%)	0 (0%)	0 (0%)
	Frag. 3	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)
	Frag. 4	1 (25%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)
	Frag. 5	0 (0%)	1 (25%)	0 (0%)	0 (0%)	3 (75%)

Table 8.3 - Two documents, showing word proportions fragments have in common

The table demonstrates areas of similarity as these are the ones with high valued cells. In particular the 4 shows that fragment 1 of document A exactly matches fragment 2 of document B, giving an area of similarity that should require further investigation. The third run of similarity is also successfully identified, giving a high 75% match between fragment 5 of both documents.

Perhaps the only area for concern is the second run of similarity, which is concealed, appearing in both fragment 2 and fragment 3 of document A, giving only a 50% and 25% match with fragment 2 of document B respectively, something that might be dismissed as coincidental.

There is a way around this problem that would catch the entire second run of similarity as a 75% (3 word) fragment. That would be to split documents A and B into every possible fragment of four consecutive words, giving a large number of overlapping fragments.

Table 8.4 shows the new fragments that would be generated from both documents. Again fragments or part fragments of two or more words common to both documents are bolded and underlined, with the overlapping four word fragments meaning that common areas will appear more than once.

	Document A	Document B
Fragment 1	aa bb cc dd	aa bb cc dd
Fragment 2	bb cc dd ee	bb cc dd uu
Fragment 3	cc dd ee ff	cc dd uu gg
Fragment 4	dd ee ff gg	dd uu gg hh
Fragment 5	ee ff gg hh	uu gg hh ii
Fragment 6	ff gg hh ii	gg hh ii vv
Fragment 7	gg hh ii jj	hh ii vv ww
Fragment 8	hh ii jj kk	ii vv ww vv
Fragment 9	ii jj kk ll	vv ww xx xx
Fragment 10	jj kk ll mm	ww xx xx yy
Fragment 11	kk ll mm nn	xx xx yy yy
Fragment 12	ll mm nn oo	xx yy yy aa
Fragment 13	mm nn oo pp	yy yy aa zz
Fragment 14	nn oo pp qq	yy aa zz ee
Fragment 15	oo pp qq rr	aa zz ee rr
Fragment 16	pp qq rr ss	zz ee rr ss
Fragment 17	qq rr ss tt	ee rr ss qq

Table 8.4 - All possible fragments of two documents

Table 8.5 shows the number of words in common for these different fragments, with the word ordering ignored. Table 8.6 shows the equivalent percentage commonality information. Cells with non-zero values are shown in bold in both cases.

		Doc. A Frag. #																
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Doc. B Frag. #	1	4	3	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	2	3	3	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	3	2	2	2	1	0	1	1	0	0	0	0	0	0	0	0	0	0
	4	1	1	1	2	2	2	2	1	0	0	0	0	0	0	0	0	0
	5	0	0	0	1	2	3	3	2	1	0	0	0	0	0	0	0	0
	6	0	0	0	1	2	3	3	2	1	0	0	0	0	0	0	0	0
	7	0	0	0	0	1	2	2	1	1	0	0	0	0	0	0	0	0
	8	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	12	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	13	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	14	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
	15	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1
	16	0	1	1	1	1	0	0	0	0	0	0	0	0	1	2	2	2
	17	0	1	1	1	1	1	0	0	0	0	0	0	0	1	2	3	3

Table 8.5 - Word count in common for all possible fragment pairings of two documents

		Doc. A Frag. #																
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Doc. B Frag. #	1	100	75	50	25	0	0	0	0	0	0	0	0	0	0	0	0	
	2	75	75	50	25	0	0	0	0	0	0	0	0	0	0	0	0	
	3	50	50	50	25	0	25	25	0	0	0	0	0	0	0	0	0	
	4	25	25	25	50	50	50	25	0	0	0	0	0	0	0	0	0	
	5	0	0	0	25	50	75	75	50	25	0	0	0	0	0	0	0	
	6	0	0	0	25	50	75	75	50	25	0	0	0	0	0	0	0	
	7	0	0	0	0	25	50	50	25	25	0	0	0	0	0	0	0	
	8	0	0	0	0	0	25	25	25	25	0	0	0	0	0	0	0	
	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	12	25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	13	25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	14	25	25	25	25	25	0	0	0	0	0	0	0	0	0	0	0	
	15	25	25	25	25	25	0	0	0	0	0	0	0	0	0	25	25	
	16	0	25	25	25	25	0	0	0	0	0	0	0	0	0	25	50	
	17	0	25	25	25	25	0	0	0	0	0	0	0	0	0	25	50	

Table 8.6 - Percentage in common for all possible fragment pairings of two documents

The tables show three isolated sections of similarity, a section of similarity being an area formed of non-zero cells with a complete surrounding of zero cells. Consecutive cells have a difference of at most one word of similarity (or 25%) giving graduated levels of change. These sections tally somewhat with the areas that a tutor would want to investigate to see if they represent plagiarism or not.

The top left area in fact represents both the first two runs of similarity as they merge into one another. The 100% similarity cell at the top left shows that the start of both documents A and B is identical. The block of four 75% cells capture the second run of identical words, eliminating the problem from Table 3 when only a maximum of 50% similarity was shown.

The bottom left area of similarity represents only noise and is the result of word 'aa' appearing in several positions in document B. Since this area contains only a maximum of 25% similarity it would likely be dismissed quickly by an investigator.

The bottom right area represents the final run of similarity, giving a maximum of 75% similarity in the final two cells.

Hence this table method appears to be a successful one for finding runs of similarity within two submissions, presenting them as areas of similarity within a table. Although some noise is evident this is likely small enough to not cause a problem.

8.3 - The Fragmentary Intercept Matrix

The previous section showed a method by which sections of similarity within two documents could be identified as part of the verification and investigation stages of the four-stage plagiarism detection process. Table 8.6, giving a percentage measure of similarity within different fragments of two documents can be considered to be a *fragmentary intercept matrix* immediately useful for identifying the fragments of two documents that are most similar. The value of each cell also identifies how similar they are at those points.

The example given in Section 8.2 is close to an approach that could be used in practice. The main difference with practical documents is that they are larger than the sample documents used. In order to more accurately identify long runs of similarity the fragments would be longer than four words in length. Such larger fragments would also make the effects of noise would be more visible throughout the matrix, meaning that only a complete cut off between zeroed and non-zeroed areas would be impossible. Instead a tutor would need to consider which areas were most significant and hence most important to look at further.

This section formalises the approach through which a fragmentary intercept matrix can be generated, including specifying the associated terminology more closely.

For comparison at fragmentary granularity a submission is seen not only as a whole, but also as a large number of overlapping fragments of continuous words. Consider a student submission. The first *fragment* is obtained from it simply by taking words from the start, up to a given *fragment length* (the example in Table 8.4 used a fragment length of 4). A second fragment could be obtained by taking the student submission, removing the first word and then repeating the fragmenting process. Words next to one another in a fragment were always next to one another in the original student submission.

In most cases it is not necessary to take every possible fragment from a student submission, since this would give too much detail for an easily readable matrix. Instead only some of the fragments are taken; for consistency purposes, these must have the same spacing between them and cover the entirety of the student submission. For this a *fragment gap* is needed, this is the value denoting how many words in the student submission there are between the first word in subsequent fragments. For a fragment gap of one, every possible fragment would be taken from the student submission. For a fragment gap of two, every alternate fragment would be taken. For a fragment gap of three, every third fragment would be taken and so on.

A representation of each fragment from the first student submission can now be compared with a representation of each fragment from the second student submission. For accurate comparison the second document is extracted using the same fragment size and gap size as the first. This comparison of two fragments gives a *fragmentary interception*, a representation of the contents of both fragments, providing for it a *fragmentary interception similarity score*, a similarity score localised to the fragmentary interception, scaled between 0 and 100, with 0 representing no similarity and 100 representing identity. A tutor then now only needs to concentrate human

examination on those fragments which, when compared, provide a higher fragmentary interception similarity score than would usually be expected.

It would be very rare to get a fragmentary interception similarity score of zero. This is due to the background effects of *noise*. Consider a simple *word count metric* that simply counts the number of words common to a fragmentary interception. There is a certain background distribution to student writing that means that certain words will appear over and over again, regardless of whom is writing. In English, these are usually *function words*, the filler words of the language, such as ‘a’, ‘by’, ‘from’ or ‘to’. The most common word in the English language is ‘the’, which alone usually accounts for around 10% of any fragment, and hence a substantial part of any noise from a fragmentary interception similarity score.

Every possible fragmentary interception similarity score can be calculated using a systematic approach and placed in a table, called a *fragmentary intercept similarity score matrix*. Those high valued areas within the matrix are then the areas that a tutor should investigate further for either possible plagiarism or legitimate common citations.

8.4 - Similarity Visualisations

Not every one finds a large matrix of numbers an easy thing to search through. To reduce this problem there has been a lot of work done on presenting such a matrix in a more eye-friendly form. Just as the reduction of two submissions into a matrix showing where they are similar is an improvement for any busy tutor, so is presenting that matrix in colour coded way.

The table can easily be presented as an image, where colour information can be used to display the extent of similarity within fragments. Each cell within the fragmentary intercept similarity score matrix can be mapped onto a grayscale value, with a fragmentary similarity score of 0 corresponding to white and a score of 100 corresponding to black. These grayscale values can be plotted onto pixels onto a graphic known as a *similarity visualisation*. This form sees an immediate contraction of the volume of data that a tutor needs to look through, since a single pixel is much smaller than the corresponding numerical equivalent. The joining together of many pixels makes some areas prominent. Such areas are known as *similarity intersections*.

Figure 8.1 shows a first example of a similarity visualisation. The visualisation has been taken from a set of student projects that were thought to be plagiarised. One chapter of the project is plotted down the y-axis. A number of possible source Web pages were taken and concatenated together and these are plotted along the x-axis. The visualisation has been produced with a fragment size of 200 words and a fragment gap of ten words. These values were determined after experimentation since they appeared to give a genuinely useful image that was fairly independent of the sources. The fragmentary metric used to generate the visualisation is not important at this stage.

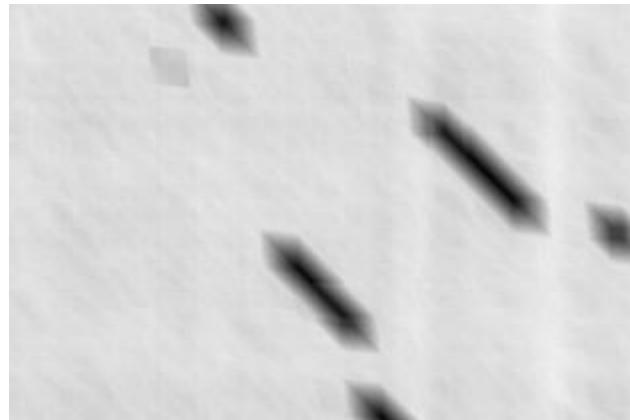


Figure 8.1 - First example of a similarity visualisation

The features of visualisations already discussed are immediately obvious here. The intense areas represent similarity and the areas that a tutor should verify and investigate can be found linked to these areas. The blackness of the areas represents identical parts of the documents at those points. The grey background represents the effects of noise. The intense areas, taken as a whole set of pixels, are each of the similarity intersections. This visualisation represents a much more clear cut example of plagiarism than would commonly be seen in practice and as such it has been chosen as a first example, but will not be discussed further here.

8.5 - Similarity Intersections

It is important to understand the different properties of similarity visualisations so that they can more easily be used to discuss similarity. The most significant areas of similarity visualisations that are important are the similarity intersections. These show three important pieces of information: the relative location of similarity, the length of the similarity and the extent of the similarity. The location can be found by measuring the start and points of the intersection, on both axes and using this as a relative point proportionally through both documents. For practical use this should be tool supported. The intense pixels represent overlapping areas but the distance between the first and last intense pixel shows the proportion of the document that is plagiarised. The intensity of the area also represents how broken up the similarity is. A black pixel will represent an area that is identical and this will be surrounded by lighter pixels, caused by the overlapping fragments. A lighter area represents either a shorter run of similarity, for instance a similarity length less than the fragment size, or a more broken up section of similarity, perhaps interspersed by a student's own words.

The properties of similarity visualisations can be explored by considering some synthetic documents. Figure 8.2 shows the similarity visualisation generated by comparing two synthetic documents. The synthetic documents are each 2000 words long and contain no similarity, apart from a 200 word stretch in the middle of both documents, where every word is common to both. This represents a short section of text copied from one document to the other. The uniqueness of the rest of the documents eliminates the problem of noise, since the usual responsible background word distribution is not preserved by the synthetic documents. The similarity visualisation has been computed by using a fragment size of 200 words and a fragment gap of 10 words, to create an image 181 pixels by 181 pixels. The similarity visualisation is shown both at its usual size and zoomed in on at the similarity intersection.

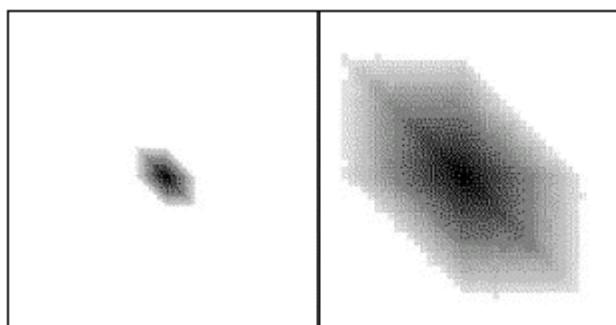


Figure 8.2 - A synthetic similarity visualisations with 200 words of similarity

The similarity intersection features an intensely black area in the centre, of a single pixel, where an exact match of 200 words is found. Radiating outwards are ever-lighter shades, until the edge pixels of the hexagon, which are imaged from fragmentary interceptions sharing only ten words in common. These correspond to the graduated number changes within the fragmentary intercept matrix.

The positioning of the similarity within the documents is also important. In the similarity visualisation, the first document is positioned along the x-axis and the second document is positioned along the y-axis. The origin point for both is in the top-left corner. The similarity intersection is situated approximately half way through the first document and half way through the second document two. It is also situated on the *primary diagonal*, the diagonal running from the origin, at the top-left, to the bottom-right.

The visualisation cannot tell us which document represents the source of the plagiarism. The similarities might be through the writer of the first document copying from the second, or vice-versa. Alternatively the writers might have collaborated on both submissions, or both copied from the same source. A similarity visualisation generated by comparing the second document with the first document would generate the same graphic, only mirrored along the primary diagonal. In this synthetic example, both similarity visualisations would be identical.

Figure 8.3 shows a number of other similarity visualisations generated from synthetic documents. All are again 2000 words long, using a fragment size of 200 words and a fragment gap of 10 words.

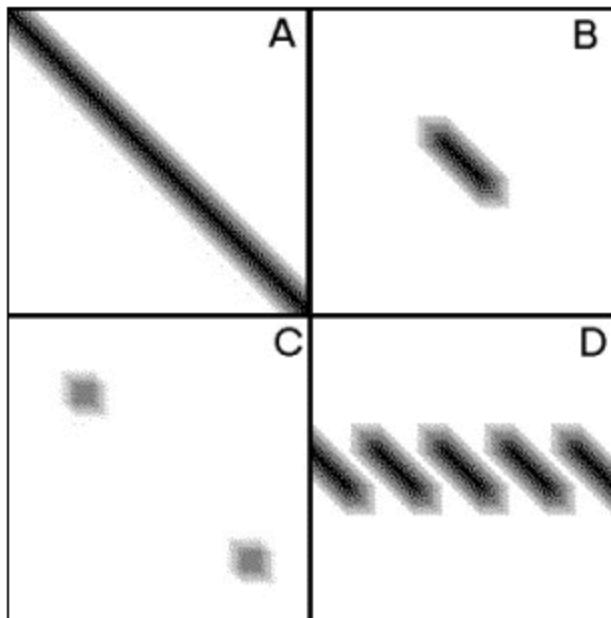


Figure 8.3 - More synthetic similarity visualisations

Similarity visualisation A shows the unlikely situation of the two documents being identical. The similarity intersection runs down the entire primary diagonal. The centre of the similarity interception shows the series of 200 word matches. The edge of it shows where just ten words are identical.

Similarity visualisation B shows the case where 400 words are identical, under much the same pattern as before. Again the similarity is positioned in the centre of both documents.

Similarity visualisation C features two similarity interceptions. Each represents a set of words identical to each document, but not to the other similarity interception. The common words are situated in the same place in both documents, one near the beginning of both, the other near the end of both. Here the central extent of the similarity interceptions is lighter than before, since at most 100 words match out of 200, demonstrating the need for careful selection of fragment sizes. However, in practice this would be sufficiently different from the background colour to stand out in most situations.

Similarity visualisation D shows a theoretical first document, along the x-axis, where the same 400 words are repeated five times, compared with a second document, along the y-axis containing these 400 words just once. There are five similarity interceptions, one for each set of matching words, in much the same shape as the similarity interception in similarity visualisation B. The leftmost and rightmost similarity interceptions start and end abruptly where the first document starts and ends. Similarity visualisation D demonstrates that one can expect to find extensive similarity only once at any given point on the x and y-axes. Later examples will show plagiarism seems to occur most often close to the primary diagonal.

A smaller fragment size would decrease the width of each of these similarity intersections, since the area through which fragmentary intersections continue to contain some similarity would decrease. It would also serve to increase the intensity of the similarity intersections in C, since the maximum proportion of similarity within a fragment would increase and hence, so would some of the fragmentary similarity scores.

A smaller fragment gap would increase the number of fragments to be compared and correspondingly, the size of the similarity visualisation. It would also increase the processing time required to generate the visualisation. Each similarity intersection would see a corresponding increase in size. On occasions an increase might make other similarity intersections visible that would be too small to see with larger fragment gaps.

The main advantage for using the similarity visualisations is that they clearly differentiate between sections of the two documents that contain similarity and sections that do not. So long as a tutor can tell which parts of the documents each similarity intersection refers to, checking similarity for potential plagiarism is much easier than by eye alone.

8.6 - Visualisations of real student submissions for intra-corporeal plagiarism

Synthetic documents are useful for verifying that the visualisation process seems to work and to assess the usefulness of and different properties of the intersections. Real student submissions can give a much better impression about how these visualisations look in practice. An effective visualisation would be one that clearly differentiates between areas of similarity and non-similarity, with any background noise not causing problems with this discrepancy. Further the metric applied at fragmentary granularity to generate the visualisation should show an intersection for every area of similarity in the two documents.

Figure 8.4 shows a similarity graphic for two student submissions that have been pre-judged by humans to contain intra-corporeal plagiarism with attempts at disguise. The first document, along the x-axis, is 2728 words long. The larger second document, along the y-axis, is 3341 words long. As in the synthetic cases, the similarity visualisation is generated with a simple word count metric, a fragment size of 200 words and a fragment gap of ten words. The size of the resulting similarity visualisation is 253 pixels by 315 pixels; the slight discrepancy from what would be expected caused by the intricacies in the parser used to produce the similarity graphic.

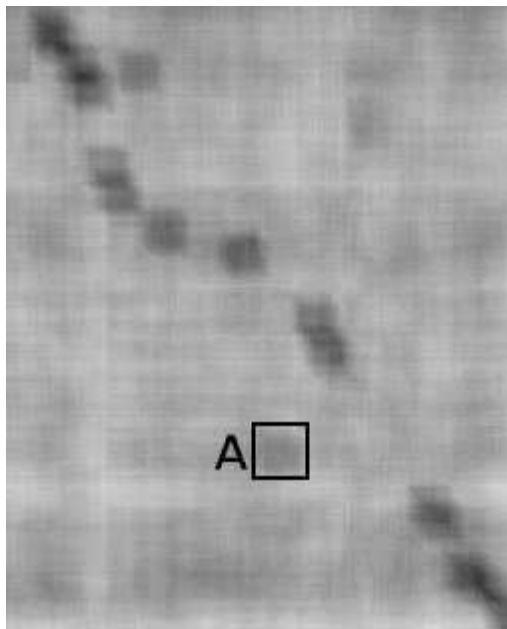


Figure 8.4 - Intra-corporeal plagiarism under the word count metric

The first thing apparent compared to the synthetic graphics, is the background noise. However the similarity interceptions, of which there are at least ten, are still clear to see. As expected, the similarity intersections appear close to the primary diagonal, where the second document seems to be a somewhat expanded version of the first. In particular, similarities in the introduction, conclusion, references and other sections of the text make this an interesting case. Although many words are changed here and there this does not fool a computerised checker, although it may get past a human marker, since the fragment size takes this into account. The similarity interception identified as A on Figure 8.4 showed, on investigation, two short sentences in each document that were identical. This suggests that the fragment size and fragment gap values were appropriate.

The similarity visualisation generated from one further fragmentary metric is worth showing. Figure 8.5 shows a refinement that reduces the noise and shapes the highlights.

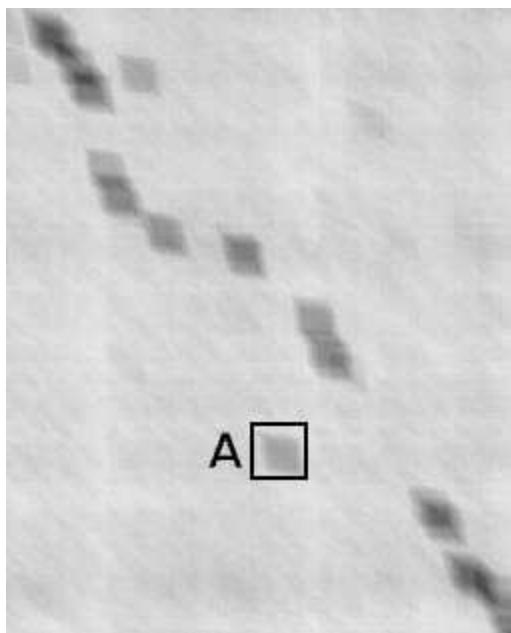


Figure 8.5 Intra-corporeal plagiarism under the word sequences metric

Figure 8.5 shows the same two student submissions as Figure 8.4, with the same fragment size and fragment gap, but a different metric. The fragmentary metric used is more computationally intensive than the last, but is responsible for a clearer similarity visualisation. The colouring is derived from the *word sequence metric* that takes the longest common ordered sequence of words within the fragment. The previous metric did not take such ordering into account. That is to say that the sequences do not have to be contiguous, but must contain the ordering. For example, the longest common ordered sequence of “apples bananas cherries damsons” and “apples apples bananas bananas damsons damsons” is “apples bananas damsons” of length 3 words.

In this case the similarity interceptions highlight much the same areas for inspection as before, but the noise level is much lower allowing the similarity interception identified as A to be much more clearly seen. This suggests that the word sequence metric is more useful than the word count metric. The tradeoff to this is that calculating the word sequence metric is more computationally intensive. The metric might also be less accurate if the similarity in the two documents is more fragmented.

8.7 - Visualisations of real student submissions for extra-corporeal plagiarism

The previous section demonstrated how similarity visualisations can aid with the verification of intra-corporeal plagiarism. They can also be very used to verify extra-corporeal plagiarism, where a source can be determined. This example represents a chapter of a final-year student project where the tutor marking it noticed stylistic differences to the rest of the document and with the aid of a Web search engine was able to trace the chapter to a Web source.

Figure 8.6 shows the similarity visualisation for the suspect chapter of the project, plotted against the Web source. It is presented using the word count metric.

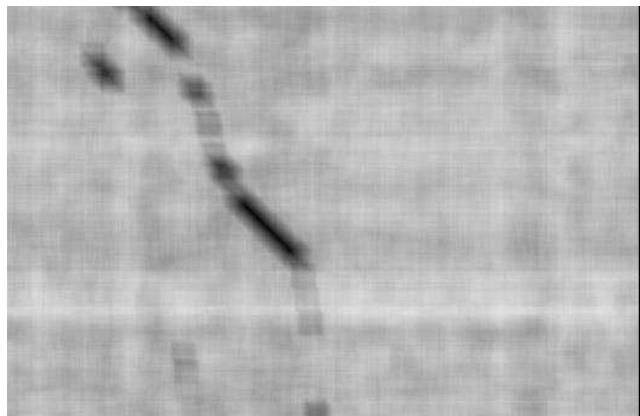


Figure 8.6 - Extra-corporeal plagiarism

The 5400 word chapter is plotted along the x-axis, the 3500 word Web source along the y-axis. Projecting the similarity intercepts down onto the x-axis reveals that around one third of the chapter is made up of similar material and the intensity of the areas reveals that the material has been copied word for word. Extracting the similarity intercepts across onto the y-axis reveals that almost the entire Web source has been used in the chapter at some point. However some parts, primarily towards the end of the Web source, have had new material introduced so that the apparent extent of the similarity is reduced. Also of interest is the first section of plagiarism within the chapter, which shows that the Web source has been slightly re-arranged. The most interesting section is the central similarity intercept of around 1000 words, which is copied verbatim; both its intensity and its positioning parallel to the primary diagonal illustrate this. The substantial similarity shown in the similarity visualisation in Figure 8.6 was later determined to be extra-corporeal plagiarism.

This approach represents one that could be used in conjunction with the limited visualisation methods used by the Web-based plagiarism detection services. These are useful for finding sources based on a Web search, but not very useful at showing where and how they are similar to the source. One way of working this might be to join all the sources identified together into a single file and produce a visualisation for the source against the text document of all the sources. This has one further advantage over the use of the similarity visualisation technique for intra-corporeal plagiarism detection. There it is impossible to know which document is the source and which is the copy, or if both were produced under close collaboration. For extra-corporeal plagiarism the source is plotted along a known axis.

8.8 - False hit verification

Section 7.5 saw four known plagiarised submissions added to a corpus. This was an attempt to error seed the corpus and demonstrate that it contained the similarity ranking list contained no missed pairs. The submissions, plagiarised by experienced tutors in the manner of a student, ranked in the top five pairs by many of the tutors which was a good sign.

Of related interest is verifying that a similarity ranking list avoid false hits. False hits are pairs of submissions that rank highly but no contain no obvious similarity. This might be because the similarity is hidden so that it is not obvious to a user. The similarity visualisations can be used to aid in this, reducing the problem to checking if a visualisation contains similarity intersections and then to verify if those intersections represent similarity. Pairs that rank highly without obvious intersections are likely to be highly fragmented.

Figures 8.7 to 8.10 show the similarity visualisations generated from the four pairs of the plagiarised submission along with its original source. They have been produced under the word sequence metric. The source is shown along the x-axis and the copy along the y-axis. The difference between height and width represents the difference in length between the source and the copy.



Figure 8.7 - Similarity visualisation for tutor 1

The similarity visualisation for tutor 1 shows contractive plagiarism. Note the area in the centre that does not contain a similarity intersection, suggesting that that section of text has simply been removed from the copy. The rest of the document contains light intersections, albeit ones that are clearly visible against the background noise. This suggests some attempts at disguise for all but the first intersection. The similarity is clearly again along the primary diagonal. Chapter 7 shows that tutor 1 spent the least time plagiarising the source document which tallies with the style of the visualisation.

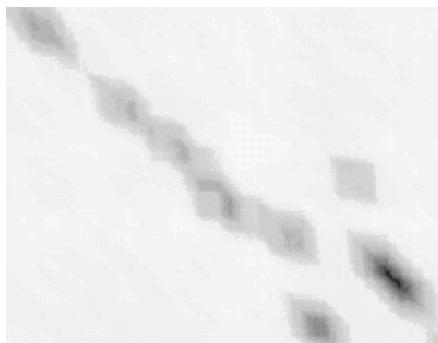


Figure 8.8 - Similarity visualisation for tutor 2

Tutor 2's source and copy pair ranked highest out of the four tutors under most of the simple metrics tested in Chapter 7. This is reflected in the visualisation with some very intense similarity intersections towards the end of the submission. The rest of the visualisation shows some attempts at re-arrangement which agrees with the plagiarism methods that the tutor listed that they had used.



Figure 8.9 - Similarity visualisation for tutor 3

Tutor 3's visualisation shows a single clear similarity intersection down the primary diagonal, again showing some attempts at disguise. It again comes highly ranked, suggesting that this strategy is not enough to fool the simple metrics. The short nature of the resulting submission might also mean that the resulting submission would not be highly marked.



Figure 8.10 - Similarity visualisation for tutor 4

The visualisation from tutor 4 shows that the copy here is the only one to represent expansive plagiarism, that is the original document has been fleshed out by with filler material. This is reflected both by the longer y-axis than x-axis and the fact that the plagiarism took 90 minutes, the longest out of all the submissions. It is also the least intense of the visualisations, reflected by the fact that it was the lowest ranked of the four, containing a discontinuity at the end of the primary diagonal where the references were rewritten.

Generally the similarity intersections are easily visible for all four of the plagiarised submissions which suggests that they are not false hits. The seeding and high ranking of the pairs over most of the simple metrics also shows that they are not missed pairs. Along with the similarity visualisations that show plagiarism by students this suggests that the visualisation technique has an additional use, not only for noticing that extent and position of similarity, but also for verifying pairs of submissions when looking for false hits.

8.9 - Conclusions

This chapter has seen the introduction of a valuable new technique for investigating similarity in pairs of student submissions or investigating any submission against a potential source. The technique has been tested on synthetic documents, on documents containing plagiarism from students and on error seeded documents highly ranked under the simple metrics of Chapter 7.

The similarity visualisations introduced are useful alone as part of increasing the efficiency of the entire detection process; the speed at which they are available allows false hits to be quickly excluded. They also increase the overall effectiveness, the visual technique is believed to be intuitive and shows exactly where submissions are similar and how similar they are at those points.

The whole technique can be machine assisted further. A prototype tool named VAST has been produced that links the visualisation directly to textual representations of the two documents. The similarity intersections can then be quickly navigated around, similarity verified and further investigated. VAST will be described further later.

Chapter 9 will further show how the detection process can be made more efficient using the visualisations as a starting point for finding the best simple metric for ranking pairs of submissions whilst avoiding missed pairs.

Chapter 9: A Visual Approach to Identify the Most Effective Simple Metric

Overview:

- Properties of similarity visualisations are presented.
- Visual metrics are introduced.
- Simple metrics are compared with a visual metric.
- An effective simple metric is found.

The simple metrics identified so far have each given an ordering of similarity but there is little indication as to which ordering is the most effective. This chapter describes how an ordering can be found that is both effective and efficient to compute.

Visualisations are argued to give the best indication of how and why two documents are similar. Properties can be read from visualisations to give an effective ordering based on visual metrics. However these visual metrics do not qualify as efficient to compute.

It is hypothesised that there may be an ordering based on simple metrics that closely correlates with the visual metrics. Such an ordering would then be effective as would be expected from the visual metrics. It would also be efficient, since efficiency is a property of simple metrics.

The hypothesis is tested on a number of corpora. These include synthetically generated corpora containing no noise, synthetic noise and stochastic noise, real text with added similarity and corpora of actual student submissions. The results conclusively favour the hypothesis and an efficient and effective simple metric is identified.

9.1 - Motivation

This chapter argues that the similarity visualisations that would be generated for every pair of submissions within a corpus provide the best mechanism with which to order those pairs by the amount of similarity they contain. That is a similarity visualisation containing a similarity intersection should be ranked higher than a similarity visualisation that does not contain an intersection. Sections 9.2 to 9.5 argue that this approach should hold, but is unfeasible for most corpora due to practical difficulties with generating visualisations.

Section 9.6 presents a number of possible measures, known as visual metrics, that could be drawn from visualisations. One is selected that seems that it could be used to order the visualisations in a similar way to a human would. Sections 9.8 to 9.12 present a number of synthetic corpora and compare the orderings generated from the chosen visual metric with the orderings from the simple metric, based on criteria from Section 9.7. Sections 9.13 to 9.15 compare the chosen visual metric with corpora of actual student submissions to see if the same results hold. The chosen simple metric is described in more detail in Section 9.16. It is intended that this should provide an

effective ordering, since it mirrors those produced from the similarity visualisations, but is also much more efficient, since it is generated from a simple metric.

9.2 - Visualisations show whether or not two submissions contain similarity

Similarity visualisations are produced to show where any two textual documents, in this context usually student submissions, are similar and how alike they are at those points. The main advantage of this is a significant time saving for a tutor who has to assess whether an intense similarity intersection within the visualisation represents plagiarism or similarity that has occurred through legitimate means.

There is a second, less obvious advantage of the visualisations. They allow a tutor to quickly assess whether two submissions resemble each other at any point or not. Since sections of similarity visualisations represent a similarity mapping at word level between two submissions the more intense areas represent parts of the texts containing at least some shared words. This can be seen by considering Figure 9.1, which shows two visualisations labelled A and B.

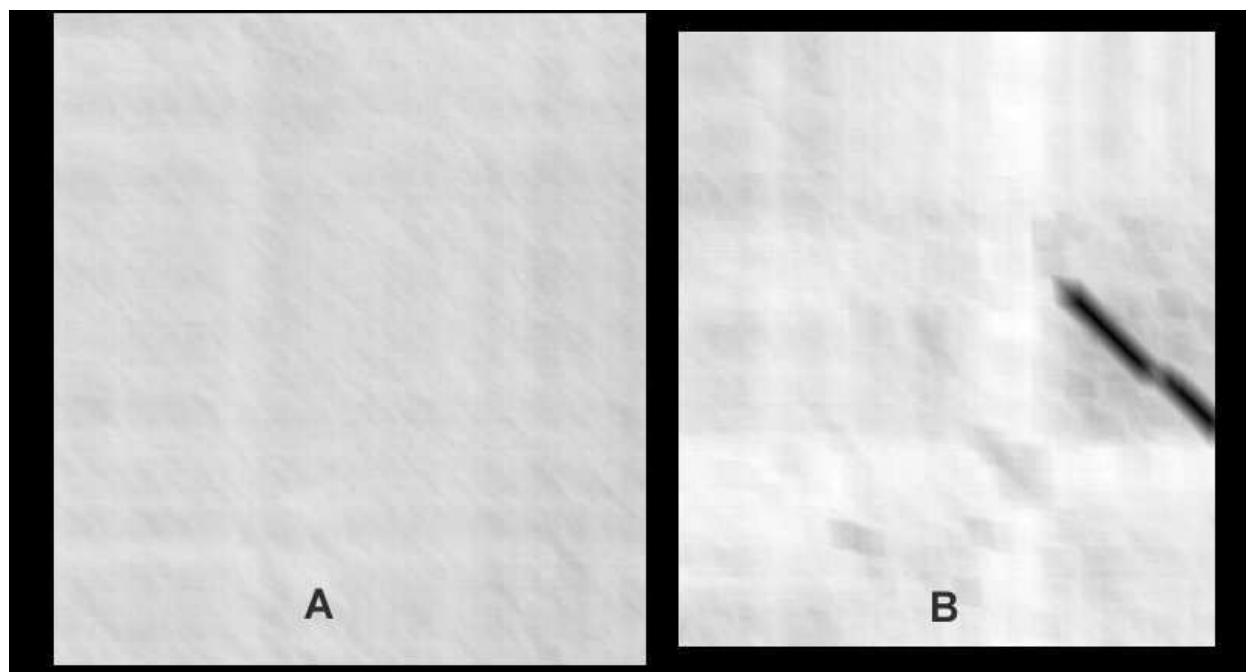


Figure 9.1 - Two similarity visualisations

Visualisation A contains no similarity intersections. Assuming the premise that these intersections reliably present similarity to users it would not be worth the time for a tutor to manually inspect this pair of submissions further.

Visualisation B contains (one or two) similarity intersections. Therefore it contains similar areas containing at least some words in common that would be worthy of manual inspection by a tutor to identify possible plagiarism.

Figure 9.2 shows representative versions two very small similarity visualisations with similar properties to those in Figure 9.1. A contains no similarity intersections, just background noise. B contains one intersection with a little bit of noise. The numbers

show the intensity of each area as a percentage of the maximum possible intensity (i.e. 0 for no commonality, 100 for completely identity). They are included for later demonstrations of different metrics.

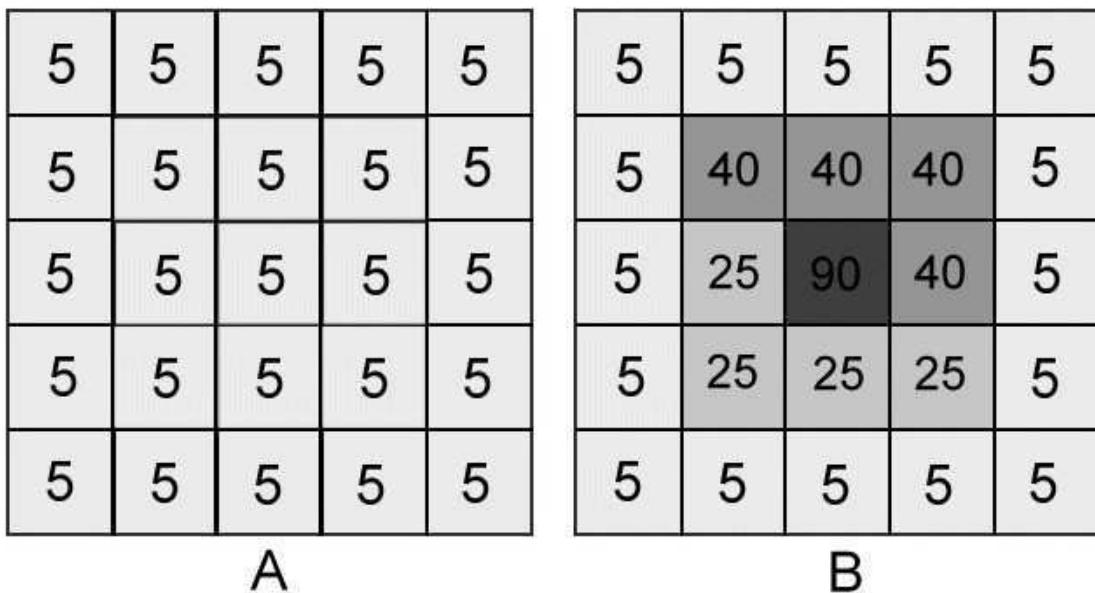


Figure 9.2 - Simple representative similarity visualisations

In order to avoid missed pairs a tutor needs to do an exhaustive search through all possible pairs of submissions for possible plagiarism. This can be done by hand using only the documents, or tool assisted with similarity visualisations linked to the appropriate texts. The VAST tool described in Chapter 10 is the only current possibility. Either way this would be a substantial drain on a tutor's resources for large corpora.

By producing a similarity visualisation for every possible pair of submission within a corpus a tutor can quickly inspect them to see if they contain any similarity intersections. Any pairs that contain no similarity intersections can immediately be eliminated from further consideration and the tutor is safe in the knowledge that they have not eliminated what would be a missed pair. Tutors can then go on and inspect the remaining pairs of submissions, aided with a tool, to eliminate false hits and give an error free plagiarism detection process. Assuming that this process holds the remaining pairs can all then be judged to each contain some level of plagiarism.

9.3 - Practical problems with this approach

There are some operational problems with using this approach with large corpora (and for small corpora automated plagiarism detection is not so much of a necessity as these can be checked by hand, possibly helped by the visualisations). Such problems will mean that this process is unlikely to be used to guarantee an error free plagiarism check in a practical context.

The first problem is the computational resources needed to produce every possible similarity visualisation. The time taken to produce a single visualisation can often be measured in minutes. The number of visualisations that needs to be produced increases exponentially as the number of submissions goes up. This equates to computations that could take days, weeks or even months, an infeasible wait within the tight constraints of the academic calendar. Section 1.3 gave details of the computational resources available, power similar to that which would be available on a standard office PC and that has affected the amount of processing possible.

The second is human ability. Manually checking all of these visualisations for similarity intersections is an onerous task. There is also likely to be some question in some borderline cases as to whether a visualisation contains an intersection or not.

These problems are not insurmountable with appropriate resolve; in the future computer systems should be powerful enough so that processing power is not a constraint. Also in practical terms tutors are unlikely to be interested in borderline similarity cases since these are hard to prove further up the chain of plagiarism judgements where potential student penalisations are delivered. But these problems are enough to suggest that such an exhaustive investigation of similarity visualisations is unlikely to be used in practice.

9.4 - Similarity visualisations can give the best ordering of similarity

There are a number of different metrics that can be used to sort a series of pairs of submissions. To detect plagiarism this needs to be a non-capricious ordering of similarity from most similar to least similar. Operationally tutors will start at the top of the list with the most similar pair of submissions, check them for plagiarism, and then work their way progressively down the list. This will continue until there is no more similarity that the tutor deems worth investigating or time resources expire.

There is no guarantee that there is no plagiarism further down the list than tutors have chosen to check. The most effective metric would be one that gave an ordering that matched the tutors expectation, ranking the most similar pair at the top of the list and the least similar pair at the bottom and the other pairs in decreasing similarity order between. This would also give a cut off point somewhere in the list, above which pairs could be considered similar and below which they could be considered suitably dissimilar. The existence of a cut off point for the most effective metric can be argued by contradiction. Suppose there is a similar pair of submissions which is ranked below the cut off point. Then either it is one position below the cut off point, in which case the cut off point can be moved one position further down to incorporate this similar pair. Or it is more than one position below the cut off point. In this case there must be a dissimilar pair between the cut off point and the similar pair. This is a contradiction, since the most effective ordering should not order a dissimilar pair before a similar pair. Hence there must be a cut off point for pair similarity.

Since even the definition of similar is debatable there is unlikely to be a single cut off point for a given corpus. Instead this will vary from user to user and corpus to corpus. The important thing is that there would be a point during a high rank to low rank inspection when a tutor decides they have checked enough pairs, which is an individual cut off point for them. For the same reason that a chosen cut off point is individual so is the definition of exactly which pairs are similar. Hence there is unlikely to be a single most effective metric. A more approachable target would be a metric that closely approximates a most effective metric.

The most computationally efficient metrics are those that require little processing. Another property of a good metric is that it should be possible to compute it in an efficient manner. It is always advantageous to process a metric that requires as little processing as possible, both to allow results to be available immediately for a user and to free up processing power for other tasks. Metrics that take hours or days to compute are not going to be acceptable, however good the ordering produced.

One way of deciding how similar two submissions are is to look at the intensity and volume of similarity intersections within the associated similarity visualisations. An ordering based on these intersections would be immediately intuitive to a tutor, with the properties of visualisations giving this an immediate reality link so that tutors can see how the ordering is produced.

For these reasons a metric derived directly from the properties of similarity visualisations can be said to be the best possible metric for assessing levels of similarity. But producing an entire set of visualisations and extracting metrics from them is just not feasible due to the computational resources required. Producing all possible visualisations for a corpus of one hundred submissions can take several months. An alternative but related approach is necessary.

9.5 - Error free metrics of low computational intensity

Metrics can be split into at least two classifications, namely simple metrics and visual metrics. Simple metrics were introduced in Section 7.2. In brief these encompass any metric that can be calculated purely based on properties of the original documents, the word simple referring to the computational intensity of processing a corpus. The count of the number of words in common between two documents would be classified as a simple metric. This might or might not be a good metric since a difference in the size of the two texts may influence this ordering.

The alternative classification would be that of visual metrics. A visual metric would be a metric produced as a result of producing the associated visualisation, or at least the underlying structure of numeric values. An example of a visual metric would be a count of the number of similarity intersections on the visualisation. Any metrics that do not fit inside the simple and visual categorisations will not be considered further here.

A benefit of the simple metrics over visual metrics is that significantly less computation is required to produce them. It takes seconds or minutes, rather than days or weeks, to exhaustively examine a corpus. This is within the acceptable waiting time for processing a corpus to give a similarity ordering. Visual metrics rate over simple

metrics by giving a much more accurate and visually demonstrable ordering of similarity. Amongst other reasons this is because the human brain is more adept at seeing patterns in visual than numeric data. But the computational resources required to produce visual metrics are significantly greater than those required to produce simple metrics.

Ideally there would be a simple metric or a combination of simple metrics that closely approximate the visual metric, thus giving an effective ordering whilst dramatically reducing the computations necessary. The best simple metric would therefore be one that gave an ordering close to the visual ordering, or whose list of ordered pairs correlated best with the list of ordered pairs from the visual metric (to the extent that the correlation would be judged statistically highly significant). This ordering would then have demonstrable and valid properties: It would be the best ordering which is computationally feasible. It would avoid false hits, those pairs flagged for inspection containing no similarity. In this context those are pairs ranked before the cut off point that should have been ranked after it. It would avoid missed pairs, much like the inverse of false hits, those pairs that should have been flagged for inspection but weren't, those ranked below the cut off point which should have been ranked before it. The most similar submissions would be ranked highest in the top portion of an ordered similarity list. A high correlation by its very nature would ensure that pairs of documents had roughly similar rankings in both lists and that there were no pairs ranked at greatly differing positions in both lists.

The rest of this chapter aims to show that the hypothesis: that there are simple metrics that closely correlate with the visual metrics, is justified. Additionally an effective simple metric or metrics will be sought. This will be accomplished by first finding a visual metric that closely approximates human judgement, then finding simple metrics that closely approximate the visual metric. For this a number of corpora will be needed including carefully constructed synthetic texts for verification and actual student submissions for further investigation.

9.6 - Choosing a good visual metric

A good visual metric should give an intuitive ordering of pairs of documents based on the properties of the associated similarity visualisations. A first approximation might be, simply, the number of and intensity of similarity intersections within the visualisation. However, translating this into a number simply using the properties of the individual documents is not an intuitively obvious process.

A number of metrics have been identified that can be computed as a side product of computing the similarity visualisations. Each visualisation is represented by an underlying two dimensional matrix where each cell shows the similarity in two fragments of the documents. Higher values are represented as darker colours on the associated similarity visualisation.

As with simple metrics it is impossible to give an exhaustive list of visual metrics. An additional metric that could be considered can always be identified. For simple metrics this could be done just by considering character chains of one character more in length. For visual metrics any other statistic property of visualisations could be considered. Hence the list of visual metrics considered is never going to be exhaustive

but it has been chosen to be representative and to give properties that can be extracted directly from the visualisations (or their underlying matrix forms) without the need for additional complex calculations.

Many of the calculations involve cell maximums or means within a given range of cells. These are standard ways of preserving the maximum information in a single numeric form. Maximums have an immediate use as representing the most similarity between two submissions (the corresponding cell minimum has no obvious use). Means represent the average similarity. A number of additional statistical properties such as the other averages, the median and mode, could also be considered, as could the inter-quartile ranges. However many of these values may prove to be bad differentials of similarity, for instance if the visualisations contain noise.

A list of metrics to be considered is given in Table 9.1. Each metric is given a name by which it will be referred to later and a description.

Visual metric name	Visual metric description
Max	The overall maximum of all the cells in a visualisation.
Mean	The overall mean of all the cells in a visualisation (total of the cells divided by the number of the cells).
Median	The overall median of all the cells in a visualisation (the cell value 50% of the way through all cells when placed in numeric order).
Mode	The overall mode of all the cells in a visualisation (the most frequent repeating cell value).
Inter-quartile range	The inter-quartile range of all the cells in a visualisation (the difference between the cells 75% and 25% of the way through all cells when placed in numeric order).
Mean of row max	Find the maximum cell value in each row of the visualisation leaving in effect a vector of values. Take the mean of that vector.
Mean of col max	Find the maximum cell value in each column of the visualisation leaving in effect a vector of values. Take the mean of that vector.
Max of row mean	Find the mean cell value in each row of the visualisation leaving in effect a vector of values. Take the maximum of that vector.
Max of col mean	Find the mean cell value in each column of the visualisation leaving in effect a vector of values. Take the maximum of that vector.

Table 9.1 - Possible visual metrics

These metrics are available for both sequenced and unsequenced visualisations. The sequenced metric is the better of the two, since it produces the clearer visualisations, minimising noise, whilst presenting the same similarity intersections to a user.

Figures 9.3 to 9.11 show examples of how the similarity scores for different possible visual metrics could be calculated. The calculations are shown on the two representative similarity visualisations introduced in Figure 9.2. Visualisation A contains no similarity intersections, only background noise. Visualisation B contains one intense similarity intersection and the same level of background noise. Since each cell contains a value between 0 and 100 each similarity score will also be valued between 0 and 100. The similarity scores will be used to demonstrate which visual

metrics can suitably differentiate between the two visualisations and how well they differentiate between them.

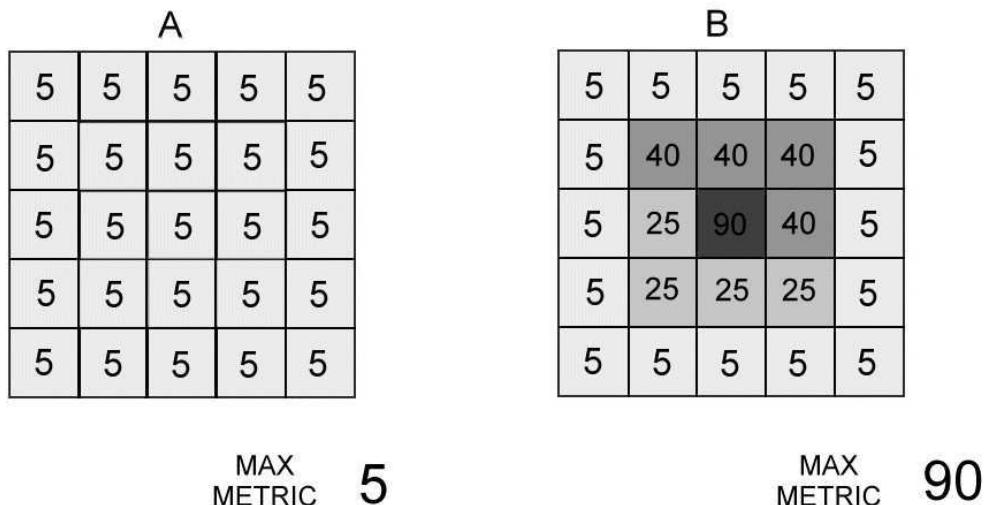


Figure 9.3 - Calculation of the max metric for two representative similarity visualisations

The max metric clearly differentiates between visualisations A and B and is easy to compute. However it would not distinguish between a visualisation containing a single similarity intersection and one containing ten similarity intersections. From that viewpoint it is not ideal.

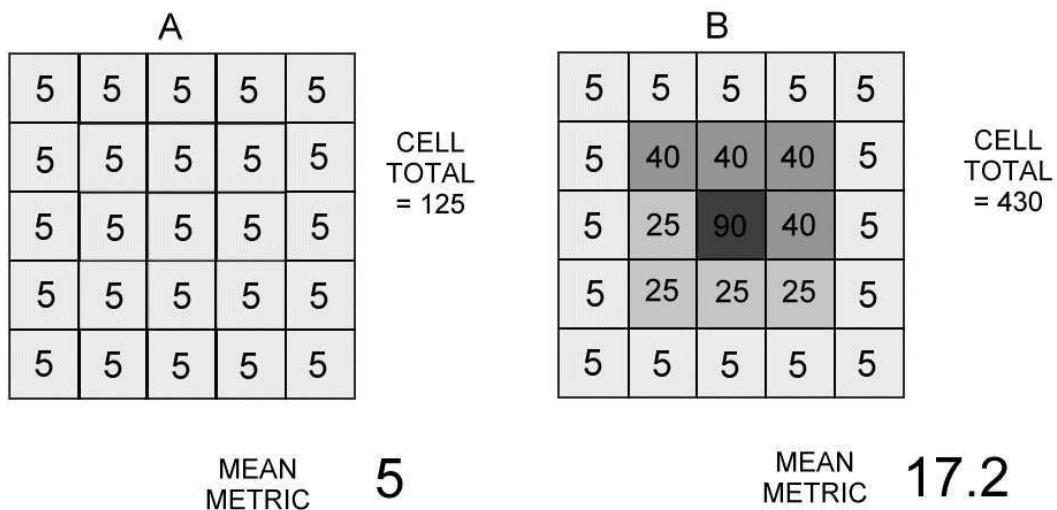


Figure 9.4 - Calculation of the mean metric for two representative similarity visualisations

The mean metric does not look to be a good discriminator since it is heavily influenced by noise. If A had higher noise, say 20%, then the two visualisations would be ordered the wrong way round.

A	B																																																		
<table border="1" style="border-collapse: collapse; width: 100%;"> <tr><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td></tr> <tr><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td></tr> <tr><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td></tr> <tr><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td></tr> <tr><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td></tr> </table>	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	<table border="1" style="border-collapse: collapse; width: 100%;"> <tr><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td></tr> <tr><td>5</td><td>40</td><td>40</td><td>40</td><td>5</td></tr> <tr><td>5</td><td>25</td><td>90</td><td>40</td><td>5</td></tr> <tr><td>5</td><td>25</td><td>25</td><td>25</td><td>5</td></tr> <tr><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td></tr> </table>	5	5	5	5	5	5	40	40	40	5	5	25	90	40	5	5	25	25	25	5	5	5	5	5	5
5	5	5	5	5																																															
5	5	5	5	5																																															
5	5	5	5	5																																															
5	5	5	5	5																																															
5	5	5	5	5																																															
5	5	5	5	5																																															
5	40	40	40	5																																															
5	25	90	40	5																																															
5	25	25	25	5																																															
5	5	5	5	5																																															
MEDIAN METRIC	MEDIAN METRIC																																																		
5	5																																																		

Figure 9.5 - Calculation of the median metric for two representative similarity visualisations

The median metric cannot discriminate between visualisations A and B and so is clearly unsuitable for further consideration.

A	B																																																		
<table border="1" style="border-collapse: collapse; width: 100%;"> <tr><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td></tr> <tr><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td></tr> <tr><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td></tr> <tr><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td></tr> <tr><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td></tr> </table>	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	<table border="1" style="border-collapse: collapse; width: 100%;"> <tr><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td></tr> <tr><td>5</td><td>40</td><td>40</td><td>40</td><td>5</td></tr> <tr><td>5</td><td>25</td><td>90</td><td>40</td><td>5</td></tr> <tr><td>5</td><td>25</td><td>25</td><td>25</td><td>5</td></tr> <tr><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td></tr> </table>	5	5	5	5	5	5	40	40	40	5	5	25	90	40	5	5	25	25	25	5	5	5	5	5	5
5	5	5	5	5																																															
5	5	5	5	5																																															
5	5	5	5	5																																															
5	5	5	5	5																																															
5	5	5	5	5																																															
5	5	5	5	5																																															
5	40	40	40	5																																															
5	25	90	40	5																																															
5	25	25	25	5																																															
5	5	5	5	5																																															
MODE METRIC	MODE METRIC																																																		
5	5																																																		

Figure 9.6 - Calculation of the mode metric for two representative similarity visualisations

The mode metric is also unable to differentiate between visualisations A and B and so is unsuitable. In most visualisations the modal cell will be one of those containing noise.

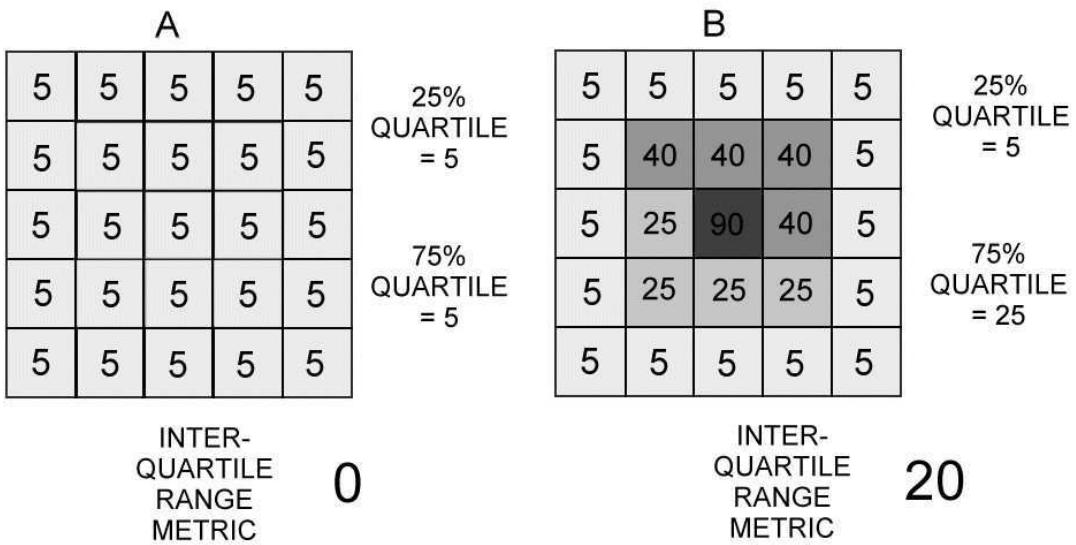


Figure 9.7 - Calculation of the inter-quartile range metric for two representative similarity visualisations

The inter-quartile range looks as though it could be a useful discriminator between two visualisations, since a visualisation containing only noise will have an inter-quartile range close to zero, whilst one containing an intersection should have a higher range. This does however depend on the size of the visualisation. The area covered by similarity intersections may well be less than one quarter of a visualisations, in which case the inter-quartile range would again approach zero and be a useless discriminator.

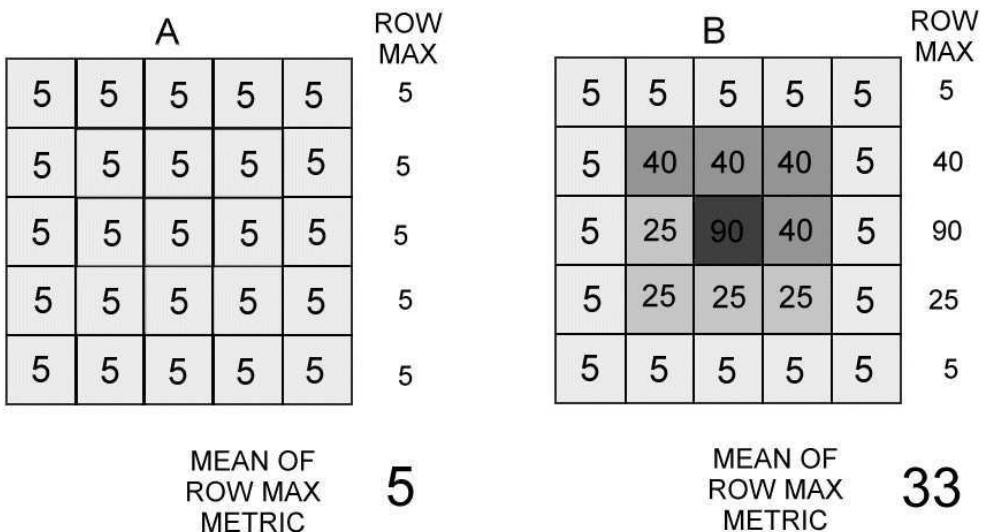


Figure 9.8 - Calculation of the mean of row max metric for two representative similarity visualisations

The mean of the row max metric looks like being a good discriminator. In Figure 9.8 visualisation A has a much lower similarity score than visualisation B. Since the mean is taken over only the number of rows, not the number of cells, the noise has little effect on the overall similarity score.

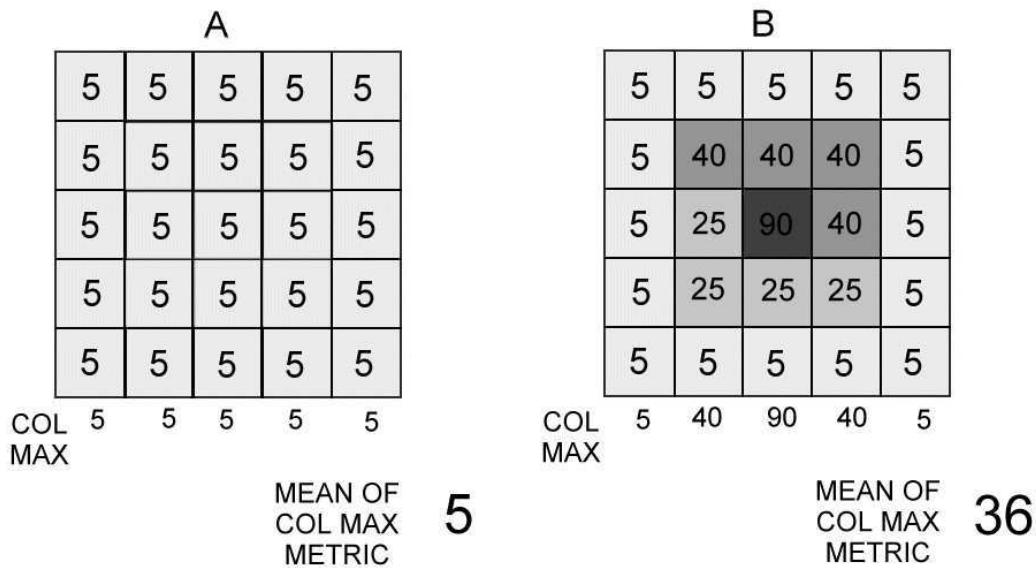


Figure 9.9 - Calculation of the mean of col max metric for two representative similarity visualisations

The mean of col max metric is calculated in much the same way as the mean of row max metric. It is interesting to note that the similarity scores returned are slightly different. Again noise has little effect on successful discrimination of two visualisations.

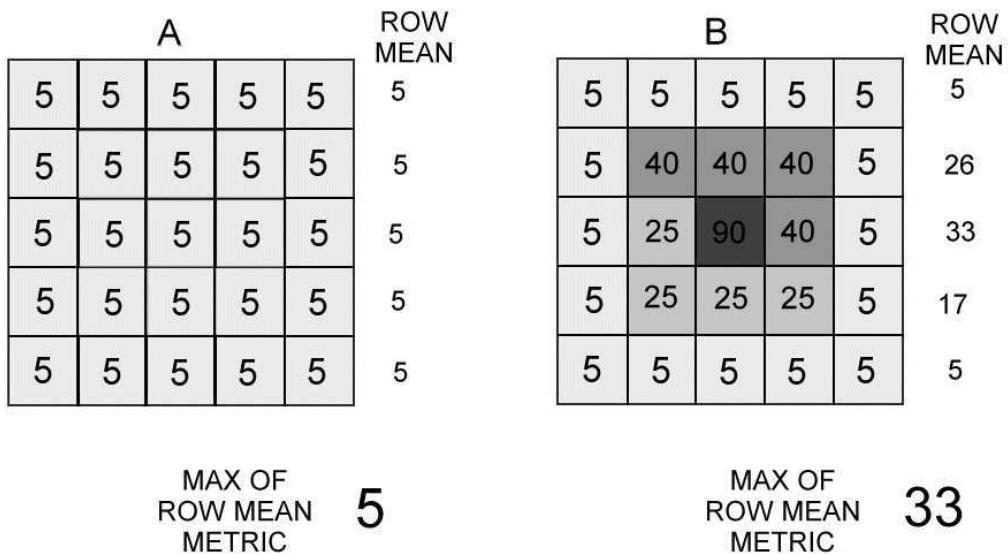


Figure 9.10 - Calculation of the max of row mean metric for two representative similarity visualisations

The max of row mean metric looks to able to separate visualisations A and B with a reasonable similarity score difference. This metric will say how intense the row is that is the most intense on average. Again note that if there are two equally intense rows (from two similarity intersections) in two parts of the visualisation this metric will only discriminate one of them.

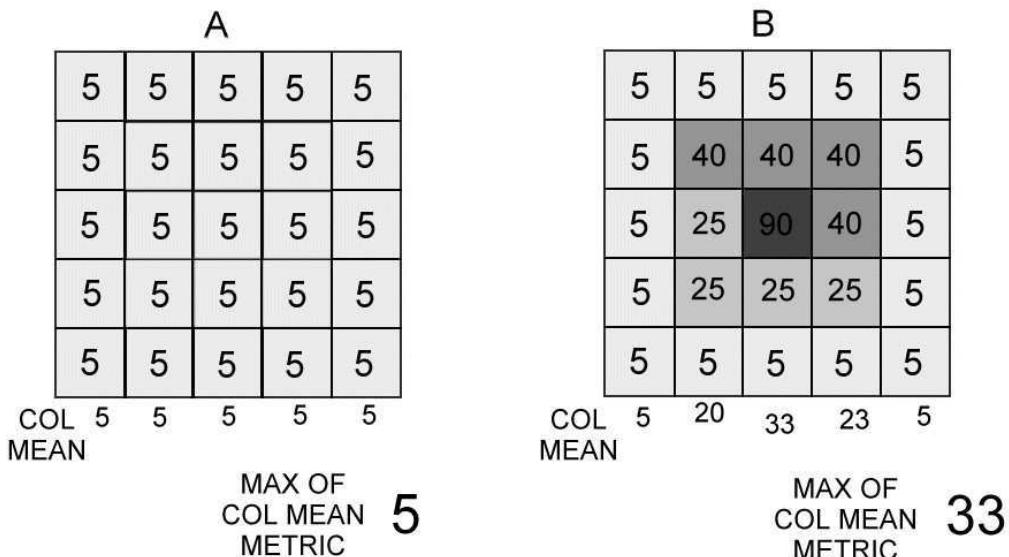


Figure 9.11 - Calculation of the max of col mean metric for two representative similarity visualisations

This metric is much like the max of row mean metric. The similarity scores calculated are identical, but this is only coincidence as these could vary slightly.

The different similarity scores are summarised in Table 9.2.

Visual metric name	Similarity score for visualisation A	Similarity score for visualisation B
Max	5	90
Mean	5	17.2
Median	5	5
Mode	5	5
Inter-quartile range	0	20
Mean of row max	5	33
Mean of col max	5	36
Max of row mean	5	33
Max of col mean	5	33

Table 9.2 - Summary of comparative scores for possible visual metrics

The metrics that best look to fit the idea of counting the number of and intensity of similarity intersections would be the mean of row max and mean of col max metrics. This is because finding the maximum element in a row or column gives a good indication of whether there is a similarity intersection in that row or column or not. Taking the mean of these values gives a single metric that is influenced by noise far less than metrics calculating the mean of every element, since prominent pixels will heavily outweigh any background noise.

Since calculating the mean of the row maximums and the mean of the column maximums will give slightly different results the best visual metric would have to be one that combined both with equal influence by taking the mean of the two (summing the two also gives the same ordering). These values can then be used to give an ordering of the pairs with the visual metric. Even if the two values only differ a little the ordering would be thrown out if the mean were not taken. This also means that it

doesn't matter whether one document is plotted horizontally and the other vertically or vice-versa as this will give the same similarity score for both.

The combined metric will be known as the *mean of max metric*. Figure 9.12 shows an example of how the mean of max metric could be calculated for a very simple similarity intersection B.

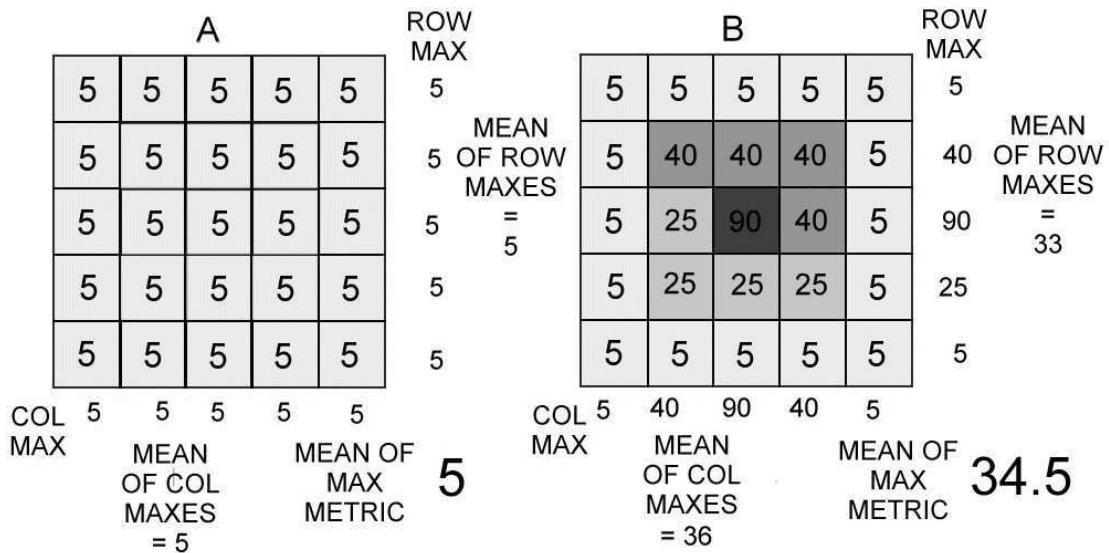


Figure 9.12 - Calculation of the mean of max metric for two representative similarity visualisations

9.7 - Correlating the simple metric with the mean of max metric

The simple metrics that would seem to be the most effective would those that best correlated with the mean of max metric. Since it is the order of the pairs that is important, not the actual values of the metrics, Spearman's rank-order correlation coefficient is used.

The correlation between each simple metric and the mean of max metric can be computed and the significance tested under a standard statistical t-test (with $n-2$ degrees of freedom) to see if there is a positive or negative correlation between the two metrics.

By definition (where n is the number of pairs, r is the observed correlation and t the t-value to test against statistical tables):

$$t = r \sqrt{\frac{n-2}{1-r^2}}$$

This can be re-arranged to give:

$$r = \sqrt{\frac{t^2}{t^2 + n - 2}}$$

The null and alternative hypothesis are:

$H_0: r = 0$ (there is no evidence of correlation between the metrics)

$H_1: r \neq 0$ (there is evidence of correlation between the metrics)

Under a two-sided hypothesis test at 5% significance (from tables) H_0 is rejected for:

$$|t| > 1.98$$

which translates to:

$$|r| > \sqrt{\frac{1.98^2}{1.98^2 + n - 2}}$$

Similarly at 1% significance (from tables) H_0 is rejected for:

$$|t| > 2.617$$

which translates to:

$$|r| > \sqrt{\frac{2.617^2}{2.617^2 + n - 2}}$$

The 1% significance level is likely to be more useful in this context since it gives a much greater assurance that a correlation is significant.

9.8 - Noise free synthetic documents

Synthetic corpora can be used to investigate the properties of the mean of max metric and give some level of assurance that the mean of max calculations have been implemented successfully.

A corpus of five synthetic documents was created. Each document was constructed to be 2000 words long. The documents were constructed so that any two documents chosen would contain a single continuous chunk of similarity of between 100 and 800 words long. They were also constructed so that any pair out of the possible ten chosen would contain a different amount of similarity.

This strategy was chosen so that the best ordering of pairs is unambiguous. The pair containing 800 identical words would be ranked first. The pair containing 100 identical words would be ranked 10th. All other pairs would be ranked between in the expected order.

The strategy used to generate the documents is shown in Figure 9.13. Each file contains four areas of similarity, each of which matches in value and ordering with exactly one other file. These are shown on the diagram as shaded areas. Any remaining space at the end of each document is kept unique and does not match any area of any other document in value or ordering. These are shown on the diagram as white areas. Identical areas on the diagram are identical in length (which is the number of words). To make the diagram easier to view they are also shown shaded with identical patterns and colours.

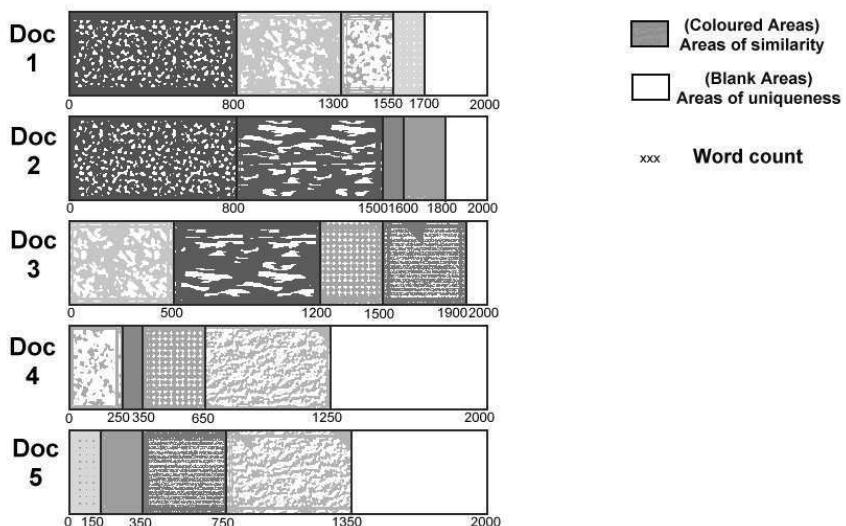


Figure 9.13 - A synthetic corpus with known best ordering of pairs

The documents were generated using unique or repeated numbers in the range 1 to 6000. No attempt was made to mimic the character and sentence distributions in the English language, so in effect each document contained one long sentence of 2000 words, each of which was between one and four characters in length.

Table 9.3 shows the number of words in common in the ten possible pairings of these five documents and the expected similarity ranks for each.

First Document	Second Document	Words in common	Expected ranking
1	2	800	1
1	3	500	4
1	4	250	7
1	5	150	9
2	3	700	2
2	4	100	10
2	5	200	8
3	4	300	6
3	5	400	5
4	5	600	3

Table 9.3 - Expected similarity ranks for synthetic corpus

Tables 9.4 to 9.6 show the similarity ranks obtained from a number of simple metrics on this corpus. They also show the correlations between the simple metrics and the expected ranking. Table 9.4 shows words 1 to words 9, a measure of the number of word chains of that length in common. Table 9.5 shows characters 1 to characters 9, a measure of the number of character chains of that length in common. Table 9.6 shows a measure of the number of sentences in common. Only words 1 to words 9, as shown in Table 9.4, can be expected to give useful results as no attempt has been made to generate documents with sensible character or sentence distributions. In particular the alphabet available for the character metric is limited to ten different characters ('0', '1', ..., '9') which will heavily influence those results.

First Document	Second Document	words chain length								
		1	2	3	4	5	6	7	8	9
1	2	1	1	1	1	1	1	1	1	1
1	3	4	4	4	4	4	4	4	4	4
1	4	7	7	7	7	7	7	7	7	7
1	5	9	9	9	9	9	9	9	9	9
2	3	2	2	2	2	2	2	2	2	2
2	4	10	10	10	10	10	10	10	10	10
2	5	8	8	8	8	8	8	8	8	8
3	4	6	6	6	6	6	6	6	6	6
3	5	5	5	5	5	5	5	5	5	5
4	5	3	3	3	3	3	3	3	3	3
Correlations		1	1	1	1	1	1	1	1	1

Table 9.4 - Words metric similarity ranks for synthetic corpus

First Document	Second Document	characters chain length								
		1	2	3	4	5	6	7	8	9
1	2	1	1	1	1	2	2	2	2	2
1	3	1	7	3	3	3	4	4	4	4
1	4	1	8	7	4	6	7	7	7	7
1	5	1	5	6	5	9	9	9	9	9
2	3	1	6	2	2	1	1	1	1	1
2	4	1	4	5	7	10	10	10	10	10
2	5	1	3	4	5	7	8	8	8	8
3	4	1	9	10	10	8	6	6	6	6
3	5	1	10	9	9	5	5	5	5	5
4	5	1	2	8	8	4	3	3	3	3
Correlations		N/A	0.16	0.35	0.43	0.94	0.99	0.99	0.99	0.99

Table 9.5 - Characters metric similarity ranks for synthetic corpus

		sentences chain
First Document	Second Document	1
1	2	1
1	3	1
1	4	1
1	5	1
2	3	1
2	4	1
2	5	1
3	4	1
3	5	1
4	5	1
Correlation		N/A

Table 9.6 - Sentences metric similarity ranks for synthetic corpus

Table 9.4 shows that the ranks obtained from words 1 to words 9 are identical and are also identical to the expected rankings. This gives a correlation of any two of these metrics of 1, or perfect correlation. This suggests that any one of words 1 to words 9 would be a good metric for ordering similarity for synthetic documents.

Although the characters metrics are less suitable it is interesting to notice that characters 6 to 9 all give identical results, as would any longer character chains. This is likely because the 'words' in the documents are modally four characters long, so these results will all incorporate part of two words. It is also interesting to see that these ranks are identical to the expected ranks with positions 1 and 2 transposed, again giving almost perfect correlation (0.99). The reason for this is down to the generation method; the similar words in documents 1 and 2 are all three characters long, whilst in documents 2 and 3 they are four characters long. This means that the length of the run of similar characters in documents 1 and 2 (800 words of 3 characters plus 801 spaces = 3201) is shorter than that in 3 and 4 (700 words of 4 characters plus 701 spaces = 3501) which is enough to cause the discrepancy. In practical terms such a one rank transposition would be unlikely to affect a tutor's likelihood of checking two submissions for similarity.

Characters 1 is simply checking that every single character that occurs in one document also occurs in the other and so cannot differentiate between the pairs. Similarly sentences is counting the number of sentences in common between two documents and since each document contains only one sentence and is different these will all return similarity of zero and hence identical rankings.

The orderings for characters 2 to 4 look highly variable and of no great use compared to the other results. The correlation of characters 5 to 9 with the expected ranking is high at 0.94 for characters 5 and 0.99 for characters 6 to 9.

It appears that any of the words metrics will give a perfect correlation for the noise free synthetic documents and any large number character metric will also give a good correlation. These results may however be influenced by the synthetic nature of the documents.

It has already been argued that the best metric for ordering similarity would be the most that most closely correlates with an ordering of the associated similarity visualisations.

Table 9.7 shows the ranks obtained from the mean of max metric (computed from the sequenced visualisations rather than the unsequenced visualisations, since these are clearer, although for synthetic documents the sequenced and unsequenced results will be identical).

First Document	Second Document	mean of max
1	2	2
1	3	4
1	4	7
1	5	9
2	3	1
2	4	10
2	5	8
3	4	6
3	5	5
4	5	3

Table 9.7 - Mean of max metric similarity ranks for synthetic corpus

The mean of max metric correlates perfectly with the characters 5 to 9 metrics. Mean of max correlates almost perfectly (0.99) with both the set of metrics words 1 to 9 and the expected ideal ordering as given in Table 9.3. The reason the correlation is not perfect in both these cases is again the transposition of the first and second placed ranks. The non-perfect correlation can be best explained by looking at the visualisations generated, which, for the most part, would immediately suggest this ordering.

Figure 9.14 shows the similarity visualisations generated for each possible pairing of the documents in the synthetic corpus.

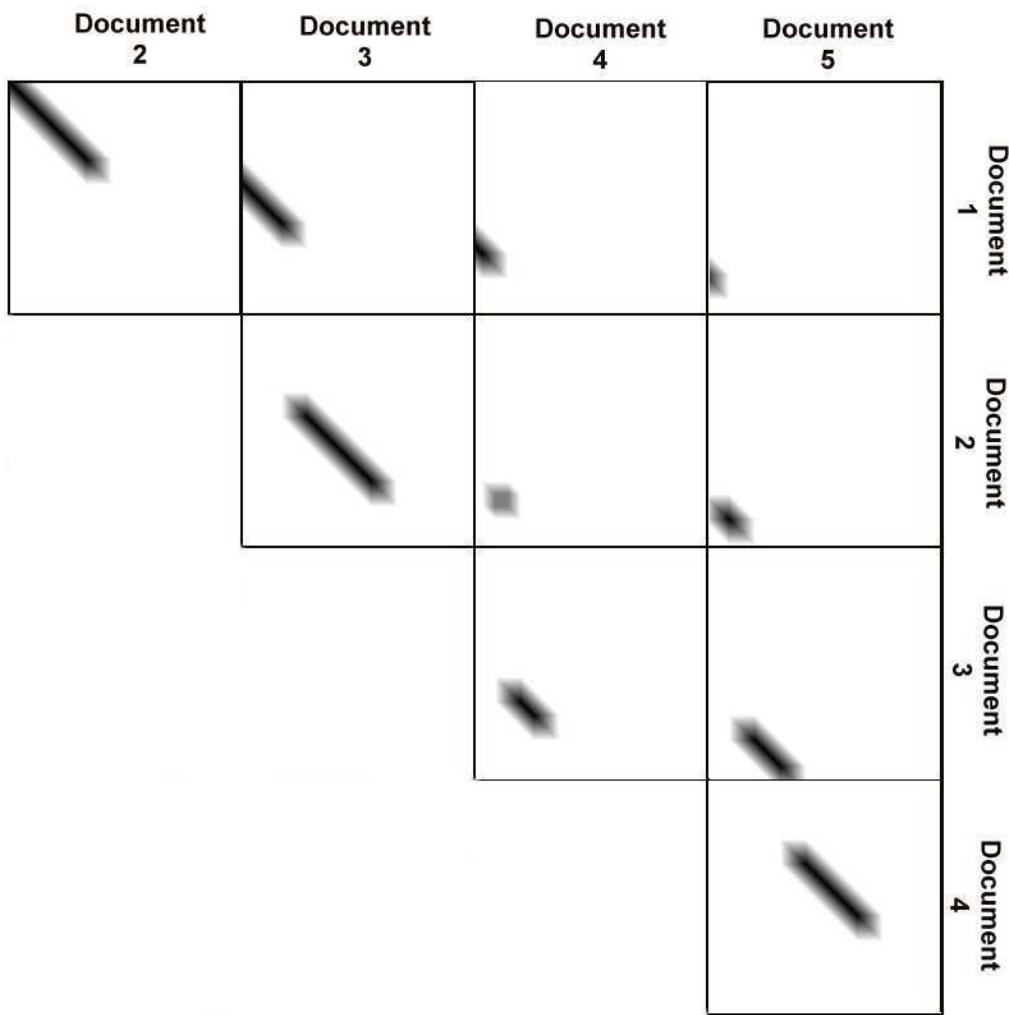


Figure 9.14 - Similarity visualisations for synthetic corpus

For the most part Figure 9.14 shows similarity visualisations where the size of the similarity intersection is related to the number of words the two documents share. However there are some exceptions, namely where an intersection meets an edge of the graphic and part of the intersection is lost.

This would explain the transposition of expected ranks 1 and 2 on the mean of max metric from the expected metric. The mean of max metric, in effect, quantifies the size of the similarity intersection and since some is missing this won't be included in the calculation. It is coincidental that these are the only ranks affected as their others that appear, visually, to be close in size.

This does not pose a problem, since it would be highly unlikely to move a highly similar pair of submissions far enough down the ranking list that a tutor would not examine them.

9.9 - Noise free borderless synthetic documents

In order to ensure correct implementation and that the assumptions about the visualisations being successfully sorted by the size of their similarity intersections is correct the similarity intersections need to avoid the borders of the visualisations. Therefore a new synthetic corpus is necessary.

The new corpus is produced by simply adding 500 unique words (unused integers) to the start and end of each of the previous synthetic documents, making each 3000 words in size. The alphabet is deliberately restricted to numeric digits to maintain consistency with the previous alphabet and to avoid any attempts to try to construct valid word distributions. The addition of unique words ensures that the generated similarity intersections will avoid the borders of the visualisations.

Figure 9.15 shows the result of the changes.

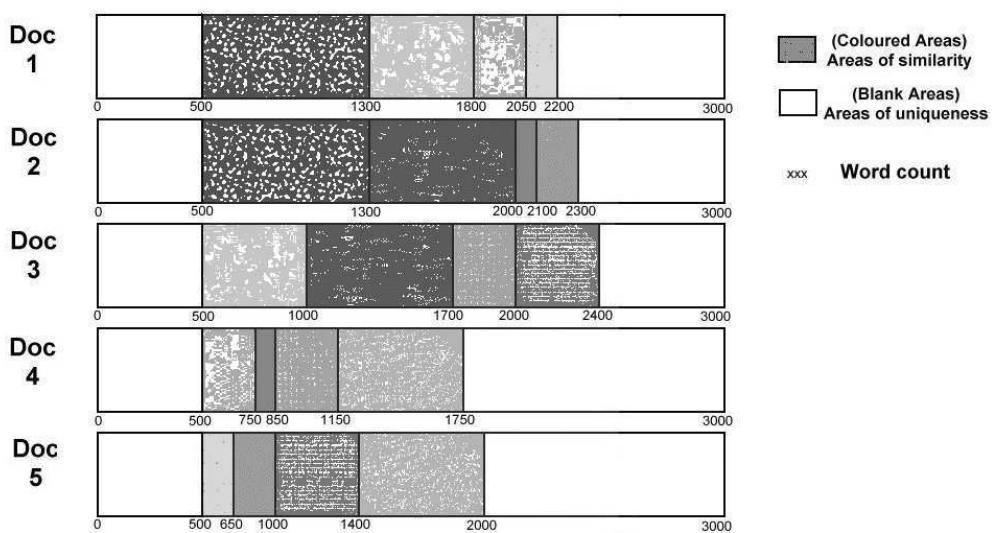


Figure 9.15 - Second synthetic corpus with known best ordering of pairs

All other details about the corpus remain unchanged from the previous synthetic corpus, including the known best ordering and expected rankings.

Tables 9.8 to 9.10 show the similarity ranks and correlations obtained from the simple metrics on the new noise free synthetic corpus.

		words chain length								
First Document	Second Document	1	2	3	4	5	6	7	8	9
1	2	1	1	1	1	1	1	1	1	1
1	3	4	4	4	4	4	4	4	4	4
1	4	7	7	7	7	7	7	7	7	7
1	5	9	9	9	9	9	9	9	9	9
2	3	2	2	2	2	2	2	2	2	2
2	4	10	10	10	10	10	10	10	10	10
2	5	8	8	8	8	8	8	8	8	8
3	4	6	6	6	6	6	6	6	6	6
3	5	5	5	5	5	5	5	5	5	5
4	5	3	3	3	3	3	3	3	3	3
Correlations		1	1	1	1	1	1	1	1	1

Table 9.8 - Words metric similarity ranks for second synthetic corpus

		characters chain length								
First Document	Second Document	1	2	3	4	5	6	7	8	9
1	2	1	1	1	1	2	2	2	2	2
1	3	1	7	3	2	5	4	4	4	4
1	4	1	5	6	5	4	7	7	7	7
1	5	1	2	4	4	7	9	9	9	9
2	3	1	6	2	3	1	1	1	1	1
2	4	1	9	7	6	10	10	10	10	10
2	5	1	4	5	7	9	8	8	8	8
3	4	1	8	10	10	8	6	6	6	6
3	5	1	10	9	9	6	5	5	5	5
4	5	1	3	8	8	3	3	3	3	3
Correlations		N/A	0.25	0.38	0.35	0.87	0.99	0.99	0.99	0.99

Table 9.9 - Characters metric similarity ranks for second synthetic corpus

		sentences chain								
First Document	Second Document	1								
1	2	1								
1	3	1								
1	4	1								
1	5	1								
2	3	1								
2	4	1								
2	5	1								
3	4	1								
3	5	1								
4	5	1								
Correlation		N/A								

Table 9.10 - Sentences metric similarity ranks for second synthetic corpus

The interesting thing about these results is that words 1 to words 9, characters 1, characters 6 to characters 9 and sentences are unchanged from the previous synthetic corpus. Hence words 1 to 9 still have perfect correlation, characters 6 to 9 still have near perfect correlation and characters 1 and sentences are still unable to differentiate between the different pairs (the table is included for the sake of completeness).

Characters 2 to characters 5 are ranked slightly differently to before due to the change in allocated words (the non-matching words at the beginning and end of the documents were allocated using unique spare four digit numbers). These metrics still appear to be much less good for ordering similarity than the other metrics.

Table 9.11 shows the most important results, the ranks obtained from the mean of max metric.

First Document	Second Document	mean of max
1	2	1
1	3	4
1	4	7
1	5	9
2	3	2
2	4	10
2	5	8
3	4	6
3	5	5
4	5	3

Table 9.11 - Mean of max metric similarity ranks for second synthetic corpus

These now correlate perfectly with the expected rankings from Table 9.3 and the words 1 to words 9 simple metrics. This suggests that, in the noise free synthetic case, any of words 1 to words 9 give a perfect sequence of similarity ranks and any of characters 6 to characters 9 give a good sequence of similarity ranks. The combined results of similarity intersections overlapping and not overlapping the edge of visualisations suggest that overlaps cause only a minor impact on a good ordering.

9.10 - The effects of noise on ordering synthetic documents

The noise free synthetic corpora give an early indication that the simple word metrics are good at ordering similarity when a set of visualisations was available. However these differ from real corpora in a couple of ways. The most immediately noticeable is a complete absence of noise in the visualisations and hence there is no noise affecting the results of the metrics.

It would therefore seem to be a good idea to introduce noise into the corpus and see if this affects the list of similar ranks and the correlations between the different metrics.

A number of new corpora have been constructed, each based on the second synthetic corpus (the one avoiding the problems of intersections overlapping the edge of the visualisation). Each contains different known amounts of added noise, the noise consistent throughout a corpus but differing across the corpora.

Noise has been added to the corpus by taking every document within it and replacing some of the words with a single known word. This word will be the same throughout every document and at every replacement, but will not already exist elsewhere in the corpus (the word '99999' was used). The noise has been generated to cover a known proportion of each document by replacing words at regularly spaced intervals. Later tests will use noise that has been randomly introduced.

To create a corpus containing n% noise every $(100/n)^{th}$ word is replaced, e.g. for 10% noise every 10th word is replaced, for 20% noise every 5th word is replaced.

So each corpus contains five documents of 3000 words, with ten possible pairings of unique similarity and a known noise level. None of the similarity intersections cross over the borders of the similarity visualisations. The synthetic corpus with 0% noise is identical to the second synthetic corpus described above.

Each of the corpora has been run processed under both the simple metrics and the mean of max metric. Tables 9.12 - 9.14 show the correlations between the simple metrics and the mean of max metric for each corpus.

Noise percentage in corpus	words chain length								
	1	2	3	4	5	6	7	8	9
0	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1	1	1
10	1	1	1	1	1	1	1	1	1
20	1	1	1	1	1	1	1	1	1
50	1	1	1	1	1	1	1	1	1

Table 9.12 - Correlations between simple words metrics and mean of max metric for corpora containing known noise levels

Noise percentage in corpus	characters chain length								
	1	2	3	4	5	6	7	8	9
0	N/A	0.25	0.38	0.35	0.87	0.99	0.99	0.99	0.99
1	N/A	-0.08	0.39	0.35	0.87	0.99	0.99	0.99	0.99
2	N/A	0.25	0.38	0.35	0.87	0.99	0.99	0.99	0.99
4	N/A	-0.08	0.39	0.35	0.87	0.99	0.99	0.99	0.99
10	N/A	-0.08	0.37	0.35	0.79	0.98	0.98	0.98	0.98
20	N/A	-0.08	0.37	0.35	0.79	0.96	0.98	0.98	0.99
50	N/A	-0.08	0.37	0.37	0.42	0.56	0.64	0.75	0.78

Table 9.13 - Correlations between simple characters metrics and mean of max metric for corpora containing known noise levels

Noise percentage in corpus	sentences chain								
	1								
0	N/A								
1	N/A								
2	N/A								
4	N/A								
10	N/A								
20	N/A								
50	N/A								

Table 9.14 - Correlations between simple sentences metrics and mean of max metric for corpora containing known noise levels

These correlations are very promising since the orderings are very close to the orderings from the second noise-free synthetic corpus, showing that the same simple metrics still make good discriminants, even when there is a substantial amount of noise in the corpus. In practice the highest level of noise (50%) used here is greater than the level of background noise likely in comparisons of real documents. Observations on visualisations from real data suggest that the background noise level is usually somewhere between the 10% and 20% levels.

As before characters 1 and sentences simple metrics rank every pair of documents equally, so the correlations returned are not applicable. They are again included for the sake of completeness. All the words metrics have perfect correlation at all the noise levels tested, so any of those would be a suitable simple metric to use approximate the mean of max metric.

Characters 2 to 9 vary in their effectiveness, but in most cases there is a sudden increase in effectiveness between characters 4 and characters 5. This is again related to the number of characters in a normal word, with five characters just overlapping two consecutive words. There is little to choose between the high valued character metrics and the word metrics in effectiveness.

The conclusion is that adding noise into the synthetic corpus does not noticeably change the ordering. The words metrics and higher valued character metrics appear to be the most suitable at ordering similarity.

Figure 9.16 shows the similarity visualisation generated from documents 1 and 2 at the different noise levels used in the synthetic corpora. That is an 800 word run of similarity within two 3000 word documents.

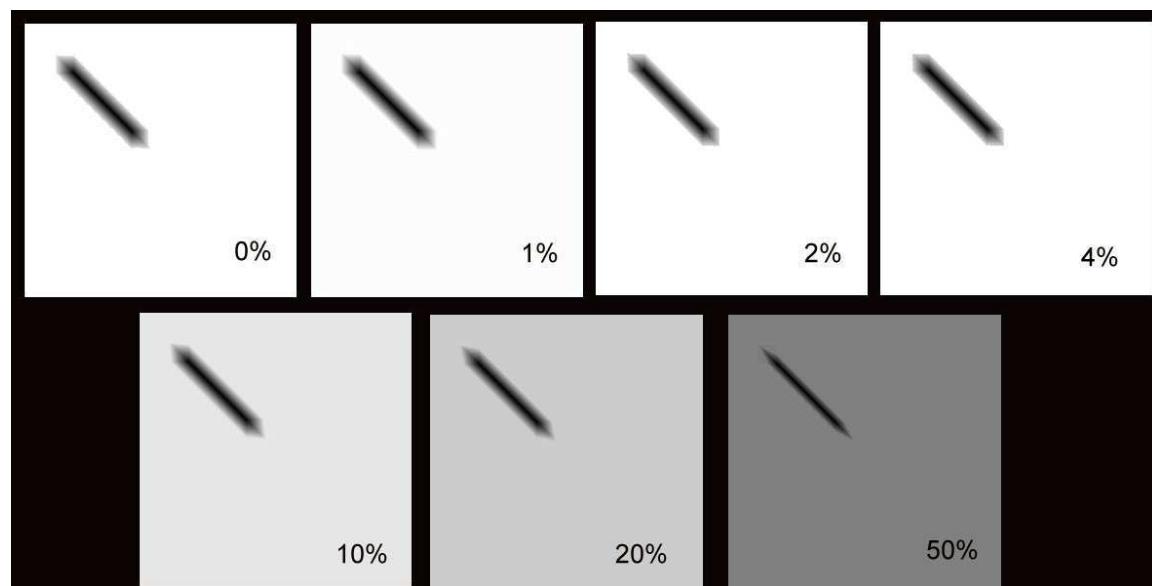


Figure 9.16 - 800 word similarity intersections at different noise levels

The graphics show that the similarity intersections are still immediately visible even at high levels of background noise. This supports the idea that the ordering is preserved.

There are two main differences, both obvious at the higher similarity level. The first is the background colour, which immediately reflects the amount of noise (or guaranteed known similarity present in the corpus). This noise level will be consistent across all the visualisations that could be generated from a single corpus. The other is the width of the intersection. As the noise level increases the width of the intersection decreases, leaving only the most central black bar. This is because the outside edges of the intersection, which would only just meet the noise level are, in effect, hidden.

9.11 - A synthetic corpus with randomly generated noise

The synthetic corpora used up until now contain synthetically generated noise. That is for a 10% noise level any consecutive ten words will contain exactly one noise generating word.

Although this is a valid test a real document is not going to conform to such a scheme. A further test would use noise generated randomly according to stochastic principles. That is a corpus of documents is generated as before, such that, for a 10% noise level, every word has a one in ten chance of being replaced by a noise generating word. By its very natures, repeating this process multiple times would give slightly different visualisations.

Figure 9.17 shows the visualisations generated from these submissions with 10% stochastic noise (the sequenced and unsequenced visualisations are identical).

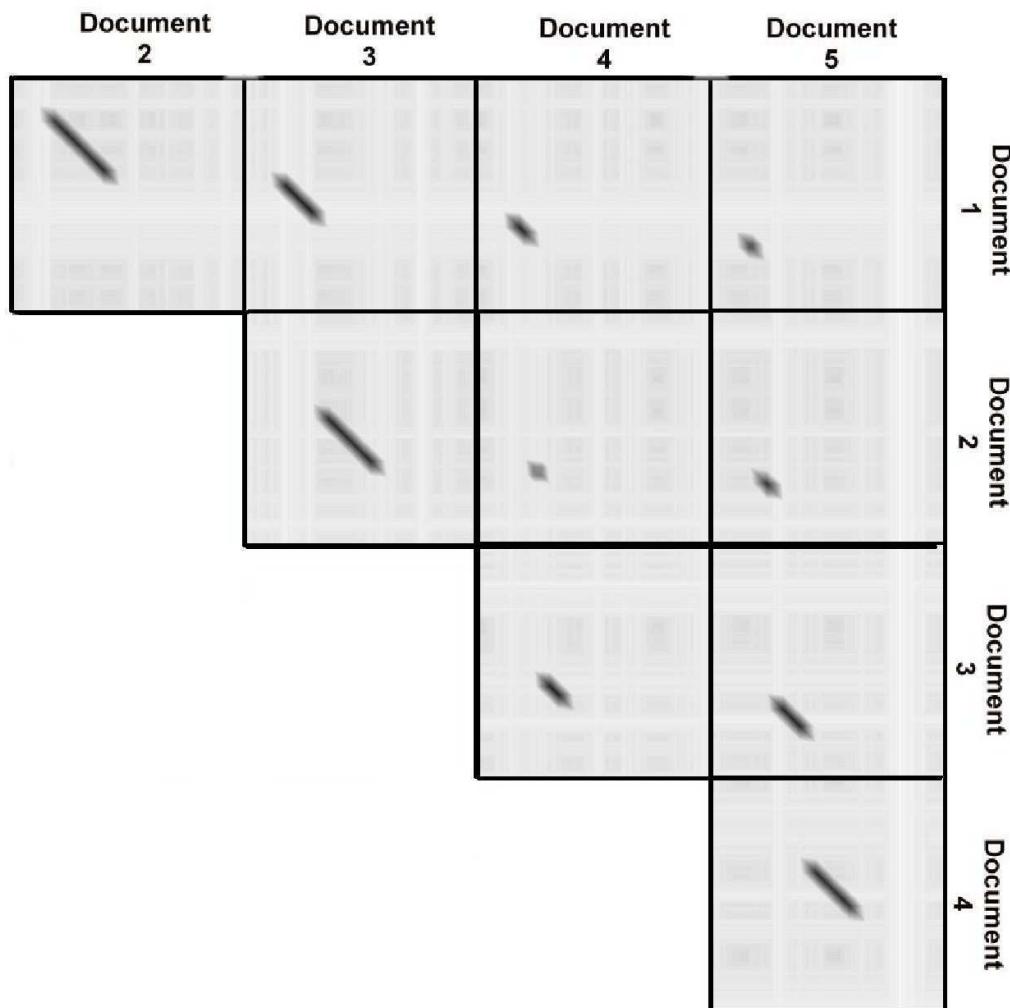


Figure 9.17 - Similarity visualisations for 10% stochastic noise synthetic corpus

The main property of the randomly generated noise is to add lighter coloured vertical and horizontal lines to the visualisations, otherwise the noise looks just like the regulated synthetic noise. This is a side effect of processing overlapping fragments of text to generate each pixel, since the number of times the noise indicating word appears will be consistent throughout the same fragment of text (same row or column) except when it is superseded by the similarity intersections.

Tables 15 to 17 show the correlations between the simple metrics and the mean of max metric for the 10% stochastic noise synthetic corpus.

		words chain length								
		1	2	3	4	5	6	7	8	9
Correlation	1	1	1	0.98	0.98	0.97	0.94	0.91	0.92	
	N/A	-0.10	0.35	0.37	0.84	0.95	0.96	0.96	0.96	

Table 9.15 - Correlations between simple words metrics and mean of max metric for 10% stochastic noise synthetic corpus

		characters chain length								
		1	2	3	4	5	6	7	8	9
Correlation	1	N/A	-0.10	0.35	0.37	0.84	0.95	0.96	0.96	0.96
	N/A	-0.10	0.35	0.37	0.84	0.95	0.96	0.96	0.96	0.96

Table 9.16 - Correlations between simple characters metrics and mean of max metric for 10% stochastic noise synthetic corpus

		sentences chain								
		1								
Correlation	1	N/A								
	N/A									

Table 9.17 - Correlations between simple sentences metrics and mean of max metric for 10% stochastic noise synthetic corpus

These results are very encouraging, since they are close to those obtained from the equivalent synthetic corpus with synthetic noise. The same metrics are still most significant. The main difference is that the correlation between the word metrics and the mean of max metric is no longer always 1, but that is to be expected, since of the number of words between each noise word is now randomly generated. The correlations are as close to perfect correlation as makes no difference. The same is true for the characters metrics, characters 5 to 9 are still highly significant.

Although only one stochastic corpus has been created the noise level inside it is very close to the noise level in real documents and the results are close enough to those from corpora containing synthetic levels of noise that this would seem to be sufficient without being overkill.

9.12 - A synthetic corpus that preserves real structure and noise levels

The synthetic corpora seen so far have all contained a known ordering of similarity that has been used to assess the value of simple metrics. However they differ in some ways from real documents. No attempt was made to generate documents following human writing style, where certain words will appear more often than others do. They have a limited alphabet of only ten characters available ('0', '1', ..., '9') which do not reflect an ordering of characters in real documents. There has been no attempt to split

them into something approximating real sentences with a linguistically valid word order. And although they have contained noise this has been either uniform or randomly generated throughout a given corpus and hence not reflective of the noise in real documents.

A further corpus has been developed that contains a known ordering of similarity but eliminates many of the earlier problems since it contains real words in a realistic ordering.

To do this a document of longer than 6000 words (the number of unique identifiers in the previous synthetic corpus) has been chosen. The actual document used was H.G. Wells 'First Man on the Moon' obtained from Project Gutenberg, an electronic source for out-of-copyright literature. The synthetic corpus has been altered by replacing a number in the synthetic corpus with the word in that numeric position in novel by Wells. In this context a word is defined as the text between two pieces of whitespace, so all punctuation has been preserved. This gives the documents a valid sentence structure that has not been seen in the synthetic corpora used so far. Continual runs of numbers have been replaced with continual runs of words, hence preserving the original linguistic properties of the document, so an area of similarity between two documents represents a run of complete sentences.

This has immediate advantages over the previous synthetic corpora. The expected ordering of similarity is still known and is as it was before. Areas of two documents will be identical similar where necessary or will be dissimilar. The only change will be levels of noise in the dissimilar areas, which will mirror the noise in two real documents. Additionally the character, word and sentence metrics can all be expected to follow regular linguistic patterns.

Figure 9.18 shows a single similarity visualisation generated from two documents in the corpus under the unsequenced metric. It is split into nine areas, labelled A to I.

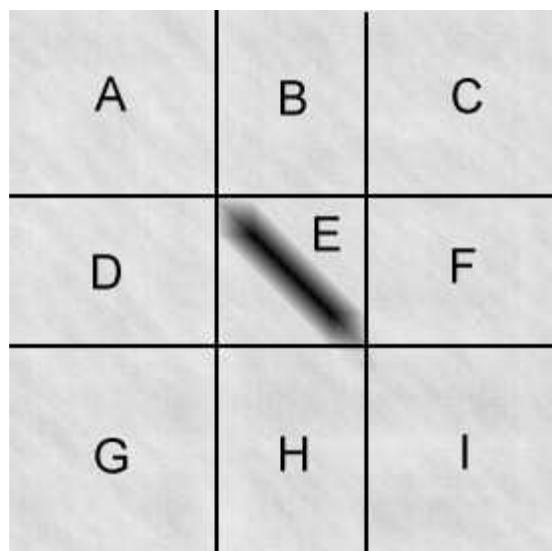


Figure 9.18: Similarity visualisation for two documents in corpus preserving linguistic properties

The most interesting area is area E. This shows the only similarity intersection in the visualisation, representing a 600 word sequence of exact similarity in both documents.

Area A represents the first 1150 words of the first document and the first 1200 words of the second documents. The background noise shows where words in both documents happen to overlap. Since the words have been extracted from different sections of the source text this overlap will be purely standard background noise with no sequences of identical words, primarily just the repeating filler words (e.g. 'the', 'of', 'and'). Areas C, G and I are similar to A. Parts of them will match identically with other documents in the corpus, but not with the other document used for this visualisation. Areas B, D, F and H represent words that would match in part with the other document (the result is area E) but these section represent purely background noise.

The visualisations for all possible pairs of documents have been produced and the corresponding mean of max metric and simple metrics generated. Figure 9.19 shows the visualisations generated from the sequenced metrics. The shape of the similarity intersections within the intersections is the same as before, so the same best ordering, which is the one generated by the mean of max metric could be expected. The difference is the variable amount of background noise.

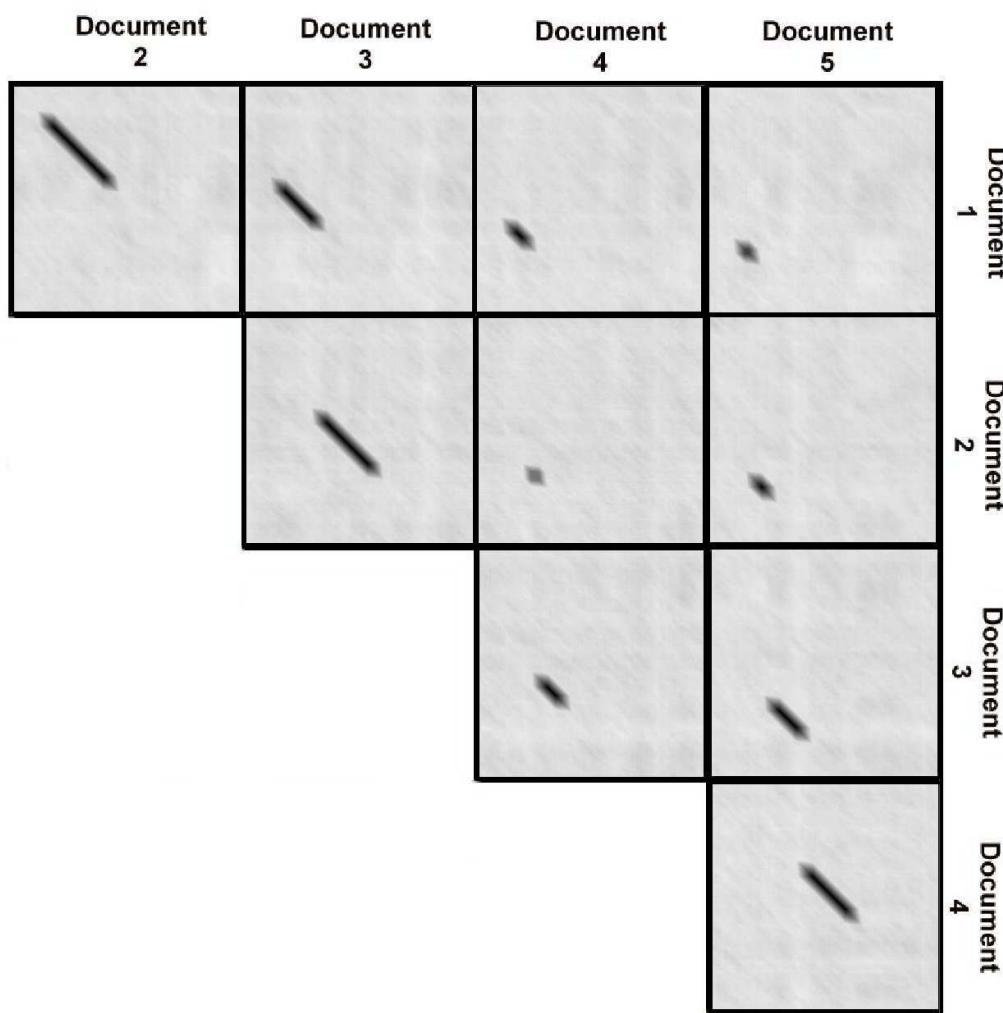


Figure 9.19: Sequenced visualisations for synthetic corpus preserving linguistic properties

Figure 9.20 shows the visualisations generated from the unsequenced metric on the same corpus.

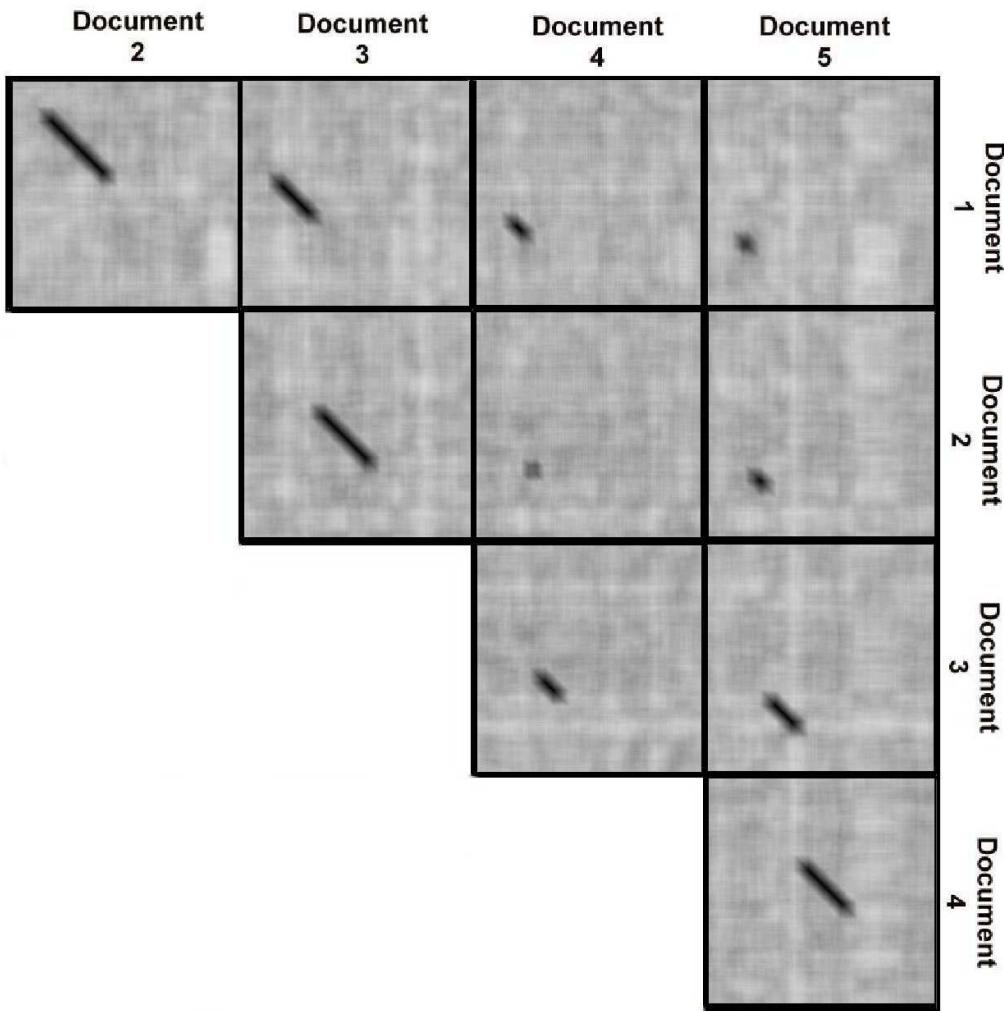


Figure 9.20: Unsequenced visualisations for synthetic corpus preserving linguistic properties

The similarity intersections are again clear but the amount of background noise is greater, supporting the view that the sequenced metric generates more useful visualisations in practice.

Table 9.18 shows the correlations between the simple metrics and the mean of max metrics for this corpus. In the table the names of the simple metrics are abbreviated for reasons of space. The mean of max metrics for both the sequenced and unsequenced mean of max metrics are included since they differ for this corpus and all other corpora maintaining standard properties of English language, such as word sequencing, that the other synthetic corpora have lost. Since $n = 10$ (for all pairings of 5 documents) H_0 is rejected at the 5% significance level for $|r| > 0.57348$ and at the 1% level for $|r| > 0.67914$. Those correlation values significant at 1% with a positive correlation are shown in the table with a grey background. Those significant at 1%

with a negative correlation would be shown inside an additional box, although there are none to show for this corpus. The most significant correlation (largest value ignoring positive or negative signs) for both mean of max metrics is additionally shown in italics and underlined. In this corpus there are multiple most significant correlations so these are all shown in this way.

	<i>chars1</i>	<i>chars2</i>	<i>chars3</i>	<i>chars4</i>	<i>chars5</i>	<i>chars6</i>	<i>chars7</i>
seq max mean	0.321212	0.854545	0.975758	0.939394	0.987879	0.987879	0.987879
unseq max mean	0.29697	0.818182	0.963636	0.963636	<u>1</u>	<u>1</u>	<u>1</u>
	<i>chars8</i>	<i>chars9</i>	<i>words1</i>	<i>words2</i>	<i>words3</i>	<i>words4</i>	<i>words5</i>
seq max mean	0.987879	0.987879	0.963636	0.987879	<u>1</u>	<u>1</u>	<u>1</u>
unseq max mean	<u>1</u>	<u>1</u>	0.987879	<u>1</u>	0.987879	0.987879	0.987879
	<i>words6</i>	<i>words7</i>	<i>words8</i>	<i>words9</i>	<i>sentences</i>		
seq max mean	<u>1</u>	<u>1</u>	<u>1</u>	<u>1</u>	0.781818		
unseq max mean	0.987879	0.987879	0.987879	0.987879	0.721212		

Table 9.18 - Correlations for synthetic corpus containing known English language sequences

The results are highly promising. The only non-significant simple metric is the characters 1 metric, which seems more random since all documents could be expected to contain the vast majority of all possible characters, even with the expanded alphabet. The other character metrics are significant, with characters 5 to 9 having perfect correlation with the unsequenced mean of max metric and characters 5 to 9 for the sequenced metric being as near perfect as makes no difference, the slight deviation likely caused by noise. All the words metrics again provide perfect or almost perfect correlation, with words 2 being the most perfect for the unsequenced metric and words 3 upwards correlating best with the sequenced metric. The sentences metric correlates significantly but the correlations are not as high as those of the words metrics or high valued character metrics. This may in part be due to the number of complete sentences in a given piece of text not being directly proportional to its length. In practical terms just changing one character in every sentence in one of two otherwise identical submissions would stop them being matched.

9.13 - Real corpora

The words metrics and character metrics with larger character chains have been shown to give lists of similar pairs of documents that are both cheap to compute and correlate well with the mean of max metrics that represent the way humans would order the pairs. This is the case on documents whether or not they contain noise and when standard linguistical properties are preserved. But there is no guarantee that these same observations hold on real student submissions.

The tests that follow on real documents have additional aims to further break down the original hypotheses. The first aim is to verify if the same simple metrics that were found to be good on synthetic documents (any words metrics and chains of character metrics encompassing two or more words) also work for real documents. The second is to find which of these metrics, if any, are the best simple metrics that most closely correlate with the mean of max metric. The third is to see if the best metric is

dependent on the corpus being processed or is largely independent for all corpora, that is, is there one simple metric that could be used in place of the mean of max metric in all cases?

9.14 - High similarity corpora

A corpus of 25 student submissions was created. This was obtained by taking part of a large corpus of student submissions known to contain shared material from a previous investigation. Pairs of submissions were taken starting from the highest ranked pair and added to the corpus until it contained 25 submissions. This ensured that the corpus would contain some similarity whilst, at 300 pairs, remaining small enough to process. The corpus contains more similarity than could be expected from a corpus made up of a complete set of student submissions. This should be reasonable for test purposes since tutors are more interested in those submissions that contain similarity than those that do not. This corpus is empathising the importance of a good ranking of the most similar submissions.

For illustrative reasons Appendix I contains the entire set of visualisations that have been generated for the high similarity corpus. This was used to verify that the top section of the ranking list should contain no missed pairs. It is also the only real corpus that is small enough for its visualisations to be usefully included in full.

Table 9.19 shows the correlations between the simple metrics and the mean of max metrics for the corpus of 25 high ranked submissions. Since $n = 300$ H_0 is rejected at the 5% significance level for $|r| > 0.11395$ and at the 1% level for $|r| > 0.14988$. Again those with a grey background have significant positive correlation at the 1% level. Those underlined and italicised are the most significant correlations. There are no negative correlations to highlight.

	<i>chars1</i>	<i>chars2</i>	<i>chars3</i>	<i>chars4</i>	<i>chars5</i>	<i>chars6</i>	<i>chars7</i>
seq max mean	0.090835	0.233338	0.292429	0.339836	0.386112	0.443046	0.486984
unseq max mean	0.432087	0.636088	0.713593	0.756325	0.783082	0.808381	0.821696
	<i>chars8</i>	<i>chars9</i>	<i>words1</i>	<i>words2</i>	<i>words3</i>	<i>words4</i>	<i>words5</i>
seq max mean	0.529184	0.565417	0.36988	0.554634	0.716885	0.716436	0.537201
unseq max mean	0.832183	0.831631	0.796426	0.875789	0.882456	0.789614	0.570454
	<i>words6</i>	<i>words7</i>	<i>words8</i>	<i>words9</i>	<i>sentences</i>		
seq max mean	0.396976	0.303698	0.282371	0.295806	0.148322		
unseq max mean	0.371896	0.277209	0.254916	0.269163	0.295251		

Table 9.19 - Correlations for high similarity corpus between simple and mean of max metrics

The results are satisfying as almost all of the simple metrics correlate significantly with the mean of max metrics at the 1% level. The words 3 metric (chains of three consecutive words) is the most significant, correlating best with both the sequenced and unsequenced mean of max metric.

The sequenced mean of max metric can probably be considered the best indicator of the two, since it produces the clearer visualisations which is the whole background for a visual ordering. For this corpus words 4 is almost as good a simple metric as words 3. If the unsequenced metric is also considered good then the high valued character metrics and the low valued word chain metrics also have a lot to recommend them, echoing the results obtained from the synthetic corpora.

The sentences metric looks unreliable. One reason for this is that this metric looks for sentences that are identical. A sentence common to two submissions but containing just one character or word change will be marked as difference and will not contribute towards a similarity score. If this is repeated throughout two otherwise identical documents a similarity score of zero will be given. This would explain a lack of an apparent ordering between the mean of max metric and the sentences metric.

9.15 - Submitted student corpora

The synthetic corpora and high similarity corpus have all shown that simple metrics that closely approximate visual metrics, namely the mean of max metric, do exist and that words metrics and character metrics with values covering multiple words appear to be the best. None of the corpora investigated so far accurately reflect the kind of corpus that could be expected to be obtained as actual student submissions, either because it is synthetic or the volume of similarity is artificially high.

This section computes the correlation between the simple and mean of max metrics on five actual corpora of student submissions.

Table 9.20 describes the five corpora. The ID is a general identifier. The name of unit is self-explanatory. The faculty is the South Bank University department offering the unit. The level is the year of study at which the student would take this unit (ignoring work placements). The year corpus collected is the year where this unit was studied. Note that where the different corpora for collected for the same unit in different years the assignment specifications were different, so plagiarism checking across the different years is not useful.

ID	Name of unit	Faculty	Level	Year corpus collected
C1	Designing Decision Support Systems	School of Computing, Information Systems & Mathematics	3	2000
C2	Designing Decision Support Systems	School of Computing, Information Systems & Mathematics	3	2001
C3	Management Skills	Business School	3	2000
C4	Management Skills	Business School	3	2001
C5	Business Information Techology	School of Computing, Information Systems & Mathematics	1	2000

Table 9.20 - Details of submitted student corpora

Table 9.21 gives some general statistics about the corpora. These include the number of submissions collected and hence the number of possible pairings (for n submissions this is $n(n-1)/2$).

ID	No. of submissions	No. of pairs
C1	125	7750
C2	133	8778
C3	172	14706
C4	131	8515
C5	131	8515

Table 9.21 - Statistics for submitted student corpora

The table demonstrates the exponential growth in the number of pairs as the number of submissions grows. Since processing a single pair can take around twenty to thirty minutes it is immediately obvious that it will be impossible to compute all possible visualisations (and hence an entire set of mean of max metrics) within an acceptable timeframe.

Table 9.22 gives the mean number of sentences, words and characters per submission in each corpus are given, along with the mean number of words per sentence and characters per word. These are rounded to the nearest integer. The latter two columns demonstrate that although the size and number of submissions may differ, the word and sentence structures vary very little from subject to subject and year to year.

ID	Mean sentences per submission	Mean words per submission	Mean characters per submission	Mean words per sentence	Mean characters per word
C1	140.47	2236	14455	16	6
C2	139.71	2142	13698	15	6
C3	74.18	1169	7786	16	7
C4	70.51	1188	7681	17	7
C5	151.83	2237	12974	15	6

Table 9.22 - Statistics for submitted student corpora

Processing the max of mean metric, a measure of the visualisation produced from every pair of submissions, is not feasible. The computational resources required to do so would be too great. Subsequently it has only been possible to calculate a low percentage of all possible pairs for each corpus. This percentage has varied between 2% and 10% depending on, amongst other things, processing time available (especially when the processing machine was not otherwise being utilised) and the size of the corpus. Some corpora have been processed at both the 2% and 10% levels and it is useful to see below that the rank ordering of the correlations barely changes, suggesting that the 2% level gives suitably good results. The pairs chosen to process were done so simply by taking those ranks exactly divisible by the percentage required on the averaged simple metrics, thus giving what could be expected to be an even spread of similarity levels.

The C1 corpus has additionally been processed in its entirety (the 100% sample level), the results of several months of calculations. This has been done to give some further assurance that the sampling is appropriate.

Table 9.23 shows the corpora as processed, including the percentage and number of pairs processed and the values for the correlation co-efficient at which the null hypothesis of no correlation would be rejected. As the number of pairs processed increases this value gets closer to zero.

Sample ID	Full Corpus ID	Percentage processed	Number of pairs processed	5% significance r value	1% significance r value
P1	C1	10	775	0.07104	0.09371
P2	C2	2	175	0.14866	0.19514
P3	C3	2	294	0.22510	0.15138
P4	C3	10	1470	0.05161	0.06814
P5	C4	10	851	0.06778	0.08946
P6	C5	2	170	0.15101	0.19791
P7	C5	10	851	0.06778	0.08946
P8	C1	100	7750	0.02249	0.02972

Table 9.23 - Submitted student corpora as processed

Tables 9.24 to 9.31 show the correlations for these corpora as processed, highlighted again using a grey background for positive correlation significant at 1%, boxed for negative correlation at 1% significance and italicised and underlined for the most significant positive or negative correlation.

	chars1	chars2	chars3	chars4	chars5	chars6	chars7
seq max mean	-7.2E-05	0.25547	0.297147	0.384093	0.463701	0.524183	0.58023
unseq max mean	-0.07036	0.205501	0.257325	0.338789	0.416432	0.469514	0.518583
	chars8	chars9	words1	words2	words3	words4	words5
seq max mean	0.622671	0.647728	0.572997	0.785282	0.746825	0.599184	0.48321
unseq max mean	0.556613	0.577905	0.606179	0.752379	0.659749	0.515612	0.408954
	words6	words7	words8	words9	sentences		
seq max mean	0.430636	0.395053	0.358786	0.341974	-0.04187		
unseq max mean	0.360745	0.329212	0.30057	0.280371	-0.0003		

Table 9.24 - Correlations for corpus P1

Here words 2 most closely correlates with both mean of max metrics, although words 3 is almost as good in the sequenced case. All correlations are significant at 1% apart from sentences and chars 1.

	<i>chars1</i>	<i>chars2</i>	<i>chars3</i>	<i>chars4</i>	<i>chars5</i>	<i>chars6</i>	<i>chars7</i>
seq max mean	0.117481	0.449648	0.538895	0.659678	0.730914	0.77283	0.79495
unseq max mean	0.04921	0.419698	0.577703	0.706283	0.766017	0.796381	0.807046
	<i>chars8</i>	<i>chars9</i>	<i>words1</i>	<i>words2</i>	<i>words3</i>	<i>words4</i>	<i>words5</i>
seq max mean	0.80347	0.799324	0.759558	0.846848	0.840136	0.692535	0.503375
unseq max mean	0.809803	0.802838	0.825424	0.854574	0.784872	0.608006	0.431675
	<i>words6</i>	<i>words7</i>	<i>words8</i>	<i>words9</i>	<i>sentences</i>		
seq max mean	0.406344	0.371654	0.344762	0.353111	0.162251		
unseq max mean	0.351453	0.311255	0.278494	0.288204	0.195985		

Table 9.25 - Correlations for corpus P2

For corpus P2 the words 2 metric once again correlates most closely with the mean of max metric. In the sequenced case words 3, words 1 and some of the high valued character metrics are also almost as significant.

	<i>chars1</i>	<i>chars2</i>	<i>chars3</i>	<i>chars4</i>	<i>chars5</i>	<i>chars6</i>	<i>chars7</i>
seq max mean	0.134952	0.232452	0.45016	0.575456	0.644826	0.688116	0.70447
unseq max mean	0.103387	0.204975	0.438384	0.580876	0.653352	0.691796	0.701574
	<i>chars8</i>	<i>chars9</i>	<i>words1</i>	<i>words2</i>	<i>words3</i>	<i>words4</i>	<i>words5</i>
seq max mean	0.71132	0.723317	0.629494	0.785121	0.762213	0.703726	0.646461
unseq max mean	0.702245	0.703928	0.673881	0.759558	0.676639	0.605424	0.544919
	<i>words6</i>	<i>words7</i>	<i>words8</i>	<i>words9</i>	<i>sentences</i>		
seq max mean	0.56651	0.510647	0.414824	0.352571	-0.02009		
unseq max mean	0.458115	0.410892	0.315279	0.247898	-0.03283		

Table 9.26 - Correlations for corpus P3

Corpus P3 shows words 2 as the most significant metric, closely followed by words 3.

	<i>chars1</i>	<i>chars2</i>	<i>chars3</i>	<i>chars4</i>	<i>chars5</i>	<i>chars6</i>	<i>chars7</i>
seq max mean	0.185076	0.2605	0.411561	0.524814	0.603267	0.657926	0.685556
unseq max mean	0.173885	0.299149	0.458565	0.577524	0.644524	0.681011	0.693931
	<i>chars8</i>	<i>chars9</i>	<i>words1</i>	<i>words2</i>	<i>words3</i>	<i>words4</i>	<i>words5</i>
seq max mean	0.70732	0.721294	0.586611	0.757517	0.757236	0.693637	0.616341
unseq max mean	0.701833	0.696933	0.676513	0.744462	0.654883	0.573113	0.48603
	<i>words6</i>	<i>words7</i>	<i>words8</i>	<i>words9</i>	<i>sentences</i>		
seq max mean	0.54574	0.478015	0.39626	0.33848	0.024715		
unseq max mean	0.410427	0.348481	0.276017	0.226561	0.013676		

Table 9.27 - Correlations for corpus P4

Corpus P4 comes from the same set of metrics as P3, the only difference is that 10% of pairs have been processed instead of just 2%. It is encouraging to see that words 2 remains the most significant metric, giving credence that taking small word chains gives good results. Words 3 also remains close. The main difference is that characters 1 becomes significant for the first time, albeit barely.

	<i>chars1</i>	<i>chars2</i>	<i>chars3</i>	<i>chars4</i>	<i>chars5</i>	<i>chars6</i>	<i>chars7</i>
seq max mean	0.775029	0.817459	0.804622	0.781464	0.770184	0.7519	0.711804
unseq max mean	0.829552	0.866979	0.84901	0.815735	0.788134	0.753854	0.703751
	<i>chars8</i>	<i>chars9</i>	<i>words1</i>	<i>words2</i>	<i>words3</i>	<i>words4</i>	<i>words5</i>
seq max mean	0.662576	0.603547	0.84795	0.786148	0.572026	0.353714	0.263555
unseq max mean	0.648063	0.582538	0.880857	0.745883	0.505261	0.279466	0.194935
	<i>words6</i>	<i>words7</i>	<i>words8</i>	<i>words9</i>	<i>sentences</i>		
seq max mean	0.224554	0.207657	0.196492	0.186698	0.015637		
unseq max mean	0.16155	0.149534	0.143757	0.139138	0.005393		

Table 9.28 - Correlations for corpus P5

In corpus P5 words 1 is the most significant metric, a result that differs from all the previous corpora. Words 2 remains close, as does characters 7. Unusually characters 1 has a very high correlation. Only the sentences metric remains insignificant.

These correlations would suggest that there isn't a single consistent simple metrics that can be guaranteed to be the most effective metric in any selected corpus, or even for a single subject. This is supported by P3/4 and P5 which were both obtained from the same unit in different years (albeit with different assignment specifications).

	<i>chars1</i>	<i>chars2</i>	<i>chars3</i>	<i>chars4</i>	<i>chars5</i>	<i>chars6</i>	<i>chars7</i>
seq max mean	0.250566	0.375698	0.40973	0.443897	0.484302	0.524905	0.545863
unseq max mean	0.290012	0.428953	0.467072	0.496691	0.528156	0.557348	0.568862
	<i>chars8</i>	<i>chars9</i>	<i>words1</i>	<i>words2</i>	<i>words3</i>	<i>words4</i>	<i>words5</i>
seq max mean	0.559241	0.575901	0.571407	0.696949	0.766865	0.662193	0.433033
unseq max mean	0.570237	0.582347	0.657437	0.735598	0.74688	0.599313	0.340997
	<i>words6</i>	<i>words7</i>	<i>words8</i>	<i>words9</i>	<i>sentences</i>		
seq max mean	0.326221	0.209006	0.143119	0.133985	-0.18481		
unseq max mean	0.240713	0.159195	0.084918	0.056124	-0.25126		

Table 9.29 - Correlations for corpus P6

Corpus P6 contains less significance than any of the other corpora, although it is encouraging to see that many of the same metrics remain highly significant, namely the high valued characters metrics and low valued words metrics. This time words 3 is the most significant, closely followed by words 2 and words 6. This is the only corpus that contains a significant negative correlation for the sentences against the unsequenced mean of max metric. There is no obvious reason for this negative correlation. The magnitude of the correlation is small enough that, although significant, inverting the pair ordering under the sentence metric as an approximation

to the mean of max metric is unlikely to be used. The negative correlation can be dismissed from consideration.

	<i>chars1</i>	<i>chars2</i>	<i>chars3</i>	<i>chars4</i>	<i>chars5</i>	<i>chars6</i>	<i>chars7</i>
seq max mean	0.20552	0.306493	0.352119	0.393358	0.441666	0.490837	0.524558
unseq max mean	0.269579	0.377522	0.438426	0.477466	0.52014	0.561658	0.587045
	<i>chars8</i>	<i>chars9</i>	<i>words1</i>	<i>words2</i>	<i>words3</i>	<i>words4</i>	<i>words5</i>
seq max mean	0.55116	0.564071	0.523862	0.686965	0.763728	0.655993	0.445144
unseq max mean	0.602921	0.606687	0.635972	0.749538	0.771233	0.618193	0.407474
	<i>words6</i>	<i>words7</i>	<i>words8</i>	<i>words9</i>	<i>sentences</i>		
seq max mean	0.347842	0.265961	0.214639	0.187191	-0.07213		
unseq max mean	0.307223	0.233019	0.189138	0.159214	-0.07444		

Table 9.30 - Correlations for corpus P7

P7 is again drawn from the same set of pairs as P6, with 10% processed instead of 2%. The additional pairs do not change the words 3 metric from being the most closely correlated with the mean of max metrics, or change words 2 and 4 as being the next best metrics. The main change is to make more of the words metrics slip into the significance regions and the previously barely negatively significantly correlated sentences metric leave. The increased sample size therefore improves the accuracy of the results, as would be statistically expected. This gives another good indication that the results between processing 2% and 10% of pairs do not differ greatly.

	<i>chars1</i>	<i>chars2</i>	<i>chars3</i>	<i>chars4</i>	<i>chars5</i>	<i>chars6</i>	<i>chars7</i>
seq max mean	0.045428	0.263767	0.302409	0.387083	0.47329	0.540712	0.596621
unseq max mean	-0.04293	0.22462	0.271612	0.354742	0.439187	0.499419	0.548374
	<i>chars8</i>	<i>chars9</i>	<i>words1</i>	<i>words2</i>	<i>words3</i>	<i>words4</i>	<i>words5</i>
seq max mean	0.641136	0.669544	0.574009	0.781651	0.768437	0.643252	0.548162
unseq max mean	0.585741	0.60651	0.598956	0.746455	0.667047	0.540839	0.453954
	<i>words6</i>	<i>words7</i>	<i>words8</i>	<i>words9</i>	<i>sentences</i>		
seq max mean	0.493654	0.449873	0.419851	0.393192	0.003063		
unseq max mean	0.407314	0.36773	0.347621	0.321546	0.013997		

Table 9.31 - Correlations for corpus P8

P8 is the 100% complete processing of corpus C1 and a follow on from the 2% processing P1 and 10% processing P2. As it is the only corpus that has been processed in its entirety it is perhaps the most important. The most crucial observation is that words 2 is again the most significantly correlated metric with both the sequenced and unsequenced mean of max metrics. Words 3 is again nearly as significant. This gives a further indication that the sampling process is a good indicator. Sentences is the only insignificant metric, suggesting that it is useless as there are few complete sentence matches. Chars 1 correlates, but barely and as before is not a good indicator of similarity.

Taking all of these results into account Table 9.32 shows the original five corpora and the most significant metric in each case. Since the metric is the same for both the sequenced and unsequenced case and whether 2%, 10% or 100% of the pairs have been processed no further distinctions are made.

Corpus	C1	C2	C3	C4	C5
Metric	words 2	words 2	words 2	words 1	words 3

Table 9.32 - Most significantly correlated metrics for real corpora

The results seem fairly conclusive, showing that words 2 is the best overall metric in the majority of cases, although on occasions the neighbouring words 1 and words 3 can provide slightly better correlation. Almost all the remaining simple metrics correlate well with the mean of max metrics, suggesting that they would be suitable, but none are optimal. The main exceptions are the single character metric and the sentence metric which give a pretty random ordering as far as correlation is concerned. Again most of the high valued character metrics and low valued word metric seem to give good results.

Taking into account the speed of processing issues, where words 1, 2 and 3 are quicker to compute than the high valued characters metrics (due to the lower number of possible chains) it would seem that the best consolidated ranking would involve any or all of these.

For ease of computation, being top ranked for three of the five corpora and the fact that it is highly significant in both the other two cases words 2 has to be the best overall simple metric. If there are suitable resources available the lists from words 1 and words 3 can also be checked but there is not likely to be much different between the three lists.

9.16 - The words pair metric

The words 2 metric has been shown to be the most effective simple metric out of those tested. Since the metric identifies the number of similar and dissimilar pairs of words in two submissions it will be better known as the *word pairs metric*.

The generic words metrics have already been specified. This is a more specific description of the word pairs metric as the most suitable simple metric found for calculating document similarity for plagiarism detection.

A method of formally calculating this word pairs metric would be:

Take two documents labelled A and B. Reduce this to a lists of all possible consecutive words pairs in alphabetically order, ignoring case and punctuation. Words are defined as the series of characters between whitespace.

From these lists compute the following values, c1 and c2 for common word pairs and u1 and u2 for unique word pairs:

c1 – the number of word pairs from document A that also exist at least once in document B (the value of c1 is the number of times the word pair exists in A).

c2 - the number of word pairs from document B that also exist at least once in document A (the value of c2 is the number of times the word pair exists in B).

u1 - the number of word pairs from document A that do not exist in document B (the value for u1 is the number of times the word pair exists in A).

u2 - the number of word pairs from document B that do not exist in document A (the value for u2 is the number of times the word pair exists in B).

Then the similarity score for the word pairs metric, a value between 0 and 100 where higher values represent greater similarity, is:

$$\frac{100(c1+c2)}{(c1+c2)+(u1+u2)}$$

Tables 9.33 to 9.35 give an example of calculating the word pairs metric for short documents A and B. Table 33 shows two documents. Table 34 splits the documents into pairs of words, sorted into alphabetically order and shows the effects both counts have on c1, c2, u1 and u2. Table 35 shows the total of these counts.

A	"The cat sat on the mat. The cat was hungry."
B	"The black cat wanted its supper as it sat on the black mat."

Table 9.33 - Two short documents

word pairs from A	word pairs from B	contributes to			
		c1	c2	u1	u2
	as it				1
	black cat				1
	black mat				1
cat sat				1	
	cat wanted				1
cat was				1	
	it sat				1
	its supper				1
mat the				1	
on the	on the	1	1		
sat on	sat on	1	1		
	supper as				1
	the black				1
	the black				1
the cat				1	
the cat				1	
the mat				1	
	wanted its				1
was hungry				1	

Table 9.34 - Counts of similar word pairs for documents A and B

totals			
c1	c2	c3	c4
2	2	7	10

Table 9.35 - Total counts of common and unique word pairs in A and B

The similarity score for the word pairs metric for documents A and B is thus:

$$\frac{100(2+2)}{(2+2)+(7+10)} = \frac{400}{21} = 19.048$$

This value is a measure on the chosen scale between 0 and 100, but does not represent a percentage of similarity. Instead it would be used as part of an ordered list of paired submissions to help decide which pairs to investigate further.

9.17 - Conclusions

Due to the sheer computational complexity it has been impossible to process in their entirety real corpora of student submissions, but a representative sample of each has shown that the word pairs metric closely correlates with the visual metrics believed to represent the ordering a tutor would use when investigating similarity. Operationally this metric can be used as part of the four-stage plagiarism detection process with little chance of error, whilst keeping processing time to a minimum. Although only a representative sample can usually be processed all indicators suggest that the results would hold if the sample size were extended or more corpora were introduced. The results also do not contradict those obtained from synthetic documents containing noise and controlled similarity levels.

One problem that has become apparent is the lack of anything resembling standard corpora for testing plagiarism detection engines. Some verified combination of documents, perhaps both synthetic and actual submissions, would be very useful for testing that detection engines work as intended. Such corpora would need to be human verified to ensure that all pairs that should be flagged as similar are known. The synthetic corpora with a known ordering and words generated from a standard text, as used in this chapter, would be a good corpus for this purpose.

Chapter 10: A Complete Detection Process

Overview:

- A complete, tool supported detection process is proposed.
- The main contributions of the thesis are reviewed.
- Future areas of research are suggested.

The thesis has seen the development and refinement of efficient and effective methods for finding plagiarism. This chapter incorporates these into an entire detection process, detailing appropriate tool support and suggesting how it can be used within the four-stage plagiarism detection process.

As this is the final chapter it will also be used to bring together the original work that has been done. It is noted that plagiarism detection in free text is still only a recent area and there is still a need for further research. A number of possible research areas are suggested.

10.1 - Motivation

This thesis has explored methods of detecting plagiarism in free text student submissions. In particular methods have been considered from viewpoints of efficiency and effectiveness.

This chapter concludes the thesis by taking the ideas already expressed and building them into a complete plagiarism detection process. Three prototype tools are described to cater for different stages of the four-stage plagiarism detection process. The most important of them is the Visualisation and Analysis of Similarity Tool (VAST). This tool links the similarity visualisations and associated text files together through an interactive interface.

Plagiarism detection is still a new area of research and although the thesis has tried to give a flavour and overview of what is going on it has only been able to focus in on a relatively small area, namely efficient and effective technical detection. A number of other areas have been identified, both during the literature review and during general exploration where research is necessary or would be beneficial. These ideas are presented in Section 10.8 and will hopefully serve as a starting point for other researchers. Section 10.7 also describes some of the main contributions to the body of detection knowledge to come out of this thesis.

The VAST tool, described in Section 10.5, has been presented in the paper 'Visualising Intra-Corporeal Plagiarism' which is included in Appendix H.

10.2 - Requirements of a complete detection process

Chapter 2 found two main requirements for a plagiarism detection process, namely effectiveness and efficiency.

Effectiveness was defined as a process that sensibly ordered pairs based on the likelihood that they contained plagiarism and the extent of such similarity. This meant that pairs containing similarity that should be investigated would be highly ranked. A side effect of the effectiveness requirement is that the rank list should contain no missed pairs, that is pairs are similar but are not ranked in the upper portion of the list.

An efficient detection process was defined as one that takes minimal time to do this. This includes both computer processing time and the time tutors spend investigating. This can be split into a number of auxiliary requirements. The computer processing necessary during the analysis stage should take a tolerable amount of time for a medium sized corpus of, say, 100 to 200 students, giving a list of pairs that would be ready almost immediately. The verification and investigation stages should be tool supported so that tutors can quickly look at similar pairs. The ranking list should avoid false hits, since these introduce extra pairs to check, although if the verification stage is successfully supported this should be a less major problem. The list returned should be non-capricious so that there can be no arguments that a student was only detected cheating due to a whimsical process. Ideally the list should be evidential, giving details why the pairs were highly ranked. In the most basic sense this could be through a percentage measure of similarity.

There are also a number of other recommendations that can be made for a detection process. There needs to be a defined sequence of steps to follow in order to successfully find plagiarism cases. Throughout this thesis the four-stage plagiarism detection process has been recommended. Such a process should be part-automated or automated at each stage. This can be done using a tool for corpus collection, one for analysis and another for the verification and investigation stages. To be appropriate the tools should be designed to be as usable as possible whilst hiding complexity from the users.

The automated detection should be supported by institutions having a clear plagiarism policy, so that students know what is and is not acceptable. There needs to be a clearly defined series of steps laid out for tutors to follow. This should ensure that all plagiarism cases are treated seriously and identically and that an institution is not liable for litigation from students.

Sections 10.3 to 10.6 describe a number of tools that can be used to support the detection process. Ways of making the process effective and efficient were described in Chapters 6 to 9.

10.3 - Student Submission System

The Student Submission System (SSS) is a prototype system based around a collection of scripts written in the TCL language. Students are directed to an HTML page that contains instructions and hidden tags detailing what can be submitted and directing them towards a pre-created storage area. Documents can be collected in formats such as RTF or Word where layout is important, or text format, for instance for source code submissions. The system checks the student's identity based on their existence on known records and details of which students have submitted work can be regularly submitted to an appropriate tutor.

The system has proven to be functional and robust, although lacks the features that would be needed for commercial or more widespread use. It has been used mainly to provide source material to test the other systems described in this chapter on.

10.4 - Text Ranker

The analysis stage of the four-stage process requires a comparison of all possible pairs of submissions under different metrics. The prototype tool Text Ranker (TRANKER) has been developed to aid with this.

TRANKER is a simple Java-based tool that at present has no user interface. Instead it is executed from the command line, with an input folder, an output file name and a choice of metrics specified as arguments. TRANKER returns a list of all possible pairs of submissions along with a similarity score for each under the chosen metric. TRANKER supports all the simple metrics that were introduced in Chapter 7. It was argued in Chapter 9 that the word pairs metric appears to be the best one for finding similarity and it is suggested that this one be used here.

The resulting output from TRANKER can be sorted in a spreadsheet, giving an ordered list of pairs. Operationally it is suggested to start at the highest ranked pair and work down the list. The VAST tool described in the following section can be used to aid with this.

Although TRANKER only really exists as a collection of source code files it has been used successfully during production of this thesis and locally to process student submissions to find similarity. There is no reason why a user interface could not be added along with automated sorting and display of results if so desired. The PRAISE tool, described in the literature review in Chapter 4, has been used to do this.

10.5 - Visualisation and Analysis of Similarity Tool

The verification and investigation stages of the four-stage Plagiarism Detection Process are the two stages that involve the most work for tutors. Hence they are the stages that could most benefit from computer assistance.

Chapter 8 described the new graphical technique of similarity visualisations. The visualisations show roughly where submissions are similar, the size of the segments for which they are similar and the extent of the similarity throughout those segments.

This original method of viewing similarity is presented as a two-dimensional image plot. The visualisations were shown to clearly differentiate between areas of two documents that did or did not contain similarity worth investigating.

The similarity visualisations in themselves are a very useful as part of the process of verifying if plagiarism has occurred. However there is no easy way for a tutor to find the parts of a pair of documents that a given similarity intersection represents. This is a problem ideally suited to computer assistance. The Visualisation and Analysis Tool (VAST) has been developed as a prototype system written in Java to guide the tutor around the documents. This means that plagiarism can be quickly investigated and false hits quickly eliminated from consideration.

Figure 10.1 shows VAST being used to investigate possible plagiarism. The display is made up of four windows, the two windows on the right containing two texts format documents. The uppermost left window contains the associated similarity intersection. The final window contains some status information. The tool also contains a number of menus for selecting files to investigate and resizing the similarity visualisation.

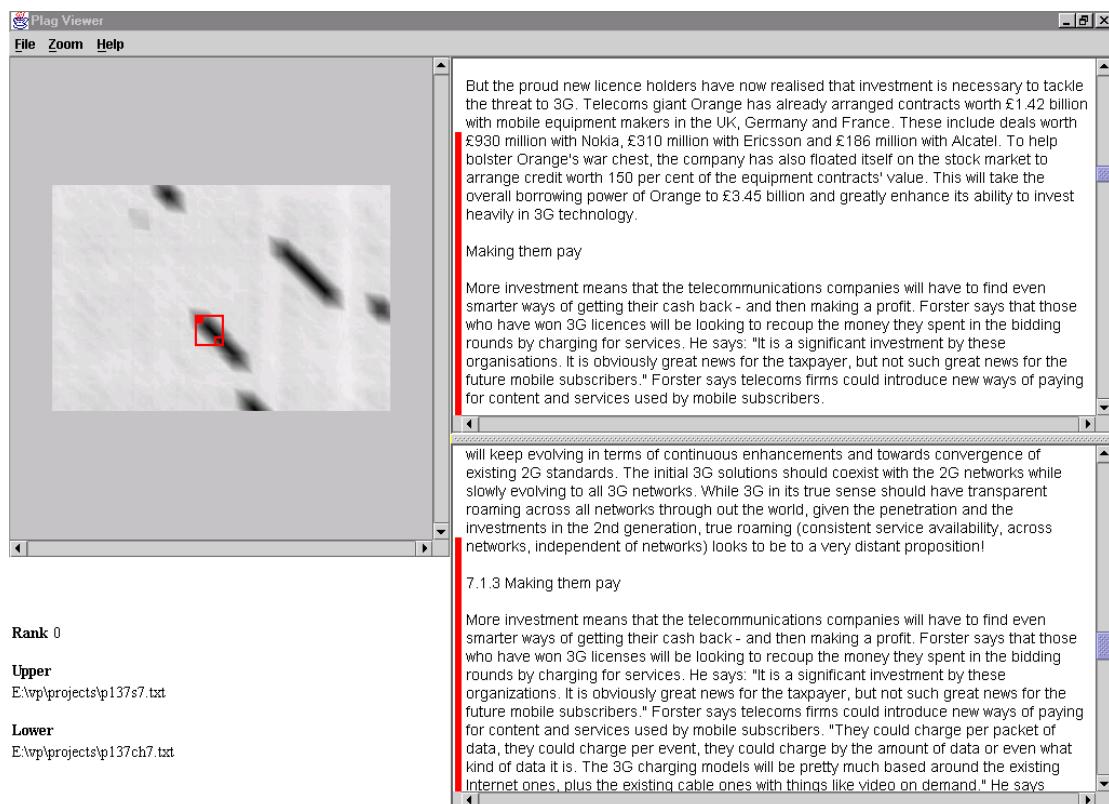


Figure 10.1 - VAST in use

Here VAST is being used to investigate a student project, where portions of a chapter have been found during a Web search. The possibly source of concatenated Web pages is shown in the upper window and across the x-axis of the visualisation. The chapter, as submitted by the student, is shown in the lower window and down the y-axis of the visualisation. The user highlights an area of the similarity visualisation that is of interest and the corresponding sections of the text are automatically navigated to in the windows on the right. This means that all the similarity intersections can be quickly verified to see if they represent intra-corporeal plagiarism.

The user selects a similarity interception using the *similarity interception selector*, the red rectangle in the navigation window. Users can click and drag the small rectangle in the top left corner of the similarity interception selector in order to move it around the similarity visualisation. They can alter the size of the selector by clicking and dragging the rectangle in the bottom right corner of it in order to view the text represented by larger or smaller similarity interceptions. The sections of the two texts associated with the area selected are marked with red side bars and can be scrolled through if they extend beyond the viewable window.

The similarity visualisation can be scrolled about if it is bigger than the window it is in. There are also options available on the menu to zoom in and out of the visualisation. This is useful if the visualisation is a non-standard size, or a user wants to find out exactly what parts of a document a small intersection represents. The zoom is merely a graphical zoom, that is for a two times zoom a single pixel is represented as four pixels, possibly with some automated smoothing. No attempt is made to recalculate localised similarity scores for the fragments and generate new pixels. This is a procedure which could not be done live and which can be a slow process for large documents, so it would slow down the use of the tool too much. Hence the zoomed visualisations can only ever be considered to be close approximations and so the marked text may be a few words out.

The section of text highlighted in VAST shows an example of extra-corporeal plagiarism, with little attempts at disguise and some small attempts at restructuring. The submitted chapter contains uncited material from several sources. Since the similarity intersections cover almost the entirety of the y-axis it can be seen that almost the entire chapter is plagiarised from different places as seen with the intersections in Chapters 8 and 9.

At present VAST only exists as a proof of concept prototype, but has been demonstrated many times at different talks and conferences and has been well received. All reactions to the tool have been favorable and its utility has been readily, though not immediately, apparent. A few prospective users were simply told that it was a tool to assist with plagiarism verification and invited to interact with it. In these cases neither the interpretation of the image nor the relationship with the text panes was self-evident and instruction had to be given before effective use could be made of it.

The majority of users have required a short verbal introduction to what the tool is intended for, what the visualisation indicates and how the selector relates to the text panes. Following this introduction all trial users have been able to successfully navigate around the documents, verifying the existence of plagiarised parts. This tends to suggest that although the tool could not be used on a 'walk up' basis only a minimal amount of instruction would be required. All of the tutors that have used the tool have indicated that they think it would be of use to them in the validation of plagiarism and most have expressed interest in having access to it when it is available.

Since the verification and investigation stages of the four-stage process are the most time consuming more tools like VAST would seem to be potential for automated anti-plagiarism techniques to be used by their intended audience. As such VAST makes a substantial contribution towards this.

10.6 - Tool use in a complete detection process

Chapter 2 described the four-stage plagiarism detection process. This process was seen as beneficial since it provides a clear process inside which suspect submissions can be found and investigated. It clearly splits the process of detection from any allocations of guilt or misconduct.

Later chapters have argued about the relative merits, technical and operational difficulties of the different stages. The opening collection stage where students submit their work electronically so that they are suitable for automated processing was seen as fairly routine. The second analysis stage was seen as essential, but it was noted that tools do exist to assist with this and many relatively basic metrics do seem to be successful in identifying some similarities.

The two later stages, verification and investigation, were seen to be the two stages where machine assistance would be most welcomed. Verification is where submissions judged to be similar are manually checked, an essential stage to ensure that no student is falsely charged with plagiarism. The investigation stage is where evidence is collected about the plagiarism suitable for some kind of internal hearing. These stages involve a lot of work for tutors, not least because the simple process of finding similarity in two submissions is believed to be an involved one, particularly if the similarity appears in different pages of the two submissions. Any kind of support to ease this process along is welcome as it is an area that has been largely ignored by those academics working on detection so far.

The tools presented in this chapter can therefore be used to support a whole detection process. SSS, although basic, is quite suitable for use in the collection stage. Likewise TRANKER can be used to support the analysis stage. Chapter 9 argued that the word pairs metric is the most appropriate to be used in TRANKER, so this should be used as the basis for a viable detection process. The words pair metric has been seen to largely avoid missed pairs and in Chapter 7 was also seen to successfully identify similarity that had been manually introduced into a corpus.

The biggest contribution towards the whole four-stage process is VAST. The importance of VAST is that it presents new view of two student submissions, as represented by the similarity visualisations, and provides an immediate and efficient way of navigating around them. Having the visualisations available has already been shown to allow false hits to be quickly eliminated from contention, since these shown to either contain or not contain similarity intersections. Likewise since these intersections can be directly selected and the appropriate areas of similarity shown a tutor can decide whether they represent undue similarity or perhaps legitimate shared citations, again allowing false hits to be eliminated and speeding through the verification stage.

VAST can also support the investigation stage. This is for much the same reasons as supporting the verification stage. Two submissions can be quickly compared and the tool used as a visual method of comparing them and marking up similarity. The tool is believed to be suitable for use in hearings where plagiarism cases are being discussed since similarity can be shown with little prior experience of using VAST being necessary. Alternatively VAST can be used as an aid to manually mark up student

submissions where such information needs to be more widely circulated. The tool can also assist in deciding on the severity of a case, since the number, location and relative size of similarity intersections should allow major and minor plagiarism to be classified.

Overall the tools can be seen to give a complete four-stage process, of which VAST is the most valuable tool of the three. It can also be integrated with the different detection tools that may in use at academic institutions. This is because it can be used to compare any provided text submissions, either as a list of suspect pairs is supplied or a corpus of submissions supplied with possible Web sources.

10.7 - Main contributions of thesis

This thesis has described and derived an efficient and effective process for plagiarism detection in free-text student submissions. The most important contributions have been:

- A non-ambiguous language with which to talk about plagiarism, described in Chapter 2.
- A part-automated process in which plagiarism can be detected, described in Chapter 2.
- A review of state of the art detection in source code, described in Chapter 3.
- An improved way of classifying detection systems, removing old inconsistencies, described in Chapter 3.
- A review of detection in free-text, an ongoing area, described in Chapter 4.
- A comparison of visual methods that can be used for plagiarism detection, described in Chapter 5.
- The development of an improved visual method for comparing two submissions and placement in a tool, described in Chapters 8 and 10.
- A demonstration that the word pairs metric is effective and error-free, in Chapters 7, 8 and 9.
- A complete part-automated efficient and effective process, in Chapter 10.
- Identification of research still to be completed, in Chapter 10.

Some of these contributions have been published, namely on the idea of similarity visualisations and the associated VAST tool, along with demonstrating error-free simple metrics and general plagiarism background. These papers are included in the appendices.

Probably the most important contribution is the answer to the main research question: 'How can plagiarism be detected effectively and efficiently?' The answer is to use a tool assisted version of the four-stage plagiarism detection process, using the word pairs metric to generate a similarity rank list and to use VAST to verify and investigate the similarity.

10.8 - Further research

Plagiarism detection is not a completed research area, in fact it is far from it. Although tools have been around for a number of years to detect source code plagiarism they are only recently appearing for free text plagiarism. As such this thesis can in many ways be considered an initial study into possible detection. During the preparation of this thesis a number of areas that are both directly and indirectly related to plagiarism have been identified where further work is needed. Some areas would be suitable for undergraduate student projects, others for research and some for further PhD theses. Most of these ideas are original and have not been previously discussed in the plagiarism literature. The areas for research are presented here, some in more detail than others. In each case some ideas about how to proceed are given.

Student Cheating

Franklyn-Stokes and Newstead provided the most involved study into how and why students in the UK cheat to date, but this was carried out before the widespread use of the Web (Franklyn-Stokes & Newstead 1995). It is believed that students use this to plagiarise, so results today could be very different. The practices of widening access into higher education and increasing financial pressures on students during their studies might also have affected students and the way in which they approach assignment specifications.

A full study of student cheating in the UK is needed. At the very least this needs to identify the types of cheating students are involved in and how frequently they are involved with it. A wide study should include a variety of different types of institutions and of students. It is also important to find out how current teaching practices could be changed based on this information. Knowing how students plagiarise could also inform how best to create tools to reduce the chances of such plagiarism being successful.

This is an area in which any number of related studies could be possible, but tracking cheating across time or through different stages of education might be of interest. One of Franklyn-Stokes and Newstead's studies compared students' perspectives of cheating with what they said they were carrying out and this could be replicated. The availability of technical methods for finding plagiarism could also be used to assess how common this is in practice. This could be further used to compare against the believed and admitted results.

Verification and Investigation of Plagiarism

VAST has been presented as one possible tool for aiding tutors in verifying and investigation similarity. Although it is believed to be better than anything else out there it is probably not the best tool that could be created.

A number of ways that VAST could be improved come immediately to mind. To start with the current system is only a prototype and a release candidate version could be prepared, perhaps to be made available over the Web for use. This could be linked with a ranking system such as SSS to allow likely pairs to be identified and appropriate visualisations generated automatically.

The similarity visualisation of a very useful way of viewing where two submissions are the same, but it is not the only possible way. Another view of the same information could be presented. One method of doing is might be by linking the text files with hyperlinks, such as the way MOSS or JPlag work for source code. This could then be linked with the visualisations to allow different ways of scrolling around the texts. The hyperlinks could be generated through a comparison of the submissions for long and similar strings, or could be perhaps marked up automatically through finding intense pixels in the similarity visualisations or high values in the underlying matrix. A percentage measure of similarity might also be useful to give a more immediate measure of how similar two submissions are. At present this needs to be estimated by looking at the visualisations.

One further addition that would be useful would be a way of using VAST to generate printed versions of the submissions that are automatically marked up with areas of similarity. Such marked up submissions would be useful for distributing, perhaps before a meeting of a board assessing plagiarism cases. One way of doing this might be to print a view of the documents automatically marked up based on the similarity visualisation, with suspect areas presented on the same page. A pre-processing stage might allow a tutor to check the similarity intersections using VAST so that areas judged not to be significant would not be included.

Similarity Visualisation Generation

The process of generating similarity visualisations is not optimised for efficiency. A mathematically interested person might well be able to improve the algorithm used.

A possible approach is that adjacent pixels in a visualisation provide much the same information. That is they probably are generated from fragments that are 90% similar. Perhaps a different underlying matrix could be prepared that is more efficiently calculable and this matrix used to generate the fragmentary intercept matrix or to directly produce the visualisations. Alternatively there might be some progressive calculations that can be made, so that a visualisation could be initially presented at a very high and approximate scale, with only those areas of interest calculated and scaled in more detail, thus giving a time saving.

Protection of Copyright

Many companies invest a lot of time and money into creating their intellectual property. They would be keen to know that it had been stolen. The same techniques used to find the level of similarity in student submissions could also be used to find similarity in intellectual property. One method might be to find a document that should be protected, then find similar documents on the Web. Visualisations could be generated for each to find if intellectual property is being used without acknowledgement or appropriate royalties being paid.

A similar and related use would relate to version control within organisations. With multiple people working on documents it is often hard to see how they have been changed. A visualisation technique like the one used in VAST could be used, probably with the visualisation inverted, so that the areas that were most different

would be most intense. These could then be used to create a master document incorporating all changes.

Usability of Detection Tools

The Web-based detection services say that they are easy to use and successful in finding plagiarism but there is little if any independent verification of this. A usability evaluation of the different tools and their different interfaces is needed. This could be used to compare the effectiveness of the different approaches, along with how easily users can adapt to the different interfaces. Any issues with successfully investigating similarity could be circulated and used to improve the tools and to improve VAST.

Comparing How Tutors Manually Investigate Similarity

There has been an assumption throughout this thesis that tutors when told that two documents require investigating by hand might not find all the similarity and might find that accurate comparisons are a slow process. This has been supported by some unpublished basic tests. An investigation might see tutors given a number of specially prepared documents, perhaps in a subject they are unfamiliar with, and asked to find which pairs contain the most similarity and where they are similar. There could be careful placement of similarity within the submissions to see if even a vigilant tutor can find it all.

A further comparison could use another group of tutors or the same tutors and ask them to investigate the similarity using VAST. The tutors could be split between Computing and non-Computing academics to see if the usability of the tool is affected by the academic discipline of the user. Some recognition of the fact that not all Computing tutors are experienced at computer use would need to be made here. The time taken to investigate similarity and the success of the investigations would give further indications that VAST is a useful tool in practice.

Authorship Analysis of Submissions over Time

Although general authorship attribution techniques are not believed to be directly applicable to plagiarism detection the techniques could be used to see if students have written a piece of work that they claim to have written. One way of doing this would be build up an archive of work from a given student over time. This would allow a number of general measures, ranging from simple numeric metrics to word usage patterns for the students to be built up. These could then be tested on future submissions as they arrive. Any that deviate greatly from the expected values could be considered suspect. This has an immediate use that since although Web plagiarism can be detected with some level of success students who are paying other people to write things for them will not be detected by such Web searches.

There are some difficulties with this approach into which thought would need to be put to see if they could be overcome. One is that this assumes that a student's writing style is consistent. This is a debatable point but there are sure to be some common features that could be found if automated methods of doing so exist. A related point is that a student's writing style is expected to improve over time. This might again make

it difficult to identify consistent features. Perhaps this could be avoided by only comparing style to recent submissions.

A further problem would be finding a suitable record of work done by a student in the first place which is not plagiarised. After all the early submissions will have nothing to compare them against. One way around this might be for a student to complete work in class that could be used as an initial corpus, but there are operational difficulties of doing all initial work within class. It is also debatable whether students under class time pressures will write in the same style as those who with a longer writing window.

Such a corpus would also be useful to assess how a student's writing style changes over time and could be used to assess changes in the volume of intra-corporeal and extra-corporeal plagiarism over time.

Plagiarism of Different Documents Types

Work on plagiarism detection has been focused on submissions that can be written in some textual form. The main areas are free text and source code. There are also other areas in which students can plagiarise and in detection methods are needed. One example is on diagrammatical designs, for instance in UML. In such diagrams components can be re-arranged, renamed and resized, which would make detection very difficult, particularly if diagrams are only submitted in graphical formats. One way might be to accept diagrams from a particular tool, reduce them to a known format in a constrained text and use appropriate metrics.

Other areas in which more expertise would be useful include plagiarism in music, which is an area that often appears in the media where one song is found to sound much like another. Plagiarised spreadsheets and scientifical data need detection methods, especially since the results can be considered to be a crucial part of a scientific experiment and are open to manipulation. Even art can be plagiarised, for instance when one painter creates a picture very like another. Some visual method of finding similarity would likely be necessary here.

Even within text there is little guarantee that a metric or method that is suitable for one sort of document will be suitable for the rest. One example of this is the difference between source code and standard free text, where different metrics are appropriate. Another area for consideration is the different between short and long documents as the visualisation methods used by VAST are only really appropriate for longer documents. Different subject areas might require different detection methods, for instance a Law essay might be considered to be made up of standard terms, whilst an English essay may be more freeform. Further short answer questions, or pieces of poetry might require a different approach to longer essays.

Some concern has been expressed that documents can be plagiarised by converting them into a foreign language and then reconverting them into English. There are services on the Web, such as babelfish.altavista.com that can do such translation. However translation is also not likely to give a sensible document without further work by students. Perhaps a metric taking the essence of documents and comparing them is another way forwards.

Different Views of the Same Document

A little bit of a tangent perhaps, but one technique that would help students create properly formulated essays with citations and would help technical detection systems minimise false hits would be some kind of mark up tags for student essays, perhaps in XML. One problem with detection is that sections of text that are quotes are not filtered out from consideration, perhaps because the techniques to find citations are difficult to apply or perhaps because this may eliminate plagiarism. A format of tags would be one solution for this, a student could tag citations and mark references in an appropriate format. The content of citations could be checked, perhaps using a database or human assistance and then these eliminated from further consideration, which would eliminate false hits by only comparing uncited areas and also find suspect areas which have been incorrectly cited.

Another use of the tags might be to mark up answers to different short question assignments. This would allow only answers to the same questions to be compared to one another and the questions ignored, which might otherwise give false hits. Different views could be generated from different tags for different purposes. A tutor could manually annotate using further tags when marking.

Clustering

VAST appears to be an effective tool when two submissions have been judged to be similar to one another. But it is only useful for comparing two things at once. Sometimes students are believed to collaborate in groups and it would be useful to identify and be able to investigate such clusters.

Visual methods of investigating clusters are needed. A refined version of the VAST display might be one possibility for three documents, showing three views at once with three linked visualisations and documents, or by showing two documents and containing tabs to switch to any other two documents. This idea could of course be extended to larger numbers of documents, but the view might not be appropriate and so other ways of visualising clusters might be necessary.

Such a method assumes that clusters can be identified. There is some further work necessary on metrics to find clusters of similar submissions.

Linguistic Techniques within Documents

Chapter 4 described two tools that attempt to find plagiarism in student submissions by plotting how the linguistic properties change as the documents progress. These could be used to find areas where the style is errant, perhaps indicating plagiarism, or used as a general writer's toolkits.

Although the techniques used did not appear to be completely successful there should be scope for further investigation of different techniques, perhaps tailored to each individual writer based on general properties of the documents pulled from a text concordance tool. A further possibility might be to join several documents by a given author together and use the tools on those. This would give a greater word count and

hence a greater chance for a style to be consistent, but would show up more areas or perhaps whole documents where the style was errant.

One metric that has not been properly tested, but appears to be a stylistic measure used by tutors investigating by eye alone, is the change of language between UK English and US English. There are various different spelling of words that could be identified. A sudden change might indicate a section of text pulled from another source. This is only one possible method that tutors use based on inspection of text and others could be identified and tested.

Glatt Word Replacement

The process of removing words from documents and asking their alleged writer to replace them accurately has been said to be one method of plagiarism detection where a source cannot be found. But there are problems with this approach. The hypothesis may be ill founded particularly when a submission has been produced under a short time span. Also there is little evidence as to where the difference between success and failure lies. An investigation might find an appropriate boundary, perhaps by testing volunteers on submissions that they have and have not written.

Another problem with the approach is that the words that are asked to be replaced are very haphazard. For instance one version of the technique says that every fifth word should be removed. This will invariably lead to a large proportion of these words being standard filler words, such as ‘the’ or ‘a’ that are easily replaced with no knowledge of the subject matter. A better method of selecting words might be possible, perhaps by looking for a proportion of words with certain properties.

Other Metrics

This thesis has tested a number of metrics but it has not tested a complete set, if such a set exists by any means. In particular the metric based on zipping files and comparing their respective sizes appeared mid way through the completion of this thesis and indications are that it might be suitable for further investigation. It might also be that there is another combination of simple metrics, perhaps weighted in some way, that can better approximate a human ordered list. There is scope for further investigation to find such metrics.

Applications to Source Code

The visualisation method introduced in this thesis, based around the idea of similarity visualisations and the associated VAST tool, has been shown to be useful for investigation within free text. It would be interesting to find out if the tool, or some refinement of it, would operate successfully on source code submissions.

A related question asks if any of the metrics that have been identified in this thesis are appropriate for source code, with or without tokenisation. In particular the word pairs metric, although simple, has been shown to be very effective in free text and could be tested on source code submissions to see if it is as effective.

Methods of Detecting Web Plagiarism

One area that this thesis has not really touched on, but is an important area of research worthy of many theses on its own, is that of effective methods of detection Web plagiarism. That is identifying the Web sources and presenting the results to the user.

The commonly used method is simply to submit parts of a document to search engines and compile the results. There is little evidence as to how successful this method is, how efficient it is and how much of the actual plagiarism is uncovered. The problem is further influenced by the fact that different search engines provide different results at different times and that pages that have been plagiarised from can disappear. Also pages can be hidden inside internal databases that will never show up on a Web search.

A number of other possible methods for finding Web sources need to be explored. One possibility is a manual search by a tutor for likely sources. This would allow the tutor to access databases. A second possibility would be to find all the URLs mentioned in student submissions and trawl them. This would allow pages that one student has cited but another has not to be identified. A third possibility would be an automatic Web search based on sections of the student submissions picked out by a suitable text concordance tool. A fourth search might be automatically done by keywords.

The results from the different searches could be compared based on their usefulness, especially if an ideal Web corpus was known. These documents could be added to a corpus of student submissions and compared to find out which submissions might have been plagiarised from which Web sources.

Recreating Documents with Web Sources

VAST has been demonstrated working mainly on intra-corporeal plagiarism. The tool can be used on extra-corporeal plagiarism, by comparing a submission with known sources, where such a source can be identified.

At present this mainly works by simply plotting one submission against one source, or concatenating a number of sources together. This can be wasteful, both computationally and visually, since large areas will be compared without any similarity.

One alternative method might be to attempt to recreate documents solely using sections of Web sources. Each would be appropriately marked up to say where it had been sourced from. This should then give a close approximation to a student submission where it has been heavily plagiarised, which should produce a series of similarity intersections down the primary diagonal of a similarity visualisation. An automated method of doing this would be useful and worth investigating.

Academic Plagiarism

This thesis has focused on detecting plagiarism by students but plagiarism in other sectors is also of interest. One area directly related is academic plagiarism, but there has been little work done to establish how common this is.

An interesting research topic would establish how common academic plagiarism is in practice and compare this with a survey finding out how acceptable such plagiarism is. It is necessary that some ideas will be duplicated between literature written by the same academic, presumably in introductory sections of subsequent papers, but there is little consensus about how frequent this is. A technical study would involve collecting a number of corpora of documents, each containing a set of documents by a given academic and calculating the percentage of self-plagiarism. Similarity visualisations could be produced to assess whereabouts in the documents such similarity occurs and if it is indeed near the beginning of the documents.

The results could be contrasted with a survey of academics, both those represented in the corpora and others, to decide how common self-plagiarism is believed to be and how much is acceptable. This could be used to inform guidelines for journal publishers and conference organisers and a self-plagiarism check could be a necessary pre-requisite before publication.

Similarity of Web pages

There are literally thousands of Web sites on the Internet that mention plagiarism in some way and it is clear that many of those offering advice offer the same points and often in the same order. University plagiarism policies are also often similar, perhaps by necessity. Furthermore many of the world's newspapers are online and searching for stories on plagiarism cases often reveals largely the same text as each, perhaps derived from a common press release or press agency story. This pattern is likely to be the same across other newsworthy areas.

It would be interesting to find out how much of Web material on plagiarism is duplicated and if perhaps there is a way of automatically creating a single Web site containing all the advice with likely duplication removed. A related side product could find out how much duplication there is and if there is any connection with referenced materials. Such surveys could be extended into other subject areas.

10.9 - Conclusions

This chapter has seen an entire tool supported plagiarism detection process put together that builds on the remainder of the thesis. The tools were developed to solve the problems associated with plagiarism detection and motivated by the results from which metrics and which comparison methods were appropriate.

Detection is far from a completed research area and this thesis has only been able to scratch the surface of what can be done and what needs to be done. It is very useful to be able to compare student submissions in an error free manner and to see why they are similar. But students are going to continue to plagiarise and many will continue to get away with it, whether it is from each other, from textbooks, from the Web, or the more worrying trend of paying people to do the work for them.

Educational reforms are needed to see that academic integrity is preserved. As governments press for ever increasing numbers of students to enter Higher Education with no more resources allocated to support them even the most efficient and effective detection processes will struggle. This may be compounded by weaker students entering who may need to cheat to be successful. Even if detection engines improve so too will the plagiarisers and the techniques that they use. Detection will be like a never ending arms race.

A known and used detection process should be an essential and compulsory part of every unit of every course if only to act as a deterrent to students considering cheating. It is not a sole solution, tutors must consider very carefully what assignment specifications they are setting and what opportunities they are giving students. But it is a partial solution and a necessary one.

Perhaps what is needed is a greater awareness that plagiarism is happening in every department in every institution. It requires a much greater number of academics from all disciplines to find sensible and complete solutions. The process presented here can only serve as a starting point but has to be more suitable than the ostrich policies many institutions are still using to this day.

References

- Altin, Jonas (2001), Information Clustering and Visualisation - An Aid for Detecting Student Plagiarism, Available from Department of Informatics, Institute of Information Technologies and Media, Mid Sweden University.
- Antosch, F. (1969), The Diagnosis of Literary Style with the Verb-Adjective Ratio, in Doležel, L. & Bailey, R. W. (Editors), *Statistics & Style*, American Elsevier Publishing Company, pp57-65.
- Anderson, J. (1998), *Plagiarism Copyright Violation and Other Thefts of Intellectual Property: An Annotated Bibliography with a Lengthy Introduction*, McFarland.
- Associated Press (2002), Teacher Resigns in Plagiarism Case - Kansas Teacher Resigns After Plagiarism Crackdown is Thwarted by the School Board, available online at http://www.abcnews.go.com/wire/US/ap20020206_1022.html.
- Atkinson, V. (2002), Why I Recommend a Considered Approach to Plagiarism, *Times Higher Education Supplement*, 26/07/02.
- Austin, M. & Brown, L. (1999), Internet Plagiarism: Developing Strategies to Curb Student Academic Dishonesty, *The Internet and Higher Education*, 2(1), pp21-33.
- Baker, B. (1993), On Finding Duplication in Strings and Software, in *Journal of Algorithms*.
- Bailey, R. W. (1969), Statistics and Style: A Historical Survey, in Doležel, L. & Bailey, R. W. (Editors), *Statistics & Style*. American Elsevier Publishing Company, pp217-236.
- Bailey, R. W. & Doležel, L. (1968), *An Annotated Bibliography of Statistical Stylistics*. Ann Arbor.
- Baron, D. (1998), Points of View - When Professors Get a A's and Machines get an F's, *Chronicle of Higher Education*, 20/11/98, pp56.
- Baruch, G. (2002), Artful Deception - If Ghostwriters Are Indispensable Why Are They So Invisible?, *Washington Post* 31/03/02, available online at <http://www.washingtonpost.com>.
- Baseley, S. (2002), Scandal of Scientists Who Take Money for Papers Ghostwritten by Drugs Companies, *The Guardian*, 07/02/02, pp4.
- Baty, P. (1999), Plagiarism Row Erupts, *Times Higher Education Supplement*, 02/07/99.
- Baty, P. (2002a), Whistleblowers: Plagiarism Scandal Returns to Haunt v-c, *Times Higher Education Supplement*, 19/11/02.

Baty, P. (2002b), Plagiarism puts Monash v-c out of job, *Times Higher Education Supplement*, 19/07/02.

Baty, P. (2002c), Cyber Cheat Network Grows, *Times Higher Education Supplement*, 01/11/02, pp4.

Bennett, P. E. (1969), The Statistical Measurement of a Stylistic Trait in 'Julius Caesar' and 'As You Like It', in Doležel, L. & Bailey, R. W. (Editors), *Statistics & Style*, American Elsevier Publishing Company, pp29-41.

Bloomfield, L. (2002), The Plagiarism Resource Site Links Page, available online at <http://plagiarism.phys.virginia.edu/links.html>.

Bodnar, C. (2002), Precedent Set in Student Plagiarism Victory, available online at <http://aix2.utttawa.ca/~fulcrum/58-03/news/Psipv.html>.

Boone, K. (1999), Preliminary Experiences with a Web-Based Coursework Submission System, Department of Computer Science, Middlesex University, London.

Bowers, N. (1997), *Words for the Taking - The Hunt for a Plagiarist*, W. V. Norton & Company, New York.

Boyer, M., Chung, T, Clark, D. & Poechman, T. (2000), Internet Plagiarism - The Results of a Discussion, available online at <http://ficse.oise.utoronto.ca/~tpoechman/plagiar.htm>.

Boywer, K. W. & Hall, L. O. (1999), Experience using 'MOSS' to Detect Cheating on Programming Assignments, *29th ASEE/IEEE Frontiers in Education Conference*, San Juan, Puerto Rico, pp18-22.

Braumoeller, B. & Graines, B. (2001), Actions Do Speak Louder Than Words: Deterring Plagiarism with the use of Plagiarism Detection Software, available online at <http://www.apsanet.org/PS/dec01/braumoeller.cfm>.

Bridgeman, S., Goodrich, T., Kobourov, S., Tamassia, R. (2000), PILOT: An Interactive Tool for Learning and Grading, *SiGCSE Bulletin*.

Buch, K. R. (1969), A Note on Sentence Length as a Random Variable, in Doležel, L. & Bailey, R. W. (Editors), *Statistics & Style*, American Elsevier Publishing Company, pp76-79.

Buckell, J. (2002), Plagiarism Tracked at 8 Per Cent, Australian IT, 11/09/02, Available online at <http://australianit.news.com.au/articles/0,7204,5071781%5e15334%5e%5enbv%5e15306-15317,00.html>.

Bull, J., Collins, C., Coughlin, E. & Sharp, D. (2001), Technical Review of Plagiarism Detection Software Report, Joint Information Systems Committee.

Buranen, L. (1999), But I Wasn't Cheating: Plagiarism & Cross Cultural Mythology, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp63-74.

Burgess, T. (2002), Cheating Row Gaining Pace, *Shore News* 25/09/02, available online at
<http://www.mytown.co.nz/story/mytstorydisplay.cfm?storyID=2946959&thecity=auckland&thepage=home&type=nzh&storytoolsnzh=1>.

Carroll, J. & Appleton, J. (2001), Plagiarism - A Good Practice Guide, available online at <http://www.jisc.ac.uk/plagiarism>.

Carroll, J. B. (1969), Vectors of Prose Style, in Doležel, L. & Bailey, R. W. (Editors), *Statistics & Style*, American Elsevier Publishing Company, pp147-155.

Carter, J. (1999). Collaboration or Plagiarism: What Happens When Students Work Together? *Proceedings of ITiCSE 1999*, pp52-55.

Chester, J. (2001a), Pilot of Free-Text Electronic Plagiarism Detection Software, available online at <http://www.jisc.ac.uk/plagiarism>.

Chester, J. (2001b), Plagiarism Prevention & Detection - Final Report on the JISC Electronic Plagiarism Detection Project, available online at <http://www.jisc.ac.uk/plagiarism>.

Christie, J. R. (1999), Automated Essay Marking – for both Style and Content, available from School of Computing and Mathematical Studies, Robert Gordon University, Aberdeen, Scotland.

Clankie, S. M. (1999), Brand Name Use in Creative Writing: Genericide or Language Right? In *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp252-262.

Clare, J. (2000), Computer plagiarism 'threatens the value of degrees', *Daily Telegraph* 3/7/2000, available online at
<http://www.telegraph.co.uk/et?ac=004228431730477&rtmo=VDw3wDqK&atmo=r3rrr3rrq&pg=/et/00/7/3/ncopy03.html>.

Clark, I. L. (1999), Writing Centres and Plagiarism, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp155-167.

Clough, P. (2000a). Plagiarism in Natural and Programming Languages: An Overview of Current Tools and Techniques, available online at http://www.dcs.shef.ac.uk/~cloughie/plagiarism/HTML_Version/index.html.

Clough, P. (2000b), Identifying Re-use Between the Press Association and Newspapers of the British Press, Available from Department of Computer Science, University of Sheffield, UK.

CNN (2001), CNN Burden of Proof - University of Virginia Tackles Cheating Heads On, Aired May 10, 2001, 12:30 ET, transcript available online at <http://groups.google.com/groups?q=cnn+burden+of+proof+university+virginia&hl=en&lr=&ie=UTF-8&oe=UTF-8&selm=9dl5ns%242ra%241%40morgoth.sfu.ca&rnum=2>.

Combrink-Kuiters, L., Elffers, H., De Mulder, R., & Van Nortwijk, K. (1999), Comparing Student Assignments by Computer, presented at *Cyberspace 1999*, 29-30 March 1999, College of Ripon & York, St. John at York, UK.

Comfort, N. (1999), Case Closed, *New Scientist* **161**(2168), pp40.

CopyFind (2002), Available online at <http://plagiarism.phys.virginia.edu/software.html>.

Cramb, A. (1999), University withholds 90 exam results over 'Internet cheating', *Daily Telegraph* 10/7/1999, available online at <http://www.telegraph.co.uk/et?ac=004228431730477&rtmo=VDw3wDMK&atmo=r0rrrq&pg=/et/99/7/10/ncheat10.html>.

Culwin, F. & Lancaster, T. (2000), A Review of Electronic Services for Plagiarism Detection in Student Submissions, *Proceedings of 1st LTSN-ICS Conference*, Edinburgh, pp54-61.

Culwin, F. MacLeod, A. & Lancaster, T. (2001), Source Code Plagiarism in UK HE Computing Schools, Issues, Attitudes and Tools, *South Bank University Technical Report SBU-CISM-01-02*.

Culwin, F. & Naylor, J. (1995), Pragmatic Anti-Plagiarism, *Proceedings 3rd All Ireland Conference on the Teaching of Computing*, Trinity College, Dublin.

Cunningham, Padraig & Mikoyan, Alexander N. (1993), Using CBR Techniques to Detect Plagiarism in Programming Assignments, available from Department of Computer Science, Trinity College, Dublin.

Currie, J. (2000), Bradford Issues Internet Plagiarism Warning, *Times Higher Education Supplement*, 28/04/04.

Curtis, P. (2002), Business Schools Condemn MBA Plagiarism, *The Guardian*, 23/10/02.

Davies, P. (2000), Who Learns From Using Continuous C.A.A.? Available online at <http://www.comp.glam.ac.uk/pages/staff/pdavies/ILTAC2000.ppt>.

Decoo, W. (2002a), *Crisis on Campus - Confronting Academic Misconduct*, The MIT Press.

Decoo, W. (2002b), How to Break That Cheating Art, *Times Higher Education Supplement*, 22/02/02.

Dee, H. (2001), The Use of Electronic Tools to Support Plagiarism Detection, available online at <http://www.comp.leeds.ac.uk/hannah/CandIT/plagiarism.html>.

Denhart, A. (1999), The Web's Plagiarism Police, available online at <http://www.salon.com/tech/feature/1999/06/14/plagiarism/index.html>.

Denning, P. J. (1995), Editorial: Plagiarism on the Web, *Communications of the ACM*, **38**(12).

DePauw (2002), Avoiding Plagiarism, available online at <http://www.depauw.edu/admin/arc/plagiarism.htm>.

DetctaCopias (2002), available online at <http://www.dcc.uchile.cl/~rmeza/proyectos/detectaCopias/index.html>.

Dettmar, Kevin J. H. (1999), The Illusion of Modernist Allusion and the Politics of Postmodern Plagiarism, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp99-109.

Diederich J., Kindermann, J., Leopold, E. & Paass, G. (2000), Authorship Attribution with Support Vector Machines, GMD - Forschungszentrum Informationstechnik, Sankt Augustin.

Doležel, L. (1969), A Framework for the Statistical Analysis of Style, in Doležel, L. & Bailey, R. W. (Editors), *Statistics & Style*, American Elsevier Publishing Company, pp10-25.

Donaldson, J. L., Lancaster, A., & Sposato, P. H. (1981), A Plagiarism Detection System, *Twelfth SIGSCE Technical Symposium*, St. Louis, Missouri, pp. 21-25.

Dryden, L. M. (1999), A Distant Mirror or Through the Looking Glass? Plagiarism and Intellectual Property in Japanese Education, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp75-85.

Ducasse, S., Rieger R., Demeyer S. (1999), A Language Independent Approach for Detecting Duplicated Code, *Proceedings of International Conference on Software Maintenance*.

Ehrlich, H. (1998), Plagiarism and Anti-Plagiarism, Available online at <http://newark.rutgers.edu/~ehrlich/plagiarism598.html>.

Evans, J. (2001), The New Plagiarism in Higher Education: From Selection to Reflection, available online at <http://www.warwick.ac.uk/ETS/interactions/vol4no2/evans.htm>.

Faber, L with C.K. & M.L. (1999), The Great Term-Paper Buying Caper – How They Do It, available online at <http://www.usnews.com/usnews/edu/college/cocheatb.htm>.

Faidhi J. A. W. & Robinson S. K. (1987), An Empirical Approach for Detecting Program Similarity and Plagiarism within a University Programming Environment, *Computing Education*, 11, pp11-19.

Farrar, S. (2002), Scientists Fail to Tackle Fraud, *Times Higher Education Supplement*, 04/01/02.

Folkers R. & Campbell M. (1999), Exclusive Poll: Cheaters Win, available online at <http://www.usnews.com/usnews/misc/press/cheating.htm>.

Foster, A. (2002), Plagiarism Detection Tool Creates Legal Quandary, *The Chronicle of Higher Education*, 17/05/02.

Fox, B. (2002), Why Copy Protection on CDs is Worthless, *New Scientist*, 09/11/02, pp9.

Franklyn-Stokes, A. & Newstead, S. E. (1995), Undergraduate Cheating: Who Does What & Why? *Studies in Higher Education*, 20(2).

Gajadhar J. (1998), *Issues in Plagiarism for the New Millennium: An Assessment Odyssey*, available online at <http://ultibase.rmit.edu.au/Articles/dec98/gajad1.htm>.

Garcia-Molina, H., Gravano, L. & Shivakumar, N., dScam: Finding Document Copies Across Multiple Databases. In *Proc. of the 4th International Conference on Parallel and Distributed Information Systems*.

Goot, J. (2002), Thin Line Splits Cheating, Smarts, available online at <http://www.wired.com/news/culture/0,1284,54963,00.html>.

Granville, A. (2002), Detecting Plagiarism in Java Code, Student Project, available from University of Sheffield, UK.

Gray, A., Sallis, P. & MacDonnel, S. (1998), IDENTIFIED (Integrated Dictionary-Based Extraction of Non-Language Dependent Token Information for Forensic Identification, Examination and Discrimination): A Dictionary Based System for Extracting Source Code Metrics for Software Forensics, *Proceedings of 1998 International Conference in Software Engineering, Education and Practice*, pp252-259.

Green, S. (1999), Standardise Penalties for Cheats, *Times Higher Education Supplement*, 09/07/99.

Greg, W.W. (1927), *The Calculus of Variants: An Essay on Textual Criticism*, Oxford.

Griggs, B. (2002), The Internet Gives College Cheaters a High-Tech Edge, *The Salt Lake Tribune* 18/11/02, available online at <http://www.sltrib.com/11182002/utah/17842.htm>.

Grossman, W. (2002), All Their Own Work?, *The Independent*, 15/04/02.

Guardian (2002), Monash Head Leaves Over Plagiarism Charge, *The Guardian*, 18/07/02.

Guiliano, E. (2000). Deterring Plagiarism in the Age of the Internet, *Inquiry*, 5(1), pp22-31.

H., D. (1999), Poaching and Plagiarising: Property, Plagiarism & Feminist Futures, In *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp111-120.

Halstead, M. H. (1977), *Elements of Software Science*, Elsevier.

Hamblen, J. O., Parker, A. & Wachtel (1998), Stephen R., A New Undergraduate Computing Arithmetic Software Library, *IEEE Transactions on Education*, 31(3).

Harnden, T. (2002), High School Cheating Row Divides US, The Daily Telegraph, 23/03/02.

Harris, J. K. (1994), Plagiarism in Computer Science Courses, *Proceedings of 1994 Ethics in Computer Age*, pp133-135.

Harris, R. (1999), Anti-Plagiarism Strategies for Research Papers, available online at <http://www.vanguard.edu/rharris/antiplag.htm>.

Haviland, C. P. & Mullin, Joan (1999), Writing Centres and Intellectual Property: Are Faculty Members and Students Differently Entitled? In *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp169-181.

Hearst, M. (2000), The Debate on Automated Essay Grading, in *IEEE Intelligent Systems*, September/October 2000, pp22-37.

Halfman, J. I. (1994), Similarity Patterns in Language, *Proceedings of IEEE Symposium on Visual Languages*, pp173-175.

Hilton, A. (2002), Study on the Use of Electronic Submission in UK Further and Higher Education, available online at <http://www.jisc.ac.uk>.

Hoad, T. & Zobel, J. (2002), Methods for Identifying Versioned and Plagiarised Documents, to appear in *Journal of the American Society for Information Science and Technology*.

Howard, R. M. (1999), The New Abolitionist Comes to Plagiarism, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp87-95.

Insights (1998), Cheating Insights From Literature, available online at <http://www.msb.edu/faculty/mchenryw/sm-s98/insights.html>.

Irving, R. (2000), Plagiarism Detection: Experiences and Issues, *JISC Fifth Information Strategies Conference: Focus on Access and Security, British Library, London.*

Irving, R., MacDonald, G., McGookin, D. & Prentice, J. (2002), *Big Brother (Glasgow University Computer Science Department's Collusion Detector System, version 2.0) User Manual*, available from Glasgow University.

Jenkins, T. (1995), Open Book Assessment in Computing Degree Programmes, *University of Leeds - School of Computing - Research Studies Series - Report 95.28.*

Johnston, C. (2001), High-Tech Alone Will Not Catch Out Net Cheats, *Times Higher Education Supplement*, 10/08/01, pp15.

Jones, E. L. (2001), Plagiarism Monitoring and Detection - Towards an Open Discussion, *7th Annual CSSC Central Plains Conference, Branson, Missouri*, April 6-7.

Jones, E. L., (2001a), Metrics Based Plagiarism Monitoring, *6th Annual CSSC Northeastern Conference, Middlebury, Vermont*, April 20-21.

Joo-Hee, L. (2001), Plagiarism Probe Launched into Professors' Paper, available online at
http://www.koreaherald.co.kr/SITE/data/html_dir/2001/11/20/200111200039.asp.

Joy, Mike & Luck, Michael (1999), Plagiarism in Programming Assignments, *IEEE Transactions on Education*, **42**(2), pp129-133.

JPlag (2001), JPlag, Available online at <http://www.jplag.de>.

Kleiner, C. & Lord, M. (1999), The Cheating Game – ‘Everyone’s Doing It’ from Grade School to Graduate School, available online at
<http://www.usnews.com/usnews/edu/college/cocheata.htm>.

Kock, Ned (1999), A Case of Academic Plagiarism, *Communications of the ACM*, **42**(7), pp96-104.

Kontaxis, K. (2001), Readability and Style: A Computational Approach, *BSc Computing Studies Project 125*, available from South Bank University, London, UK.

Kroeber, K. (1969), Perils of Quantification: The Exemplary Case of Jane Austin’s ‘Emma’, in Doležel, L. & Bailey, R. W. (Editors), *Statistics & Style*, American Elsevier Publishing Company, pp197-213.

LaFleur, Robert A. (1999), Literary Borrowing and Historical Compilation in Medieval China, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp141-150.

LaFollette, M. C. (1992), *Stealing into Print - Fraud, Plagiarism and Scientific Misconduct in Publishing*, University of California Press.

Lamb, Robyn (2000), High-Tech Plagiarism a Problem at University of Maryland, available online at <http://www.studentadvantage.com/lycos/article/0,4683,c4-i83-t379-a125964,00.html>.

Larochelle, G. (1999), From Kant to Foucault: What Remains of the Author in Postmodernism? In *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp121-130.

Lathrop, A. & Foss, K. (2000), *Student Cheating and Plagiarism in the Internet Era – A Wake Up Call*. Published by Libraries Unlimited Inc.

LeClercq, T. (1999), Confusion and Conflict about Plagiarism in Law Schools and Law Practice, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp195-203.

Lee, J. A. N. (1995), Of ‘Babbage and Kings’ and ‘How Sausage Was Made: And Now for the Rest of the Story, *IEEE Annals of the History of Computing*, **17**(4), pp7-2.

Leight, D. (1999), Plagiarism as Metaphor, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp221-229.

Lesko, J. P. (1996), Plagiarism and Questionable Appropriation of Text by Non-native Speaker Students in Taught Postgraduate Courses: Views and Experiences of Postgraduate Staff, *Proceedings of the Edinburgh Linguistics Department Conference '96*, pp142-149.

Linley, A. (1952), *Plagiarism and Originality*, Harper & Brothers Publishers, New York.

Lingua Franca (2000), Cribs Cheats and Plagiarism, available online at <http://www.abc.net.au/rn/arts/ling/stories/s149143.htm>.

Livingston-Webber, Joan (1999), GenX Occupies the Cultural Commons: Ethical Practices and Perceptions of Fair Use, In *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp262-272.

Loftus, M. & Smith, A. K. (1999), What Started in Elementary School... Adults Cheat Too, available online at <http://www.usnews.com/usnews/edu/college/cocheatc.htm>.

Lyon, C, Malcolm, L. & Dickerson, B. (2001), Detecting Short Paragraphs of Similar Text in Large Document Collections, *Proceedings of Conference on Empirical Methods in Natural Language Processing, June 2001*.

MacKenzie, D. (1991), Europe Lays Down the Law on Software, *New Scientist*, **30**(1774), pp20.

Major, L. E. (2002). Web of Deceit, *Guardian Education*, 08/01/02, pp9.

Mallon, T. (2001), Stolen Words - The Classic Book on Plagiarism (2nd Edition), Harvest Book, Harcourt Inc.

Mann, C. (1998), Who Will Own Your Next Good Idea?, available online at <http://www.theatlantic.com/issues/98sep/copy.htm>.

Martin, B. (1992a), Plagiarism by University Students: The Problem and some Proposals, *Tertengala (University of Wollongong Students' Representative Council)*, 20/07-03/08/92, pp20.

Martin, B. (1992b), Scientific Fraud and the Power Structure of Science, *Prometheus*, **10**(1), pp83-98.

Martin, B. (1994), Plagiarism: A Misplaced Emphasis, *Journal of Information Ethics*, **3**(2), pp36-47.

Martin, B. (1997), Academic Credit Where It's Due, *Campus Review*, **7**(21), pp11.

Maslen, G. (2002), Net Plagiarism Scam Revealed, *The Times Higher Education Supplement*, 27/09/02, pp16.

McCabe, D. (2002), New Research on Academic Integrity: The Success of 'Modified' Honor Codes, available online at <http://www.collegepubs.com/ref/SFX00515.shtml>.

Medori, J., Atwell, E., Gent, P. & Souter, C. (2002), Customising a Copying-Identifier for Biomedical Student Reports: Comparing Simple and Smart Analyses, School of Computing, University of Leeds.

Miles, T. & Burleigh, J. (2001), Yard Probe as A-Level Paper is Sold for £400, *Evening Standard*, 14/06/01, pp7.

Monostori, K., Zaslavsky, A. & Schmidt, H. (2000), Parallel & Distributed Overlap Detection on the Web, *Applied Parallel Computing. New Paradigms for HPC in Industry and Academia · 5th International Workshop, PARA 2000 Bergen, Norway, June 18-20, 2000*, pp.206-214.

Moon, J. (1999), How To...Stop Students from Cheating, *Times Higher Education Supplement*, 03/09/99.

Morgan, C. J. & Foster, W. T. (1992), Student Cheating: An Ethical Dilemma, *Proceedings of 1992 Frontiers in Education Conference*, pp678-682.

Morton, A. Q. (1978), *Literary Detection: How to Prove Authorship and Fraud in Literature and Documents*, Bowker.

MOSS (1994), Measure of Software Similarity, available online at <http://www.cs.berkeley.edu/~aiken/moss.html>.

Mosteller, F. & Wallace, D. L. (1984), *Applied Bayesian and Classical Inference. The Case of the Federalist Papers, 2nd Edition of Inference and Disputed Authorship: The Federalist*. Springer-Verlag.

Muller, C. (1969), Lexical Distribution Reconsidered: The Waring-Herdan Formula, in Doležel, L. & Bailey, R. W. (Editors), *Statistics & Style*, American Elsevier Publishing Company, pp42-56.

Newstead, S. E., Franklyn-Stokes A & Armstead P. (1996), Individual Differences in Student Cheating, *Journal of Educational Psychology*, **88**(2), pp229-241.

Okklemo, S. (2001), Other People's Essays, available online at http://www.hero.ac.uk/studying/archive/other_people_s_essays908.cfm.

Ottenstein, K. J. (1977), An Algorithmic Approach to the Detection and Prevention of Plagiarism, *SiGCSE Bulletin* **8**(4), pp. 30-41.

Oxford (2002), Plagiarism and the Internet, available online at <http://www.oucs.ox.ac.uk/ltg/reports/pappendix1.html>.

Palmer, C. & Jones, P. (2001), Plagiarism and the Academic Essay: A Story of Mystery, Treachery and Betrayal? University of Lancaster & University of Greenwich.

Parker, Alan & Hamblen, James O. (1989), Computer Algorithms for Plagiarism Detection, *IEEE Transactions on Education*, **32**(2), pp. 94-99.

Paull, H. M. (1928), *Literary Ethics*, Kennikat.

Pearson, G. (2002), Electronic Plagiarism Seminar, available online at <http://www.lemoyne.edu/library/plagiarism.htm#young1>.

Pitts Jr, L. (2002), Reputation vs Character: The Lessons of Piper High, available online at http://seattletimes.nwsource.com/html/editorialsopinion/134479551_pitts23.html.

Prechelt, L., Guido, M. & Phlippsen, M. (2000), JPlag: Finding Plagiarisms Among a Set of Programs, *Technical Report 2000-1*, Facultat fur Informatik, Universitat Karlsruhe, Germany.

Prechelt, L., Guido, M. & Phlippsen, M. (2001), JPlag: Finding Plagiarisms Among a Set of Programs with JPlag, submitted to *Journal of Universal Computer Science*, Facultat fur Informatik, Universitat Karlsruhe, Germany.

QAA (2002), Distance Learning Guidelines, available online at <http://www.qaa.ac.uk/public/dlg/guidelin.htm>.

Rahman, L. (2002), FreeStyler - A Text Forensic Tool, *BSc Computing Studies Project 264*, available from South Bank University, London, UK.

Randall, M. (1999), Imperial Plagiarism, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp131-140.

Ribler, R. L. (1997), Visualizing Categorical Time Series Data with Application to Computer and Communications Network Traces, PhD dissertation, Virginia Polytechnic Institute and State University, Blacksburg, Virginia.

Ribler, R. L. & Abrams M. (2000), Using Visualisation to Detect Plagiarism in Computer Science Classes, *Information Visualisation 2000*, pp173-177.

Rieger, M. & Ducasse S. (1998), Visual Detection of Duplicated Code, *Proceedings of ECOOP Workshop on Experiences in Object-Oriented Re-Engineering*.

Ringle, K. (2001), Plagiarism Redux, or Just a Blunder?, *International Herald Tribune*, 09/02/02.

Robinson, Sally S. & Soffa, M. L. (1980), An Instructional Aid for Student Programs. *SiGCSE Bulletin 12(1)*, pp. 118-129.

Roy, Alice M., (1999), Whose Words These Are I Think I Know: Plagiarism, the Postmodern & Faculty Attitudes, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp55-61.

Ryan J. J. C. H. (1998), Student Plagiarism in an Online World. available online at http://www.asee.org/prism/december/html/student_plagiarism_in_an_onlin.htm.

Rose, Shirley K. (1999), The Role of Scholarly Citations in Disciplinary Economics, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp241-249.

Runk, D. (2002), Plagiarism of Sermons is an Issue for Clergy, *The Associated Press*, Available online at <http://www.registerguard.com/news/20020316/15c.rel.pulpitplagiarism.0316.html>.

Sanders, R. (1998), Online Plagiarism Detector Helps CS Professors Bust Cheating Programmers, available online at <http://www.coe.berkeley.edu/EPA/EngNews/98S/aiken.html>.

Sanderson, M. (1997), Duplicate Detection in the Reuters Collection, *Technical Report TR-1997-5, Department of Computing Science, University of Glasgow*, 1997.

Sallis, P., Aakjeer, A. & MacDonnal, S. (1996), Software Forensics: Old Methods for a New Science, *Proceedings of 1996 International Conference on Software Engineering*, pp481-485.

Samuel, E. (2002), Rising Star of Electronics found to have Fabricated his Ground-Breaking Results, *New Scientist* 05/10/02, pp4-5.

Samuelson, P. (1994), Self-Plagiarism or Fair Use? *Communications of the ACM*, 37(8).

Satterwhite, R. & Goerin, M. (2000), Downloading Detectives: Searching for On-Line Plagiarism, available online at

http://www.coloradocollege.edu/Library/Course/downloading_detectives_paper.html.

Saxon, S. (2000), Comparison of Plagiarism Detection Techniques Applied to Student Code, Part II Computer Science Project, Trinity College, Cambridge.

ScienceDaily (2002), Plagiarism-Detection Software Stems Students' Use of Paper Mills, available online at

<http://www.sciencedaily.com/releases/2002/05/020501073710.htm>.

Seymour, U. (2002), Name That Tune, PC Advisor, 17/11/02.

Shamoon, L. & Burns, D.h H. (1999), Plagiarism, Rhetorical Theory and the Writing Centre: New Approaches, New Locations, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp183-192.

Sheard, J., Dick, M., Markham, S., MacDonald, I. & Walsh, M. (2002), Cheating and Plagiarism: Perceptions and Practices of First Year IT Students, *Proceedings of ITiCSE '02, June 24-26, Aarhus, Denmark*, pp183-187.

Shivakumar, N. & Garcia-Molina, H. (1995), SCAM: A Copy Detection Mechanism for Digital Documents, *Proceedings of 2nd International Conference in Theory and Practice of Digital Libraries, Austin, Texas, June 1995*.

Shivakumar, N. & Garcia-Molina, H. (1998), Finding Near Replicas of Documents on the Web, *Proceedings of Workshop on Web Databases, in conjunction with EDBT '98, March 1998*.

Si, A., Leong, H. V. & Lau, W. H. R (1997), CHECK: A Document Plagiarism Detection System, *SAC 1997*, pp70-77.

SIM (1989), The Software and Text Similarity Engine, available online at <http://www.cs.vu.nl/~dick/sim.html>.

Simmons, S. C. (1999), Competing Notions of Authorship: A Historical Look at Students and Textbooks on Plagiarism and Cheating, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp41-51.

Singh, S. & Tweedie, F. J. (1995), Neural Networks and Disputed Authorship: New Challenges, *IEEE Proceedings of Artificial Neural Networks*, **409**, pp24-28.

Special, D. (2002), Temptation to Cut and Paste, Daily Yomiuri, available online at <http://222.yomiuri.co.jp/newse/20020402wo61.htm>.

Spence, R. (2001), *Information Visualisation*, published by Addison-Wesley.

Spigelman, C. (1999), The Ethics of Appropriation in Peer Writing Groups, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp231-240.

Standler, Ronald B. (2000), Plagiarism in Colleges in the USA, available online at <http://www.rbs2.com/plag.html>.

Stearns, Laurie. (1999), Copy Wrong: Plagiarism, Process, Property and the Law, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp5-17.

Stefani, L. & Carroll, J. (2001), A Briefing on Plagiarism, LTSN Generic Centre Assessment Series No. 10.

Stephens, S. D. (2000), Using Metrics to Detect Plagiarism, Department of Computer & Information Systems, Florida A&M University.

Stoerger, S. (2002), Plagiarism, available online at <http://www.web-miner.com/plagiarism>.

Sutherland, J. (2000), US students log on to the Internet for a cheat's charter, *Guardian*, 22/6/2000, available online at <http://www.guardianunlimited.co.uk/Archive/Article/0,4273,4032413,00.html>.

Swan, J. (1994), Touching Words: Helen Keller, Plagiarism, Authorship. in Woodmansee, M. & Jaszi, P. (editors), *Construction of Authorship: Textual Appropriation in Law and Literature*, Duke University Press, pp57-100.

Swearingen, C. J., (1999), Originality, Authenticity, Imitation and Plagiarism: Augustine's Chinese Cousins, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp19-30.

Telegraph (2002), Alarm at Plague of Plagiarism, *The Daily Telegraph*, 23/03/02.

THES (2002), Support for Monash v-c After Plagiarism Report, *Times Higher Education Supplement*, 26/06/02.

Thompson, W. (2002), A Very Lonely Enterprise, *The Times-Picayune* 03/04/02, available online at <http://www.nola.com>.

Tysome, T. (2002), Troubled Wales is Hit by Plagiarism Claim, *Times Higher Education Supplement*, 08/11/02.

Utley, A. (2002), Time Runs Out for Net Cheats, *Times Higher Education Supplement*, 09/08/02, pp3.

Various (2000), Plagiarism, available online at
<http://www.lgu.ac.uk/deliberations/forums/plagiarism.html>.

Verco, Kristina L. & Wise, Michael J. (1996), Plagiarism a la Mode: A Comparison of Automated Systems for Detecting Suspected Plagiarism. *The Computer Journal* **39**(9), pp741-750.

Verco, K. L. & Wise, M. J. (1996a), Software for Detecting Suspected Plagiarism: Comparing Structure and Attribute-Counting Systems, *Proceedings of First Australian Conference on Computer Science Education*, July 3-5 1996, Sydney, Australia.

Voumard, P. R. (1994), Computer Based Individual Assignment System, *Proceedings of First International Conference in Multimedia Engineering Education*, Melbourne, Australia, pp249-254.

Watson, A. (2001), A Gift for Language, *New Scientist*, **172**(2321), pp25.

Whale, G. (1990), Identification of Program Similarity in Large Populations. *The Computer Journal* **33**(2), pp140-146.

White, Edward M. (1999), Student Plagiarism as an Institutional and Social Issue, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp205-210.

Williams, C. B. (1969), A Note on the Statistic Analysis of Sentence Length as a Criterion of Literary Style, in Doležel L. & Bailey R. W. (Editors), *Statistics & Style*, American Elsevier Publishing Company pp69-75.

Williams, L. (1999), But Isn't That Cheating? *Proceedings of 29th ASEE/IEEE Frontiers in Education Conference*, San Juan, Puerto Rico, Session 12b9.

Wilson, Henry L. (1999), When Collaboration Becomes Plagiarism: The Administrative Perspective, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp211-218.

Wise, Michael J. (1996), YAP3: Improved Detection of Similarities in Computer Program and Other Texts, *1996 SiGCSE Technical Symposium*, Philadelphia, USA, pp. 130-134.

Woodward, W. & Lomax, S. (2001), A-Level Maths Paper on Sale to Pupils, *The Guardian*, 14/06/01, pp3.

Wools, D. (1999). Computer Programs Can Catch Out Copycat Students but the Smart Internet Plagiarist is still Difficult to Detect, *Times Higher Education Supplement* 03/09/99.

Writing Tutorial Services (2001), Plagiarism: What It Is and How to Recognise and Avoid It, available online at <http://www.indiana.edu/~wts/wts/plagiarism.html>.

Zelbroski, James T. (1999), Intellectual Property, Authority and Social Formation, in *Perspectives on Plagiarism and Intellectual Property in a Postmodern World*, Buranen, L. & Roy, A. M. editors, State University of New York Press, pp31-39.

Appendix A: Glossary

Academic credit - A qualification or other award earned by a student for completing assignment specifications to a suitably high standard.

Academic institution - A place of learning where tutors teach students.

Academic plagiarism - Plagiarism carried out by academics, for instance copying journal articles and submitting them as their own work for possible career development.

Analysis stage - The second stage of the four-stage plagiarism detection process. Here all submissions are compared with each other (for intra-corporeal plagiarism detection) or the external sources such as the Web (for extra-corporeal plagiarism detection) to find submissions that are similar to each other or the Web sources.

Assignment specification - The description of a task that students must carry out successfully to gain academic credit.

Attribute counting metrics - A count of some property of a single document which might involve tokenisation. This has been redefined to remove the inconsistencies from the literature but is not considered a sensible classification.

Attribute counting systems - A system that uses only a combination of attribute counting metrics to find similar submissions. This is not considered a sensible classification.

Authorship attribution - The branch of linguistics that aims to calculate the author of a work based on knowledge of works by other known authors. This is not appropriate for plagiarism detection since there is no corpus of known work by a given student.

Cluster - A set of two or more student submissions that are all judged to be similar to one another (e.g. for a cluster of three submissions, submission 1 is similar to submission 2, submission 2 is similar to submission 3 and submission 1 is similar to submission 3).

Characters Metric - A simple metric that measures the number of sequences of characters of a chosen length two documents have in common.

Cheating - Unauthorised behaviour that is going against student etiquette when trying for an academic award or to gain an advantage over other students. Examples include plagiarism, use of cribs in exams and paying someone to complete an assignment specification on your behalf.

Closeness Calculation - A computationally part of automated plagiarism detection where a single number is generated from a number of different metrics to decide how similar two submissions are.

Collaboration - Where two students discuss and work on an assignment specification together but complete their final submissions independently.

Collection stage - The first stage of the four-stage plagiarism detection process. This is where students submit their work to an electronic system so it can later be analysed for similarity.

Collusion - Where two students discuss and work on an assignment specification together and complete elements of their final submissions together. This might be judged to be intra-coral plagiarism.

Computational intensity- A measure of the computer processing power needed to complete a given task.

Confirmation stage - The third stage of the four-stage plagiarism detection process. Here a tutor checks the pairs of student submissions that have been judged to be similar to see if they represent plagiarism or they represent legitimate shared citations or false hits. The tutor decides which pairs will go on to be investigated further.

Contractive plagiarism - Plagiarism where the source is larger than the copy and hence the source has been reduced in some way to create the student submission.

Copying - The definitely unacceptable part of the collaboration collusion copying continuum. This is where two students have produced submissions so closely that they could be judged identical or one student has completed the work and the other has handed it in as their own. Copying is an example of student plagiarism.

Corpal Metrics - A multi-dimensional metric that is a measure of a property of an entire corpus, for instance the proportion of submissions using a given keyword.

Corpora - The plural of corpus. A number of sets of linked student submissions.

Corpus - A set of student submissions that are linked in some way, for instance they are all handed in to meet the same assignment specification.

Correlation - A measure between -1 and +1 of the commonality between two ordered lists, with -1 representing a reversed order, +1 representing identity and 0 representing no correlation.

Direct copy - Two student submissions that are identical to one another with no attempt at disguise. One is a direct copy of the other.

Disguise - Where a student has attempted to change a source and hand it in as their own submission so that the use of the original source won't be noticed.

Document - Any text that can be examined for similarity. Those collected from students as answers to an assignment specification are known as student submissions.

Effective plagiarism detection system - A detection process that produces a sensible ordering of similarity within the similarity rank list, in particular avoiding false hits and missed pairs.

Efficient plagiarism detection system - A process that requires as few human resources as possible to identify and investigate potential plagiarism. A secondary requirement is that computer resources needed should be limited.

Electronic submission - An alternative to paper based submission. Students submit their work in electronic format, such as MS Word, RTF, program source code or plain ASCII text for marking.

Essay banks - Web sites that store a large number of student submissions that other students can then download, modify and submit as their own work. They can be provided free or on a fee basis and are an extension of pre-Web services that offered to supply essays on different subjects for a fee in the back of magazines.

Expansive plagiarism - Plagiarism where the source has been extended, either by adding new thoughts or adding filler words and phrases to make a student submission.

Extra-corporeal plagiarism - Plagiarism where the plagiarism source is outside the corpus of student submissions, for instance a Web site or material from a book.

False hits - Pairs of submissions that are ranked high enough for a tutor to investigate them but are judged to be dissimilar, thus being a waste of tutor time.

Filler words - Those words in the English language that occur regularly and fill in the gaps between the major word types, such as 'and', 'or', 'the', 'of'.

Fingerprint - In this context a metric of values that represent a single submission, for a singular metric, or similarity in a pair of submissions, for a paired metric.

Four-stage plagiarism detection process - The series of steps collection, analysis, confirmation and investigation through which a set of student submissions can have plagiarised cases identified through a part-automated route.

Fragmentary granularity - A metric which is applied on sections of two student submissions at a time, not on the whole thing, for instance to identify those areas of the submissions that are plagiarised.

Fragmentary granularity matrix - The underlying table of numbers in a similarity visualisation representing the intensity of the corresponding pixel at each point. Each number is generated using a fragmentary granularity metric on fragments obtained from appropriate positions in both documents.

Fragments - The result of breaking up student submissions into smaller chunks.

Free text - Submissions presented in a natural language form using words and phrases, for instance student essays.

Free text plagiarism - Plagiarism that has been done in natural language, for instance, altering the words of another writer and presenting it as your own work.

Gross granularity - A metric which has been applied on the entirety of one submission against the entirety of another submission to give a single similarity score.

Hybrid metric systems - A system that a combination of both attribute counts and structure metrics to find similar submissions. This has been defined to remove the inconsistencies from the literature but is not considered a sensible method of classification.

Intra-corporeal plagiarism - Plagiarism entirely within a corpus, primarily meaning two students who have copied from one another.

Investigation stage - The fourth and final stage of the four-stage plagiarism detection process. This is where pairs of similar submissions have been found and they have been confirmed by human inspection to be similar and possible cases of plagiarism. In this case further evidence is collected, such as student interviews and marked up copies of the submissions and penalties are given.

Linguistics - The branch of science examining properties of words and word use.

Marquee cursor- The mouse controlled selection device used in the VAST prototype to control which parts of the texts associated with the similarity visualisation to study.

Mean of max metric - A visual metric that has been identified to give an ordering of similarity that clearly differentiates between pairs containing similarity and pairs that don't. The metric finding the most intense pixel in every row and column of a similarity intersection and taking the mean intensity of those pixels.

Metric - Where all possible pairs of student submissions are compared under certain conditions and a numeric value is generated for each pair to show the extent of similarity. The rules used to produce the numeric similarity score is known as the metric.

Missed pairs - A pair of submissions that contains plagiarism but is not automatically ranked in the upper portion of an ordered list of similar pairs and hence not investigated further by a tutor.

Mosaic plagiarism - Plagiarism where chunks from different sources are used and rearranged in a way that could be considered like a mosaic is created from combining and arranging different pictures.

Multi-dimensional metrics - A metric that is calculated; based on some property of more than two documents; for instance the length of the longest substring common to all the documents being considered. This could be useful for clustering similar submissions.

Multiply sourced - A student submission or external source that has been used in multiple student submissions.

Noise - The background colouring on similarity visualisations, produced by the standard distributions of English language that will always find a low level of similarity in any two fragments, for instance by use of the filler words, such as 'the'.

Normalisation - The process whereby metric results are scaled within known values, say 0 and 100, so they can be more directly comparable.

Other detection systems - A catch-all group for systems that cannot be classified as attribute counting systems, structure metric systems or hybrid systems. This has been defined to remove the inconsistencies from the literature but is not considered a sensible method of classification.

Ostrich plagiarism policy - Where an academic institution states that plagiarism does not exist in their institution and has no formal way of dealing with it.

Paired Metric - A metric which gives a measure of some property of two submissions considered alongside each other, for instance the length of the longest common substring in both. The simple metrics used are all examples of paired metrics.

Paraphrasing - Using the ideas of another but rewriting them in your own words without suitable and continual acknowledgement.

Plagiarism - Taking the words or ideas of another and presenting them as your own without suitable acknowledgement.

Plagiarism detection engine - A tool used to assist a tutor in automatically finding plagiarism in some way.

Plagiarism detection system - An alternative, more traditional way, of referring to a plagiarism detection engine.

Plagiarism source - Where a student submission that is judged to be plagiarised has been copied from.

Primary diagonal - The diagonal going from the top left to bottom right of a similarity visualisation along which most plagiarism seems to occur. This represents parts of documents in approximately the same position in both.

Proactive plagiarism policy - A policy of an academic institution where plagiarism is actively sought out on a regular basis, perhaps by using automated detection methods and cases are followed up when they are found.

Professional plagiarism - Plagiarism in a professional setting, for instance copying an internal report or company Web page from another source or using a service that writes standard CVs or job applications.

Reactive plagiarism policy - The academic policy where plagiarism is not actively sought out but is taken seriously and followed up when it is identified during the course of marking.

Root word - A base word by which a given word could be represented. The ways of finding this word vary but might include taking the word it would be found under in a thesaurus, then standardising this word to a standard tense.

Sentence Metric - A simple metric that measures the number of sentences in common between two documents.

Similarity - Where two submissions have words or ideas in common they are said to be similar. When they have been looked at by a tutor they may also be judged to be plagiarised.

Similarity rank - The ordering of pairs of submissions in a list from most to least similarity under a given metric.

Similarity rank list - The list of ordered pairs of student submissions from most to least similarity under a given metric.

Similarity score - A numeric value signifying how similar two student submissions are under a given metric. Here 0 represents no similarity, 100 represents complete identity and values in between represent different levels in between.

Similarity system - A system that automatically produces similarity scores for pairs of submissions in a corpus and allows a similarity rank list to be generated ordering the pairs of submissions from most to least similarity.

Simple metrics - A metric from which a similarity score can be produced directly from a representation of two submissions without needing intermediate processing stages, such as processing localised similarity scores for fragments and combining them.

Similarity intersection - The intense areas of a similarity visualisation that correspond with runs of similarity in the two associated documents.

Similarity visualisations - A graphic showing areas of similarity in two documents. The colour of each pixel of the graphic is generated by using a fragmentary metric on overlapping fragments from equivalent parts of both documents. The associated graphic should then develop intense areas where the documents contain runs of similarity.

Singularly sourced - A plagiarism source that has been copied from once only.

Singular metric - A metric which measures a property of an individual submission, for instance the proportion of times the word 'the' is used in free text.

Source code - A language such as C or Java where the number of and combinations of words and characters available are constrained so that it is interpretable as instructions by a machine.

Source code plagiarism - Plagiarism of source code submissions, where two students have handed in programs where one has been derived from the other in some way. Detecting this is a well understood area since the constrained language reduces the number of possibilities that must be checked.

Structural Metrics - A metric that measures a property of one or more submissions where knowledge of the structure of the documents is needed.

Structure metrics - A count of some property of a pair of documents that might involve tokenisation. This has been redefined to remove the inconsistencies from the literature but is not considered a sensible classification, in fact the inclusive redefinition makes these the same as with paired metrics.

Structure metrics systems - A system that uses only a combination of structure metrics to find similar submissions. This has been redefined to remove the inconsistencies from the literature but is not considered a sensible classification.

Student plagiarism - Plagiarism by students, where a student hands in a submission which contains the words or ideas of another student or ideas found in another source.

Student submissions - Work handed in by a student for academic credit that is an attempt to meet an assignment specification. These submissions may be later judged to be plagiarised.

Stylistics - The branch of linguistics that analyses how, when and why particular words or phrases are used by different writers.

Summarising - A plagiarism technique where the source is reduced into a student submission.

Superficial Metrics - A measure of similarity that can be gauged by looking at submissions without any knowledge of their linguistical properties.

Synthetic corpus - A corpus of documents that have been generated using synthetic means by taking sequences of words or characters in a known and defined order.

Thesaurising - A technique for plagiarism where words in a source are replaced by synonyms or changed in such a way that the submission makes the same points but the intention is that the plagiarism will not be discovered.

Tokenisation - A technique for plagiarism detection where certain terms are replaced by relevant tokens before automated detection commences.

VAST - The Visualisation and Analysis of Similarity Tool. A prototype tool that displays two documents and the associated similarity visualisation and allows a user to navigate around the documents by focusing in on areas of the visualisation. They then verify if areas represented as similar contain plagiarism or not.

Visual metrics - A metric which is based on some property of the similarity visualisation that would be generated for a given pair of student submissions.

Visualisation - The technique where data is translated into a diagrammatic notation to make it easier to view. Similarity visualisations are an example of a visualisation technique.

Web plagiarism - Plagiarism where the source document is a page on the World Wide Web. This can be 'copied and pasted' into a student submission with the minimum possible work.

Web-based plagiarism detection services - Services on the Web that take in student submissions and return a list of Web sources that the submission may have been plagiarised from, commonly by presenting the student submissions with appropriate sections hyperlinked to external sources. The self reported market leader is plagiarism.org.

Weighting - The process by which the values of different metrics are scaled to influence the closeness calculation to a greater or lesser degree.

Words Metric - A simple metric that measures the number of sequences of words of a chosen length in common between two documents.

Words Pair Metric - A simple metric that measures the number of sequences of word pairs in common between two documents. Identified as the most effective simple metric.

A Descriptive Taxonomy of Student Plagiarism:

Fintan Culwin & Thomas Lancaster
Center for Interactive Systems Engineering
School of Computing
South Bank University
London SE1 0AA

Abstract

The lack of agreement on a descriptive taxonomy of plagiarism may be preventing academic researchers and tutors from effectively communicating about the nature of plagiarism and may also be impeding the development of effective anti-plagiarism policies and tools. An initial descriptive taxonomy is proposed which attempts not to preclude any further discussion of policy and tools.

The Need For A Taxonomy

The Oxford English Dictionary defines **plagiarism** as:

“The wrongful appropriation of purloining and publication as one's own, of the ideas, or the expression of the ideas (literary, artistic, musical or mechanical etc.) of another”.

The OED definition of plagiarism is predicated upon what might be described as individual scholarly plagiarism. The topic of University level taught-course plagiarism (student plagiarism) is more complex and hence requires a larger and more comprehensive definition. A complete definition of student plagiarism is also a descriptive taxonomy of the types and, to an extent, the causes of plagiarised submissions.

Although the literature on student plagiarism is not particularly extensive, and largely anecdotal, it is clear from examining it that the term plagiarism is used by different authors to mean significantly different things [V&W96][Wis96] [C&N95]. It is also apparent that there is an impression amongst academic staff that the incidence of plagiarism is dramatically increasing and that this growth is caused by the extensive recent expansion of Internet and Web based

information services [Gaj98][Rya98][Sta00]. This in turn has raised interest in free-text anti-plagiarism policies and automated systems to detect it [Den99].

It is against this background that we offer this taxonomy of plagiarism with the intention that it will allow tutors to communicate more accurately and effectively about the incidence and nature of plagiarism incidents. It is also intended to promote and guide research into plagiarism with the intention of gaining a clearer understanding of its causes and nature, so as to inform effective anti-plagiarism policy and practise. In many places the description of the taxonomy raises many questions that are suitable for empirical investigation and we have tried not to pre-judge or predict any answers that may be obtained. However this taxonomy is informed by ongoing research into automated free text plagiarism systems which may already have unduly influenced our perception of the problem and hence of the taxonomy. In places some suggestions for possible automated approaches, or comments upon the likely effectiveness of approaches, have been included. As a descriptive taxonomy it attempts to provide a set of terms which allows the nature of plagiarism to be clearly and unambiguously exposed. In so far as is possible this paper simply presents the taxonomy without making any comment upon the implications for institutional policy. All of these issues, particularly that of automated detection, are being addressed in other work that is being conducted by the authors.

We feel that this introduction must end with an important warning. As a descriptive taxonomy this work may inform the development of an institutional plagiarism policy but is not, of itself, such a policy. For example at the end of the taxonomy the reasons why a student may plagiarise are described in order to enumerate them. An institutional policy informed by this enumeration would have to indicate how each motivation was to be decided and what the institution's policy towards it would be. In particular we feel it is essential to point out that such a policy must, in the UK context, clearly separate the detection of plagiarism from the decision of intent and both of these from the institutional verdict and extent of any penalty.

The Taxonomy

The first term in the taxonomy has already been introduced in the section above. The common understanding of the word plagiarism is divided between **professional plagiarism** and **student plagiarism**. Professional plagiarism refers to the act of an individual in the context of their professional activities appropriating without acknowledgement the intellectual activity of another. The definition of student plagiarism involves the same concept of appropriation without acknowledgement, but places it within the context of educational assessment. Although an

individual student may plagiarise another individual's work when engaged in original research, this would seem to lie more within the concept of professional plagiarism. Student plagiarism per se involves an undergraduate, or taught postgraduate, student misrepresenting the ownership of the content of a **submission** that is produced in response to an **assignment specification**, and which gains them academic credit towards an award. It is this aspect of attaining academic credit that differentiates between professional and student plagiarism, although we suggest that the qualification is only required if the distinction is not clear from the context. It is also the case that a social element, involving a **group** of students working upon the same or similar specification is usually, though not necessarily, involved in situations that lead to student plagiarism. This social aspect leads to the collection of a set of submissions produced by the group, some of which are, at least in part, potentially plagiarised.

A **corpus** is a set of documents sharing some kind of common bond. The documents in a corpus are expected to share subject matter as well as document structure, e.g. a set of essays on the Russian Revolution, or a set of C programs implementing a binary search. Within the context of student plagiarism the set of submissions that are made in response to a particular assignment specification can be described as an **assignment corpus**. This may require further qualification if a particular assignment specification is presented on more than one occasion and/or in more than one institution. One other significant qualification of the term corpus in the context of student plagiarism is the collected submissions of an individual student throughout their academic progress. This collection could be described as a **student's corpus** and may be of interest as a plagiarised submission within it might be expected to show significant stylistic differences from the rest of the submissions.

Intra-corporeal and **extra-corporeal** plagiarism refer to the location of the **source** and (possibly only partial) **copy** of plagiarised submissions, as illustrated in Figure 1. Submissions 1 and 2 illustrate **direct** intra-corporeal plagiarism with 1 as the source and 2 as the copy. Submissions 3 and 4 illustrate direct **mutual** intra-corporeal plagiarism. Mutual plagiarism is where the source and the copy cannot be decided, as opposed to has not yet been decided, and may be caused by excessive collusion during the completion of the assignment.

Submission 5 is an example of extra-corporeal plagiarism, being a copy of resource 6 which has not been produced in response to this use of the assignment specification and so is not included in the corpus. Submissions 7 and 8 also appear as mutual intra-corporeal copies but this is caused by them both being indirectly plagiarised from resource 9 which is outside the corpus. In this situation the authors of the plagiarised submissions may be totally unaware of the existence of a

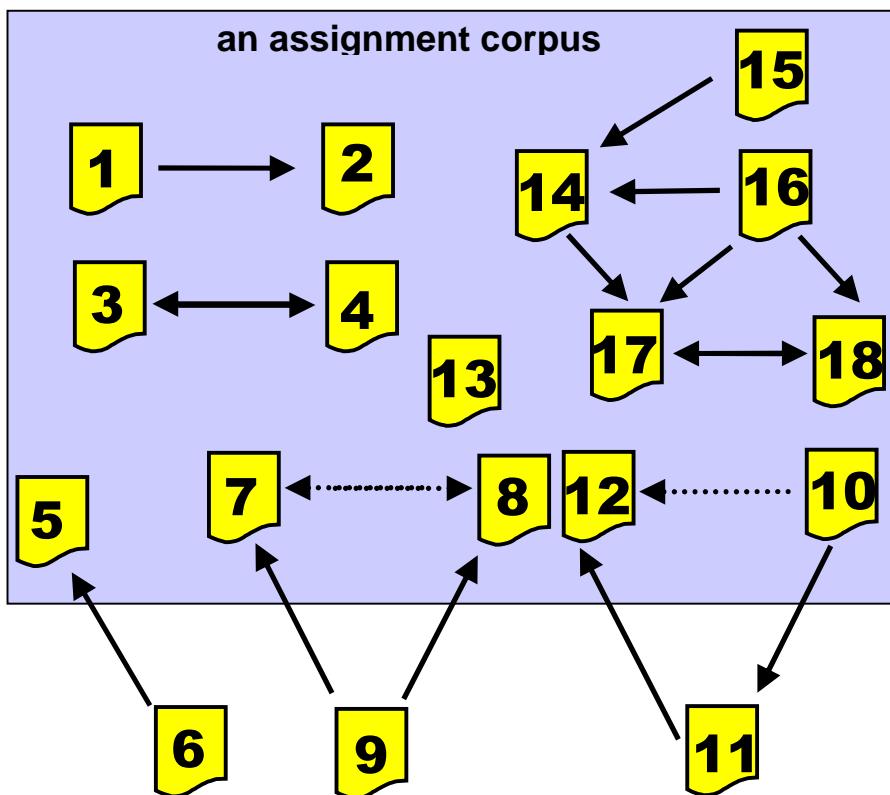


Figure 1: Intra- and extra- corporal plagiarism.
(arrows always point from the source to the copy)

copy within the assessment corpus and this may have the consequence illustrated in 10, 11 and 12. Here 12 is an **indirect intra-corporeal** plagiarised copy of submission 10, which occurred by the extra-corporeal copy 11 being the source of copy 12. This might possibly occur when an assignment specification is being used on more than one occasion and/ or in more than one institution. Submission 13 is a **singleton** and may represent an original submission or a plagiarised submission whose source is unknown. Finally in this part of the taxonomy submissions 14 to 18 form a **cluster** of plagiarism, with the direction of copying as shown, and with submission 15 forming an **outlier** from the cluster.

Submission 14 also illustrates the possibility of a submission being plagiarised from multiple sources, known as a **multiply sourced** copy. However, for ease of analysis a multiply sourced submission can always be broken down into a number of simple relationships between source/ copy pairs. For example submission 14 can be viewed as a pair of source-copy relationships between 15 and 14 and between 16 and 14. Hence a cluster can always be considered as a set of pair relationships which leads into consideration of the precise nature of the **plagiarism relationships** between pairs, which in turn will inform the extent of intra-corporeal plagiarism within a cluster.

The derivation of a diagram such as that presented in Figure 1, but restricted to intra-corporeal plagiarism, requires that every submission is compared, in some way, with every other submission and a **judgement** made regarding the **significance** of any **plagiarism indicators** between them. To accomplish this some quantitative measure (**metric**) of each submission has to be made and the resulting metric, or set of metrics, used as the basis of the judgement.

In considering the nature of plagiarism indicators one distinction is the **granularity** used for the measure. A metric can either be applied to the entire submission or can be applied to a part of it: i.e. section, page, paragraph or sentence, collectively known as **fragments**. When the entire submission, known as **gross granularity**, is used as the basis for measurement the evidence for any resulting detected plagiarism may be difficult to demonstrate. For example a metric may be based upon a histogram of letter frequencies and an **automated judgement** may be based upon the closeness of two histograms. Assuming that this was not a false hit, that is an incorrect automated judgement, the reasons why the judgement was made may not be immediately apparent and any tool implementing the system would be unable to provide assistance.

Alternatively a metric derived from measurements based upon **fragmentary granularity** may allow the visual similarity to be easily demonstrated. For example a metric based upon the occurrence of uncommon words in a sentence may lead to an automated judgement that two sentences are similar, which can readily be demonstrated for human confirmation. In this scenario the decision of which words are uncommon could be made with respect to a histogram of all words used in all submissions in the corpus, an example of a **corpal metric**.

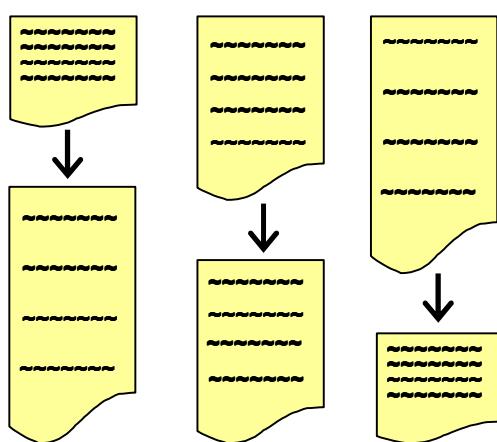


Figure 2: Source & destination extents.

Where fragmentary granularity is used in the analysis the **extent** of the plagiarism between two submissions can be measured. The considerations which inform this definition are illustrated in Figure 2. In this diagram the upper images represent three different source submissions, each of which has an associated copy below it. In the left-hand pair the entire extent of the source has been fragmented and included within the larger copy, hence the extent of plagiarism **on** the source is high but the extent **within** the copy is low. The right-hand pair illustrate the opposite extreme where there is minor plagiarism on the source but which results in extensive plagiarism within the copy. The middle pair illustrate the situation where both source and copy extent are moderate. These considerations can be characterised on the graph shown in Figure 3.

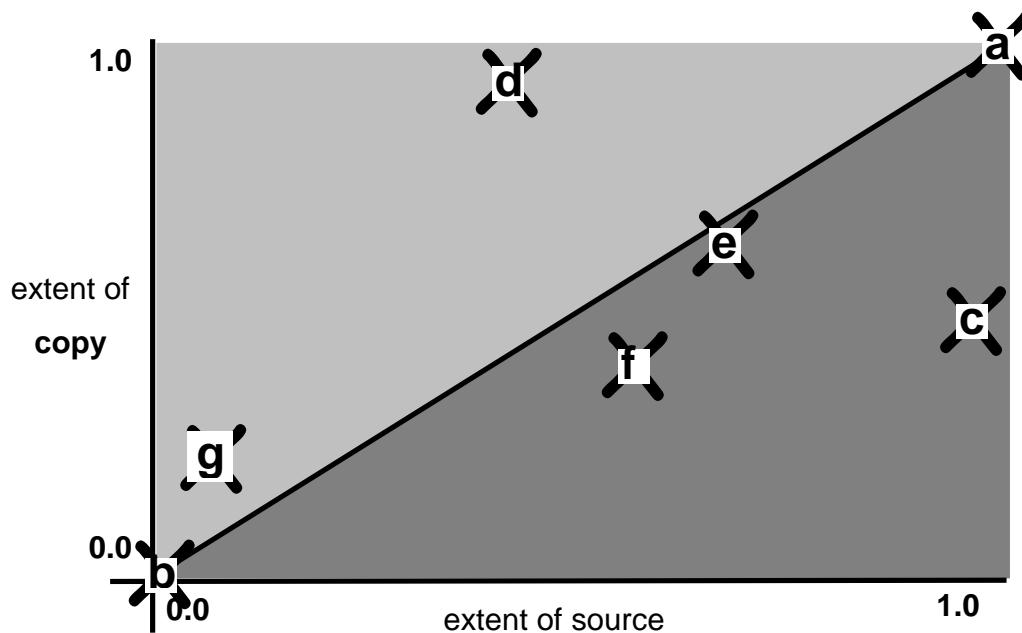


Figure 3: Extent relationships.

Each point on the graph indicates a relationship between two submissions. The point labelled **a** indicates a situation where all of a source document has been supplied as all of a copy and indicates (irrespective of the internal sequence) identical submissions, whereas point **b** indicates a situation where there is no connection between a pair of submissions. The straight line which connects these two points defines the set of pairs of submissions which are equal in size, according to the fragmentary measure which is being used. The (darker) area below the line thus represents pairs where the size of the copy is greater than the size of the source. For example

point **c** is a pair where approximately 100% of the source has been taken to provide 50% of the copy. Correspondingly the (lighter) area above the line represents pairs where the size of the copy is less than the size of the source. For example point **d** is a pair where approximately 50% of the source has been taken to provide 100% of the copy. Accordingly pairs below the line may represent **expanding plagiarism** and those above the line represent **contracting plagiarism**.

It might be supposed that contracting pairs are indicative of more extensive plagiarism and hence points towards the top and right of the graph indicate increasing apparent plagiarism. Points **e** and **f** have been included to illustrate the **boundary** of plagiarism, if **e** is adjudged to be a plagiarism relationship and **f** is adjudged not to be then the boundary must lie somewhere between the two submissions. However the nature of the curve that separates significant plagiarism from non-significant similarity is not clear. Figure 4 illustrates two possibilities, the dotted line indicates a simple straight line boundary. The curved line defines an area above it where the relationships are deemed to be plagiarised. There is no a-priori reason why either of these, or any other similar, boundaries should be favoured over any other.

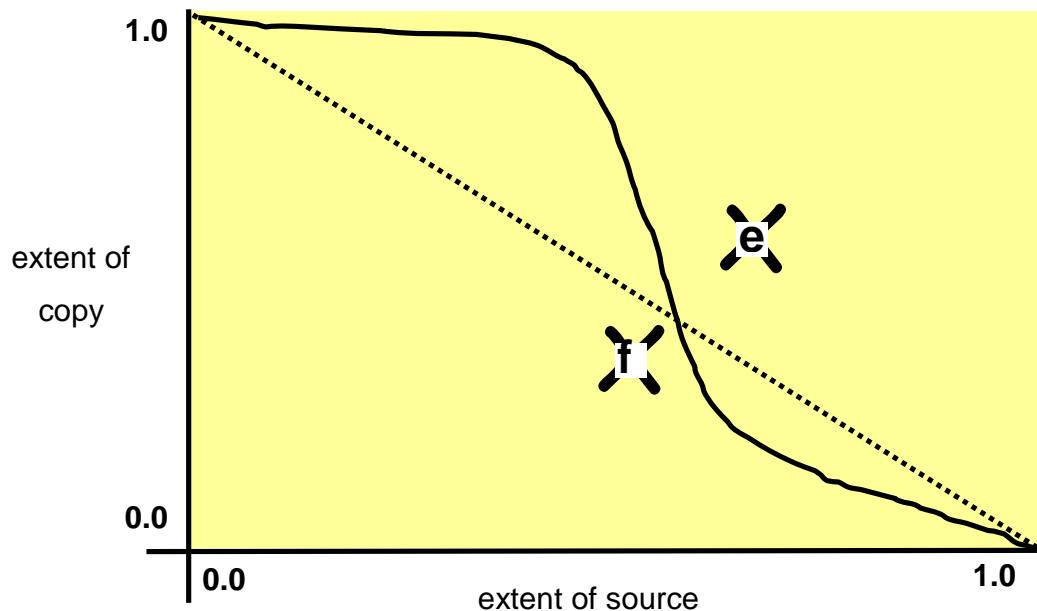


Figure 4 Boundary curves.

In addition to using this space for the classification of individual relationships the characteristics of all the relationships within an assignment corpus can be used to describe the extent of plagiarism within it. For example the average point from the corpus could be used to give a crude indicator of the **amount** of plagiarism within a corpus. Additionally, the overall location and shape of the area

defined by all relationships may also give an indication of the originality of the cohort as a whole. Both of these measures could be used in comparison with a similar group in other institutions, the same group in another subject or successive groups taking the same unit and assessment, to make comparative statements about the extent of plagiarism.

The point labelled **g** on Figure 3 is included to warn against an over-mechanistic interpretation of the extent relationships. It defines a relationship where only a small extent of the source is included as a small part of the copy, and might usually be adjudged to be either a coincidence or minor plagiarism. However it may be the case that although it is small in its extent it contains a crucial part of the assignment and hence may indicate significant plagiarism. For example in a programming assignment this may be the source code, or for a science investigation this may be the experimental data and analysis. Accordingly plagiarism of this nature is classified as **crucial plagiarism**. This is likely to be a significant factor where the costs of producing the plagiarised material are very high relative to the costs involved in attempting to disguise it.

A **false hit** is a situation where an automated judgement indicates that two unrelated submissions are plagiarised and can be compared to a **missed pair** where an (automated or manual) plagiarism judgement should be made but is not. Figure 5 attempts to clarify this consideration by considering only two possible relationships copied or independent and two possible judgements plagiarised and innocent. The four possible outcomes are indicated as two correct outcomes, detection or exoneration and the two incorrect outcomes described above.

		<i>relationship</i>	
		copied	independent
<i>judgment</i>	plagiarised	correct detection	false hit
	innocent	missed pair	correct exoneration

Figure 5 Relationships and Judgments

The identification of corresponding fragments in pairs of submissions depends upon a judgement made upon metrics derived from each fragment. It is possible that identical fragments will be found in two submissions, and it is also possible that some of these may be coincidental. It is also

possible that some attempt will be made to **disguise** the copied fragments. One possible disguise may be the **sequence** in which the copied fragments appear in the plagiarised submission, although an investigation based upon appropriate fragmentary granularity should not be troubled by this. Where the pairs are expansive then this strategy may result in **mosaic plagiarism**, which is where a contiguous fragment from the source is further fragmented and scattered through the copy, possibly with resequencing. Here again an investigation with appropriate fragmentary granularity should not be troubled by this, with the appropriate size here being the size of the fragments in the copy.

These considerations may suggest that an effective investigation should be based upon a lower, rather than a higher, fragment size. However one other disguise strategy may be **re-fragmentation**, where fragments from the source are either combined or further divided in the copy. This in turn suggests that an effective investigation may have to use a range of fragment sizes to look for all possible disguises.

Before leaving consideration of the nature of the metrics that might be used to assist in automated plagiarism detection a distinction needs to be made between **superficial** and **structural** metrics. These differ in the extent that they attempt to look beneath the surface form of the submission. Many of the metrics suggested so far are superficial attribute counting metrics, e.g. sentence length, character histograms etc. It is also possible to derive metrics which are based upon a consideration of underlying aspects, e.g. the sentence parsing complexity. It would seem possible that structural metrics would be more difficult to disguise and would also be indicative of an individual's writing style and hence of use in detecting plagiarism with a student's corpus. Structural metrics have been shown to be particularly useful in constrained-text (e.g. program source code) plagiarism detection [V&W96].

Without undue speculation into the motives of plagiarists: under some circumstances **paraphrasing** and **summary** may also be considered as attempts to disguise. Paraphrasing involves changing the form of words used to express meaning between the source and the copy, without significantly adding to the meaning or the argument that supports it. The most superficial level of paraphrasing may simply involve **thesaurising** the source by the replacement of words by synonyms, including restructuring the fragment to require an antonym. A closely related disguise might be to restructure the fragment so that a different form of the same root word is required. Summary can be thought of as contractive paraphrasing with the intention of preserving, possibly only parts of, the meaning whilst dispensing with some, or all, of the supporting argument.

In the context of student plagiarism, defined above as involving undergraduate or postgraduate taught courses, it is not expected that the ideas or arguments presented by the students be particularly original. Hence the boundary between legitimate repetition of an argument and paraphrasing per se may only be defined by the intent to deceive, and as such may become a matter of judgement by the tutor. In the context of automated detection the extraction of meaning from free text is one of the aspirations of artificial intelligence research. Although some progress is being made such technology is currently impractical for applications such as this [Mur00]. Without a mechanism for the extraction of meaning it would seem that the automatic detection of paraphrasing is not possible.

It may be possible for the **superficial paraphrasing**, as described above, to be automatically investigated but the costs, in terms of preparing and operating the detection system, would be significantly higher than other, simpler systems. It can be argued that paraphrasing requires more engagement with the material being plagiarised than the other disguise mechanisms and so if an automated anti-plagiarism system encourages paraphrasing instead of simple copying it would have had some indirect benefit. This cost/ benefit payoff would also have to be considered when deciding between superficial and structural metrics.

The discussion above has focussed upon intra-corporeal plagiarism as this is the most amenable to automated investigation because, by definition, the source and the copy are both available for investigation. Extra-corporeal plagiarism includes **Web plagiarism**, where the source of the plagiarised submission is located and downloaded from a Web resource. It may be possible for Web plagiarism to be detected by making use of Web facilities. It is likely that a student would have located the resource to plagiarise via one of the major Web search engines and this can be countered by attempting to locate possible sources using the same search engine technology. To facilitate this it may be appropriate to identify the least common words in a submission and then to automatically feed them through the search engines, retrieve the documents returned and include them in the corpus. It might then be the case that the source resource has been retrieved and will be flagged for investigation in the usual manner.

This approach would not be effective if the student were to obtain a submission from one of the Web sites which offer to supply a solution written to the assignment specification for a fee, known as **commercial plagiarism**. However such a submission would be likely to exist as an isolated and distanced sentinel on an intra-corporeal distribution graph. It might be the case that submissions obtained in this way would have some characteristics that distinguished them from the submissions produced by the rest of the group. As an obvious example in the UK context, a submission purchased from a US site would have US spelling which few if any of the other

submissions would be expected to have. This might also be expected to be the case for any non-Web source resources such as published documents or books.

An isolated singleton approach to detection might also be useful if a student corpus were being maintained (a corpus comprising of all the submissions that a student has made during their institutional academic progress). Some of the metrics used to measure the differences between submissions from different students, e.g. a common word frequency histogram or sentence length distribution, might be expected to be a relatively constant attribute of an individual's writing style. A plagiarised submission would be unlikely to have the same pattern of characteristics. That is, although it may by chance have a similar sentence length distribution it would be unlikely to also have a similar word frequency histogram. It would be an interesting empirical question to determine how much of such variability exists in student submissions and an interesting matter of policy to decide how much of a variation in an outlier could be adjudged to indicate plagiarism.

One possibly useful variant of this approach would be to use **overlapping fragments** to attempt to detect distinct differences in style within an individual submission. An example of an overlapping fragment might be a span of 500 characters. By measuring a fragment of (say) the first 500 characters of a submission, then (say) characters 5 to 505, and then characters 10 to 510, any plagiarised passages of approximately 500 characters in length might be detectable. The fragment which contains the plagiarised material would be expected to have different characteristics which could be demonstrated using the same mechanisms as suggested above.

Returning to intra-corporeal plagiarism, the mechanism by which a submission becomes plagiarised might also benefit from some terminological clarity. Pairs of purely intra-corporeal plagiarised submissions may originate due to one being the source and another being the copy, or may originate from mutual plagiarism. Where there is a source/ copy relationship this can occur through **theft**, **gift** or **accident**. Theft occurs where the plagiarist gains access to the submission of another student and submits it, without the knowledge of the originator. This category includes **opportunistic theft** where, for example, a floppy disk containing the submission is left in a workstation, as well as **deliberate theft** where, for example, a floppy disk may be stolen from a workstation whilst the originator is distracted or absent. Gifting occurs where the originator intentionally and knowingly allows the plagiarist access to their submission; the reasons for this may include friendship, payment or intimidation. Accidental occurrences of source copy pairs have been claimed by students using automated handin systems and have been explained as the inadvertent submission of a the wrong file, possibly because the machine had previously been used by another student to complete their handin.

Mutual plagiarism may occur when two students, or two groups of students, collude too closely on the preparation of a submission. This introduces the collusion continuum shown in Figure 6 which attempts to define the relationship between **collaboration**, which may be encouraged, **collusion**, which may be suspect and **copying**, which should always be an academic offence.

Collaboration may be defined as students actively supporting each other and in some circumstances this may actually be required in order for the assessment to be completed. For example a software development assignment may require different students, or groups of students, to independently develop constituent parts of a single artefact. During the completion of the assignment it may only be possible for one group to determine if their part will work in situ by having collaborative access to the parts being produced by the other groups. Collusion may be defined as allowing access to parts of the submission, or to original material which directly informs parts of the submission. For example in a software development assignment a pair may collude by jointly developing a high level design before working independently on the detailed design and implementation. Copying is clearly defined as the presentation, possibly with some attempt at disguise, of material that was not produced by the identified author, or authors, of the submission.

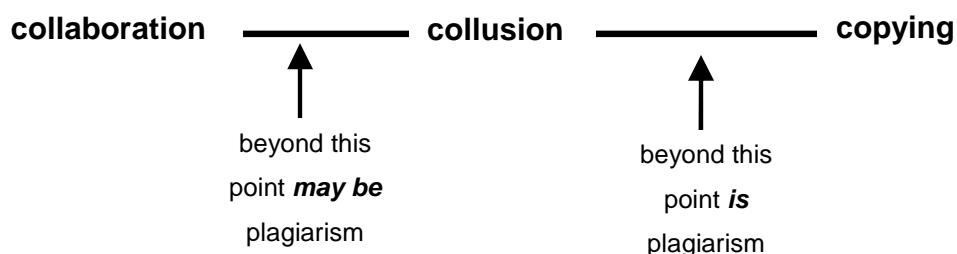


Figure 6 Collusion, collaboration and copying continuum

The presentation of the continuum in Figure 6 indicates that the possibility of plagiarism starts somewhere between collaboration and collusion and will definitely occur somewhere between collusion and cheating. The precise point at which plagiarism occurs on this continuum will vary with the subject of the assignment, the nature of the assignment and policy of the institution or individual tutor concerned. We would suggest that it is essential that students are given some clear and unambiguous indication of where along this continuum an offence will be adjudged to have occurred.

The nature of the subject of the submission may also be of significance here. In some subjects, for example Law, the verbatim citation of references is required. This would not automatically be

considered as plagiarism as the culture of the subject would indicate that this is acceptable and the citations would be clearly identified and sourced. It might be expected that automated plagiarism detection system should be constructed so as to identify and exclude correctly sourced passages. However there is the possibility of a copy of a submission in such a culture can be identified as plagiarised by virtue of the selection of (correctly referenced) citations that have been chosen: a situation known as **reference plagiarism**.

The motivation to plagiarise may be **intrinsic**, where the student does not have the ability, competence or organisation to complete the assignment unaided; or **extrinsic** where the student has the capacity but is prevented for some other reason from completing it. One intrinsic situation that is particularly notable is where a tutor sets an assignment specification which is impossible for a significant proportion of the group to complete and leads to an outbreak of **mass plagiarism**, where a large proportion of the group resorts to cheating. This may be caused by a specification being too difficult for the majority of the students to be able to complete without assistance or by there being inadequate resources, leading to a situation where there is an over-reliance upon the few available resources. Extrinsic plagiarism include situations where mitigation of external circumstances should be claimed by the student but for some reason is not. Extrinsic plagiarism might also include **innocent plagiarism** where there is a honest belief that the behaviour is acceptable or even expected.

One further anti-plagiarism measure which might be regarded as prevention rather than detection is to **escrow** the submission during its development. This can be the non-public escrowing of drafts during development with the escrow archive being opened only if a plagiarism investigation is required. Alternatively public escrowing may involve placing the drafts of the submission into a public area. This latter possibility may involve the use of computer supported co-operative working (CSCW) systems that could be combined with the summative assessment of process as well as product. So far as can be established no plagiarism prevention escrow system has been used and although a private system may result in a plagiarised archive and a public system may result in a race to plagiarise, it is an area of empirical investigation that might be worthy of investigation.

Our final taxonomic distinction relates to a classification of institutional plagiarism policies which we would characterise as ostrich, reactive and proactive. An **ostrich policy** may, in many cases, be synonymous with the absence of a policy. In these situations the processes to be followed when plagiarism is detected would be unclear and tutors reporting upon incidents may get the impression that the institution had rather they had not located it. It is possible that this may actually be the unwritten culture of the institution with staff discouraged from reporting any

plagiarism that they come across, sometimes accompanied by an official denial that any plagiarism takes place. A **reactive policy** has amongst other things, such as clear advice to students concerning the meaning of copying, a clearly defined and jurisprudently appropriate procedure to be followed when an incident is detected. A **proactive policy** extends a reactive policy to indicate passive or active measures to prevent or detect plagiarism. A **passive measure** may for example include a prohibition on reusing a coursework specification within a defined period. An **active measure** may require a team of markers to allocate submissions randomly between themselves so that students cannot know in advance which tutor will be marking a particular submission.

Glossary

accident, where plagiarism is detected on a *submission* that should not have been included in the *corpus*, or where a duplicate submission exists in the *corpus*.

active measure, a part of a *proactive policy* that attempts to detect, and hence deter, plagiarism.

amount, the *extent* of plagiarism within a *corpus*

assignment corpus, the collection of all *submissions* produced in response to an *assignment specification* (aka in context simply as *corpus*).

assignment specification, a statement of what is required for a student, or group of students, to gain academic credit.

automated judgement, a *judgement* made by a computerised plagiarism detection system.

boundary the point where the *source* and *copy* *extents* becomes *significant*.

cluster, a group of *submissions* in a *corpus* which are all inter-connected by plagiarism relationships.

collaboration, where students assist each other in the preparation of *submissions* in a way that is acceptable, given the nature of the *submission* and subject

collusion, where a student has access to material which will form part of another's *submission* and which may consequently unduly influence the contents of their *submission*.

commercial plagiarism, where the plagiarist pays for a *submission* which is then presented as their own work.

contracting plagiarism, where the *source* is larger than the *copy*.

copy, the *submission* made (possibly only in part) from a plagiarised *source*.

copying, where a student includes material in their *submission* where an acceptable degree of assistance has been given, or includes material which should be referenced or attributed and which is not.

corpal metric, a measurement made on the whole *corpus*.

crucial plagiarism, where a small *extent* of plagiarism may be *significant*.

disguise, the changes made to plagiarised material in an attempt to prevent detection.

escrow, where drafts of *submissions* can be placed in a public or private repository as a defence against a possible subsequent allegation of plagiarism.

extent, the proportion of the *source*, or of the *copy*, which is involved in a *plagiarism relationship*.

extra-corporeal plagiarism, where the *source* of a *copy* is not located in the same *corpus*

extrinsic, where a student plagiarises even though they would be able to complete the *specification*.

expanding plagiarism, where the *source* is smaller than the *copy*.

false hit, an invalid judgement of *significant plagiarism*.

fragment, a *granularity* of less than a complete *submission*.

gift, where the *source* material is used with the author's permission.

granularity, the extent of *submission* upon which an individual measurement is made

gross granularity, a *granularity* of a complete *submission*.

group, a number of students completing an *assignment specification* to the same deadline.

innocent plagiarism, where a student is unaware that plagiarism is an academic offence.

intra-corporeal plagiarism, where the *source* and the *copy* are located in the same *corpus*.

intrinsic, where a student plagiarises because they are unable to complete the *specification*.

judgement, a decision, possibly automated, that the *plagiarism relationship* is, or is not, *significant*.

mass plagiarism, where the *amount* of plagiarism within a *corpus* is considered excessive.

metric, a quantitative measurement of an individual *plagiarism relationship*.

missed pair, a *significant plagiarism relationship* within a *corpus* which is not detected.

mosaic plagiarism, *expanding plagiarism* where fragments of the *source* are scattered through the *copy*.

multiply sourced, a *copy* which has more than one *source*.

mutual plagiarism, a plagiarism relationship between two *submissions* which has no identifiable *source* and *copy*.

ostrich policy, where an institution has no policy for the processing of plagiarism, possibly because plagiarism is deemed not to be happening.

outlier, a member of a *cluster* which is connected to only one other member of the group.

overlapping fragments, a process of detecting plagiarised material within a submission by making measurements on *fragments* that slide along the submission with one *fragment* separated from the next by less than the length of a *fragment*.

paraphrasing, a *superficial disguise* where the form of words is changed,

passive measure, a part of a *proactive policy* that attempts to prevent plagiarism.

plagiarism, the presentation of ideas or material produced by another person as if it were one's own.

plagiarism indicators, the measurable characteristics of the source, or of the copy, which evidence the plagiarism

proactive policy, an institutional policy which adds to a *reactive policy* measures to prevent or detect plagiarism.

professional plagiarism, *plagiarism* in the context of an individual's employment.

plagiarism relationship, the measurable characteristics between the source and the copy which evidence the plagiarism

re-fragmentation, a disguise strategy where fragments are further split or combined.

reactive policy, an institutional policy which states only what will happen from the point where plagiarism is detected.

reference plagiarism, where material is included in a submission and correctly attributed but the choice of material to present is copied from another submission.

significance, the point at which a judgement is made that the *extent* of the plagiarism indicates an academic offence has taken place.

singleton, a *submission* in a *corpus* which has no plagiarism relationships with any other *submission* in the same *corpus*.

source, the material from which a plagiarised copy is made.

student plagiarism, *plagiarism* in the context of gaining academic credit, also known in context simply as plagiarism.

student's corpus the collection of all *submissions* produced by an individual student during their academic progress.

structural, a *metric* which measures a non-superficial aspect of a *submission*.

submission, material produced by a student in response to an *assignment specification*.

summary, contractive paraphrasing.

superficial, a *metric* which measures a surface aspect of a *submission*.

theft, where the source material is used without the author's permission.

thesaurising, a superficial disguise involving synonyms or antonyms.

Web plagiarism, extra-corporeal plagiarism where the source is obtained from the World Wide Web.

Conclusion

As stated in the introduction there are many places in this description of a possible taxonomy where more questions are raised than are answered, or even elaborated. We are aware that much of the terminology included, and the description of it, is influenced by the perspective that we have as computing academics involved in the development of automated plagiarism detection tools. In particular we are concerned that this perspective may cause us to be blind to terms, or refinements of terms, that need to be included to make the taxonomy more complete and also

more meaningful to other disciplines. Nonetheless we are not aware of any other attempted taxonomy and would present this as a starting point in the hope that it proves sufficiently useful to allow those concerned with student plagiarism to communicate more effectively.

The authors would like to thank Pete Chalk of South Bank University: London and Jim Benson of the Royal College of Physicians: London for the comments that they made on drafts of this paper.

References

- [C&N95] Pragmatic anti-plagiarism
Fintan Culwin & Jeff Naylor
Proc. 3rd Conference on the Teaching of Computing, DCU Dublin IE 1995.
- [Den99] 'The Web's Plagiarism Police'
Andy Denhart, Salon Technology (1999),
<http://www.salon.com/tech/feature/1999/06/14/plagiarism/index.html>
[correct on 28/3/00]
- [Gaj98] 'Issues in Plagiarism for the New Millennium: An Assessment Odyssey'
Joan Gajadhar, The Open Polytechnic of New Zealand (1998),
<http://ultibase.rmit.edu.au/Articles/gajad1.html>
[correct on 28/3/00]
- [Mur00] 'Automating Content Integration with Autonomy'
Gerry Murray, International Data Corporation (IDC)
<http://www.autonomy.com/tech/idc.pdf>
[correct on 28/3/00]
- [Rya98] 'Student plagiarism in an online world'
Julie J. C. H. Ryan (1998)
http://www.asee.org/prism/december/html/student_plagiarism_in_an_onlin.htm
[correct on 28/3/00]
- [Sta00] "Plagiarism in Colleges in USA"
Ronald B. Standler (2000)
<http://www.rbs2.com/plag.html>
[correct on 28/3/00]
- [V&W96] 'Software for Detecting Suspected Plagiarism:
Comparing Structure and Attribute Counting Systems'
Kristina L. Verco & Michael J. Wise, University of Sydney, Australia, (1996).
Proc. 1st Australian Conference on Computer Science Education,
Sydney, Australia, July 1996.
- [Wis96] 'YAP3: Improved Detection of Similarities in Computer Program and Other Texts'
Michael J. Wise, University of Sydney, Australia,(1996).
Proc. ACM SIGCSE '96, Philadelphia, USA, Feb 1996.

Plagiarism, Prevention, Deterrence & Detection

Fintan Culwin & Thomas Lancaster

South Bank University, UK.
fintan@sbu.ac.uk & lancaste@sbu.ac.uk
<http://cise.sbu.ac.uk/>

Summary

Many tutors believe that plagiarism, especially copying material from the Web, is a significant and increasing problem in UK higher education institutions. A number of academic and commercial groups are researching the nature and extent of the problem and are developing software tools and systems for plagiarism detection. Recognising that prevention is better than cure, this paper commences by reviewing the advice that has been given by various institutions and agencies on how to specify assignments that are less prone to plagiarism. However, the evidence on the ground is that these precautions do not always prevent cheating and so effective detection systems are also needed. The major part of this paper will introduce a four-stage plagiarism detection model and describe some of the tools that can be used within it. Hopefully the deployment of an effective system will also have a significant deterrent effect.

Biography

Dr. Fintan Culwin is the Head of the Centre for Interactive Systems Engineering (CISE) based at the South Bank University's School of Computing, Information Systems & Mathematics. Mr. Thomas Lancaster is a Research Assistant at CISE and a PhD Student.

Keywords

Plagiarism, Web plagiarism, student cheating, copying, plagiarism detection

Introduction

Any Internet search engine is likely to yield thousands of pages containing the word 'plagiarism'. For example AltaVista (<http://www.altavista.com/>) in April 2001 supplied a list of 83,467 such pages and we have noticed that, since January 2000, the number of sites has grown dramatically. This growth in interest reflects the growth in concern expressed by academics in a number of forums including the ILT's first annual conference 'Learning Matters' [8] and a number of workshops held since. Explicit UK based research evidence related to the nature and scale of the problem was last conducted in 1995 [10], prior to the widespread availability of the Web. This study reported that over 75% of students admitted to cheating and 50% to plagiarising in some way and it would seem that this proportion is likely to have increased with the growth of the Web.

This apparent growth has been discussed at length in the literature, including the British media [2, 3]. Lathrop and Foss' book 'Student Cheating and Plagiarism in the Internet Era' is a good starting point for anyone wanting to find out more about plagiarism detection and prevention [12]. A short Web page covering some similar material to the book and containing other useful links is available at

http://www.asee.org/prism/december/html/student_plagiarism_in_an_onlin.htm. The 'Plagiarism and How To Avoid It' Web site maintained by David Gardner at <http://ec.hku.hk/plagiarism/introduction.htm> offers a similar general introduction to the major issues.

Plagiarism Prevention

The Web is a valuable source for anti-plagiarism advice which can teach both students and tutors what plagiarism is, how to recognise it and how to avoid it.

The following pages are good examples of those that inform students what plagiarism is and how to cite material properly:

<http://www.indiana.edu/~wts/wts/plagiarism.html>
<http://www.writing.nwu.edu/tips/plag.html>

Culwin and Naylor advocate describing a continuum ranging from **co-operation** which is encouraged, through **collaboration** which is reluctantly accepted through to **copying** which is unacceptable [9]. This is shown in Figure 1. Co-operation is explained as talking about a problem and sharing ideas. Collaboration is explained as showing or sharing material that might be included in a final version. Copying is explained as presenting material that was written or developed by another person, possibly with some disguise. Many tutors have noted that the position on

this continuum that divides acceptable from unacceptable behaviour is, to some extent, culturally defined and hence must be made explicit to all students at the start of their courses.

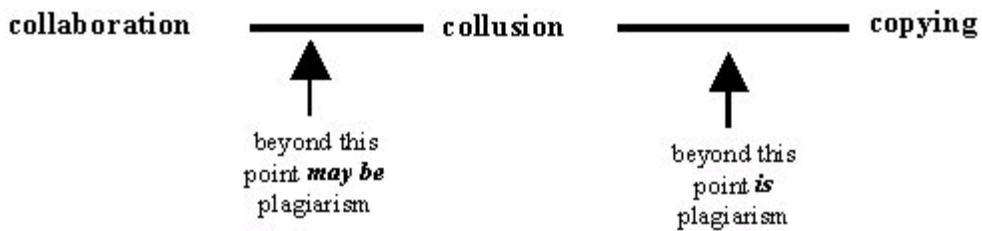


Figure 1: Co-operation/Collaboration/Copying Continuum

Students have been known to use the online essay banks to avoid writing a submission and it is worth being familiar with these. Well known examples include:

<http://www.schoolsucks.com/>
<http://www.cyberessays.com/>
<http://www.gcseworld.co.uk/>
<http://www.planetpapers.com/>
<http://www.netessays.net/>
<http://www.essaydepot.com/>

Many pages give advice to tutors on writing assignment specifications that make plagiarism difficult. Suggestions include:

- Never reuse an old assignment specification as previous submissions have a habit of being handed in again.
- Ask students to supply photocopies of any references used as part of an appendix. This ensures that all their references are genuine.
- Check the Web before the specification is finalised so that easily available sources are known. Tutors should also be familiar with books and journal articles on the subject.
- Set the assignment specification on a very unique or recent event on which there is unlikely to be much material available.
- Alternatives to the standard essay, such as case studies, present more difficulties in locating suitable material to plagiarise.
- Group assignments with individual summaries can make it harder for the whole group to agree to plagiarise.
- Assessed work produced in class, possibly with preparation allowed beforehand, reduces the opportunities to plagiarise.

It has also been suggested that students could be taken through some of the submissions in the online essay banks, with methods of correct citation and avoiding plagiarism stressed. Many students fail to recognise that citing, when appropriate, strengthens their submissions. This might also have an added bonus of making students less likely to use essay banks if they know that their tutors are aware of them.

Additional deterrent methods that have the advantage of helping students to improve but are more demanding of tutor time include:

- Ask for an early draft of the submission or a plan, so that problems can be caught early and improvements can be suggested.
- Collect in an annotated bibliography before the submission is due. This can be hard to construct from a supplied paper and ensures that the students have done some work before the submission date.
- Viva all the students in order to check what they have learned and that they are familiar with the ideas in the submission.
- Get students to give a presentation either individually, or in groups, on their subject area. It is hard to talk about and field questions on an area that you are not familiar with.

When marking submissions tutors could look for pointers that might suggest that some of the work is not original:

- Unusual references. Many outdated references, or references that are not available locally suggest that the paper may not be freshly written.
- Dramatic change in levels of writing ability. Those well-written sections may not be written by the student. Changes in tense or voice can also be suspicious.
- Analogies with non-local or non-current events, such as talking about the local weather in Texas, or President Clinton.
- American spellings, it should be no surprise that most available on-line material is Americanised.

This list of prevention techniques is not intended to be exhaustive or prescriptive, merely an overview, from a UK perspective, as prevention is covered in more detail elsewhere. More information can be found in Austin and Brown's paper [1]. Web sites containing additional suggestions include:

- <http://alexia.lis.uiuc.edu/~janicke/plagiary.htm>
- <http://www.virtualsalt.com/antiplag.htm>
- <http://www.wiu.edu/users/mfbhl/wiu/plagiarism.htm>
- <http://www.plagiarized.com/>

The Four-Stage Plagiarism Detection Process

Culwin and Lancaster defined a Four-Stage Plagiarism Detection Process, as shown in Figure 2, which includes the potential for incorporating automated plagiarism detection [4]. They also provided an in-depth discussion of the issues that an institution considering using such an automated process need to be aware of [7].

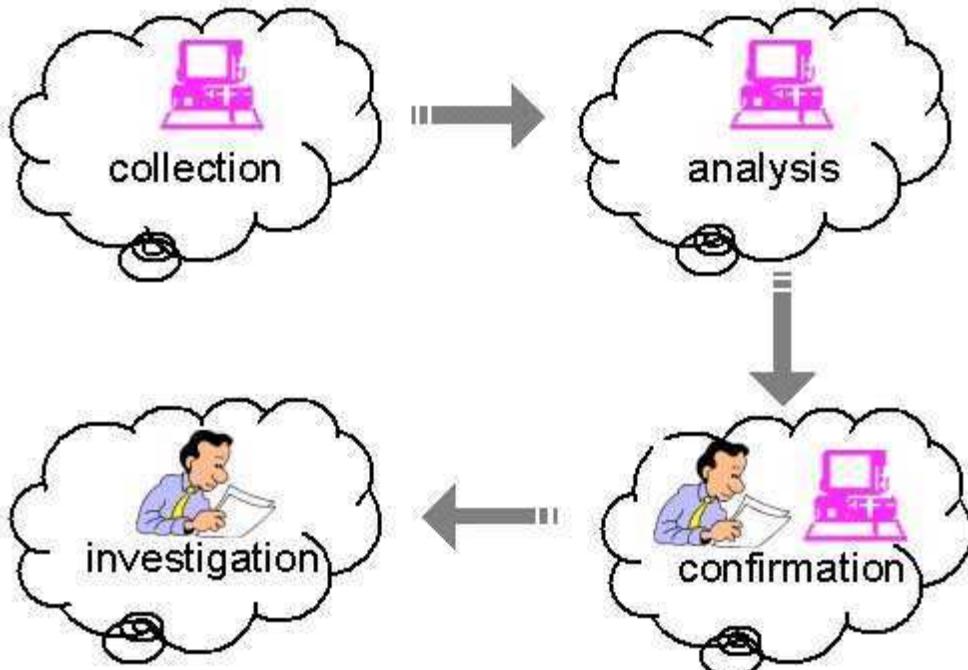


Figure 2: Four-Stage Plagiarism Detection Process

The process can be used to find **intra-corporeal plagiarism**, close similarity within a **corpus** or set of student submissions, such as a set of essays on a similar theme. It can also be used to find **extra-corporeal plagiarism** from outside the corpus, such as copying from a book, or the Web. Such definitions were standardised by Culwin and Lancaster in their plagiarism taxonomy [4]. The wording used throughout this paper is consistent with that taxonomy.

The four-stage process comprises **collection** where a corpus of student submissions is obtained, preferably in a machine-readable format. This requirement is essential for fully automated plagiarism as scanning printed submissions or typing hand written ones is not cost effective. Following this is the **analysis** phase, where the submissions are compared with each other and with documents obtained from the Web and the list of those submissions, or pairs, that require further investigation is produced. This list should be annotated in some way to indicate the extent or nature of the detected similarity. No automated system can reliably report that detected similarity represents plagiarism, and so a human has to provide **verification**, checking that the similarity is not acceptable. An example of acceptable similarity might be shared common citations. Those submissions that are now considered plagiarised would be subject to **investigation** where the students have to be shown the evidence of plagiarism and given the opportunity to explain it. This stage also involves the process of deciding culpability and possible penalties.

Collection tools

Most institutions using plagiarism detection systems have their own locally developed collection systems, for instance the School of Computing at South Bank University uses a simple series of interactive Web pages to accept a submission and return a receipt number which is also e-mailed to the student. This has the added advantage of reducing the workload on the school clerical staff who would otherwise have to accept all the submissions by hand.

All submission systems should comply with the Data Protection Act and give students an explicit indication of this. Students should also be warned that their submissions will be subject to automated plagiarism detection at the time their work is collected.

The only known collection tool is the Nottingham University developed Coursemaster, developed from a pre-Web system called Ceilidh. It is an integrated system for managing a course Web site providing facilities for the distribution of lecture notes, monitoring registers and class progress as well as allowing students to submit their work and providing some built in plagiarism detection. New UK academic users will pay prices starting at £1000. More details are available at <http://www.cs.nott.ac.uk/CourseMaster>.

Detection Tools

Culwin and Lancaster reviewed Web-based plagiarism detection services that were available in the spring of 2000 [6]. They found that the services were usually able to detect deliberately plagiarised documents submitted to them, but they were too costly for regular institutional use. In addition the precise mechanisms by which students submitted their work were cumbersome and in most cases the format of the results returned to the tutors were not helpful for further investigation of apparent undue similarity.

A selection of the services currently available for plagiarism detection are listed below. These include both the original commercial and the newer free services. The latter tend to be limited versions of the commercial services and depend on a tutor being suspicious enough of a piece of work to obtain an electronic version of it for analysis. Plagiarism.org <http://www.plagiarism.org/>

Plagiarism.org claims to be the market leading detection service. Students submit their work by pasting it into a text box and the results are e-mailed to the tutor. A separate local collection may still be needed. If the tutor chooses to investigate further the submission is presented back to the tutor as a series of hyperlinks to Web pages or other student submissions that are believed to be similar. The main problem with the service is the cost of \$1 per submission.

Paperbin <http://www.paperbin.com/>

(previously known as Integriguard <http://www.integriguard.com/>)

Paperbin is a similar service charging \$4.95 per tutor per month for any number of submissions. Integriguard always seems like a poor cousin to plagiarism.org mainly because it only checks a portion of each submission and could quite possibly miss plagiarised material. Results are provided only by e-mail in a format with limited readability.

Copycatch.com <http://www.copycatch.com>

This is a service very much along the lines of plagiarism.org, but with lower subscription costs and the additional benefit of accepting files in popular word processing formats, such as MS Word. As at the time of writing this document the system doesn't seem to be available.

CopyCatch <http://www.copycatch.freeserve.co.uk/vocalyse.htm>

Not to be confused with the detection system of the same name, CopyCatch is a UK developed detection tool (single license user cost £250). It is an intra-corporeal plagiarism detection system, designed to compare all submissions against all others and report those above a given threshold of similarity.

Text Ranker <http://cise.sbu.ac.uk/>

This is a similar system under development at South Bank University. It returns a list of pairs of student submissions ordered from most to least similarity under metrics chosen by the user to be most important. Material from the Web or other sources can be added to the corpus to approximate extra-corporeal detection.

Essay Verification Engine (EVE) <http://www.canexus.com>

The Essay Verification Engine is similar to Paperbin but is operated locally. Segments from each student submission in text, MS Word or Word Perfect format within a collected corpus are run through Web search engines. A report for each submission is supplied in text format. The system does not check for intra-corporeal plagiarism. The system is of limited use in investigating which Web sites are likely to have been plagiarised from and does not identify where the similarity occurs. This can be particularly irritating if a submission is multiply sourced or where it only contains a very short segment from a given Web page. Plagiarism might also be missed during the approximated checks.

FindSame <http://www.findsame.com>

This is a useful free service for a tutor that suspects that a submission was plagiarised but does not recognise the source. FindSame provides a text box or file upload capabilities for tutors and returns an immediate Web page of hyperlinked results, in a similar manner to plagiarism.org. It could be used as a cheap and more complete plagiarism detection system in conjunction with EVE for analysis and FindSame for verification and investigation.

HowOriginal.com <http://www.howoriginal.com>

This is the free version of Paperbin and is very much styled along the same lines as FindSame with a limit of 1000 characters. This means that only one or two paragraphs can be checked at a time. Submission is by text copy and paste. Tutors supply the text and an e-mail address and are mailed back sometime later with details about how to collect the results. This is just the standard and limited Paperbin text report with lots of additional advertising. The

whole procedure is very unsatisfying; tutors with such a short section of text to check might be better advised to paste the section into a search engine and obtain an immediate response.

Plagiserve <http://www.plagiserve.com/>

Plagiserve is a new free service, requiring instructors to register before using it. Tutors then paste a text file into a text box and are later e-mailed back with a Web address where they can receive the results. The presentation format is the standard set of hyperlinks, rather than these being connected to external pages they are connecting to a lower frame which contains alternate sections of source and copy text, allowing the contents of multiple Web pages to be shown on a single page. Operationally the service is very similar to FindSame with an additional integral database of likely source materials. The main difference is the delay in getting the results.

Verification tools

Tools to aid in the verification and investigation stages of the Four-Stage Plagiarism Detection Process are just being developed. It is in the verification part of the process that automated assistance is likely to be most beneficial and so more tools might be expected in the near future. It is unlikely that specific tools could be developed for the investigation stage as existing workflow management tools would seem to be sufficient.

Visualisation and Analysis of Similarity Tool (VAST) <http://cise.sbu.ac.uk/>

The Visualisation and Analysis of Similarity Tool (VAST) presents a graphical representation of the similarity between two student submissions plotted against one another [5]. The similarity intersections, the intense areas of the graphic, can be selected and the appropriate sections of both submissions quickly displayed in windows on the screen. The tool has been shown to be an improvement over the regular approach of trying to find similarity in two student submissions by eye alone [11].

WordCheck <http://www.wordchecksystems.com/>

Although WordCheck is commonly included in listings of plagiarism detection tools it is considered more suitable for linguistic analysis of how similar two texts are. It could perhaps be used for investigation if both a potential source and copy were known in order to investigate if one text was a derivation of the other. However, there are few advantages to this over a non-computer aided approach.

Glatt Plagiarism Services <http://www.plagiarism.com/>

This site is devoted to the Glatt tests, where every fifth word of a student submission is replaced by identically sized spaces. Students are expected to be able to accurately replace the omitted words if they originally wrote the paper and to have a lower performance if they copied it. A limited free Web based version and a commercial version of the software is available, but it is unclear how much trust could be placed in the results. Glatt also offers commercial courses in plagiarism for both tutors and students.

Current Initiatives and Active Research Groups

The recent media coverage of plagiarism has sparked interest throughout the UK academic community. Most notable is a Joint Information Systems Committee (JISC) project involving an evaluation of Web based plagiarism detection services. Five institutions, representing the diversity of UK higher education, are involved in a trial using plagiarism.org. The trial is intended to reveal the operational issues involved in the large-scale deployment of a plagiarism detection system as well as evaluating its effectiveness in detecting and discouraging plagiarism. Other JISC studies are looking at plagiarism detection software from a social and technical viewpoint and the more specialised area of computer program source code plagiarism detection. The project is intended to be finished in June 2001 and the results will be presented at a workshop for the higher education community. More details are available at <http://www.jisc.ac.uk/jciel/plagiarism>.

The Centre for Interactive Systems Engineering (CISE) at South Bank University is researching efficient and effective methods of intra-corporeal plagiarism detection, along with ways of visualising the similarity between submissions in order to assist human evaluation. Additional work is aimed at identifying the extent of and motivations for student cheating and plagiarising. The CISE Web site is at <http://cise.sbu.ac.uk/>.

The University of Hertfordshire has a Plagiarism Detection Research Group, but their Web site only says they have ideas about plagiarism detection; there is no indication whether the research is ready for publication or not. The group have their home page at <http://homepages.feis.herts.ac.uk/~pdgroup>.

The Plague group at Monash University is developing tools for plagiarism detection in both source code and free text. Their site is at <http://www.csse.monash.edu.au/projects/plague>.

Rob Irving at Glasgow University has adapted a source code plagiarism detection system to work with free text. His Web site is available at <http://www.dcs.gla.ac.uk/~rwi>.

Researchers at Sheffield University are starting a new METER Project based around text similarity analysis, with plagiarism applications. Details of their research is at <http://www.dcs.shef.ac.uk/nlp/funded/meter.html>.

Conclusions

It is likely that student plagiarism will continue to be a problem, despite the best effort of tutors to minimise it and some of the tools and services presented here are becoming an integral part of academic life. Those institutions without a pro-active anti-plagiarism policy, one where plagiarism is both actively sought out and taken seriously when discovered, are likely to be under increasing scrutiny from both the media and academia itself.

There are now effective tools available to identify Web sources and find similar student submissions. The current drawback is at the human driven stages of verification and investigation and that is where additional effort could most usefully be employed. Current tool developments, such as VAST are needed to change that.

Some have speculated that a better understanding of why students cheat would better aid both prevention and detection of plagiarism. Tutors need to be aware that some students have always cheated. Some will, probably, always try to cheat. The challenge, for academics is to find effective ways to encourage students to work for and value the awards they gain whilst deterring those that might be tempted to cheat and detecting those who do.

References

- [1] Austin M. & Brown L., Internet Plagiarism: Developing Strategies to Curb Student Academic Dishonesty. *The Internet and Higher Education*, 2, 1, p21-33 (1999).
- [2] Clare J., Computer plagiarism 'threatens the value of degrees'. *Daily Telegraph* 3/7/2000, Available online at <http://www.telegraph.co.uk/et?ac=004228431730477&rtmo=VDw3wDqK&atmo=rrrrrrq&pg=/et/00/7/3/ncopy03.htm> (2000), [Accessed 3/4/2001].
- [3] Cramb A., University withholds 90 exam results over 'Internet cheating'. *Daily Telegraph* 10/7/1999, Available online at <http://www.telegraph.co.uk/et?ac=004228431730477&rtmo=VDw3wDMK&atmo=rrrrrrq&pg=/et/99/7/10/ncheat10.html> (1999), [Accessed 3/4/2001].
- [4] Culwin F. & Lancaster T., A Descriptive Taxonomy of Student Plagiarism. Awaiting publication, available from South Bank University, London (2000).
- [5] Culwin F. & Lancaster T., Visualising Intra-Corporeal Plagiarism. Accepted for IEEE Information Visualisation 2001, 25-27 July, London, UK, available from South Bank University, London (2000).
- [6] Culwin F. & Lancaster T., A Review of Electronic Services for Plagiarism Detection in Student Submissions. 8th Annual Conference on the Teaching of Computing, organised by the LTSN Centre for Information and Computer Sciences (2000).
- [7] Culwin F. & Lancaster T., Plagiarism Issues in Higher Education. To be published in *Vine*, 123 (2001).
- [8] Culwin F. & Lancaster T., Pro-Active Anti-Plagiarism in Action, an Initial Report. The ILT's first annual conference, Learning Matters, York, (2000).
- [9] Culwin F. & Naylor J., Pragmatic Anti-Plagiarism. Proceedings 3rd All Ireland Conference on the Teaching of Computing, Dublin (1995).
- [10] Franklin-Stokes A. & Newstead S., Undergraduate Cheating: Who Does What & Why? *Studies in Higher Education*, 20, 2, p159-172 (1995).
- [11] Lancaster T. & Culwin F., Towards an Error Free Plagiarism Detection Process. Accepted for ITiCSE 2001, 25-27 June 2001, Canterbury, UK, available from South Bank University, London (2001).
- [12] Lathrop A. & Foss K., Student Cheating and Plagiarism in the Internet Era – A Wake Up Call. Published by Libraries Unlimited Inc (2000).

Plagiarism Issues for Higher Education

Fintan Culwin & Thomas Lancaster,
School of Computing, Information Systems & Mathematics,
South Bank University,
London, SE1 0AA,
United Kingdom.

fintan@sbu.ac.uk, lancaste@sbu.ac.uk
<http://cise.sbu.ac.uk>

Abstract

Academic institutions are finding they have to operate under a pro-active anti-plagiarism policy, where plagiarism is actively sought out as a serious breach of acceptable academic behaviour. This paper considers the reasons that institutions need such a policy and the issues they should be aware of when implementing one.

Introduction

The ever-growing amount of information on the Internet and the ease with which students have access to this information has produced an increasing temptation for students to download Web material, possibly modify it and hand it in as their own unaided work. Taking the words or ideas of another person and using them without proper acknowledgement is a commonly accepted definition of plagiarism. *Student plagiarism* is further defined as plagiarism with the intent of gaining academic credit. Academic institutions should be concerned about this since student plagiarism appears to be on the increase [1, 10, 14].

This paper considers why academic institutions are having to employ already scarce resources to combat plagiarism. It also looks at the issues that institutions should be aware of when trying to operate a pro-active anti-plagiarism policy.

The Problem of Student Plagiarism

Students intending to gain academic qualifications are expected to demonstrate appropriate levels of attainment and ability through coursework and examinations. This requires students to produce submissions that meet a given *assignment specification* which is then marked by a tutor to confirm that the work reaches the required standard. In many, if not the majority, of institutions students are also required to confirm that the submission is the result of their own, unaided work. Students who falsely give this declaration are playing a part in reducing the value of the qualifications awarded by the academic institution. Knowing that other students are cheating, but are not being punished for it, can be infuriating to other students, who may themselves be discouraged from putting appropriate effort into their own submissions.

The Franklyn-Stokes and Newstead studies in 1995, taken as the British higher education baseline, showed that over half of all students are willing to confess to having plagiarised at some stage of their academic course [9]. This included not only *intra-corporeal plagiarism*, copying from other students on the same course, but also *extra-corporeal plagiarism*, such as close paraphrasing from a book or submitting an assignment produced in response to an identical specification in a previous year. Such definitions were standardised by Culwin and Lancaster in their plagiarism taxonomy [5]. Newstead, Franklyn-Stokes and Armstead found that students cheated mainly due to time pressure, or because they didn't believe that they could get marks as high as they wanted without resorting to cheating [13]. They also found that a significant proportion of students cheated because they knew of other students cheating and hence believed it was the norm.

The current proportion of students who are plagiarising is expected to be higher than in the Franklin-Stokes and Newstead study. Their figures were produced before the growth of the Web which provides students with easy access to vast amounts of copyable information. This growth of *Web plagiarism* has been discussed at length in the literature [1, 10, 14]. Some students assemble a submission by amalgamating extracts taken from a number of different sites. Others copy assignments directly from sites supplying free pre-written essays [17, 22]. Any Americanisms in these essays can be swiftly corrected with the aid of a UK spell checker and cultural references altered with a minimum of effort.

Critics have attempted to discuss why Web plagiarism is a particular problem and how it can be countered. Ryan gives examples of pointers from her own experience and looks at how a tutor can use Web search engines to attempt to find out where the plagiarism has been sourced from [14]. Gajadhar discusses the belief of some students that material on the Internet is ‘free’ and can be reused without citations and concludes that many institutions are ignoring the problem because they don’t know how to cope with it [10]. Austin and Brown discuss how assignment specifications can be set that minimise the ways in which students can cheat and how students can be made aware that plagiarism is unacceptable [1]. A search for plagiarism on the Internet will also throw up a large number of Web sites that offer advice and give their solutions to the problems [11, 19, 20]. All comment about the magnitude of the problem and how it is an issue worth serious attention.

Collapsing a Cheating Culture

Much of the rigorously researched evidence for the extent of plagiarism is pre-Web and hence obsolete. But the problem is receiving an increasing amount of attention from different quarters. The press seem to find a story on plagiarism every week [3, 4, 16]. Academics discussing the problem say they routinely come across plagiarism cases. Academic organisations are finding it necessary to be seen to be taking plagiarism seriously. The Joint Information Systems Committee (JISC) has held workshops [2] and started to fund anti-plagiarism research. The Committee of Vice Chancellors and

Principals (CPVP) is also investigating the extent and impact of plagiarism as well as technology to detect it. The sudden rise in the number of journal and conference submissions also suggest that plagiarism is a hot issue and most of these sources suggest that it is both prevalent and endemic [12, 15].

Institutions seem to be struggling to update their policies to be seen to be taking plagiarism seriously in light of this interest. Policies vary from *reactive* policies, where plagiarism is treated as a serious academic offence when it is found, but not actively sought out, to *pro-active* policies, where an effort is made by institutions to find and reduce plagiarism by technical and other means. Many institutions are finding it necessary to replace a reactive policy with a proactive one to break the seemingly growing culture that thinks cheating in our academic institutions is acceptable.

A pro-active policy involves standard plagiarism prevention measures. These range from not issuing the same assignment specification year after year, researching the subject area to check that model essays are not prevalent on the Internet and requiring drafts of submissions to minimise the opportunities for students to cheat [11].

Institutions are having to employ increasingly technical solutions in order to implement their pro-active anti-plagiarism policies, including using Web-based plagiarism detection services. Culwin and Lancaster reviewed a number of these, finding each to provide much the same service, with sections of a student submission hyperlinked to Web pages containing similar text [7]. A smaller study by Denhart reached similar conclusions [8]. The main problem with such solutions was seen to be that they were economically unfeasible. The market leader plagiarism.org charges \$1 for each paper that is submitted to it. The site was also found to be suitable for accepting submissions in plain text only and not the word processing formats most students use. It did not make the degree of similarity between two submissions clear to the user. However plagiarism.org [21] is being trialled by a number British institutions in a current JISC project.

Institutions with only a reactive anti-plagiarism policy have access to a similar service. Findsame.com [18] offers limited free detection for single submissions, via copy and paste into a text box, which can be useful to find Web plagiarism. This very much places the burden of work on the tutor responsible for marking the student submission. For a complete check they would have to put all the submissions of their students through the detection engine on an individual basis and check all the results. However, this service is valuable to tutors and much more useful than individually submitting lines of a submission into a search engine in an attempt find matching text.

Four-Stage Plagiarism Detection Process

The Four-Stage Plagiarism Model, as shown in Figure One, is designed to illustrate how an automated pro-active approach to plagiarism detection can be implemented.

In the *collection* stage, students submit their work to the system, usually via a Web front end. Next, in the *detection* stage, the collected work is run through a detection engine, which produces a list of those student submissions, or pairs of submissions, which appear to be the most similar. The human led stage of *confirmation* follows, where a human manually verifies if the similarity reported represents plagiarism. The system may be erroneously reporting similarity, this is known as a *false hit*. The similarity may also have come about by two people legitimately quoting from the same cited sources, which would not constitute plagiarism. Any similarity still thought to be plagiarism is passed on to the further *investigation* stage which might result in a penalty.

The main issues in the collection stage all stem from the fact that it is *logistically complex*. It is necessary to know who is going to submit work and what the work is that they are submitting. Any submission system has to take into account that the work will have deadlines which can differ from student to student should the circumstances arise. Work may need to be collected in a variety formats, e.g. Rich Text or MS Word formats for a report type question, or a standard text file for programming source code. Whatever

format the submissions are collected in, they need to be converted into plain text for machine analysis. Tutors will need to be informed as to which students work has and has not been collected so they can follow this up if necessary.

Where a collection system completely replaces a paper based submission there is also the necessity to distribute the collected work amongst the markers, either in printed or electronic form. The machine collection also has the additional advantage of decreasing the workload of the department's clerical staff. The submissions need to be stored securely, complying with Data Protection Act requirements, which also mean that students must be clearly be informed why their work is being collected on-line and how the information from processing them will be utilised.

Detection is the *computationally intensive* stage, since a large number of student submissions have to be compared against one another to detect intra-corporeal plagiarism, and against external sources to find extra-corporeal plagiarism. The main problem is to find what possibly disguised parts of submissions are stored where. In addition to locating true hits, detection systems also have to be non-capricious by attempting to ensure that they avoid *missed pairs*. A missed pair is a pair of submissions with significant similarity, but which the detection engine does not flag as such. These considerations have to be balanced against the storage and computational requirements of the detection engine. Additionally, submissions could be processed individually as they come in or batch processed, either when a whole set or *corpus* of submissions is ready or at regular time intervals.

Any detected similarity must be confirmed by tutors in what could be said to be a *judgementally complex* stage of the detection process. To be competent tutors have to ensure that they are consistent, non-capricious and coherent. This suggests that anyone who is involved in the plagiarism detection process would require training and would require a good knowledge about what undue similarity actually is.

The final stage, investigation, is *jurisprudentially complex*. If a submission has been found to contain evidence suggesting plagiarism a decision has to be made whether to exonerate or penalise the student who submitted it. The students have a right to present circumstances that mitigate the cheating and possibly to appeal against any penalty. This process, in the UK, must ensure that students are presented with the evidence that suggests an academic offence has been committed and given the opportunity to defend themselves. Moreover the people involved in collecting the evidence must be distinct from those making the judgement. If this due process is not followed it is possible that a leave to appeal to the civil courts could be obtained and subsequently damages awarded against the institution.

Any department considering the introduction of a pro-active policy must take care to compare the costs incurred, primarily valuable staff time, with the benefits obtained, mainly ensuring that only students doing their own work receive academic credit.

The issues presented in this article have been informed by the authors' own Visualisation and Analysis of Similarity Tool (VAST), designed to vastly decrease the amount of work tutors need to do to confirm and investigate possible plagiarism in the human led stages of the Four-Stage Plagiarism Detection Process [6]. Figure Two shows VAST being used to identify plagiarism in student submissions. The pronounced areas in the graphic on the left represent excessive undue similarity; the similarities in the highlighted area are shown in the windows on the right.

The current JISC project, primarily investigating plagiarism.org, but also producing a *Which style* report on other detection engines is investigating some collection issues. However, it is not concerned about the confirmation and investigation stages. This is despite these being, possibly, the most crucial stages in the whole procedure and certainly the most time consuming for a human investigator.

The Quality Assurance Agency (QAA) Code of Practice states that institutions should ensure that assessment is conducted with rigour, fairness and due regard for security,

whilst preventing fraudulent activities, such as impersonation and students submitting work that is not their own. Institutions wanting to take the QAA code seriously need to have a pro-active anti-plagiarism policy.

Conclusions

Many stakeholders are convinced that plagiarism is a serious academic problem and institutions will have to demonstrate that they are tackling it. The use of automated systems is expected to become more and more common, but they need to be informed by a greater understanding of how and why students cheat.

There is certainly a lack of consistency across the whole Higher Education sector. Some institutions have a pro-active anti-plagiarism policy, some a reactive policy and a few still claim, but cannot prove, that none of their students cheats. The standards across the sector need to be equal, so that students cannot assume that they can cheat by moving to a more lenient institution. The QAA guidelines alone are not sufficient for this and further guidance on plagiarism policies, plus guidance on issues of legality, such as jurisprudence, the Data Protection Act and copyright, is urgently needed.

The best possible solution would simply see the culture of cheating eliminated; few people think this is possible, even if a combination of technical solutions placed alongside carefully designed courses and assignment specifications can reduce it to a minimum. Institutions that do act proactively run the risk of reducing their student numbers and income in the short term. Hopefully, publicity of the value of their awards long-term should make up for it.

References

- [1] Austin M. & Brown L., *Internet Plagiarism: Developing Strategies to Curb Student Academic Dishonesty*. The Internet and Higher Education, 2, 1, p21-33 (1999).

[2] Barrie J. *Presentation on iParadigms and plagiarism.org*. At JISC Plagiarism Workshop, South Bank University, London (July 7, 2000).

[3] Clare J., *Computer plagiarism 'threatens the value of degrees'*. Daily Telegraph 3/7/2000, Available online at
<http://www.telegraph.co.uk/et?ac=004228431730477&rtmo=VDw3wDqK&atmo=rrrrrrr q&pg=/et/00/7/3/ncopy03.html> (2000), [Accessed 26/1/2001].

[4] Cramb A., *University withholds 90 exam results over 'Internet cheating'*. Daily Telegraph 10/7/1999, Available online at
<http://www.telegraph.co.uk/et?ac=004228431730477&rtmo=VDw3wDMK&atmo=rrrrrrr q&pg=/et/99/7/10/ncheat10.html> (1999), [Accessed 26/1/2001].

[5] Culwin F. & Lancaster T., *A Descriptive Taxonomy of Student Plagiarism*. Awaiting publication, available from South Bank University, London (2000).

[6] Culwin F. & Lancaster T., *Visualising Intra-Corporeal Plagiarism*. Accepted for IEEE Information Visualisation 2001, 25-27 July, London, UK, available from South Bank University, London (2000).

- [7] Culwin F. & Lancaster T., *A Review of Electronic Services for Plagiarism Detecting in Student Submissions*. 8th Annual Conference on the Teaching of Computing, organised by the LTSN Centre for Information and Computer Sciences (2000).
- [8] Denhart A., *The Web's Plagiarism Police*. Available online at <http://www.salon.com/tech/feature/1999/06/14/plagiarism/index.html> (1999) [Accessed 26/1/2001].
- [9] Franklin-Stokes A. & Newstead S., *Undergraduate Cheating: Who Does What & Why?* Studies in Higher Education, 20, 2, p159-172 (1995).
- [10] Gajadhar J., *Issues in Plagiarism for the New Millennium: An Assessment Odyssey*. Available online at <http://ultibase.rmit.edu.au/Articles/dec98/gajad1.htm> (1998) [Accessed 26/1/2001].
- [11] Harris R., *Anti-Plagiarism Strategies for Research Papers*. Available online at <http://www.vanguard.edu/rharris/antiplag.htm> (2000) [Accessed 26/1/2001].
- [12] Lancaster T. & Culwin F., *Towards an Error Free Plagiarism Detection Process*. Accepted for ITiCSE 2001, 25-27 June 2001, Canterbury, UK, available from South Bank University, London (2001).

[13] Newstead S., Franklyn-Stokes A. & Armstead P., *Individual Differences in Student Cheating*. Journal of Educational Psychology, 88, 2, p229-241 (1996).

[14] Ryan J. J. C. H., *Student plagiarism in an online world*. Available online at http://www.asee.org/prism/december/html/student_plagiarism_in_an_onlin.htm (1998) [Accessed 26/1/2001].

[15] Sheard J., Dick M. & Markham S., *Is it Okay to Cheat? The Views of Postgraduate Students*. Accepted for ITiCSE 2001, 25-27 June 2001, Canterbury, UK (2001).

[16] Sutherland J., US students log on to the Internet for a cheat's charter. Guardian, 22/6/2000, Available online at <http://www.guardianunlimited.co.uk/Archive/Article/0,4273,4032413,00.html> (2000), [Accessed 26/1/2001].

[17] *Cyber Essays*. Available online at <http://www.cyberessays.com> [Accessed 26/1/2001].

[18] *Digital Integrity Search*. Available online at <http://www.findsame.com> [Accessed 26/1/2001].

[19] *Plagiarism and Anti-Plagiarism*. Available online at <http://newark.rutgers.edu/~ehrlich/plagiarism598.html> [Accessed 26/1/2001].

[20] *Plagiarism: What It is and How to Recognize and Avoid It*. Available online at <http://www.indiana.edu/~wts/wts/plagiarism.html> [Accessed 26/1/2001].

[21] *Plagiarism.org*. Available online at <http://www.plagiarism.org> [Accessed 26/1/2001].

[22] *School Sucks*. Available online at <http://www.schoolsucks.com> [Accessed 26/1/2001].

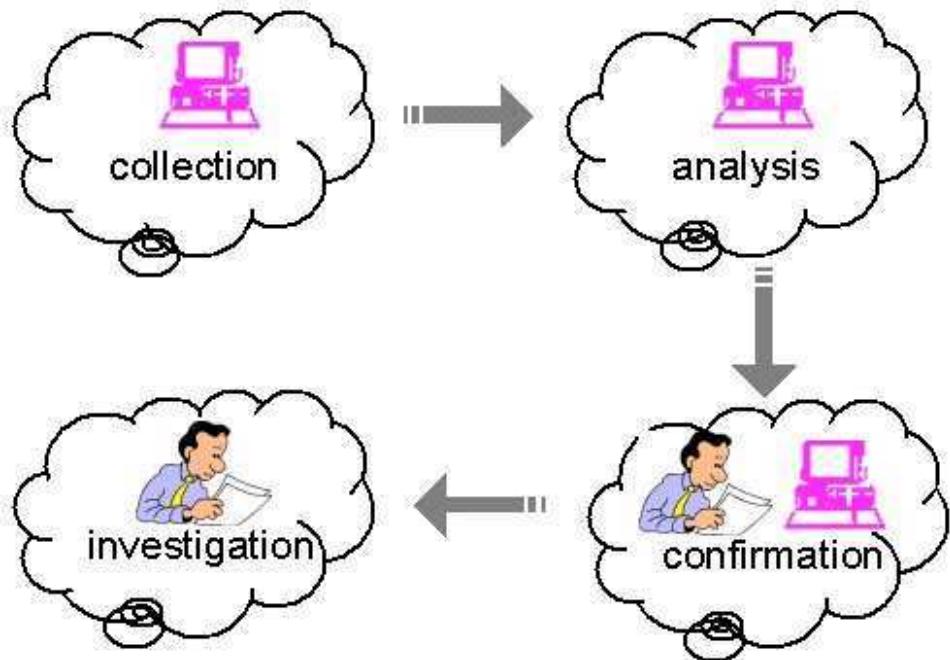


Figure One: Four-Stage Plagiarism Detection Process

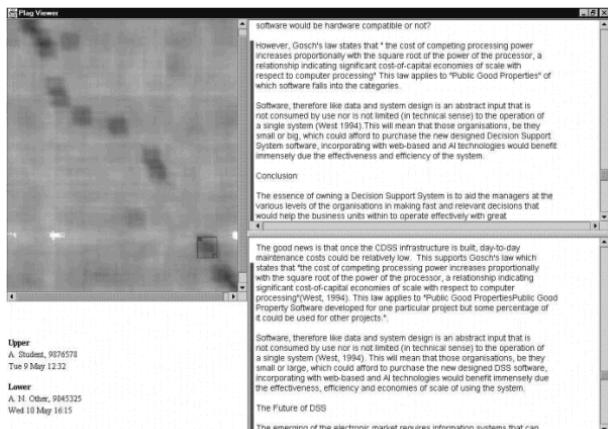


Figure Two: VAST used to investigate student submissions

A REVIEW OF ELECTRONIC SERVICES FOR PLAGIARISM DETECTION IN STUDENT SUBMISSIONS

Fintan Culwin

Centre for Interactive Systems Engineering
School of Computing
South Bank University
London SE1 0AA
fintan@sbu.ac.uk

<http://www.scism.sbu.ac.uk/~fintan>

Thomas Lancaster

Centre for Interactive Systems Engineering
School of Computing
South Bank University
London SE1 0AA
lancaste@sbu.ac.uk

ABSTRACT

Student plagiarism is an ever-increasing problem for academic institutions. A growing number of students are using material from the Web in their submissions, without properly acknowledging the source. This paper reviews the need for widespread plagiarism detection systems and evaluates available Web based detection services. Four services are discussed: the Measure of Software Similarity (MOSS) service for program source code and the plagiarism.org, Integriguard and copycatch.com services for free-text submissions. The downloadable Essay Verification Engine (EVE) tool for free-text detection of Web plagiarism is also evaluated. The paper finds that all five could be invaluable resources for academic institutions as they strive for a pro-active anti-plagiarism policy. The paper concludes by looking at the authors' current work to combat plagiarism.

Keywords

Plagiarism, cheating & copying, forensic linguistics

1. OVERVIEW

Plagiarism (the presentation of another person's ideas or material as if it were one's own) has always been an issue of concern to academic institutions. Recently, the number of plagiarism incidences reported by the media has increased and it is likely that other incidents are going undetected by institutions that do not like to think that they have a problem. Extensive plagiarism undermines the value of qualifications awarded by an institution and infuriates those students who gain qualifications by legitimate means. Hence effective anti-plagiarism policies and systems are essential to guarantee

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

8th Annual Conference on the Teaching of Computing,
Edinburgh

© 2000 LTSN Centre for Information and Computer Science

academic quality.

Plagiarism itself is not a new concern. In 1995, Culwin and Naylor outlined the problem and described a software system to check for similarities in program source code submissions [2]. More recently World Wide Web sites have appeared that claim to look for similarities in free-text submissions. It can be said that these are a necessary evil; the growth of the Web has provided students with access to large amounts of source material that they can *copy and paste* directly into an essay. Whole sites exist where complete assignments can be downloaded and handed in by students as their own work [4].

Such *Web plagiarism* is an increasing complication to tutors, already watching for duplicated material in student submissions. Many students realise that the likelihood of being detected is low, since work is often spread between many markers who are unlikely to mark a pair of directly plagiarised submissions. As academic institutions are run more like mass industries, tutors who would previously have known their students cannot be expected to learn the details of so many students' writing styles. The ease with which an essay or section of a Web page can be loaded into a word processor and reformatted also makes it harder for tutors to notice duplicate material unaided.

This paper looks at the increasing amount of academic resources available to aid tutors detect plagiarism. Support for finding similarities in both the (long established) constrained text and the (more recent) free-text areas of writing are examined. The paper ends with a look at the future of anti plagiarism measures.

2. CONSTRAINED TEXT

Until relatively recently, the processing power of computers has made it difficult to search for plagiarism within anything other than constrained text. As a result there are plenty of systems to find similarities in the major type of constrained text, program source code. Verco and Wise reviewed many of these systems [5] and Wise developed his own YAP3 system for just this purpose [6]. Culwin

and Naylor also developed a TEAMHANDIN system [2].

This paper is not a technical review of how such systems work, but they will usually tokenise a source code submissions in such a way, that, for example, renaming a variable throughout, or splitting a print statement over two lines, can be detected.

A more recent addition to the field is the Measure of Software Similarity (MOSS) service, operated by the University of Berkeley [10]. This allows tutors to collect a corpus of source code submissions locally and then submit it to the MOSS server to be checked. The source code would usually be collected through a local Web interface, to minimise clerical support time, but could be collected on disk, or by other methods.

The MOSS service is currently free to instructors on programming courses (although this may only be for a limited time) and setting up a new account is straightforward. The system works with C, C++, Java and ML source code, all of which are commonly taught languages in undergraduate programming courses. The instructor can supply source code expected to be common to each submission (e.g. supplied starter source code) and this can be discounted. The sensitivity of the analysis can also be controlled.

After processing, the server sends an e-mail giving the location of a Web page where the results can be found. Each pair of submissions within the corpus is ranked and the tutor can compare submissions side by side, with those parts of the submissions that MOSS believes to be similar colour coded. The tutor can then decide whether to take the case further.

The MOSS service has been in use at South Bank University, and other UK institutions, for some time with successful results.

3. FREE-TEXT

Looking for similarities between free-text submissions (most notably essays) has only become feasible in the last couple of years, with dramatic drops in the price of processing power and memory. There are now three Web sites and a downloadable tool designed to look for plagiarism in student submissions.

Denhart [3] compared plagiarism.org [11] with the downloadable Essay Verification Engine (EVE) [8] by submitting an essay made up of sentences from four well-known authors, with some amount of paraphrasing and re-ordering. He found that plagiarism.org decided that the essay was original, but EVE found most of the plagiarised material on the Web.

This section repeats the process with a carefully constructed sample text, designed to test the systems. The sample starts with two sentences each from Charles Babbage [13] and James Gosling

[14], both famous works widely available on the Web. In addition it contains a paraphrased sentence from an essay on the Year 2000 (Y2K) problem, obtained from one of the free essay sites [12] and a short sentence from a newly indexed personal Web site.

The services compared are plagiarism.org and EVE, as above, plus Integriguard [9] and copycatch.com [7], both looking for the same market as plagiarism.org. This section looks primarily at how effective each is at identifying the plagiarised material, but also considers how usable each is for tutors and students alike.

3.1 Integriguard

Integriguard [9] provides a ninety-day free trial of their service, after which instructors are charged 59.40USD per year. The trial account is straightforward to set up, giving each supervisor a unique number, although the password is displayed on-screen in unencrypted form, which is not ideal. After that the instructor simply gives their identification details to the students, who must register on an associated student site. Students then submit their work by copying and pasting it into a text box. A few minutes later the results of the analysis of that submission are e-mailed to the tutor.

The service claims to test each submission by comparing it against its database of writings. The returned e-mail showed that all six sentences of the sample text had been considered, and that none of them were in the Integriguard database. However five of the six sentences were accompanied by addresses of Web locations, where duplicates of the material can be found. Located sentences are given individually and with their surrounding context so that tutors can decide if they are properly referenced.

The sites found were of interest. For Babbage the search engine found the original site, a second site that quotes a small section of Babbage in agreement and, inexplicably, an adult entertainment site. Sentence two, also from Babbage, found the same three sites and an additional one. Material from the much-reproduced Gosling paper was found on eleven sites and nine sites respectively. Sentence five, a sentence from the Y2K paper that was actually in context a legitimate quote from a Y2K expert, found the original source for the quote, as well as ten essay sites which stored the paper. The final sentence, from the recently published personal Web site, was found to be original.

The whole process seems generally unsatisfactory, the presentation on the site and in each e-mail is poorly laid out and there is no guarantee that the work students are submitting to Integriguard is the same work they independently submit to the departmental office. Web sites are listed across multiple lines, meaning that a user has to cut and

paste the pieces into a browser to check out the source material. No effort is made to distinguish legitimately cited material and so a tutor would need to be very wary before starting proceedings against a student based on the evidence returned by Integriguard. The sample checked is also likely to miss small amounts of plagiarised material. This method would also generate a lot of e-mails for a tutor with a large class.

3.2 Plagiarism.org

Plagiarism.org [11] allow users to submit up to five papers as part of a free trial, initiated by supplying an e-mail address and filling in a questionnaire. Unlike Integriguard, plagiarism.org requires an instructor to give their address, telephone number and academic affiliation during the registration process. It also freely displays the password being chosen. The (text format only) file can be submitted by way of a text box or file upload. Students register and use a separate site as part of the paid-for service, which could not be tested. Tutors produce a separate submission area for each assignment they set.

Site design is generally effective and seems more up-to-date than Integriguard (dated 1998). It is also more commercially motivated, providing information as to what plagiarism is and links to press sites praising the plagiarism.org service. The system claims to work by comparing each submitted file to its database, using its own registered algorithm, as well as searching the Web. If a match is found details of a tutor with a matching assignment are said to be returned, the paper itself is not released.

Results from the service came after about 24 hours, via an e-mail with a pointer to a Web page. The paper was flagged as having some plagiarised material, presenting the entire submission on screen, with sections of it hyperlinked to source material on the Web. The system found the entire section of Babbage, linking back to our original source and the entire section of Gosling, linking to a different site. It also found the paraphrased section of the Y2K report, highlighting only those parts that remained unaltered and linking to one of the essay sites. Plagiarism.org flagged the material from the personal Web site as original.

The presentation of both the site and the plagiarised results is very good, although there is a notable delay in waiting for the results. The plagiarism.org service seems to work well, but would be unusable for most academic institutions, the price being 1USD per uploaded paper.

3.3 Copycatch.com

Copycatch.com [7] is the newest of the Web based plagiarism detection services. It is currently supported by advertising on the student site and hence provides a free service. It checks that only

tutors are using the service, contacting the departmental office before allowing an instructor to register, a process that can take up to two working days. Much like the other services, the system checks papers against its internal database and the wider Web. Unlike the other services, submissions can be made in many common word processing formats as well as plain text, although this doesn't take away the need to hand in a local, printed copy. However the submission system refused to accept the test essay in plain text format. A version encoded in Microsoft Word format was accepted successfully.

Once registered as an instructor, a tutor can add details of a new assignment, including a cut off date, to the system. Tutors then pass on identification details to their students, who have to submit their papers to the system before the cut off. There is no leeway to this date, late papers will simply not be checked for plagiarised material. When the deadline is reached copycatch.com processes the corpus and e-mails the tutor with a list of the number of lines in each essay and the number of lines matched. More detailed results are made available on the copycatch.com website for 48 hours.

The results are presented much like those of plagiarism.org, with the textual representation of the paper presented in a frame, with sections highlighted as html links to indicate a web source. The tests found parts of all four sections, as well as unrelated parts of the linking text, where the similarity was purely coincidental. Most of the Babbage paper was found at the original source, although for some reason one word was not highlighted. The section from the Gosling white paper was found, but as part of a set of lecture notes at a US institution. Sections of the Y2K paper were found on two different sites (even though the entire text was present on both sites). One of the located sites was where the original author had posted the information (which then found its way onto various essay sites). The system also found part of the material from the personal Web site.

Copycatch.com seems to be the most useable of the three services, although there is little to choose between any of them in terms of results. At present, being free, copycatch.com wins hands down on value for money.

3.4 Essay Verification Engine

The Essay Verification Engine (EVE) [8] unlike the other services listed is not Web based. Instead it is a downloadable tool (cost 19.99USD with a limited free trial) which checks the contents of an essay against some of the Internet search engines. It is designed mainly to be used when Web plagiarism is suspected, to aid a tutor in discovering the source.

By means of a simple wizard interface, users select files that they believe may contain plagiarised

materials. These can be in Microsoft Word, RTF or plain text formats. They can also select the strength of the test as high, medium or low. A medium strength test took about ten minutes on a fast machine, requiring continual access to the Internet.

The results are presented in a pop-up window on screen, with sections of the essay highlighted, along with matching Web links. These can be saved as an RTF file, or printed out for future reference, although in these latter cases it is not always easy to see which sections of text are linked with which Web site.

The results from EVE are the least convincing of all the services, finding plagiarised material in only the Babbage document and the Gosling Java white paper. For each of these, only some of the document is marked as plagiarised, probably revealing something about EVE's string search strategy. However EVE did find multiple references, four sites for Babbage, including the original and six sites for Gosling, four of which are mirrors of the white paper and two of which quote sections from it.

EVE is probably useful for the isolated cases where plagiarism is suspected but the source cannot be traced, but, in general, one of the Web based services would seem to be more effective.

4. COMPARATIVE EVALUATION

Whilst the results returned by the plagiarism detection services are broadly comparable, they represent different ends of the spectrum for tutors in terms of ease-of-use and effectiveness. There are also security and operational issues to be considered.

From an HCI perspective, the administrative interfaces to plagiarism.org, copycatch.com and Integriguard are very similar. Each uses a standard navigation bar to the left of the Web browser window, with related information displayed on the right. The copycatch.com interface is arguably the best of the three, allowing tutors to easily register themselves and establish assignment cut-off dates. The plagiarism.org interface suffers from being cluttered making it can be difficult to find the required section of the site. The Integriguard interface seems more geared towards advertising its service than aiding instructors to use it.

All three services work along the same principles. A tutor registers a submission, in some cases with a cut off date. Students then register on a designated student section of the site and submit their paper. The results are then sent to the tutor. In the case of Integriguard and plagiarism.org, processing can be done at any time after the paper is received. For copycatch.com, all submissions are batch processed after the due date. The source code detection service, MOSS, uses a similar batch-processing process, only in this case submissions

are collected locally and bulk submitted to MOSS. This means that late submissions can be given the same amount of automated scrutiny as submissions received on time.

The submission process for students ranges from poor to satisfactory. Plagiarism.org and Integriguard provide only a simple text box for a student to cut and copy their text into. Copycatch.com allows students to select a file from their local hard drive in popular word processing formats and submit that instead. None of the methods are geared towards helping students unfamiliar with computer operation and tutors may find themselves having to provide support for the submission process.

Aside from this, the workload for tutors using the services is low. They merely need to set up the service to accept papers, ensure that all students expected to submit assignments do so and interpret the results at the end. The Web-based results from plagiarism.org and copycatch.com are much more readable than the e-mail returned by Integriguard. For security purposes, viewing the results from copycatch.com requires knowledge of the instructor's password, viewing those from plagiarism.org (or MOSS for that matter) does not.

From an overall effectiveness viewpoint, the results from copycatch.com, plagiarism.org and MOSS are all comprehensive and well presented. Integriguard suffers since only a fraction of the essay is ever checked for Web plagiarism. The same is true for the EVE tool, although it is possible to increase the number of tests at the expense of a much longer processing time. For free-text plagiarism detection, there is little to choose between plagiarism.org and copycatch.com; both seem effective at finding suspected plagiarism and communicating this to tutors, although copycatch.com is presently free, as is MOSS, for source code.

Since the services are all based in the US, there might be implications with regard to the Data Protection Act. These are not considered within the context of this paper.

5. CONCLUSIONS

The area of free-text anti-plagiarism is still in its infancy and will almost certainly continue to receive plenty of media attention as more incidents arise. Some universities do not yet appreciate the full scope of the problem, we encourage them to operate a pro-active anti-plagiarism policy and to take steps to actively seek out plagiarism in student submissions.

We, the authors, are continuing to research and develop software to detect free-text plagiarised submissions. The existing Web based systems are able to trace some simple cases of plagiarism, but we believe that more sophisticated linguistic metrics could be more effective. Our initial investigations are

proving successful and suggesting that more powerful detection systems are possible. The existing services are costly (integriguard.com, plagiarism.org) or may start to charge at any time (copycatch.com, MOSS). As a result we foresee the need for a plagiarism Web server based in the UK available for free academic use.

One problem that we have foreseen, is that it is currently difficult to discuss plagiarism, since the term is ill-defined and the language limited. We have prepared a descriptive taxonomy of plagiarism that aims to aid discussions in the area [1]. Further work will aim to encourage departments to implement a pro-active anti-plagiarism policy, where plagiarism is not only taken seriously when found, but is actively sought out during all coursework activities.

Finally, whilst there seems to be little to choose between the services on offer at present, we would encourage all academic institutions to show to re-evaluate their plagiarism policies and seriously consider using one of these services as a matter of course. In particular the MOSS service is an asset to the academic computing community. Whilst it might charge in the future, it is currently free for academic use and deserves to be more widely used.

6. REFERENCES

- [1] Culwin F. & Lancaster T., *A Descriptive Taxonomy of Plagiarism*. Awaiting publication, available from South Bank University, London (2000).
- [2] Culwin F. & Naylor J., *Pragmatic Anti-Plagiarism*. Proceedings Third Conference on the Teaching of Computing, DCU Dublin IE (1995).
- [3] Denhart A., *The Web's Plagiarism Police*. Available at <http://www.salon.com/tech/feature/1999/06/14/plagiarism/index.html> (1999).
- [4] Gajadhar J., *Issues in Plagiarism for the New Millennium: An Assessment Odyssey*. Available at http://www.asee.org/prism/december/html/studen plagiarism_in_an_onlin.htm (1998).
- [5] Verco K. & Wise M., *Software for Detecting Suspected Plagiarism: Comparing Structure & Attribute Counting Systems*. Proceedings First Australian Conference on Computer Science Education, Sydney, Australia (1996).
- [6] Wise M., *YAP3: Improving Detection of Similarities in Computer Program and Other Texts*. Proceedings ACM SIGCSE '96, Philadelphia, USA (1996).
- [7] *Copycatch.com*. Available at <http://www.copycatch.com>.
- [8] *Essay Verification Engine*. Available at <http://www.canexus.com/eve/>.
- [9] *Integriguard*. Available at <http://www.integriguard.com>.
- [10] *Measure of Software Similarity*. Available at <http://www.cs.berkeley.edu/~moss/general/moss.html>.
- [11] *Plagiarism.org*. Available at <http://www.plagiarism.org>.
- [12] *Year 2000 Computer Problem (Y2K Bug)*. Available at <http://www.cheater.com/homework/Homework/Schoolsucks/Uploads/Random2/10791413.htm>.
- [13] Babbage C., *Passages from the life of a Philosopher, Chapter VIII*. Available at <http://www.fourmilab.ch/babbage/lpae.html>.
- [14] Gosling J. & McGilton H., *The Java Language Environment – A White Paper*. Available at <http://www.javasoft.com/docs/white/langenv/index.html>.

JISC

Joint Information
Systems Committee



SOUTH BANK UNIVERSITY
· LONDON ·

SCHOOL OF COMPUTING, INFORMATION SYSTEMS & MATHEMATICS

Source Code Plagiarism in UK HE Computing Schools, Issues, Attitudes and Tools

by

Fintan Culwin, Anna MacLeod & Thomas Lancaster

Technical Report SBU-CISM-01-01
September 2001

Contents

Introduction		Page 3
1. Source Code Plagiarism Detection Systems		Page 4
2. The Questionnaire Survey		Page 5
2.1 Methodology		Page 5
2.2 The Results		Page 6
2.3 Free Response Comments		Page 8
3. Internet Hosted Similarity Detection Engines		Page 9
3. 1 Qualitative analysis		Page 10
3.2 Quantitative analysis		Page 13
4. Other Issues		Page 16
5. Conclusions and recommendations		Page 18
Appendix A	Source code plagiarism publications list	Page 19
Appendix B	The questionnaire	Page 22
Appendix C	Detailed survey results	Page 25

Source Code Plagiarism in UK HE Computing Schools, Issues, Attitudes and Tools

Fintan Culwin, Anna MacLeod & Thomas Lancaster
Centre for Interactive Systems Engineering (CISE)
School of Computing Information Systems & Mathematics (SCISM)
South Bank University (SBU)

{fintan macleod lancastre}@sbu.ac.uk
Web server: cise.sbu.ac.uk

Introduction

In December 2000 the Joint Information Systems Committee (JISC) commissioned CISE to produce a report on 'Plagiarism Detection Methods in Computer Programming'. This project was a constituent part of the JISC Committee for Integrated Environments for Learners (JCIEL) electronic plagiarism detection project (see www.jisc.ac.uk/jciel/plagiarism).

The purpose of the report was to inform JCIEL of the extent and nature of source code plagiarism and advise upon any resources that might need to be developed or deployed. It was also commissioned with the intention of making the technology more widely known within the UK HE academic computing community.

The report was completed in May 2001 and consists of four parts. It commences with a short literature survey intended to establish a context for the subsequent parts of the report. The second part summarises the results of a questionnaire based survey of UK computing departments, designed to establish perceptions of the extent of, and attitudes towards, source code plagiarism. The third part gives a quantitative and qualitative analysis of two widely known and freely available similarity detection engines, MOSS and JPlag. The final part comments on a number of miscellaneous issues and the report finishes with a number of conclusions and recommendations to JCIEL.

The report was presented to JCIEL at its September 2001 meeting and was published as SBU SCISM Technical Report 2001-01, copies of which were sent to all UK HE computing schools. An updated copy of the report (hopefully incorporating more replies to the questionnaire) will be maintained on the CISE web server whose address is given above.

1. Source Code Plagiarism Detection Systems

A list of references that have been consulted in the preparation of this report is given in Appendix A.

Computer systems for source code plagiarism detection have been in existence for over twenty years (Halstead 1977, Ottenstein 1977) and are now routinely used in many academic institutions. The first systems were based upon attribute counting algorithms, extracting various superficial metrics from source code submissions, for example a count of the use of a particular reserved word, and then flagging those pairs of submissions which were significantly close for tutors to inspect further. Later developments saw the introduction of structure metric systems where each submission is reduced to a series of identifiers, or tokens, representing, for example, a function call or a variable declaration. Pairs of tokenised submissions are then recursively searched for the longest common token sequences and the proportion of the submissions matched used as a similarity metric. Structure metric systems have been shown to be more effective than attribute counting systems (Verco & Wise 1996) and all existing systems use those principles.

The earliest attribute counting system, developed by Ottenstein (Ottenstein 1977), used Halstead's software science metrics (Halstead 1977), and primarily counted the uses of operators and operands within a piece of code. Such metrics were originally developed to allow automated marking of source code for style. Subsequent research has introduced new metrics and tried to find the most effective combinations (Robinson & Soffa 1980). Parker and Hamblen reviewed the metrics that seem to be successful for finding plagiarism in source code written in different programming languages (Parker & Hamblen 1989).

Structure metric systems have developed from Donaldson, Lancaster and Sposato's early hybrid system (Donaldson et al 1981). More recent examples include Plague (Whale 1998) and YAP3 (Wise 1996). Joy and Luck showed how students could plagiarise and how they could be detected with reference to their own SHERLOCK system (Joy & Luck 1999).

Academic institutions have access to both their own locally developed systems, including TEAMHANDIN (Culwin & Naylor 1995) and Irving's Big Brother system (Irving 2000). They also have access to services provided over the Internet. Best known is Aiken's Measure of Software Similarity (MOSS). No technical details of MOSS are available, only an independent critique (Bowyer & Hall 1999) describing one approach for using the system. The JPlag system, accessible on the Web and described from a technical perspective (Prechelt et al 2000 & 2001) offers similar functionality. (See section 3 of this report.)

Aside from JPlag little research continues into source code plagiarism detection, since it is a well-understood problem involving the analysis of constrained text.

Recent developments in structure metric systems employ sophisticated pattern matching algorithms, developed as part of the human genome project, to establish similarity between pairs of programs.

2. The Questionnaire Survey

2.1 Methodology

The questionnaire reproduced in Appendix B was prepared both in text form and as an interactive Web page. A list of contacts was prepared; starting with a list supplied by the Learning and Teaching Support Network centre for Information and Computer Sciences (LTSN ICS) and supplemented by personal contacts and Web searches for the e-mail address of the computing head of school. An e-mail was sent early in December 2000 to the people on the list asking for the questionnaire to be completed.

For institutions who had not responded towards the end of January 2001 a reminder e-mail was sent to the same contact. In mid-February 2001 every institution that had not responded was contacted by telephone and asked to complete the survey. A total of 55 out of approximately 110 HE level computing schools completed the questionnaire. It is intended to continue collecting responses and the Web hosted version of this report will be updated with additional data as it becomes available.

The instructions for completing the survey emphasised that the data obtained would be processed anonymously, and the Web system was so successful in this respect that questions concerning differences between pre-92 and post-92 Universities cannot be answered. Despite this a small number of institutions refused to complete the questionnaire on the grounds that they regarded the information as being too sensitive to reveal. Another small group of institutions became embroiled in internal discussions regarding the information to be supplied and have not yet responded. The reasons why other institutions have not responded is unknown. It is presumed that the request was never transmitted to the appropriate person or that the request was not regarded as sufficiently important.

Two institutions (both Russell group) responded in detail, describing how their students were required to regularly attend small group tutorials and describe the work that they had done. Consequently they did not regard source code plagiarism as a problem as students who cheated would be unable to adequately explain the code they presented and so would be caught out. One institution responded that they emphasised the principle of software reuse to the extent that using other student's source code was regarded as totally legitimate. The same institution commented that the appropriate response to student cheating would be to educate them as to why it was unacceptable.

A version of the questionnaire was prepared and trailed with a small number of Further Education computing lecturers. The opinion obtained was that the nature of assignments that FE students were given made plagiarism much less prevalent. Additionally, the smaller number of students in a cohort made detection much more likely and the appropriate response was seen as more educative than punitive. Because of these initial results no large scale survey of FE staff was conducted. However, it is felt that the substantive results of this report on the characteristics of available technology would be applicable to any interested FE tutors.

2.2 The Results

The detailed results of the questionnaire are given in Appendix D, only an executive summary of the outcomes is presented here.

Questions 1 & 2 asked about the prevalence and extent of plagiarism on initial courses. The majority of the responses indicated that it occurred some, most or almost all the time and that it usually involved less than 20% of the cohort.

Questions 3 & 4 asked the same questions about subsequent courses and the responses indicated that plagiarism was a little less prevalent and involved a smaller proportion of students.

A personal communication from Alex Aiken asserts, on the basis of his experience of operating the MOSS server, that for any (USA) corpus 10% of submissions will be plagiarised.

Question 5 asked about the attitude towards plagiarism as a problem and the modal response was that it was 'a routine headache' with equivalent numbers indicating that it was a 'minor nuisance' or 'bad and getting worse'. Only one response was prepared to say that it was 'not a problem'.

The evidence of previous UK surveys on student behaviour (Newstead et al 1996) suggests that tutors under-estimate the extent of cheating behaviour. If this finding is applicable to source code plagiarism then a possible interpretation of the implications of questions 1 to 5 is that plagiarism is both more extensive and more prevalent than indicated.

Questions 6, 7 & 8 indicated that 36 institutions routinely check submitted source code but half of these (17) relied upon visual inspection with only 4 making use of a public service such as MOSS. Of those that do not use any system the usual (11/17) reason was that no one had ever thought of using one.

Question 9 indicated that four fifths of institutions have a written policy on source code plagiarism. However almost all of these are reactive policies that state what is to be done when plagiarism is found. Very few are pro-active in requiring all work to be checked.

Question 10 suggests that the overwhelming majority of institutions would possibly, probably or certainly use a detection service. Very few (4/55) stated that they would not use one.

Question 11 asked about attitudes towards plagiarism detection and the modal response was that it is required to validate the effectiveness of an assessment. Almost all the other responses indicated that it is either an unfortunate requirement or that without it examinations would have to be exclusively used.

Question 12 asked tutors to indicate why they thought students resorted to plagiarism. No tutor stated that the assignment specifications were too complex (although this might be the favoured response of students). The modal response was that students were too disorganised closely followed by the response that programming was too difficult for students to learn. Many respondents commented at the end of the questionnaire that they would have liked to make multiple responses to this question.

Questions 13 & 14 asked about the number of students in the school and the number caught plagiarising in the last academic year. The response to question 14 was very patchy with responses ranging from a very small fraction (0.0014%) to approximately 13% with an average of approximately 5%. There was small negative correlation (-0.20) between size of school and rate of offending. This might suggest that cheating is less common in larger schools or that it is easier to avoid being detected although this is likely to be statistically insignificant.

Questions 15 & 16 were free response questions asking what the modal penalties were for first time and repeat offenders. The responses were categorised and indicated that the standard penalty for a first time offender (38/50) was related only to the piece of work which was copied. Either the mark for that component was zeroed and/or a rework was required. For repeat offenders a more severe penalty was indicated, with half of all respondents stating that expulsion was a possibility. However many stated that such cases were uncommon.

Questions 17 asked which programming languages any proposed system should be configured to check. The most popular response was Java with almost as many requiring C or C++. This was followed by Visual Basic, Pascal, Prolog, Ada, Delphi, Haskell, Scheme and LISP which all achieved more than one mention. Ten other languages (listed in Appendix D) all gained a single mention.

2.3 Free Response Comments

The questionnaire concluded with a request for respondents to add any other comments they might like to make.

Several respondents commented that 'collusion' or 'substantial collaboration' is much more common than outright plagiarism as such and that theft of floppy disks or listings from printers is an increasing problem. Accordingly only those cases that are 'in your face' cheating are taken forward to formal action.

Some tutors faced with the extent of cheating and the effort required to take a case through disciplinary proceedings either ignore it or take informal action. However, it was noted that informal actions cannot be used as part of a subsequent formal procedure where, as question 16 suggests, expulsion would otherwise be a possibility.

Comments were made that students do find the problems set too difficult, but that problems of comparable difficulty were set five or ten years ago and did not present so many problems. This might be correlated with the increasing number of students who are having to work to support themselves and causing increased absence from lectures and tutorials, a group which were noted by one respondent to be more likely to cheat.

The institutional response is mentioned by some respondents with those who report plagiarism seen negatively by their managers and one tutor claiming that they were reprimanded for setting work that was too difficult. Others state that after making a clear case no action resulted from the disciplinary process. This is possibly because registry staff did not understand the nature of software development or because students flatly deny any wrongdoing despite the evidence and without an admission the institution feels it cannot proceed.

Two institutions have responded that in response to the problems of detection and processing they are abandoning continuous assessment and resorting to laboratory based exams, even though this is regarded as educationally less valid.

On a more positive note some institutions are implementing the ideas of pair programming from the extreme programming community in the hope that it will mitigate the problem and others are considering differentiated assessments and measuring value added per individual student.

The culture of cheating is mentioned by one response making the connection between the acceptance of cheating in the first year leading to cheating on final year projects. Other cultural comments relate to different expectations of students who arrive from non-UK institutions and their not realising that plagiarism is unacceptable.

3. Internet Hosted Similarity Detection Engines

A Web search, requests on e-mail lists and discussions with attendees at the 2001 SIGCSE symposium indicated that there were only two widely known, freely available source code similarity detection engines, MOSS and JPlag. There are a large number of private engines, produced and used within a single institution and tightly coupled with teaching or teaching administration systems, for example the BOSS system developed by Mike Joy at Warwick. There are also a smaller number of more generic engines in development, for example Big Brother developed by Rob Irving at Glasgow.

This evaluation is restricted to the two freely available engines and has been produced after using both systems, reading all available documentation and e-mail discussions with the service providers.

MOSS (Measure Of Software Similarity) is the better known of the two engines. It was developed by and is maintained by Alex Aiken at the University of California, Berkeley campus. Its home page is located at

<http://www.cs.berkeley.edu/~aiken/moss.html>

JPlag was developed by Guido Malpohl and is maintained by Guido Malpohl and Lutz Prechelt and at the University of Karlsruhe. Its home page is located at

<http://www.JPlag.de>

Both of the engines are supplied and supported by the maintainers as a service to the academic community, with the passive support of their institutions. There is no formal guarantee of continued availability or statement of the level of support available. However, from experience, e-mail queries have been promptly responded to and both maintainers state that it is their intention to provide the service indefinitely. Neither supplier charges for the service but both require users to register in order to ensure that only bona fide academics are using the facilities.

3.1 Qualitative analysis

Rather than describe each service separately, both engines will be evaluated side by side using a number of criteria. Table 1 contains a summary of the evaluation.

	MOSS	JPlag
Languages supported	8	4
Algorithm used	Not stated	Token pattern matching
Independent verification	No	No
Self validation	No	Yes
Collection tool	No	No
Multiple File submissions	Yes	Yes
Platforms supported	Unix	Any
Submission method	Command line Stand alone & from browser	Command line Stand alone & from browser
Reporting method	Remote Web pages	Remote or local Web pages
Report contents	Ordered list Unparsed files	Submissions processed Histogram Clustered Ordered list
Metrics produced	percentage extent tokens matched lines matched	percentage similarity
Visualisation tool	Cross linked listings	Cross linked listings
Security	User id & e-mail	User id & password
Faq & support	Minimal	Minimal
Miscellaneous	Self adjusting Counter intuitive	N° of pairs can be specified

Table 1 Qualitative analysis summary

Languages supported. The programming languages currently supported by the engines are listed in Table 2. Although MOSS supports a larger number of languages results from the UK survey, as given above, indicate that Java and C/C++ predominate demand. Both services report that it would be relatively easy to support additional languages.

	Java	C++	C	Pascal	Ada	ML	Lisp	Scheme
MOSS	✓	✓	✓	✓	✓	✓	✓	✓
JPlag	✓	✓	✓					✓

Table 2 Programming languages supported by the engines

Algorithm used. Details of the algorithm used by the MOSS engine are not given in order to ensure that it is not circumvented. The only details given are that it actually examines program structure and does not rely upon superficial metrics. It is believed to be tokenisation followed by fast sub-string matching. The JPlag engine's algorithm is described in a technical report (Prechelt et al 2000) and is also tokenisation followed by sub-string pattern matching.

Independent verification. A search of the ACM and IEEE digital libraries and other available journal search engines resulted in only one publication that contained either 'MOSS' or 'JPlag' in its title, keywords or abstract. This paper (Boywer & Hall 1999) proved to be a descriptive paper relating how the MOSS engine was used in a single institution for analysis of a single submission. As such it was not considered adequate verification of the efficacy of the service. Both services report anecdotal comments from their users on the efficacy of the engines and of their value in plagiarism detection and deterrence.

Self validation. There has been no publication by MOSS of any attempt to validate the engine. JPlag has published a technical report (Prechelt et al 2000) and a paper (Prechelt et al 2001) describing a validation exercise. The exercise consisted of submitting four different corpora of student submissions to the engine and investigating the effects of tuning various parameters. Additionally for one of the corpora a number of deliberately plagiarised submissions were added and the engine was shown to be capable of detecting all of them. It would be valuable to conduct a replication of the black box aspects of this study with both the MOSS and JPlag engines, and a start at this has been made in the quantitative analysis study reported below in section 3.2.

Collection issues. Neither service supplies or supports a collection tool requiring the user to arrange for the corpus of submissions to be assembled in some way. Both tools support both single file and multiple file submissions, with the multiple files having to be placed into their own sub-directory in both cases.

Platforms supported: Submissions are sent to MOSS by executing a PERL script, which itself relies upon the existence of an environment that supports uuencode, mail and either zip or tar utilities. Although this does not preclude any platform, in practice it limits the system to Unix environments. JPlag, minimally, relies upon a Java run time system being available and so should run on almost any platform.

Submission method: MOSS only supports submission by execution of the PERL script from the command line. JPlag can also execute from the command line for maximum flexibility, but additionally it can be run as a stand-alone program or as an applet within Netscape (explicitly not Internet Explorer, Mosaic, Opera, etc.).

Reporting method: After processing the submissions MOSS sends an e-mail to the address from where the submission was made. The e-mail contains the URL of a page on the MOSS server where the results can be consulted. There is no provision for the results to be downloaded (for reasons of operational efficiency) and the results remain available for 14 days. When JPlag is used from the command line the results are placed directly onto the machine that was used with parsing and processing logs shown on the terminal. When JPlag is used stand

alone or within Netscape the results and the logs are available on a private page on the JPlag server, with an option for them to be downloaded as a zip archive.

Report contents: The main report page supplied by MOSS consists of an ordered list of pairs followed by a list of the submissions which could not be parsed, but which have been included in the processing with possibly inaccurate results. The report supplied by JPlag consists of a list of all the files which were processed, this may not be the same as all the files which were submitted as files which could not be parsed are excluded from processing. The list of excluded files is not contained in the report but it is available in the parsing log. The JPlag report also contains a histogram showing the number of pairs of submissions in each 10 percentile range and an ordered clustered list of pairs. The ordered clustered list commences with the most similar pair and then lists all other files, in order of similarity, to the first member of the pair which has the fewer matches with other submissions. This constitutes a cluster, and is followed by all other clusters.

Metrics produced. The JPlag engine produces a single metric for each pair, percentage similarity, which is defined as the proportion of the combined submissions which were shown to be identical. Full details of the metric are contained in (Prechelt et al 2000) and (Prechelt et al 2001). The MOSS engine produces four metrics for each pair. A percentage extent for each submission, presumably the same as the JPlag percentage similarity but reported for each submission individually. These are accompanied by the number of tokens which have been matched and the number of lines that have been matched. MOSS suggests that the tokens matched metric is the most indicative and the list is ordered by these values.

Visualisation tool. The visualisation tools for both engines are essentially identical having been first developed by JPlag and then adapted for use by MOSS. They consist of a HTML page containing three frames. One, at the top, contains a table which lists by line numbers, and colour codes, the fragments in each submission which were shown to be identical. Each line of the table is a hyperlink which, when activated, will cause the corresponding parts of the submissions, displayed alongside each other in the two remaining frames, to be shown. The corresponding parts of each submission are themselves appropriately colour coded and contain a graphical hyperlink that can be used to synchronise the view in the other frame. This latter mechanism allows the user to scroll through one of the submissions and easily locate the identical fragment in the other.

Security. As mentioned above it is necessary to register with each service before first use as a precaution to ensure that only academic staff have access to it. Registering with MOSS requires an e-mail address to be supplied and a numeric user-id is provided, this id has to be included within the script used to submit the program listings. The JPlag registration process requires the user to

indicate a password, and also issues a numeric id. The id is then used as a parameter to the URL (or encoded within the JAR archive supplied via this URL for standalone use) and the password has to be supplied with every submission.

FAQ and support. Both systems include a short FAQ and will respond to individual queries by e-mail.

Miscellaneous. One feature of the MOSS system not described above is the self-adjusting nature of the processing. Although it is possible for the user to indicate a 'base file' which was supplied to the students and which will cause the system to exclude its contents from consideration, a similar process operates automatically with code that appears in a high proportion of submissions being excluded from consideration. This has the counter-intuitive consequence that two submissions which are 100% identical on visual inspection are reported as being less than 100% in the results and may actually be reported as being less similar than a pair that are not 100% identical. However a command line option can be used to turn this feature off.

The command line options for the JPlag tool allow the user to indicate the number of pairs to be returned or the minimal percentage similarity to be reported upon. The MOSS tool seems to return either 500 pairs or fewer depending upon the possible number of pairs or the extent of the similarity. Both tools contain an option that allows the sensitivity of the matching to be tuned by indicating the minimum number of tokens in a match to be specified.

3.2 Quantitative analysis

The intention of this analysis was to investigate the extent to which the two available detection engines, MOSS and JPlag, agreed with each other. As any decision to ask an individual student to explain undue similarity in their work starts with the list of pairs produced by a detection engine, it would seem essential that the order of this list be as accurate as possible. Although it is difficult, if not impossible, to validate the ordering in any absolute manner, it might be hoped that two independent engines would produce a largely similar list.

To accomplish the analysis a corpus of Java source code obtained from the 2000/2001 SBU first year BSc computing students completing a specification known as waypoint 5, was submitted to each engine. The specification was in two parts. The first of which required the students to construct an accompanying class for a pre-supplied class hierarchy that allowed the user to initialise the state of a new instance by inputting values for its attributes from the terminal. Once this has been accomplished the second part required them to use this facility in a more complex specification which maintained a collection of objects.

The waypoint 5 specification closely shadowed the contents some taught material known as waypoint 4, and students were encouraged to adapt the provided waypoint 4 material when completing waypoint 5. Hence it was anticipated that there would be a large degree of similarity within the entire cohort. Only submissions for the second part of the specification were used in the qualitative analysis as the first part had proved so uncomplicated that all submissions were highly similar. A typical submission was approximately 200 lines long and contained a main() method and six to eight subsidiary methods.

A total of 102 submissions were available at the time of the investigation of which 88 were determined by JPlag to be syntactically correct and processed. For the same 102 submissions MOSS reported that only 4 had syntactic problems, which were processed with a warning that the results for those files may be inaccurate. As neither of the engines actually uses a compiler to ensure that a submissions will compile this inconsistency must be caused by differences between the parsers used. Unfortunately there was no time before the production of this report to investigate this further and the analysis was restricted to the 88 submissions which were acceptable to both engines.

The MOSS engine returned a list of the 500 pairs of submissions that had the highest values on the tokens matched metric; there was no apparent mechanism to obtain any more pairs. To facilitate processing the two extent values, that is the percentage of each file that was implicated as being similar, were combined to give a single percentage extent metric. The third metric, lines matched, was also used in the analysis. 2000 pairs of submissions were requested from the JPlag engine each of which had a single percentage similarity metric.

Consequently for a pair of submissions returned from either MOSS or JPlag there could be only the JPlag percentage similarity metric if it was reported by JPlag and not by MOSS. Alternatively if it was reported by MOSS and not by JPlag there would only be the MOSS percentage extent, tokens matched and lines matched metrics. Finally if the pair were reported by both engines then all four metrics would be available.

To facilitate discussion the four metrics will be identified as j1, m1, m2 and m3 corresponding to the percentage similarity, percentage extent, tokens matched and lines matched metrics respectively. The MOSS site suggests that the m2 metric is the most indicative of undue similarity.

percentage extent and tokens matched (m1 and m2)	0.729471
percentage extent and lines matched (m1 and m3)	0.74123
tokens matched and lines matched (m2 and m3)	0.83344

Table 3 Moss metric correlations

The first stage in processing was to compute the correlations between the three MOSS metrics for all 500 pairs. The results are given in Table 3 and the values,

particularly the m2 & m3 correlation, suggest that metrics are measuring approximately the same factors within the submissions.

The analysis continued by taking approximately the top 100 pairs from all four lists and combining them into a consolidated list by removing duplicates. It was not possible to take exactly the top 100 pairs due to sequences of pairs having the same value, in these cases more than 100 pairs were taken. If all four metrics were measuring the same factor in the same manner this should result in a consolidated list of approximately 100 pairs. The process produced a list that contained approximately 240 pairs, already indicating a degree of dissimilarity between the individual lists.

In the consolidated list there were 63 pairs which were contained within the top 100 pairs from the JPlag list that were not included within the top 500 pairs of the m2 MOSS list. There were also 28 pairs from the top 100 pairs obtained from the MOSS lists which were not contained in the top 2000 pairs from the JPlag list. When these unmatched pairs were removed from the this resulted in a pruned consolidated list that contained only 150 pairs. The Spearman rank correlation coefficients computed from this list are given in Table 4.

percentage extent and tokens matched (m1 m2)	0.467093
percentage extent and lines matched (m1 and m3)	0.458347
tokens matched and lines matched (m2 and m3)	0.573241
percentage extent and percentage similarity (m1 & j1)	0.348746
tokens matched and percentage similarity (m2 & j1)	0.087894
lines matched and percentage similarity (m3 & j1)	0.00194

Table 4 Spearman rank correlations for the 100 pair pruned list

Of the correlations between m and j factors only that between m1 and j1 was significant at the 1% level. (All the m intercorrelations were significant at this level.) The lack of significance between the favoured m2 and j1 metrics gives no evidence that the engines are in accord with each other. The significance of the m1 and j1 metrics is explicable when the nature of the two metrics, essentially the proportion of the files which are shown to be identical, is considered.

The conclusion from this part of the analysis is that, particularly when the 91 excluded pairs are taken into account, there does not appear to be a large degree of consensus between the two engines. However it could be argued that this lack of agreement might be caused by considering the top 100 matches from each metric and a greater degree of agreement might be obtained if the range were more restricted.

Consequently the analysis was repeated using approximately the top 25 entries from each list. This produced a consolidated list containing 71 pairs and a pruned consolidated list that contained 50 pairs. The Spearman rank correlation coefficients computed from this list are given in Table 5.

percentage extent and tokens matched (m1 m2)	0.399856
percentage extent and lines matched (m1 and m3)	0.341224
tokens matched and lines matched (m2 and m3)	0.596879
percentage extent and percentage similarity (m1 & j1)	0.363745
tokens matched and percentage similarity (m2 & j1)	0.088211
lines matched and percentage similarity (m3 & j1)	-0.18824

Table 5 Spearman rank correlations for the 25 pair pruned list

The level of significance of these correlations is identical to those of the previous analysis (the first four are significant at the 1% level, the remaining two are not). These results would tend to suggest that, if anything, there is less agreement between the two engines at the very top of the similarity list than in the upper part of the list.

Assuming that suspicious pairs are examined in the sequence indicated by the engine and that the resources for physical inspection are limited. The implication of these findings is that the chances of a pair being inspected, and hence the chance that a student will be asked to explain undue similarity, depends upon which engine has been used. This would seem to be unduly capricious. However the analysis has only been conducted upon a relatively small first year corpus that was expected to contain a high degree of similarity and so would need to be repeated with more corpora before this conclusion could be confirmed.

The providers of the services would like it to be pointed out that the tools are not intended as tests for plagiarism. They supply an ordered list of apparent similarities that allow a tutor to more efficiently locate the pairs that should be examined to determine if they require further investigation.

4. Other Issues

The UK QAA computing benchmark statements (available from www.qaa.ac.uk/crntwork/benchmark/computing.pdf) includes, for both threshold and modal levels, the following statement:

Produce work involving problem identification, the analysis, the design and the development of a system, with accompanying documentation. The work will show problem solving and evaluation skills, draw upon supporting evidence and demonstrate a good understanding of the need for quality

The British Computer Society (BCS) Guidelines on Course Exemption & Accreditation (available from www.bcs.org.uk/educat/guide.htm) includes the following statement:

All exempt courses will: . . . have a strong emphasis on design throughout the course . . . [stress] the need for . . . an emphasis on design and the development of appropriate practical skills.

This strong emphasis upon the design process and its accompanying documentation includes the production designs expressed using notations such as Jackson Structured Programming (JSP) schematics or Unified Modelling Language (UML) notations. These should be a integral part of any software development submission and be accorded a significant proportion of marks. As with program source code, the production of these artefacts is a time consuming and laborious, yet very necessary, process and a superficial disguise of a plagiarised copy is for some students a very attractive alternative.

From experience in SBU SCISM the submission of plagiarised designs is not as prevalent as the submission of plagiarised source code but is significant. However, as far as is known, no similarity ranking engines exist to support detection of cheating in this aspect of software development.

The visualisation tools supplied as part of the MOSS and JPlag systems are considerably more effective for verifying the existence of undue similarity than unaided manual inspection. However they lack the ability to scale the visualisation from a detailed view to an overview and are limited to examining only a pair of submissions, providing no support for the investigation and validation of a cluster. Again practical experience of source code plagiarism detection suggests that the vast majority of similarities are part of a cluster rather than a simple pair.

The only known approach to improved and alternative visualisation tools are the Composite Categorical Patterngrams (CCP) reported by Ribler and Abrams (Ribler & Abrams 2000) which allow clusters of plagiarised submissions to be detected and investigated.

5. Conclusions and recommendations

The survey provides a snapshot of the extent of and attitudes towards source code plagiarism provided by approximately 50% of UK HE schools of computing. It suggests that there is a definite problem and that automated techniques to assist detecting similarity could be more widely used.

While the MOSS and JPlag services remain available under the current conditions of use there seems to be little point in JISC replicating either service or supplying an alternative. However this recommendation is made without regard to the requirements of UK data protection and copyright legislation and should the export of identifiable data outside the UK (or EU) prove to be problematic then this conclusion may need to be revised.

A watching brief should be maintained upon both services and JISC may have to consider taking action if one or both of them becomes unavailable at some time in the future.

The UK computing academic community owes a debt of gratitude to Guido Malpohl and Alex Aitken, and their institutions, for developing and supporting the services.

The JPlag service would seem to be the easiest to use and produces more comprehensible results. However it supports a smaller number of languages and will not process programs which do not parse. Whilst the MOSS server will process programs which do not parse the results produced in these circumstances may be invalid. It is not JISC's purpose to endorse either of these services but to make their existence and characteristics known to the UK community.

The reasons why some files that are successfully parsed by MOSS are refused by JPlag and the reasons why the results returned from each system are apparently widely different need further investigation. This would seem to be suitable for a final year undergraduate or master's level project.

Appendix A Source code plagiarism publications list

This list contains an entry for every reference that was consulted in the preparation of this report. It is not claimed that this is an exhaustive literature survey and the authors would be most interested in learning of any other suitable entries.

Kevin W. Bowyer and Lawrence O. Hall 1999
Experience Using "MOSS" to Detect Cheating on Programming Assignments
Proc. 29th ASEE/IEEE Frontiers in Education Conference, 13b18-22

Janet Carter 1999
Collaboration or Plagiarism: What Happens when Students Work Together?
Proc. ITiCSE 1999, Cracow, Poland, pp52-55.

Paul Clough 2000
Plagiarism in Natural and Programming Languages: an Overview of Current Tools and Technologies
http://www.dcs.shef.ac.uk/~cloughie/plagiarism/HTML_Version/index.html

Fintan Culwin. & Jeoff Naylor 1995
Pragmatic Anti-Plagiarism
Proc. 3rd All Ireland Computer on the teaching of computing, DCU Dublin

Padraig Cunningham & Alexander N. Mikoyan 1993
Using CBR Techniques to Detect Plagiarism in Programming Assignments
Available from Department of Computer Science, Trinity College, Dublin

Donaldson, John L., Lancaster, Ann-Marie & Sposato, Paula H. 1981
A Plagiarism Detection System
Proc. Twelfth SIGSCE Technical Symposium, St. Louis, Missouri, pp. 21-25.

Faidhi J. A. W. & Robinson S. K. 1987
An Empirical Approach for Detecting Program Similarity and Plagiarism Within a University Programming Environment
Computer Education, v 11 No. 1

Andrew Gray, Philip Sallis & Stephen MacDonnel 1998
IDENTIFIED (Integrated Dictionary-based Extraction of Non-language-dependent Token Information for Forensic Identification, Examination and Discrimination): A Dictionary-based System for Extracting Source Code Metrics for Software Forensics
Proc. 1998 International Conference in Software Engineering, Education and Practice, pp252-259.

James O. Hamblen, Alan Parker & Stephen R. Wachtel 1998
A New Undergraduate Computer Arithmetic Software Library
IEEE Transactions on Education, Volume 31, No. 3

James K. Harris 1994
Plagiarism in Computer Science Courses
Proc. 1994 Ethics in Computer Age, pp133-135.

Jonathan Isaac Helfman 1994
Similarity Patterns in Language
Proc. IEEE Symposium on Visual Languages, pp173-175

Rob Irving 2000
Plagiarism Detection: Experiences and Issues
Presented at JISC Fifth Information Strategies Conference:
Focus on Access and Security, London

Jankowitz H. T. 1988
Detecting Plagiarism in Student Pascal Programs
The Computer Journal Vol. 31 No. 1

Mike Joy & Michael Luck 1999
Plagiarism in Programming Assignments
IEEE Transactions in Education, Vol. 42, No.2, pp129-133

G. R. Lund 1995
Controlling Plagiarism in Student Programs
Proc. 3rd All Ireland Computer on the teaching of computing, DCU Dublin

Newstead S.E., Franklin-Stokes A., Armstead P. 1996
Individual Differences in Student Cheating,
Journal of Educational Psychology Vol. 88 No 2. 229-241.

Ottenstein, Karl J. 1977
An Algorithmic Approach to the Detection and Prevention of Plagiarism
SIGCSE Bulletin vol. 8 No. 4, pp. 30-41.

Alan Parker and James O. Hamblen 1999
Computer Algorithms for Plagiarism Detection
IEEE Transactions on Education, Volume 32, Number 2, pp94-99.

Lutz Prechelt, Guido Malpohl & Michael Phillipson 2000
Jplag: Finding Plagiarisms among a Set of Programs
Universitat Karlsruhe, Germany, Technical Report 2000-1

Lutz Prechelt, Guido Malpohl & Michael Phillipson 2001
Finding Plagiarisms Among a Set Of Programs with Jplag
Submission to Journal of Universal Computer Science

Ribler R. L. & Abrams M. 2000
Using Visualisation to Detect Plagiarism in Computer Science Classes
IEEE 0-7695-0804-9/2000, pp173-177.

Robinson, Sally S. & Soffa, M. L. 1980
An Instructional Aid for Student Programs
SIGCSE Bulletin Vol. 12 No. 1, pp. 118-129.

Philip Sallis, Asbjorn Aakjeer and Stephen MacDonnal 1996
Software Forensics: Old Methods for a New Science
Proc. International Conference in Software Engineering, Education and Practice, pp481-485

Robert Sanders 1998
Online Plagiarism Detector Helps CS Professors Bust Cheating Programmers
Berkeley <http://www.coe.berkeley.edu/EPA/EngNews/98S/EN1S/aiken.html>

Steve Saxon 2000

Comparison of Plagiarism Detection Techniques Applied to Student Code
Trinity College, Cambridge Part II Computer Science Project

John Traxler 1995

Cheating In Pascal Programming Assessments with Large Classes
Proc. 3rd All Ireland Computer on the teaching of computing, DCU Dublin

Kristina L. Verco & Michael J. Wise 1996

Plagiarism a la Mode: A Comparison of Automated Systems for Detecting Suspected Plagiarism
The Computer Journal, Vol. 39, No. 9, pp741-750.

Kristina L. Verco & Michael J. Wise 1996

Software for Detecting Suspected Plagiarism: Comparing Structure and Attribute-Counting Systems

Proc. First Australian Conference on Computer Science Education, Sydney Australia

Whale G. 1990

Identification of Program Similarity in Large Populations
The Computer Journal, Vol. 33 No. 2

Laurie Williams 1999

But Isn't That Cheating?
Proc. 29th ASEE/IEEE Frontiers in Education Conference, San Juan, Puerto Rico

Michael J. Wise 1996

YAP3: Improved Detection of Similarities in Computer Programs and Other Texts
Proc. SIGCSE '96, Philadelphia, USA, Feb 15-17 1996, pp130-134

Appendix B

The questionnaire

For questions 1 to 12 please delete the responses that DO NOT apply until ONLY ONE response is left:

Question 1 I think that outbreaks of source code plagiarism, involving any proportion of students, on initial programming courses is:

- * Extensive almost all the time
- * Prevalent most of the time
- * Moderate some of the time
- * Occasional on occasion
- * Rare almost unknown

Question 2 I think that the proportion of students involved in a typical outbreak of source code plagiarism on initial programming courses is:

- * more than 50%
- * between 35% and 50%
- * between 20% and 35%
- * between 10% and 20%
- * less than 10%

Question 3 I think that outbreaks of source code plagiarism, involving any proportion of students, on subsequent programming courses is:

- * Extensive almost all the time
- * Prevalent most of the time
- * Moderate some of the time
- * Occasional on occasion
- * Rare almost unknown

Question 4 I think that the proportion of students involved in a typical outbreak of source code plagiarism on subsequent programming courses is:

- * more than 50%
- * between 35% and 50%
- * between 20% and 35%
- * between 10% and 20%
- * less than 10%

Question 5 I think that the problem of source code plagiarism is:

- * bad and getting worse
- * under control
- * a routine headache
- * a minor nuisance
- * not a problem

Question 6 Does your institution routinely check all submitted source code for indications of copying?

- * yes (please answer question 7 and ignore 8)
- * no (please answer question 8 and ignore 7)

Question 7 (if your answer to Question 6 was yes) The checking system used is:

- * non-automated, by hand and eye
- * a public service such as MOSS
- * a service developed and/or operated in house
- * a part of an integrated teaching environment
- * some other technique (please specify at the end)

Question 8 (if your answer to Question 6 was no) A checking system is not used because:

- * nobody has ever started to use one
- * the consequences of using one are thought horrendous
- * a decision has been taken not to use one
- * it is believed that no or little cheating takes place
- * the group sizes are so small it is not needed

Question 9 Does your school/department have a written policy on source code plagiarism:

- * no
- * yes - a pro-active policy that requires work to be checked
- * yes - a re-active policy that states what to do when it is found

Question 10 If a national source code plagiarism detection service was established do you think that your school would use it?

- * no - we do not use one and would not use one
- * no - we are happy with our current system
- * no - due to cost and staffing issues
- * possibly - it would depend upon the type of service
- * probably - providing it met our needs
- * certainly - when will it start?

Question 11 Which of the following statements best describes your attitude towards source code plagiarism detection?

- * it is not needed
- * it is not an activity that we should get involved in
- * it is an unfortunate requirement
- * it is required to validate the effectiveness of assessment
- * without it examinations are the only valid form of assessment

Question 12 I think that the major reason why students resort to plagiarism is:

- * programming is too difficult for them
- * the programs they are asked to do are unreasonably complex
- * there is too much pressure from other subjects
- * there is too much pressure from work and/or family
- * they are too disorganised to complete the work in time

Question 13 The number of FTE students in my school is approx. _____

Question 14 The number of students caught plagiarising source code in the last academic year was approx. _____

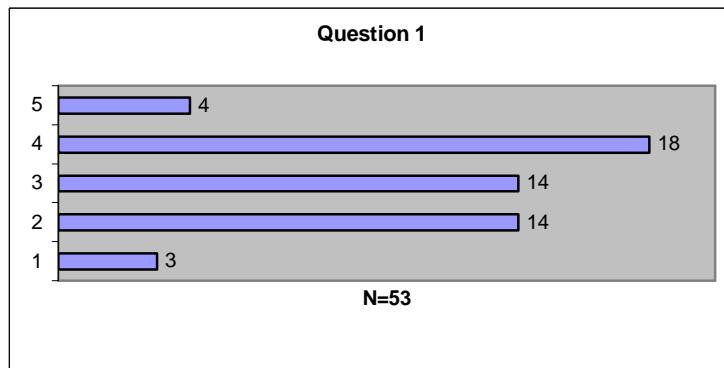
Question 15 The modal penalty for a first time offender who was adjudged to have committed significant plagiarism would be _____

Question 16 The modal penalty for a repeat offender who was adjudged to have committed significant plagiarism would be _____

Question 17 If a national service were to be set up, in order from most to least important, the programming languages needed would be _____

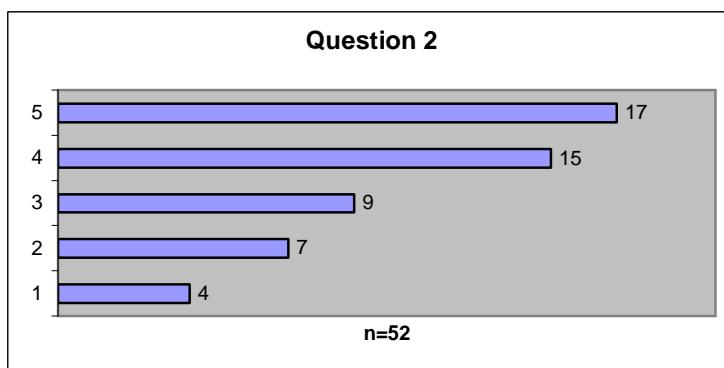
Please add any further comments here.

Appendix C Detailed survey results



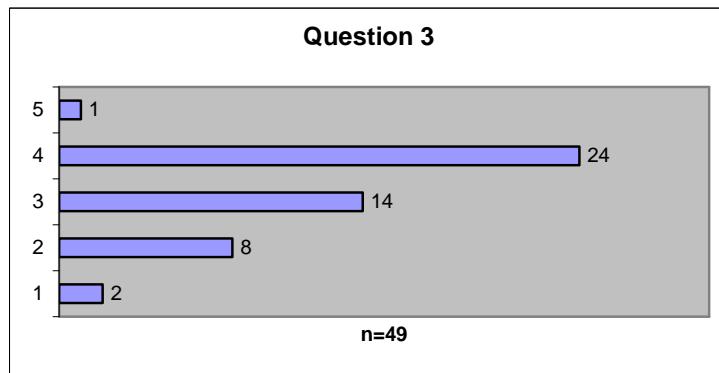
I think that outbreaks of source code plagiarism, involving any proportion of students, on initial programming courses is:

- 5 rare almost unknown
- 4 occasional on occasion
- 3 moderate some of the time
- 2 prevalent most of the time
- 1 extensive almost all the time



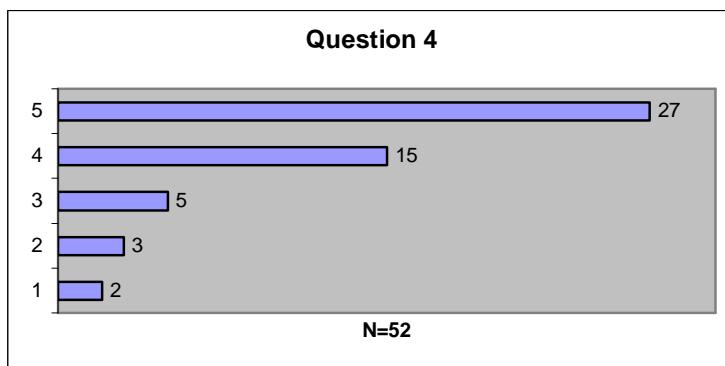
I think that the proportion of students involved in a typical outbreak of source code plagiarism on initial programming courses is:

- 5 less than 10%
- 4 between 10% and 20%
- 3 between 20% and 35%
- 2 between 35% and 50%
- 1 more than 50%



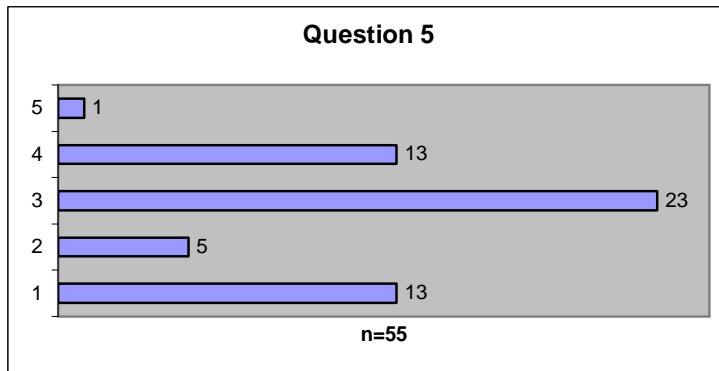
I think that outbreaks of source code plagiarism, involving any proportion of students, on subsequent programming courses is:

- 5 rare almost unknown
- 4 occasional on occasion
- 3 moderate some of the time
- 2 prevalent most of the time
- 1 extensive almost all the time



I think that the proportion of students involved in a typical outbreak of source code plagiarism on subsequent programming courses is:

- 5 less than 10%
- 4 between 10% and 20%
- 3 between 20% and 35%
- 2 between 35% and 50%
- 1 more than 50%



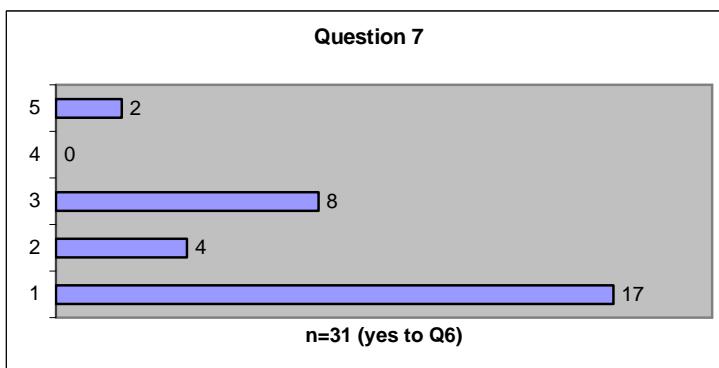
I think that the problem of source code plagiarism is:

- 5 not a problem
- 4 a minor nuisance
- 3 a routine headache
- 2 under control
- 1 bad and getting worse

Question 6

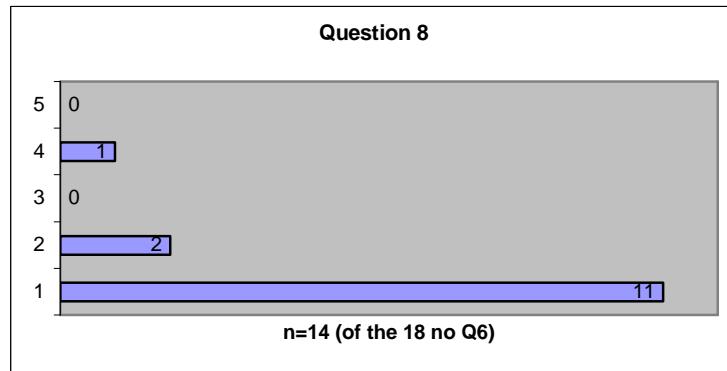
Does your institution routinely check all submitted source code for indications of copying?

- yes 36
- no 18



The checking system used is:

- 5 some other technique (please specify at the end)
- 4 a part of an integrated teaching environment
- 3 a service developed and/or operated in house
- 2 a public service such as MOSS
- 1 non-automated, by hand and eye



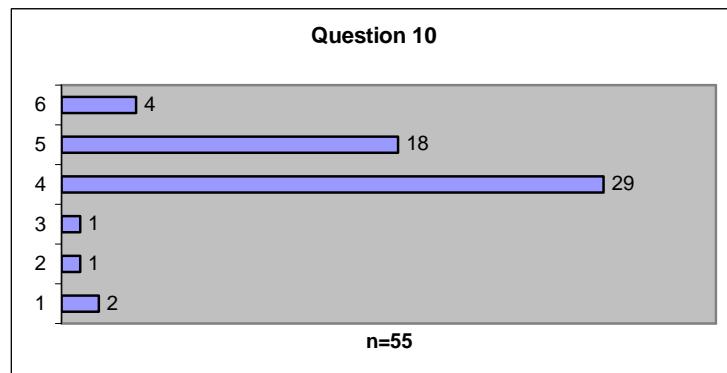
(if your answer to Question 6 was no) A checking system is not used because:

- 5 the group sizes are so small it is not needed
- 4 it is believed that no or little cheating takes place
- 3 a decision has been taken not to use one
- 2 the consequences of using one are thought horrendous
- 1 nobody has ever started to use one

Question 9

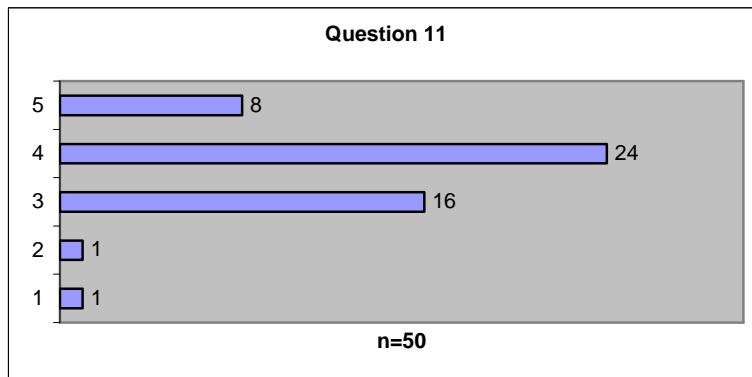
Does your school/department have a written policy on source code plagiarism:

no	10
yes - a pro-active policy that requires work to be checked	4
yes - a re-active policy that states what to do when it is found	39



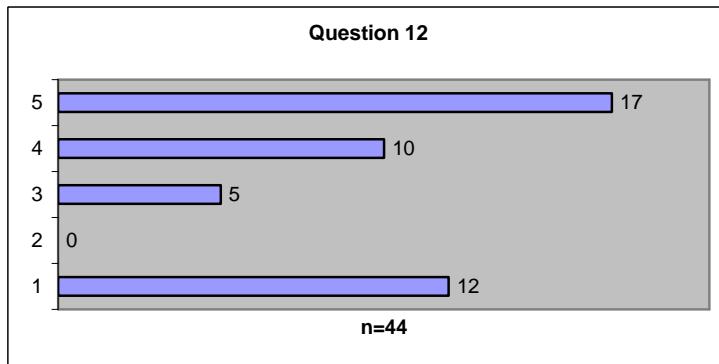
If a national source code plagiarism detection service was established do you think that your school would use it?

- 6 certainly - when will it start?
- 5 probably - providing it met our needs
- 4 possibly - it would depend upon the type of service
- 3 no - due to cost and staffing issues
- 2 no - we are happy with our current system
- 1 no - we do not use one and would not use one



Which of the following statements best describes your attitude towards source code plagiarism detection?

- 5 without it examinations are the only valid form of assessment
- 4 it is required to validate the effectiveness of assessment
- 3 it is an unfortunate requirement
- 2 it is not an activity that we should get involved in
- 1 it is not needed



I think that the major reason why students resort to plagiarism is:

- 5 they are too disorganised to complete the work in time
- 4 there is too much pressure from work and/or family
- 3 there is too much pressure from other subjects
- 2 the programs they are asked to do are unreasonably complex
- 1 programming is too difficult for them

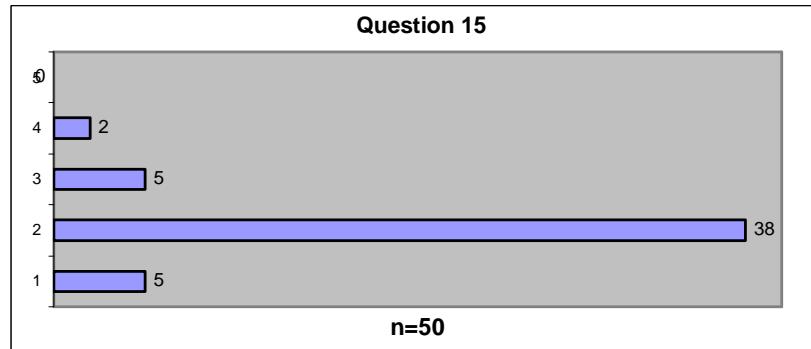
Questions 13 & 14

Q 13 The number of FTE students in my school is approx. _____

Q14 The number of students caught plagiarising source code in the last academic year was approx.

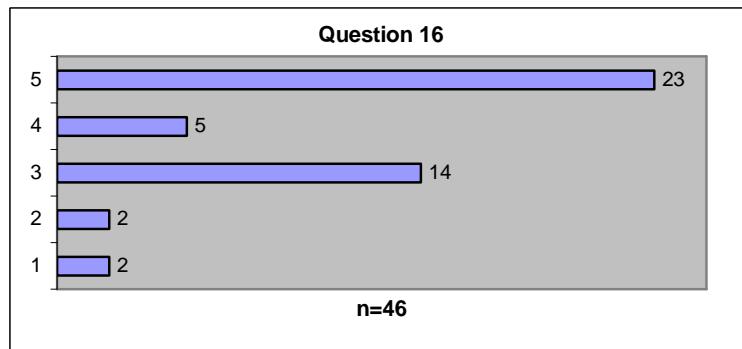
Q13 produced answers in the range 20 (actually a cohort on a course) to 1800. Q14 gave very varied answers, the modal appearing to be that exact records were not kept.

For the values which were given the percentage rate of offending ranged from a fraction (0.0014) of a percent to approximately 13%, with an average rate of approximately 5%. There was a small negative correlation (-0.20) between the size of a school and the percentage rate of offending.



The modal penalty for a first time offender who was adjudged to have committed significant plagiarism would be:

- 5 possible expulsion
- 4 course level penalty (e.g. repeat year)
- 3 unit penalty (e.g. fail part of year)
- 2 coursework penalty (e.g. zero work submitted & repeat)
- 1 warning only



The modal penalty for a repeat offender who was adjudged to have committed significant plagiarism would be:

- 5 possible expulsion
- 4 course level penalty (e.g. repeat year)
- 3 unit penalty (e.g. fail part of year)
- 2 coursework penalty (e.g. zero work submitted & repeat)
- 1 warning only

Question 17

If a national service were to be set up, in order from most to least important, the programming languages needed would be. (Languages listed without regard to priority).

Java	46
C++	27
C	13
VB	14
Pascal	6
Prolog	5
Ada	3
Delphi	3
Haskell	3
Lisp	2
Scheme	3
Eiffel	1
javascript	1
ML	1
Modula 2	1
Occam	1
Oracle	1
Perl	1
QBasic	1
SML	1
SQL	1
VHDL	1

Towards an Error Free Plagiarism Detection Process

Thomas Lancaster & Fintan Culwin

Centre for Interactive Systems Engineering

School of Computing, Information Systems & Mathematics

South Bank University

London SE1 0AA

United Kingdom

{fintan, lancaste}@sbu.ac.uk

ABSTRACT

For decades many computing departments have deployed systems for the detection of plagiarised student source code submissions. Automated systems to detect free-text student plagiarism are just becoming available and the experience of computing educators is valuable for their successful deployment.

This paper describes a Four-Stage Plagiarism Detection Process that attempts to ensure no suspicious similarity is missed and that no student is unfairly accused of plagiarism. Required characteristics of an effective similarity detection engine are proposed and an investigation of a simple engine is described. An innovative prototype tool designed to decrease the workload of tutors investigating undue similarity is also presented.

1 BACKGROUND

University computing academics have been concerned with developing and deploying systems and tools for the detection of plagiarised programming submissions since at least the 1970s [9]. Recently, a number of well publicised incidents have suggested that the widespread availability of word processing and electronic information systems have caused an increase in the proportion of students, in all subjects, who are plagiarising free-text (e.g. essay) submissions.

In 1995, over half of students surveyed by Franklyn-Stokes and Newstead admitted to plagiarising [7]. Today, all anecdotal evidence suggests that this figure is higher [8, 10] and this is likely due to the growth of students adding ‘free’ material from the Web to their submissions.

Against this background automated tools to locate plagiarism in student submissions are a necessary evil. Computing academics, because of their experience and subject-specific skills, are at the forefront of both the development of these tools and the

deployment of systems that employ them. This paper will describe a four-stage model for pro-active anti-plagiarism systems, the validation of a similarity detection engine and a tool to assist with the validation of suspect pairs of submissions.

2 PLAGIARISM DETECTION PROCESS

Two types of plagiarism detection services are currently available. The first detects *Web-based plagiarism*, (e.g. the apparent market leader plagiarism.org), which claim to detect material that has been copied from the Web. Reviewers have found that the services do locate plagiarism, but are limited, too technical in their approach to the end-user and too expensive for many institutions [4, 6]. John Barrie, of plagiarism.org, reported that tutors using his service find large amounts of plagiarism, even when students are pre-warned that their work will be checked by the automated system [1].

Systems to detect and verify *intra-corporeal plagiarism*, such as the Visualisation and Analysis of Similarity Tool (VAST) are also becoming available [5]. Intra-corporeal plagiarism is defined by Culwin and Lancaster in their *plagiarism taxonomy*, as plagiarism within a class set, or *corpus* [2]. This is where students copy from one another, or collaborate closely enough that their submissions are near identical.

Systems such as VAST work on the principle of similarity analysis and are used within the Four-Stage Plagiarism Detection Process, shown in Figure 1. Most of this process can be entirely automated, or tool assisted. A corpus of student submissions is first *collected* in a machine-ready form. The corpus is then *analysed* in an attempt to find submissions that contain large amounts of similarity. Pairs of submissions can be ranked on a number of metrics to give a *consolidated similarity rank* for each. Those pairs with high ranks are then *confirmed* by humans to ensure that they do not represent *false hits*, where the similarity is not as great as the ranking would suggest. Those with low ranks are expected to contain little similarity but verifying that there are no *missed pairs* in this range, that is pairs ranked low but containing similarity, is difficult. Pairs with high ranks can then be *investigated* to find sufficient evidence to warrant the student being required to explain evident plagiarism.

The automated process has been shown to be generally reliable. The *similarity detector* used gives reasonable results, which seem to rank similar pairs highly and avoid the lack of consistency associated with human judgement [3]. The

similarity visualisations, graphics providing a visual representation of the similarity in two documents, have been found to reduce the time tutors need to spend on verifying plagiarism, aiding the human-led stages [5]. The only problem with the approach at present seems to be getting some level of assurance that the results are error free; that is they avoid false hits and missed pairs.

This paper reports on investigative work done when plagiarised submissions were added to an existing corpus of student submissions which was then subject to plagiarism analysis. The intention was to give some assurance that the system would avoid missed pairs. The paper also illustrates how VAST can be used to quickly verify that those pairs with high ranks are not false hits. It shows how the whole Four-Stage Plagiarism Detection Process can help tutors to easily find all of the plagiarism that their students are committing, thus providing an error free, effective, plagiarism detection process.

3 CONSOLIDATED SIMILARITY RANK

A corpus of 125 student submissions was collected from a Computing unit at the authors' institution and submitted to the similarity detector. The corpus gave a total of 7750 possible pairs, each of which had to be investigated.

Every pair of submissions was given a *similarity score* using 19 simple metrics that compared the counts of words, word pairs, word triples, single characters, character pairs, character triples, single sentences and twelve other related metrics, then ordinally ranked under each metric to give a *similarity rank*. The ranks were combined using a non-weighted linear function to give each pair of submissions a *consolidated similarity rank*. The highly ranked pairs contained a large of amount of similarity that may indicate plagiarism.

Although the highly ranked pairs could be manually verified to show that they did not contain any false hits, confirming that there were no missed pairs would be more difficult. Checking 7750 pairs by eye for similarities does not seem feasible and it has been shown that tutors finding similarity are very inconsistent, best classifying similarity using a descriptive ordinal scale of major, minor and no apparent similarity [3]. An alternative, less exhaustive, approach was to introduce plagiarised submissions into the corpus and demonstrate that the resulting plagiarised pairs had sufficiently high consolidated similarity ranks for further inspection.

Time constraints mean that tutors can usually only further investigate those pairs of submissions with major similarity. Deliberately plagiarised pairs would be most useful if they were representative of this classification. For the consolidated similarity ranks to be appropriate the newly plagiarised pairs should be ranked somewhere within the highly similar top portion of the *consolidated similarity ranking list*.

Four tutors were each given a different source submission to plagiarise in the manner of a student with an impending deadline, who was not willing to write the assignment from scratch and planned to fool a human marker. The submissions chosen were around the median length of 2234 words, but not verified for content or quality. Tutors were asked to record the methods used to plagiarise and the time taken plagiarising, as shown in Table 1

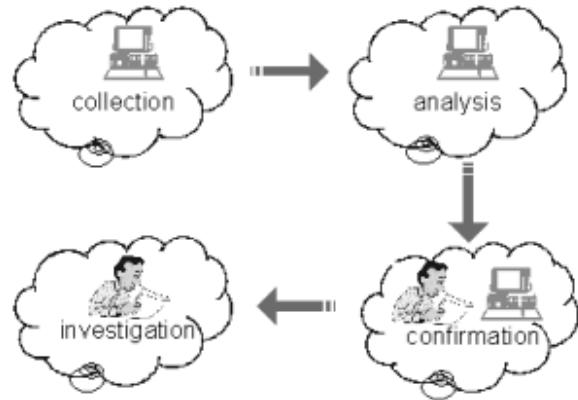


Figure 1. Four -stage plagiarism detection process

Table 1. Plagiarism methods

	Tutor 1	Tutor 2	Tutor 3	Tutor 4
Changed formatting	✓	✓	✓	✓
Changed headings	✓	✓	✓	✓
Changed header/footer	✓	✓	✓	✓
Rephrased sentences	✓	✓	✓	✓
Changed references		✓		✓
Added filler phrases				✓
Search and replace on key words				✓
Reordered sections		✓		
Removed sections		✓	✓	
Time taken	30 min	45 min	75 min	90 min

Table 2. Ranks for plagiarised pairs

	Tutor 1	Tutor 2	Tutor 3	Tutor 4
Consolidated	3	2	32	15
Words 1	4	1	3	5
Words 2	4	1	2	5
Words 3	4	1	2	8
Words 4	4	1	2	12
Words 5	4	1	2	23
Words 6	4	1	3	38
Words 7	4	1	3	43
Words 8	4	1	3	49
Words 9	4	1	3	47
Sentences	2	1	28	32
Characters 1	322	175	3523	4596
Characters 2	3	1	2381	4
Characters 3	24	1	3	9
Characters 4	6	1	3	7
Characters 5	4	1	3	6
Characters 6	4	1	2	5
Characters 7	4	1	2	5
Characters 8	4	1	2	5
Characters 9	4	1	2	5

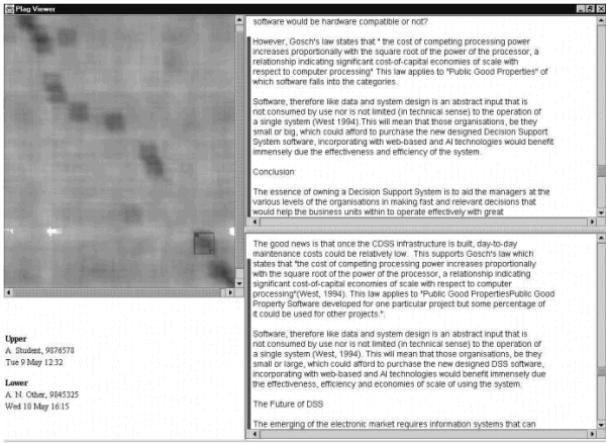


Figure 2. VAST used to investigate submissions

The extended corpus, now containing 129 submissions and 8256 pairs, was run through the similarity detector and the ranks for each source/copy pair, for each metric, are shown in Table 2, along with the consolidated rank for each. The ranks shown represent the placement of the pair within all 8256 possible. The metrics used included word chains of length one to nine, these (denoted Words 1 to Words 9), character chains of length one to nine (Characters 1 to Characters 9) and exactly matching sentences (Sentences).

The results were good for the submissions Tutors 1 and 2 plagiarised with least effort, ranking them in 3rd and 2nd place and would be investigated by a vigilant tutor. They were less good for the submissions altered by Tutors 3 and 4 and ranked in 32nd and 15th place. With tool support the 15th place submission might be investigated further. The plagiarised submission in 32nd place would likely not be noticed, unless it was found by chance whilst marking.

The possible missed pair is explained by the influential untypical metrics. The single character rank (Characters 1) seems inappropriate for all tutors and the pairs of characters rank (Characters 2) is disparate for Tutor 3. The metrics ranking all Tutors in the top five plagiarised submissions (Words 1-2, Characters 6-9) seem to be best.

The results suggest that the best consolidated ranking list would be generated from using only some of the metrics, possibly weighted. Some computationally intensive metrics, like Words 8 and 9 are so similar one of them redundant. Metrics that seem bad at finding one type of plagiarism may be good at finding another; excluding them might mean that plagiarism is missed. Consolidating using only the top ten ranks associated with a given pair of submissions seems to remove the bias that disparate metrics added and could be confirmed by investigating other corpora empirically. A good consolidated similarity ranking list might also need a short computation time.

4 SIMILARITY VISUALISATIONS

An effective similarity detector must have no missed pairs. Occasional false hits are much less problematic since, as Figure 1 shows, tutors have to manually verify highly ranked pairs of submissions. Investigating false hits can be time consuming and



Figure 3. Similarity visualisation for tutor 1

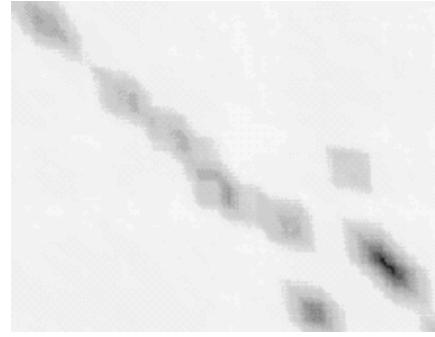


Figure 4. Similarity visualisation for tutor 2



Figure 5. Similarity visualisation for tutor 3



Figure 6. Similarity visualisation for tutor 4

can potentially discourage tutors from using automated plagiarism detection systems.

The Visualisation and Analysis of Similarity Tool (VAST) provides a graphical representation of the similarity between two submissions and has been shown to decrease the time tutors need to spend on confirmation and investigation [3]. Figure 2 shows VAST in action on two actual student submissions found to contain large amounts of intra-corporeal plagiarism.

The *similarity visualisation* is shown in the window on the top left. A rectangular area of the visualisation can be selected with the *marquee cursor*, shown towards the bottom right. The sections of the submissions associated with the size and location

of the marquee cursor are highlighted in the two text windows on the right. The fourth window contains status information.

VAST was used to verify that the plagiarised submissions produced by the tutors were similar to the source and confirm that they did actually represent plagiarism. The similarity visualisations generated by VAST for each tutor are shown in Figures 3 to 6. The source is shown along the x-axis and the copy along the y-axis. The difference between height and width represents the difference in length between the source and the copy. All the copies but one were smaller than the source, representing *contractive plagiarism*. The use of a lot of padding by Tutor 4 resulted in *expansive plagiarism*.

The intensity of each pixel is decided by comparing equal sized fragments from both the source and copy, for the longest, non-contiguous, list of words. The resulting darker areas of the visualisations, known as *similarity intersections*, are of interest as these indicate areas of distinct similarity standing out against the background noise of the English language.

The series of similarity intersections down the *primary diagonal* of each visualisation would leave tutors with little doubt that the submissions were plagiarised. They would then be able to use VAST to confirm the similarities and talk to any students concerned to determine which was the source of the plagiarism and which was the copy, as in most cases this is not obvious.

Of interest is how the similarity visualisations relate to the plagiarism methods used and the time spent. Figure 5 is the most intense of the visualisations, shown that whilst a lot of language has been removed, there is still a great deal in common between the source and copy. Figure 6 is the least intense, showing that a lot of padding has been added to the submission. The discontinuity to the end of the similarity intersection shows where the references have been rewritten.

Figure 4 is interesting, due to the discontinuities near the start and end of the primary diagonal. The initial gap shows where a new section about the Web was added to the copy. The similarity intersections off the diagonal near the end show how the concluding section of the source submission was moved earlier into the copy, with the previous end of one section of the source used as the new conclusion.

The similarity visualisations allow pairs of student submissions to be quickly checked for false hits, thus helping give an error free plagiarism detection process.

5 CONCLUSIONS

VAST has been shown to be a valuable addition to a tutor's anti-plagiarism arsenal and been very favourably received during hands-on demonstrations. The similarity intersections that appear in similarity visualisations provide a quick and easy way for a tutor to verify the locations of excessive similarity in both submissions. The visualisations are quick to scrutinise to discount false hits.

VAST is only of use when a previous process finds pairs for which similarity visualisations are worth generating. The inclusion of deliberate plagiarism into a corpus demonstrated that consolidated similarity ranks that avoid missed pairs are available, even if the prototype is not optimised and further metrics could be investigated. The combination of no missed pairs and no false hits gives an error free detection process.

The plagiarism strategies used varied significantly with some tutors trying to fool both a human marker and an automated similarity analyser. However there is little evidence that these strategies represent how students actually plagiarise. Such evidence might help in the selection of appropriate metrics, but students are unlikely to be forthcoming. Perhaps the only way to find such information would be to closely investigate plagiarised submissions with the help of VAST in order to discover just how the plagiarism had been carried out.

The similarity visualisations pointed conclusively to extensive plagiarism, but smaller amounts of plagiarism can be seen as smaller similarity intersections. Using such tools as part of a pro-active anti-plagiarism policy should be a target of every institution. Students must understand that plagiarism is both academically unacceptable. The experience of computing tutors in the operation and deployment of source code anti-plagiarism systems would be of value at institutional and wider levels.

REFERENCES

- [1] Barrie J. Presentation on iParadigms and plagiarism.org. At JISC Plagiarism Workshop, South Bank University, London (July 7, 2000).
- [2] Culwin F. & Lancaster T. A Descriptive Taxonomy of Student Plagiarism (2000). Awaiting publication, available from South Bank University, London.
- [3] Culwin F. & Lancaster T. Variability of Free-Text Similarity Assessment (2000). In preparation, draft available from South Bank University, London.
- [4] Culwin F. & Lancaster T. A Review of Electronic Services for Plagiarism Detection in Student Submissions (2000). *Proceedings of 1st LTSN-ICS conference*, Edinburgh August 2000, p 54-61.
- [5] Culwin F. & Lancaster T. Visualising Intra-Corporeal Plagiarism (2000). To be presented at 5th International Conference on Information Visualisation, University of London, July 2001.
- [6] Denhart A. The Web's Plagiarism Police (1999). Available WWW: <http://www.salon.com/tech/feature/1999/06/14/plagiarism/index.html>.
- [7] Franklin-Stokes A. & Newstead S. Undergraduate Cheating: Who Does What & Why? (1995) *Studies in Higher Education*, 20, 2, p159-172.
- [8] Gajadhar J. Issues in Plagiarism for the New Millennium: An Assessment Odyssey (1998). Available WWW: http://www.asee.org/prism/december/html/student_plagiari_sm_in_an_onlin.htm.
- [9] Ottenstien K.J. An Algorithmic Approach to the Detection and Prevention of Plagiarism (1976). SIGCSE Bulletin, Vol8 issue 5.
- [10] Ryan J. J. C. H. Student plagiarism in an online world (1998). Available WWW: http://www.asee.org/prism/december/html/student_plagiari_sm_in_an_onlin.htm.

Visualising Intra-Corporeal Plagiarism

Fintan Culwin

*School of Computing, Information Systems & Mathematics,
South Bank University, London, SE1 0AA, United Kingdom.*

fintan@sbu.ac.uk and lancaste@sbu.ac.uk

Thomas Lancaster

Abstract

This paper describes VAST, a prototype Visualisation and Analysis of Similarity Tool, which tutors can use to investigate student submissions for intra-corporeal plagiarism. VAST displays a pair of student submissions and a graphical representation of their similarity, allowing tutors to navigate directly to areas of potential plagiarism. It improves on the human eye approach by identifying similarities a tutor might otherwise miss and saving investigative time. VAST is demonstrated using noise-free synthetic texts and actual student submissions containing intra-corporeal plagiarism.

Associated ideas, including similarity visualisations, similarity intersections and the Four-Stage Plagiarism Detection Process are also introduced.

as *Web plagiarism* [1]. Students have also been known to copy from other reference works, an example of *extra-corporeal plagiarism*, where the *plagiarism source* is outside a *corpus*, or collected and related group, of student submissions. Web plagiarism is itself an example of extra-corporeal plagiarism. Students also copy from one another, a process known as *intra-corporeal plagiarism*.

A number of Web based services are able to detect Web plagiarism. Culwin and Lancaster investigated a number of such services and found that they were able to find incidences of Web plagiarism within submissions made to them, but the services varied greatly in the quality of replies and the amount of plagiarism detected [2]. A smaller investigation by Denhart had similar conclusions [3]. Such *Web-based plagiarism detection services* can also be used to detect intra-corporeal plagiarism, with each submission added to an internal database and compared with previous submissions, in much the same way that a submission can be compared with a database of Web pages.

In its most basic form, a *similarity analyser* takes a pair of student submissions, extracts from them some representative information, such as an ordered sequence of uncommon words, then compares the extracted information in order to compute a *similarity score*. This measure is then communicated to a user of the system, who can then examine closely those pairs of student submissions with a high similarity score and manually decide whether plagiarism has occurred.

Figure 1 shows the *Four-Stage Plagiarism Detection Process* through which a corpus of student submissions can be collected and any plagiarism found and verified. It is primarily designed to be used in conjunction with automated systems.

The first stage is *corpus collection*. With the currently available Computing and Information Technology infrastructure, documents can easily be collected electronically and prepared for processing.

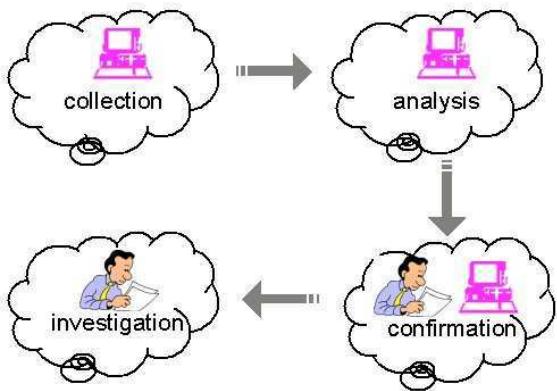


Figure 1. Four-stage plagiarism detection process

Next, the corpus is put through *analysis*, a stage roughly similar to the similarity analyser described above. This paper deals mainly with intra-corporeal plagiarism, so it is the finding of similarities within a corpus that will be described here. In contrast, the Web-based plagiarism detection services follow a similar pattern, but mainly look for Web plagiarism. It is relatively straightforward, if computationally intensive, to find documents within a corpus that are similar. Such similarities might be an overly large shared vocabulary, or a series of sections presented in the same order throughout two or more student submissions. Measures of similarity can be obtained using a number of simple metrics, ranging from word frequency distribution to document structure. Such measures can be useful individually, but are most usually combined to give a single numeric measure of similarity to a pair of student submissions; this measure is known as a *similarity score*.

The list of pairs of student submissions can now be sorted on this similarity score. Those with a high ranking are instantly suspect, although at this stage there is no evidence that plagiarism has occurred. Each highly ranked pair of student submissions now goes through a stage of *confirmation*. This is where a human manually verifies that each suspect pair does not represent a *false hit*, which is a pair that is ranked highly, but contains little similarity. There is no related method to confirm the non-existence of *missed pairs*, which are pairs ranked low, but which contain similarities, other than going through all results by hand. This is unlikely to be possible, since just 100 student submissions generate a corresponding 4950 possible pairings. The number of possible pairings increases at an exponential rate compared to the number of student submissions to be checked.

The final stage of the Four-Stage Plagiarism Detection Process is *investigation*. This is where a human manually checks those pairs which are highly ranked, but confirmed to not be false hits, in order to assess whether the

similarity constitutes plagiarism, or acceptable similarity, such as a properly cited common quotation. With the evidence collected, appropriate measures should then be taken.

The process that has been tested here seems to work relatively well. The collection stage is routine. Metrics have been found that rank pairs highly with high levels of similarity, whilst being found to usually avoid false hits. Investigations have revealed that the results of applying the metrics have been used successfully to find intra-corporeal plagiarism.

The current bottleneck in this system is in the human led stages, confirmation and investigation. The manual confirmation and investigation of suspect pairs increases the work for the tutor in charge of the unit. Previously many such pairs would simply have been missed, since the technology was not present to detect intra-corporeal plagiarism and the pressures of academia work against the likelihood of tutors discovering plagiarism by eye.

Whilst a computer cannot be used alone to aid a tutor with verification, it can certainly be used to assist in the process. Studies have shown that users are more productive when using tools that employ visual methods, such as presenting data to users in a graphical form. Representing data by graphical means is not a new concept; presenting data in terms of graphs and charts has long since been commonplace since it is more easily interpreted by humans. Tutors using a system to investigate similarities need also to know which sections of a pair of documents are most.

Hence there is a need for a tool to allow tutors to investigate pairs of student submissions, which is useable by non-computer specialists and makes it possible to navigate documents to those places which contain most similarity. This paper covers the development of such a prototype Visualisation and Analysis of Similarity Tool, known forthwith as VAST, showing how it has been successfully used to investigate the presence of intra-corporeal plagiarism in student submissions in the authors' institution.

2. Fragmentary Granularity Matrix

A similarity analyser takes in two student submissions and returns a similarity score, which a tutor can use to decide whether manually investigating the submissions for possible plagiarism is worth the time involved. Usually the metrics used to compute the similarity score are applied at *gross granularity*. This means that a representation of the whole of the first student submission is compared with a representation of the whole of the second student submission. Such a comparison generates a single similarity score for each metric applied; these

scores are then combined in some fashion to give the similarity score returned to the user.

In order to implement a visualisation system, a single numeric similarity score is not sufficient. What is needed is a method by which a number of such similarity scores can be returned for a pair of student submissions, providing a piecemeal representation of its similarity.

This thought process has led to a visualisation system, where student submissions that rank highly under gross granularity are further compared at *fragmentary granularity*. In this case each submission is seen not only as a whole, but also as a large number of overlapping fragments of continuous words. Consider a student submission. The first *fragment* is obtained from it simply by taking words from the start, up to a given *fragment length*. A second fragment could be obtained by taking the student submission, removing the first word and then repeating the fragmenting process. Words next to one another in a fragment were always next to one another in the original student submission.

In most cases it is not necessary to take every possible fragment from a student submission, since this would give too much detail for useful visualisation. Instead only some of the fragments are removed; for visualisation purposes, these must have the same spacing between them and cover the entirety of the student submission. For this a *fragment gap* is needed, this is the value denoting how many words in the student submission there are between the first word in subsequent fragments. For a fragment gap of one, every possible fragment would be removed from the student submission. For a fragment gap of two, every alternate fragment would be removed. For a fragment gap of three, every third fragment would be removed and so on.

A representation of each fragment from the first student submission can now be compared with a representation of each fragment from the second student submission. This gives a *fragmentary interception*, a representation of the contents of both fragments, providing for it a *fragmentary interception similarity score*, a similarity score localised to the fragmentary interception. A tutor then now only needs to concentrate human examination on those fragments which, when compared, provide a higher fragmentary interception similarity score than would usually be expected.

It would be very rare to get a fragmentary interception similarity score of zero. This is due to the background effects of *noise*. Consider a simple *word count metric* that simply counts the number of words common to a fragmentary interception. There is a certain background distribution to student writing that means that certain words will appear over and over again, regardless of whom is writing. In English, these are usually *function*

words, the filler words of the language, such as ‘a’, ‘by’, ‘from’ or ‘to’. The most common word in the English language is ‘the’, which alone usually accounts for around 10% of any fragment, and hence a substantial part of any noise from a fragmentary interception similarity score.

An example may make the process of generating fragmentary interception similarity scores clearer. Table 1 shows a case of *expansive plagiarism*, plagiarism with word additions and perhaps minor rearrangements. Table 1 shows two documents, in this case phrases, the second a slightly expanded version of the first. For the purposes of tracing, the first phrase is labelled S1 and the second phrase is labelled S2.

Table 1. Expansive plagiarism phrases

S1 = “the cat sat on the mat”

| S2 = “the black cat sat on the bathroom mat” |

A simple count reveals that 100% (6 of 6) of the words in S1 are also in S2. In the other direction 75% (6 of 8) of the words in S2 are also in S1. Under this simple word count metric, the ordering of the words is unimportant, but the word frequency is. Averaging these values would give a similarity score between S1 and S2 of 87.5. Using a different metric might give a different similarity score.

In Table 2, the phrases from S1 and S2 have been split into fragments. The splitting process, in this example, used a fragment size of four words and a fragment gap of one word. The fragments have been labelled according to which phrase they were removed from and their position within the phrase. The first two characters in the new label denote the source phrase, the third character denotes the position of the first word of the fragment within the source phrase.

Table 2: Expansive plagiarism fragments

S11 = “the cat sat on”	S21 = “the black cat sat”
S12 = “cat sat on the”	S22 = “black cat sat on”
S13 = “sat on the mat”	S23 = “cat sat on the”
	S24 = “sat on the bathroom”
	S25 = “on the bathroom mat”

Table 3 is derived by comparing every fragment from S1 with every fragment from S2, using the word count metric. There are a total of fifteen comparisons to make to generate the fragmentary intercept similarity scores, comparing every fragment from S1 against every fragment from S2. Comparing S11 (“the cat sat on”) and S21 (“the black cat sat”) gives three words in common between the two, which is the number given in parentheses. This value is scaled between 0 and 100, by simply multiplying by 4, to give the main value in the cell, the fragmentary

intercept similarity score of 75. Comparing S21 and S11 under this metric would give the same value, so there is no need to calculate a backwards score and average the results, but under some metrics, for instance where word positioning is important, this might be needed. The rest of the table, known as a *fragmentary intercept similarity score matrix* is determined in a similar fashion.

Table 3: Fragmentary Intercept Similarity Score Matrix

	S11	S12	S13
S21	75 (3)	75 (3)	50 (2)
S22	75 (3)	75 (3)	50 (2)
S23	100 (4)	100 (4)	75 (3)
S24	75 (3)	75 (3)	75 (3)
S25	50 (2)	50 (2)	75 (3)

Due to the nature of the comparisons of adjacent cells, the count of similar words never varies by more than one and in many cases is identical. Since this example is necessarily simple, the similarity score is high in every case. The two cases where it is 100 are of particular interest. The fragments S12 and S23 are identical and show a place in the two documents that should be of great interest to a tutor. The fragments S11 and S23 are also identical under this metric, where word ordering is unimportant. This demonstrates that there is always a need for tutors to manually validate all reported similarities.

3. Similarity Visualisations

The similarity comparison using fragmentary granularity allows the pinpointing of similar areas of two texts in a much better way than a simple examination by tutors. But it is still not the basis of a viable tool, since tutors are expected to hunt through a matrix of numbers – a process which can be daunting and time consuming in itself. To have a usable system, the similarity score matrix must be presented in a more eye-friendly form.

The table can easily be presented as an image, where colour information can be used to display the extent of similarity within fragments. Each cell within the fragmentary intercept similarity score matrix can be mapped onto a grayscale value, with a fragmentary similarity score of 0 corresponding to white, and a score of 100 corresponding to black, and plotted onto a *similarity visualisation*. The joining together of many pixels makes some areas prominent; these areas are known as *similarity intersections*.

Figure 2 shows the similarity visualisation generated by comparing two synthetic documents. The synthetic documents are each 2000 words long and contain no

similarity, apart from a 200 word stretch in the middle of both documents, where every word is common to both. This represents a short section of text copied from one document to the other. The uniqueness of the rest of the documents eliminates the problem of noise, since the usual responsible background word distribution is not preserved by the synthetic documents. The similarity visualisation has been computed with using a fragment size of 200 words and a fragment gap of 10 words, to create an image 181 pixels by 181 pixels. The similarity visualisation is shown both at its usual size and zoomed in on at the similarity intersection.

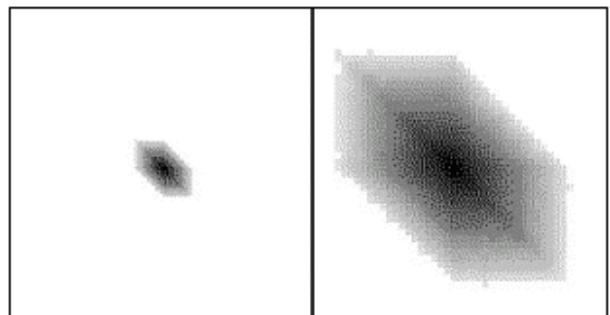


Figure 2. Synthetic similarity visualisations – 200 words

The similarity intersection features an intensely black area in the centre, where an exact match of 200 words is found. Radiating outwards are ever-lighter shades, until the edge pixels of the hexagon, which are imaged from fragmentary interceptions sharing only ten words in common.

The positioning of the similarity within the documents is also important. In the similarity visualisation, the first document is positioned along the x-axis and the second document is positioned along the y-axis. The origin point for both is in the top-left corner. The similarity intersection is situated approximately half way through the first document and half way through the second document two. It is also situated on the *primary diagonal*, the diagonal running from the origin, at the top-left, to the bottom-right.

The system cannot tell us which document represents the source of the plagiarism. The similarities might be through the writer of the first document copying from the second, or vice-versa. Alternatively the writers might have collaborated on both submissions, or both copied from the same source. A similarity visualisation generated by comparing the second document with the first document would generate the same graphic, only mirrored along the primary diagonal. In this case, both similarity visualisations would be identical.

Figure 3 shows a number of other similarity visualisations generated from synthetic documents. All are again 2000 words long, using a fragment size of 200 words and a fragment gap of 10 words.

Similarity visualisation A shows the unlikely situation of the two documents being identical. The similarity intersection runs down the entirety of the primary diagonal. The centre of the similarity interception shows the series of 200 word matches. The edge of it shows where just ten words are identical.

Similarity visualisation B shows the case where 400 words are identical, under much the same pattern as before. Again the similarity is positioned in the centre of both documents.

Similarity visualisation C features two similarity interceptions. Each represents a set of words identical to each document, but not to the other similarity interception. The common words are situated in the same place in both documents, one near the beginning of both, the other near the end of both. Here the central extent of the similarity interceptions is lighter than before, since at most 100 words match out of 200, demonstrating the need for careful selection of fragment sizes. However, in practice this would be sufficiently different from the background colour to stand out in most situations.

Similarity visualisation D shows a theoretical first document, along the x-axis, where the same 400 words are repeated five times, compared with a second document, along the y-axis containing these 400 words just once. There are five similarity interceptions, one for each set of matching words, in much the same shape as the similarity interception in similarity visualisation B. The leftmost and rightmost similarity interceptions start and end abruptly where the first document starts and ends. Similarity visualisation D demonstrates that one can expect to find extensive similarity only once at any given point on the x and y-axes. Later examples will show extensive plagiarism seems to occur most often close to the primary diagonal.

A smaller fragment size would decrease the width of each of these similarity intersections, since the area through which fragmentary intersections continue to contain some similarity would decrease. It would also serve to increase the intensity of the similarity intersections in C, since the maximum proportion of similarity within a fragment would increase and hence, so would some of the fragmentary similarity scores.

A smaller fragment gap would increase the number of fragments to be compared and correspondingly, the size of the similarity visualisation. Each similarity intersection would see a corresponding increase in size. On occasions an increase might make other similarity intersections

visible that would be too small to see with larger fragment gaps.

The discussion of the visualisation of synthetic documents prepares for the interpretation of a similarity visualisation derived from a pair of true-life documents, which have been judged by humans to be examples of intra-corporeal plagiarism, with an attempt at disguise. Figure 4 shows a similarity graphic for two such documents. The first document, along the x-axis, is 2728 words long. The larger second document, along the y-axis, is 3341 words long. As in the synthetic cases, the similarity visualisation is generated with a simple word count metric, a fragment size of 200 words and a fragment gap of ten words. The size of the resulting similarity visualisation is 253 pixels by 315 pixels; the slight discrepancy from what would be expected caused by the intricacies in the parser used to produce the similarity graphic.

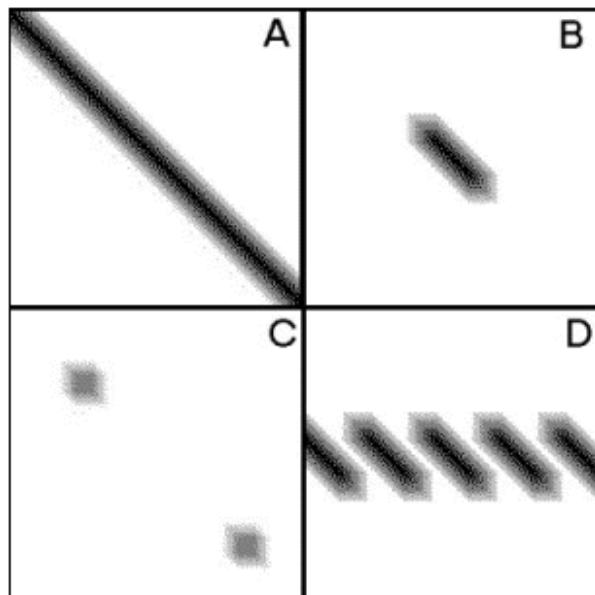


Figure 3. More synthetic similarity visualisations

The first thing apparent compared to the synthetic graphics, is the background noise. However the similarity interceptions, of which there are at least ten, are still clear to see. As expected, the similarity intersections appear close to the primary diagonal, where the second document seems to be a somewhat expanded version of the first. In particular, similarities in the introduction, conclusion, references and other sections of the text make this an interesting case. Although many words are changed here and there this does not fool a computerised checker, although it may get past a human marker, since the fragment size takes this into account. The similarity interception identified as A on Figure 4 showed, on investigation, two short sentences in each document that

were identical. This suggests that the fragment size and fragment gap values were appropriate.

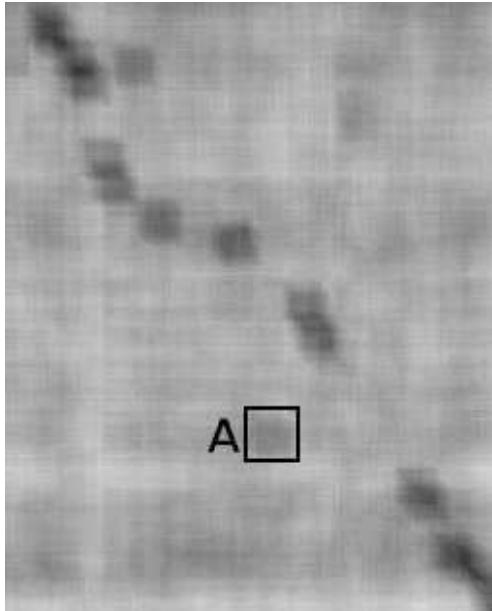


Figure 4. Intra-corporeal plagiarism - word count

The similarity visualisation generated from one further fragmentary metric is worth showing. Figure 5 shows a refinement that reduces the noise and shapes the highlights. It shows the same two student submissions as Figure 4, with the same fragment size and fragment gap, but a different metric. The fragmentary metric used is more computationally intensive than the last, but is responsible for a clearer similarity visualisation. The colouring is derived from the *word sequence metric* that takes the longest common ordered sequence of words within the fragment. The previous metric did not take such ordering into account. That is to say that the sequences do not have to be contiguous, but must contain the ordering. For example, the longest common ordered sequence of “apples bananas cherries damsons” and “apples apples bananas bananas damsons damsons” is “apples bananas damsons” of length 3 words.

In this case the similarity interceptions highlight much the same areas for inspection as before, but the noise level is much lower allowing the similarity interception identified as A to be much more clearly seen. It might be possible to use graphical techniques to filter the noise from the graphic still further, however this is sufficient to demonstrate the principal behind the graphics.

VAST uses a combination of the fragmentary intercept similarity score matrices generated using the word count and word sequence metrics, combining them with colour information to generate a similarity visualisation. This will be seen in Figure 6.

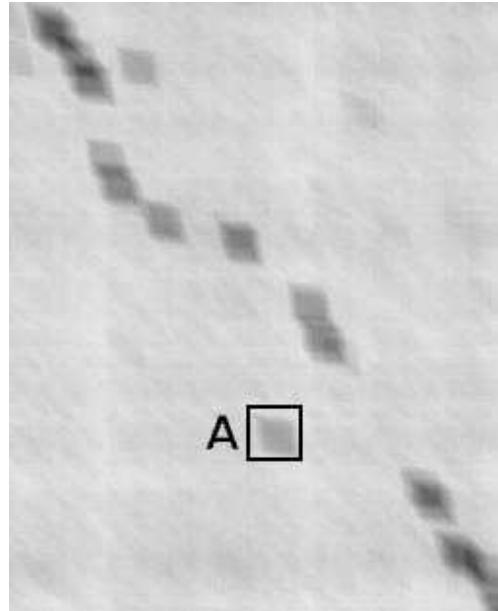


Figure 5: Intra-corporeal plagiarism – word sequences

4. Visualisation and Analysis of Similarity Tool

The similarity visualisations in themselves are very useful as part of the process of verifying if plagiarism has occurred. However there is no easy way for a tutor to find the parts of a pair of documents that a given similarity interception represents. This is a problem ideally suited to computer assistance, with a prototype navigation system available to guide the tutor around the documents.

Figure 6 shows the Visualisation and Analysis of Similarity Tool (VAST) being used to navigate around the two student submissions suspected of intra-corporeal plagiarism in the previous section. The user highlights an area of the similarity visualisation that is of interest and the corresponding sections of the text are shown in the windows on the right. The similarity visualisation can be scrolled about if it is bigger than the window it is in. The section of text by the area selected is marked with a red side bar highlighted, so that it can be scrolled through if it extends beyond the viewable window. The user can then carry out the final two stages of the Four-Stage Plagiarism Detection process, confirming that each similarity interception does not represent a false hit and investigating whether the similar sections are plagiarism, or if they contain legally cited materials.



Figure 6. Visualisation and analysis of similarity tool

The user selects a similarity interception using the *similarity interception selector*, the red rectangle in the navigation window. Users can click and drag the small rectangle in the top left corner of the similarity interception selector in order to move it around the similarity visualisation. They can alter the size of the selector by clicking and dragging the rectangle in the bottom right corner of it in order to view the text represented by larger or smaller similarity interceptions.

The student submissions used to generate the similarity visualisation in Figure 6 are the same as in Figures 4 and 5. Both the word count and word sequence metrics are used and displayed in the single similarity visualisation, using colour information to display both. Notably the red and green components of the image are set using the word count metric and the blue component set using the word sequence metric. The coloured graphic seems to be more effective in this circumstance.

The section of text highlighted in VAST shows an example of intra-corporeal plagiarism with disguise and restructuring. The essence of the content is essentially the same, with the same quotes, references and points made in the same order. However the paragraphs have been broken in different places, abbreviations replaced with full wording and section headings altered. There has also been some simple thesaurising, such as replacing 'large' with 'big'.

The plagiarism, although disguised, looks relatively straightforward to find in this case. For a tutor struggling only with two paper submissions it is much harder. Since the length of the submissions is different, the similarity intersections will not be on the same page and a tutor will have a needle in a haystack like job to find them. VAST is a big help in the necessary investigation, with the suspect similarity interceptions easily highlighted by a user for examination.

5. Extra-Corporeal Extension

The previous section demonstrated how VAST can aid with the verification of intra-corporeal plagiarism. It can also be very used to verify extra-corporeal plagiarism, where a source can be determined. This example represents a chapter of a final-year student project where the tutor marking it noticed stylistic differences to the rest of the document and with the aid of a Web search engine was able to trace the chapter to a Web source.

Figure 7 shows the similarity visualisation for the suspect chapter of the project, plotted against the Web source.

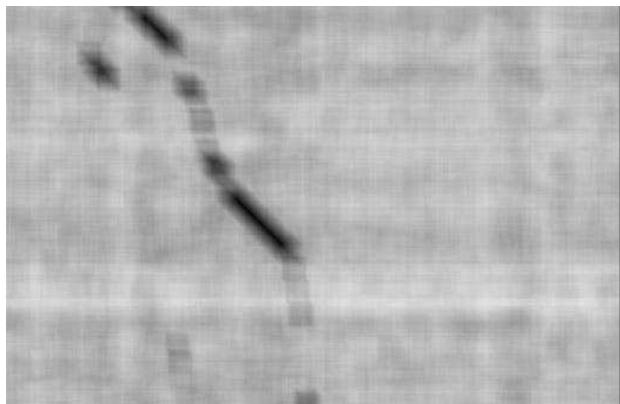


Figure 7. Extra-corporeal plagiarism

The 5400 word chapter is plotted along the x-axis, the 3500 word Web source along the y-axis. Projecting the similarity intercepts down onto the x-axis reveals that around one third of the chapter is made up of similar material and the intensity of the areas reveals that the material has been copied word for word. Extracting the similarity intercepts across onto the y-axis reveals that almost the entire Web source has been used in the chapter at some point. However some parts, primarily towards the end of the Web source, have had new material introduced so that the apparent extent of the similarity is reduced. Also of interest is the first section of plagiarism within the chapter, which shows that the Web source has been slightly re-arranged. The most interesting section is the central similarity intercept of around 1000 words, which is copied verbatim; both its intensity and its positioning parallel to the primary diagonal illustrate this.

The substantial similarity shown in the similarity visualisation in Figure 7 was later determined to be extra-corporeal plagiarism.

6. User Reviews

VAST is currently being developed as a proof of concept prototype. As such much of the functionality that would be needed for a production version has not even been fully specified. Despite this it has been shown to a number of academic tutors from a number of different

institutions, including a demonstration at the 2000 ITiCSE conference in Helsinki.

All reactions to the tool have been favorable and its utility has been readily, though not immediately, apparent. A few prospective users were simply told that it was a tool to assist with plagiarism verification and invited to interact with it. In these cases neither the interpretation of the image nor the relationship with the text panes was self-evident and instruction had to be given before effective use could be made of it.

The majority of users have required a short verbal introduction to what the tool is intended for, what the image indicates and how the rectangular highlight relates to the text panes. Following this introduction all trial users have been able to use the image and the highlighter to successfully navigate around the documents, verifying the existence of plagiarised parts. This tends to suggest that although the tool could not be used on a 'walk up' basis only a minimal amount of instruction would be required. All of the tutors that have used the tool have indicated that they think it would be of use to them in the validation of plagiarism and most have expressed interest in having access to it when it is available. One South Bank tutor, who had assisted with the collection of a corpus of documents, was invited to comment upon the prototype at an early stage and was somewhat disappointed that it would not be immediately available.

It is intended that a small-scale study will be conducted in the near future with tutors being required to annotate pairs of plagiarised documents, one group purely by eye and one with access to the tool. The tutors will then be asked to make a simulated decision to exonerate, warn or start a formal investigation of the pair of students concerned. An informal observation has indicated that moving the relative locations of plagiarised sections makes it more difficult for a tutor to detect them. If this is the case then VAST should prove to be of significant benefit.

7. Conclusion

Even if there are methods to easily detect plagiarism in free-text submissions, the process of human verification is still costly, in terms of time and effort expended. The visualisation method presented here has many advantages over relying on the human eye alone. Primarily it provides a graphical representation of two texts, which may be source and copy, or the result of collaboration, where similar areas are visibly highlighted for human verification. The associated tool VAST allows those areas to be easily displayed on screen so that a human judgement can be made.

The success of this method depends on choosing appropriate values for the fragment size and fragment gap, so as not to overwhelm the tutor with too much information, or to miss any similarity in the submissions caused by examining fragments that are either too small or too large. The ideal sizes may vary from submission to submission, or even within different sections of a submission. This process of choosing correct values could be machine assisted and is something that requires further investigation. The current values have been demonstrated to find and display fragmentary intersections to a VAST user.

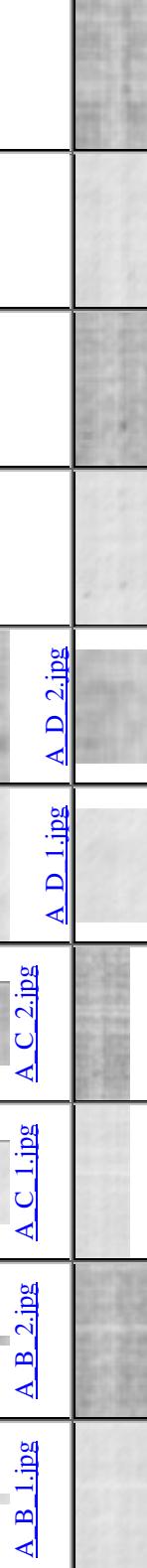
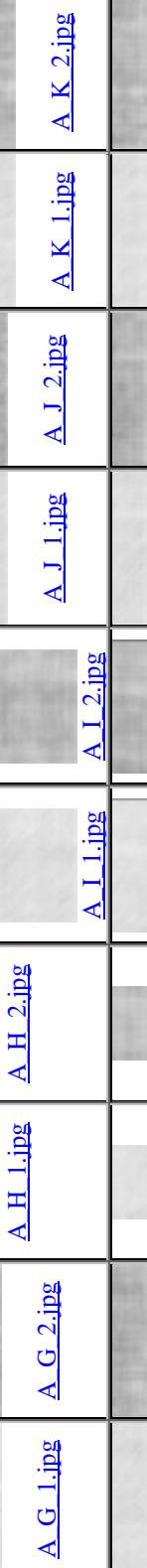
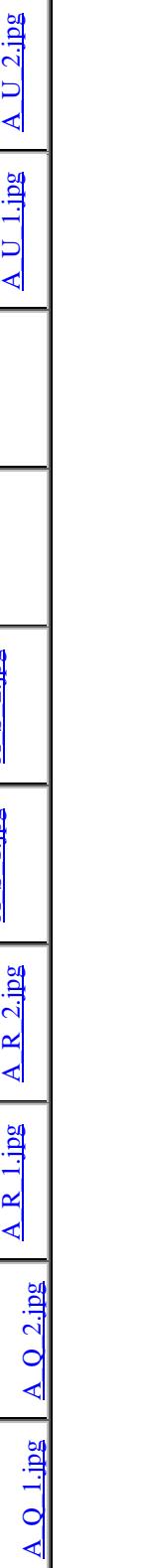
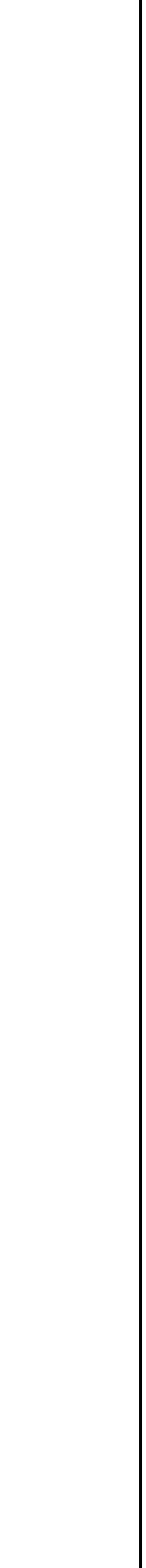
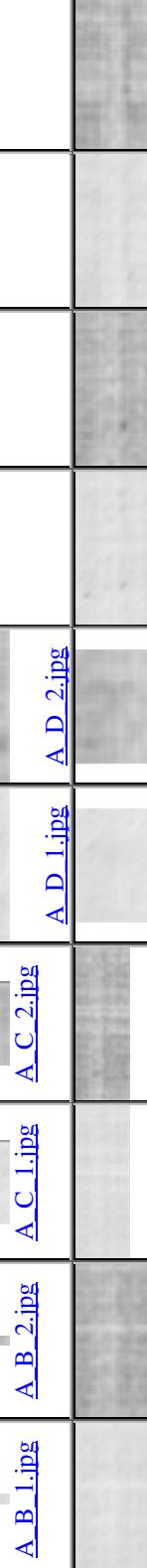
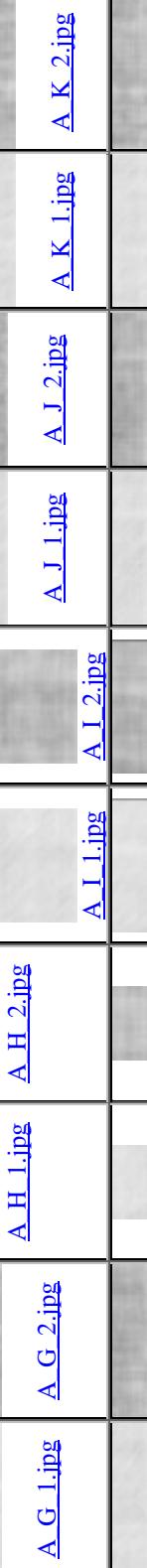
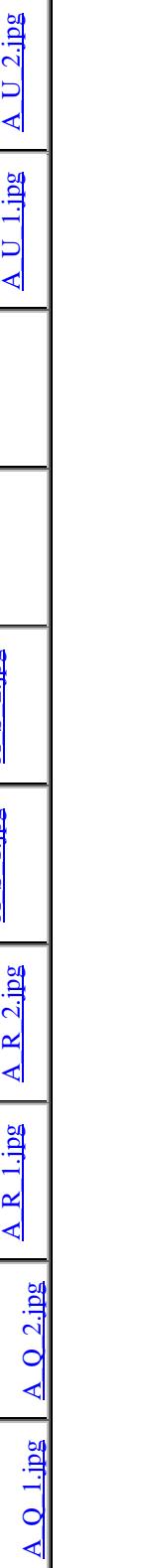
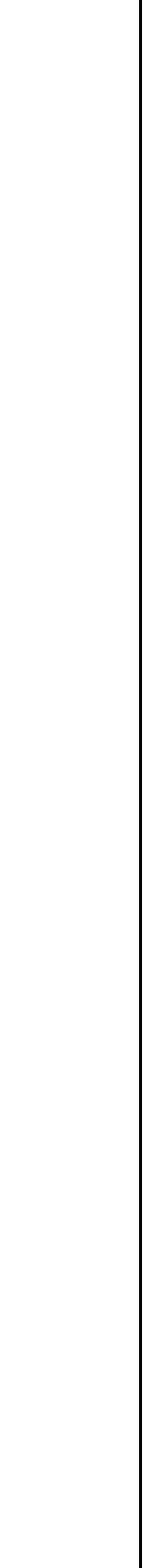
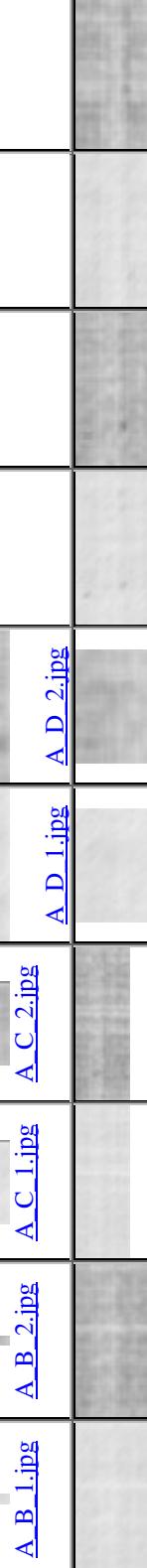
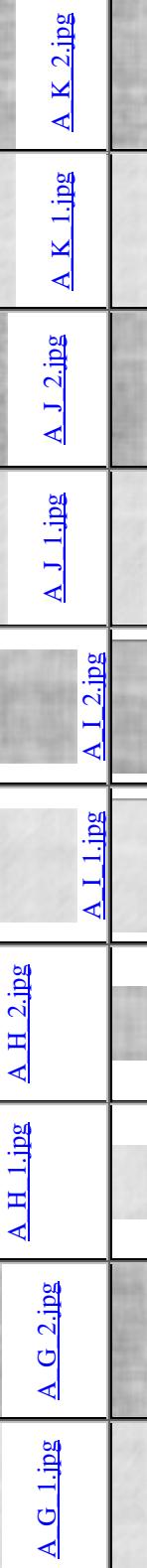
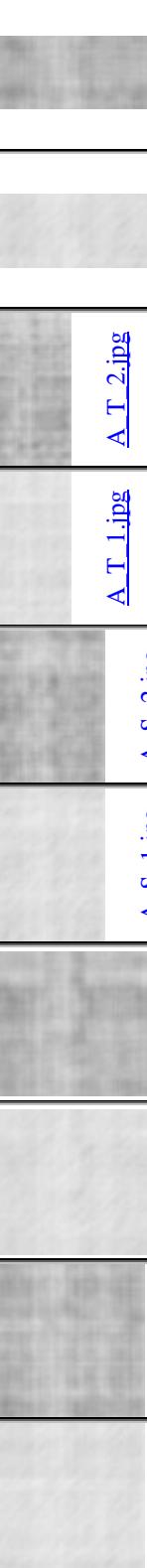
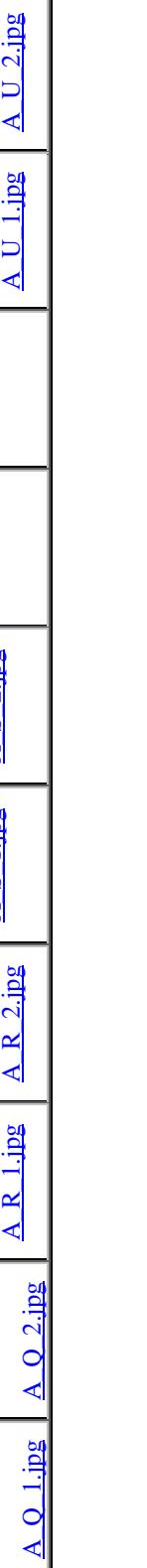
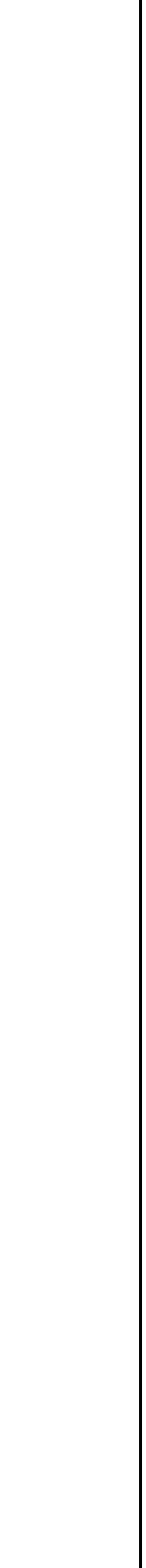
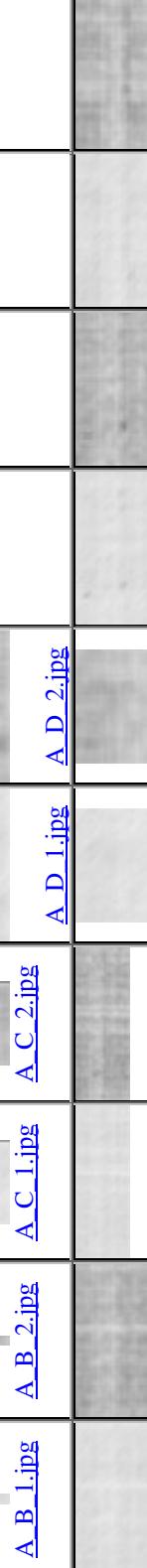
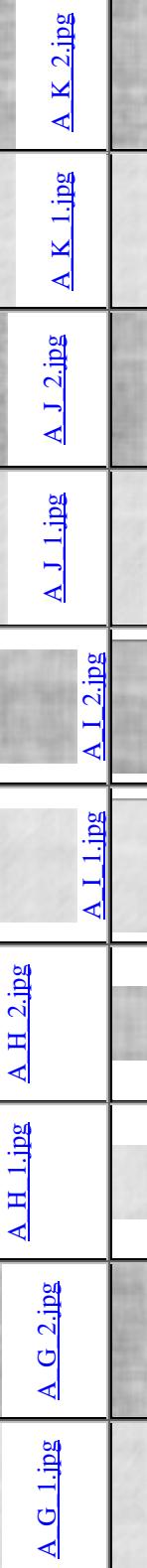
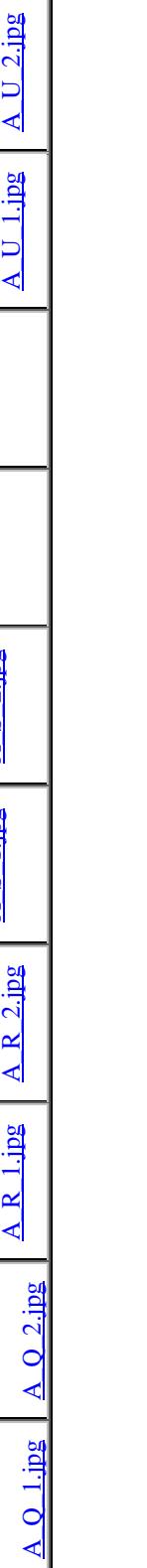
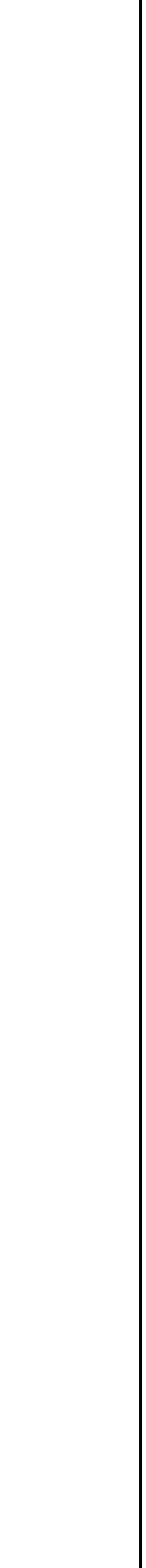
One problem outside the current scope of VAST is that of *multiply sourced plagiarism*. This is where different sections of a student submission have been plagiarised from different sources, for example, one section from the Web and one section from another student. A method is needed to display all possible similarity visualisations for a given student submission, possibly on a single similarity visualisation. The prototype version of VAST also needs a front end to allow tutors to select the similarity visualisation that they wish to examine.

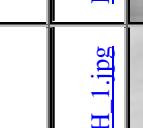
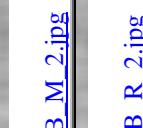
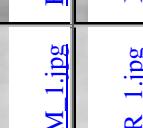
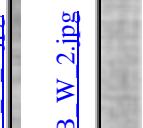
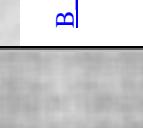
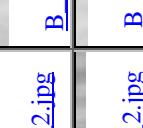
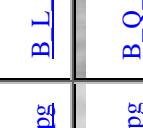
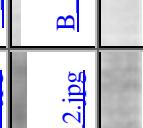
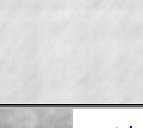
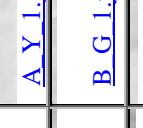
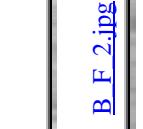
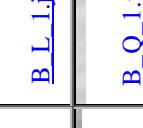
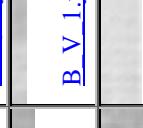
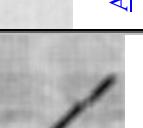
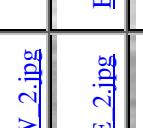
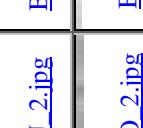
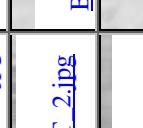
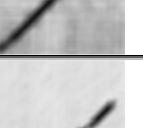
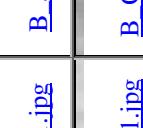
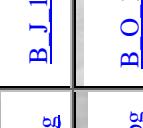
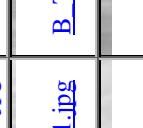
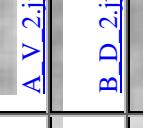
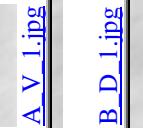
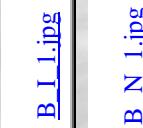
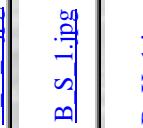
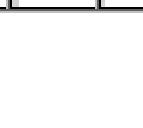
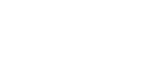
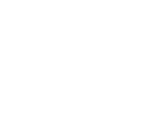
The system can be combined with a pro-active plagiarism policy, where plagiarism is not only taken seriously when found, but also actively sought out, say by routinely sending all student submissions to a plagiarism detection service. This allows a tutor to quickly illustrate plagiarism to a suspected student, or the part of an academic institution responsible for determining the penalty for such plagiarism. The tool at present is most suitable for detecting and verifying intra-corporeal plagiarism, but could be used to verify extra-corporeal plagiarism, where the source can be determined by other means. There are many services available to detect Web plagiarism. The tool could be refined and used in conjunction with them to find more than just intra-corporeal plagiarism.

8. References

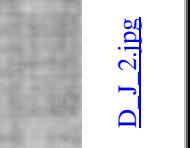
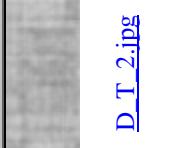
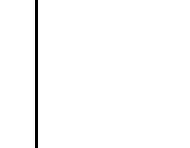
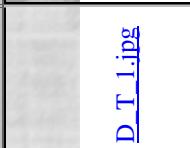
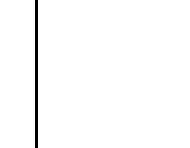
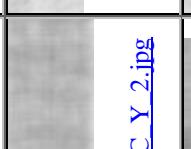
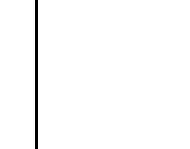
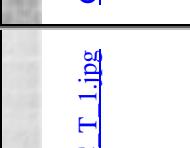
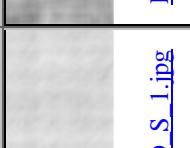
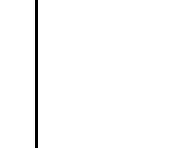
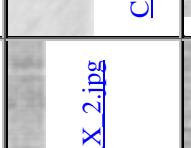
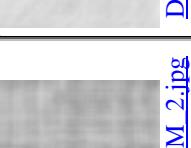
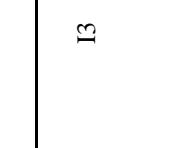
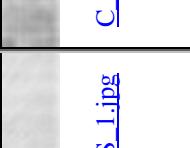
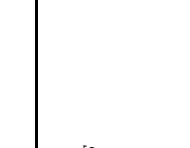
- [1] Culwin F. & Lancaster T., *A Descriptive Taxonomy of Student Plagiarism*. Awaiting publication, available from South Bank University, London (2000).
- [2] Culwin F. & Lancaster T., *A Review of Electronic Services for Plagiarism Detecting in Student Submissions*. 8th Annual Conference on the Teaching of Computing -organised by the LTSN Centre for Information and Computer Sciences (2000).
- [3] Denhart A., *The Web's Plagiarism Police*. Available at <http://www.salon.com/tech/feature/1999/06/14/plagiarism/index.html> (1999).
- [4] Franklin-Stokes A. & Newstead S., *Undergraduate Cheating: Who Does What & Why?* Studies in Higher Education, 20, 2, p159-172 (1995).

Appendix I: Visualisations for All Pairs in the High Similarity Corpus

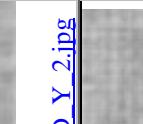
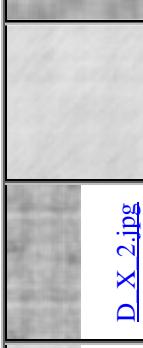
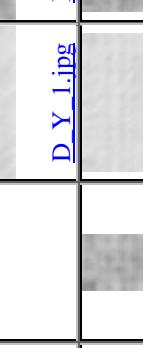
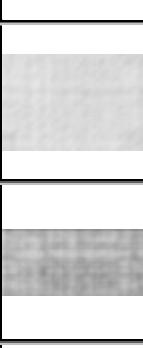
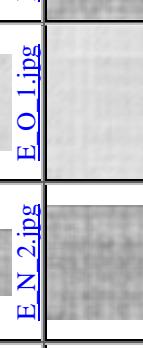
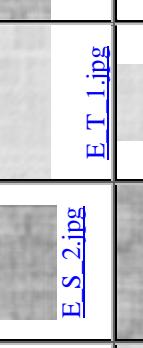
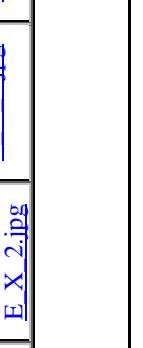
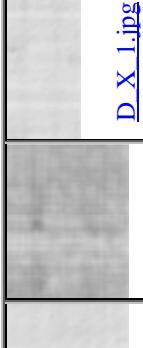
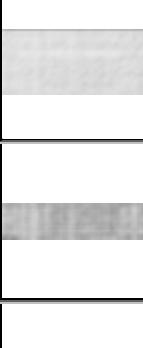
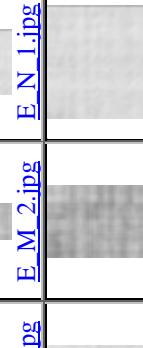
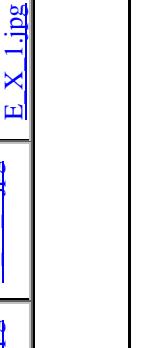
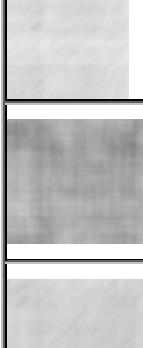
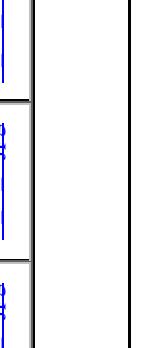
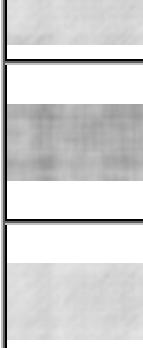
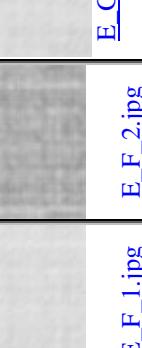
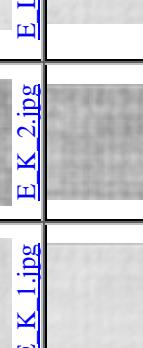
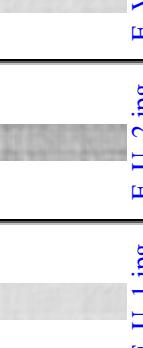
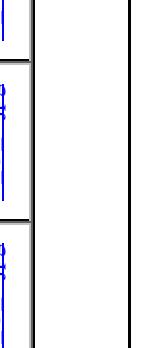
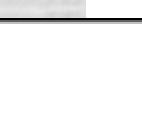
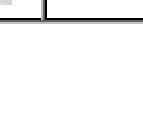
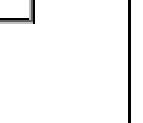
									
									
									
									

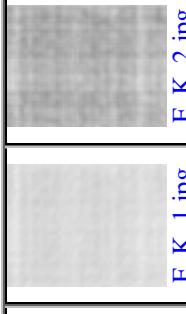
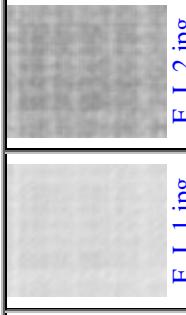
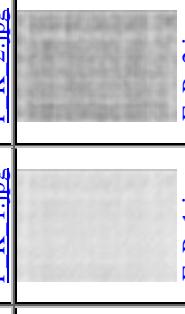
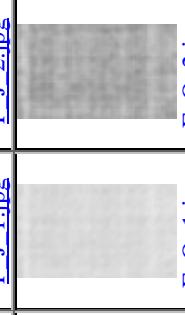
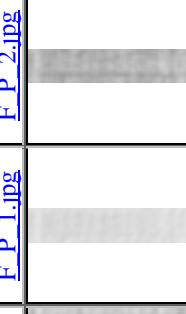
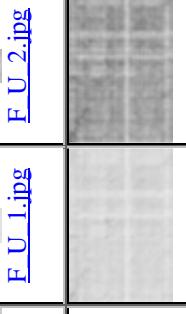
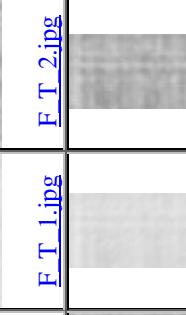
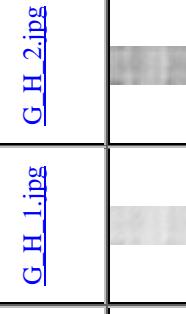
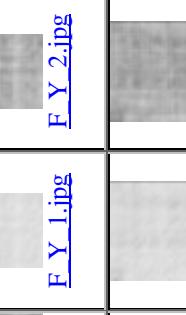
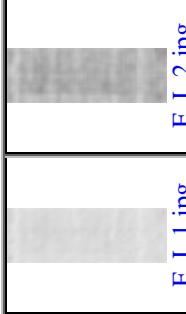
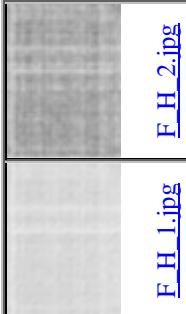
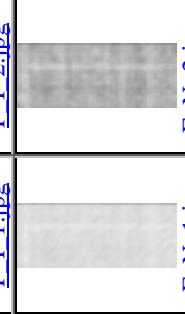
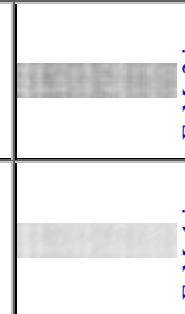
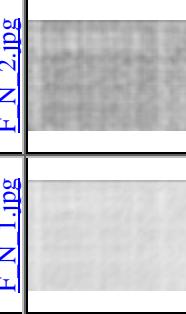
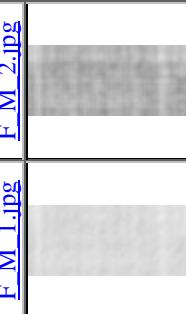
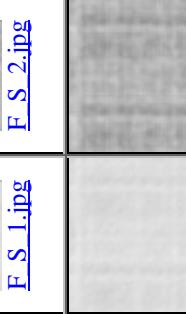
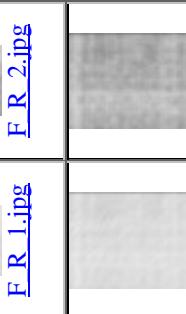
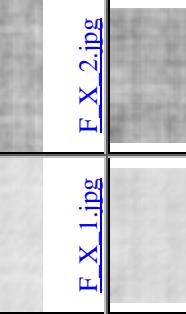
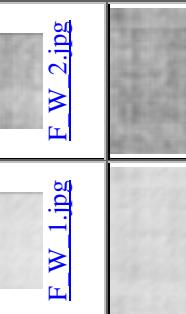
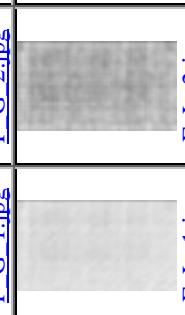
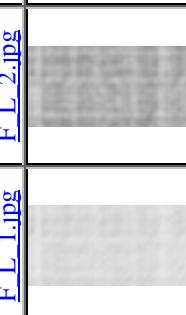
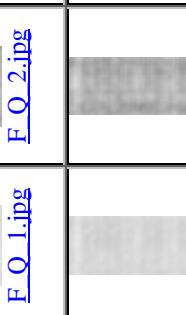
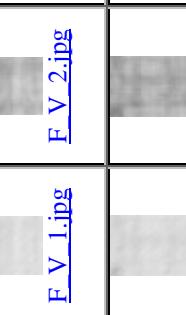
Appendix I: Visualisations for high similarity corpus

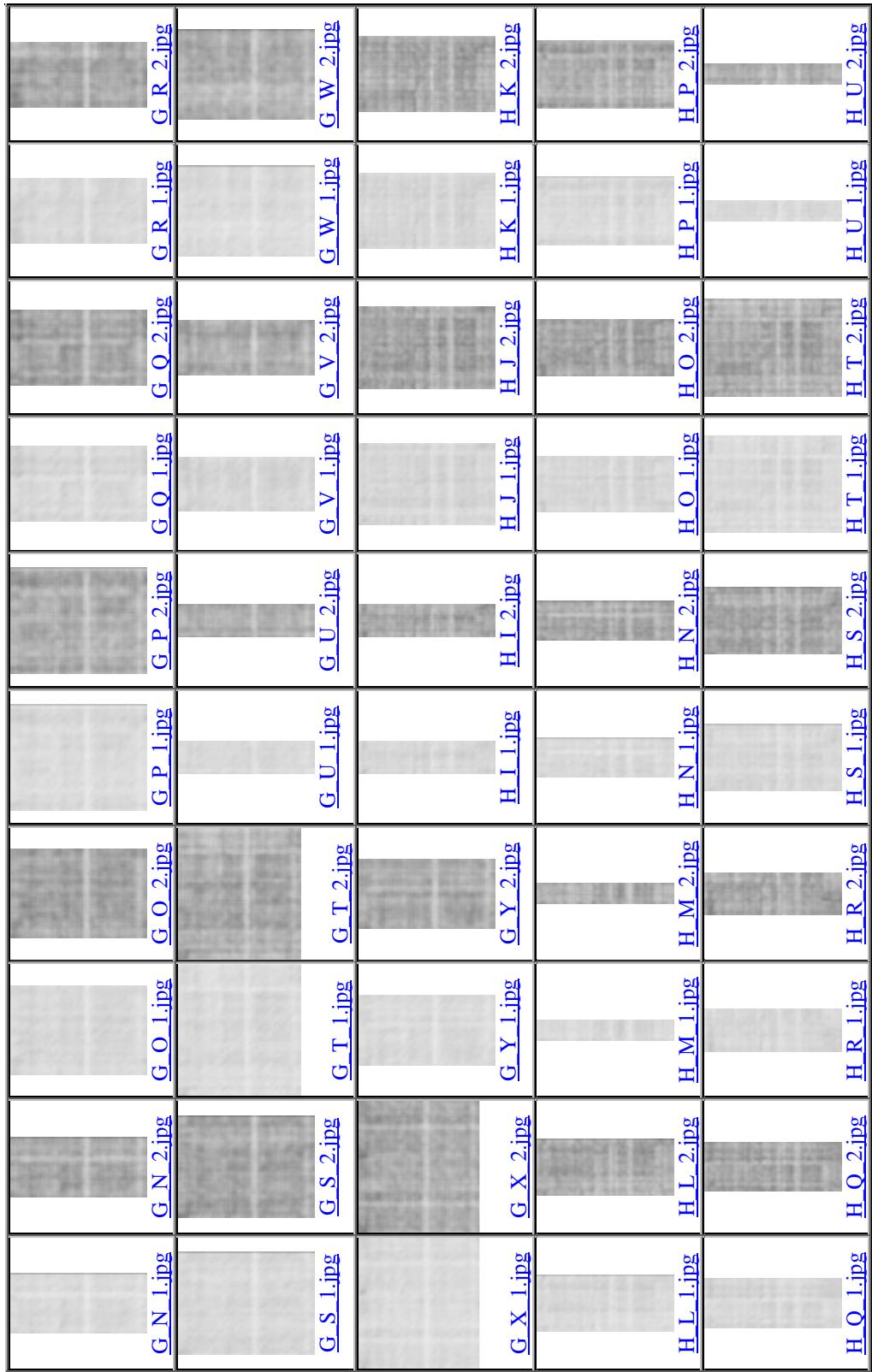
						
						
						
						
						
						

Appendix I: Visualisations for high similarity corpus

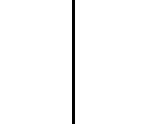
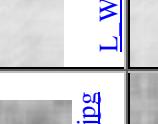
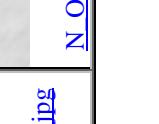
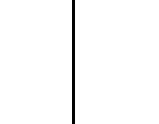
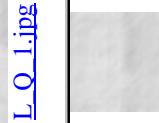
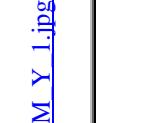
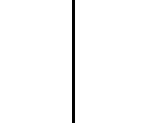
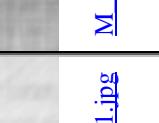
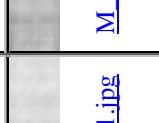
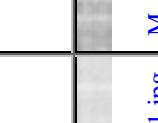
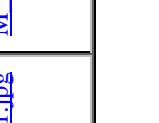
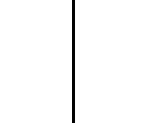
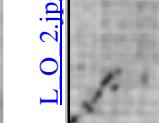
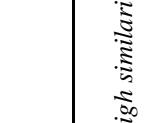
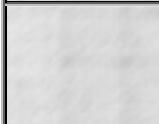
							
							
							
							
							
							
							
							

Appendix I: Visualisations for high similarity corpus

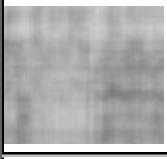
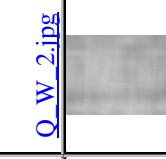
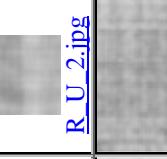
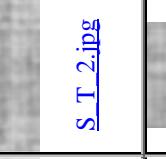
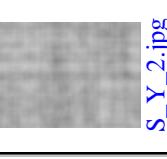
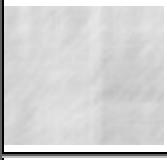
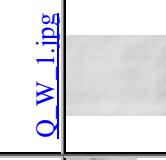
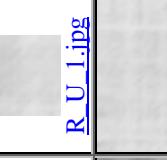
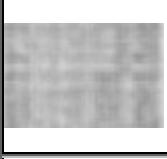
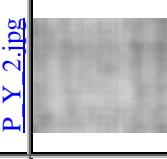
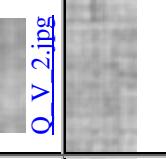
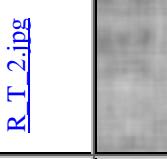
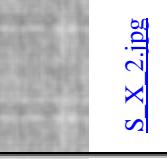
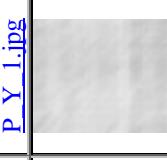
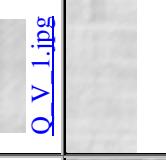
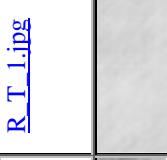
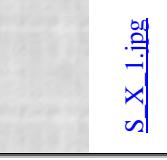
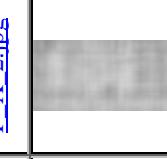
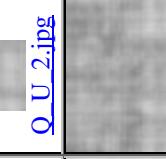
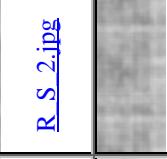
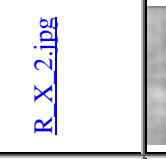
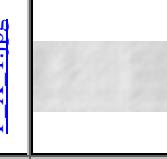
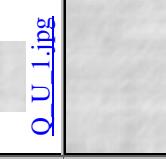
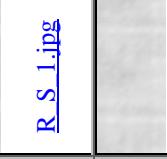
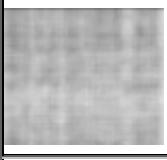
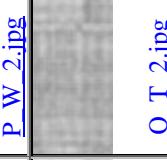
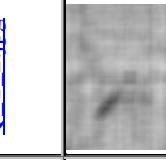
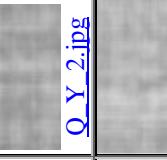
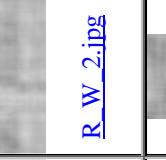
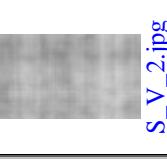
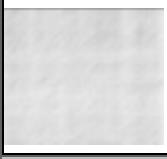
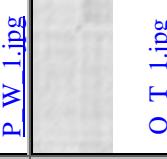
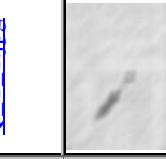
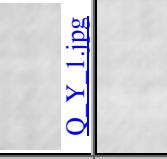
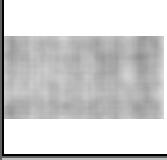
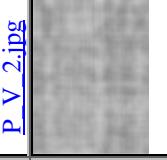
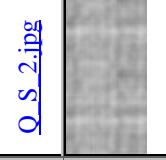
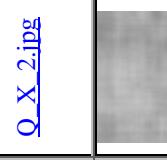


Appendix I: Visualisations for high similarity corpus

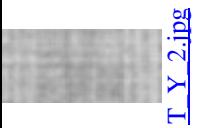
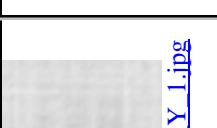
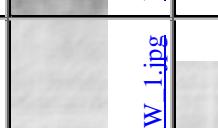
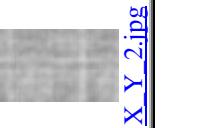
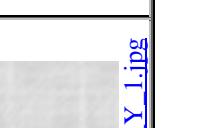
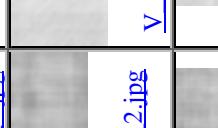
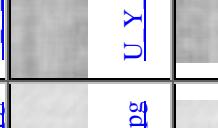
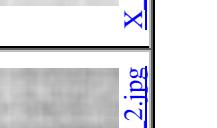
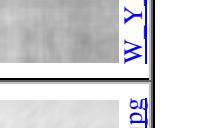
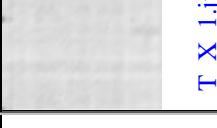
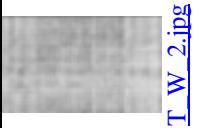
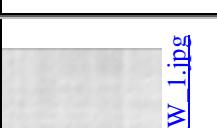
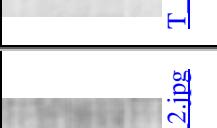
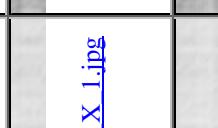
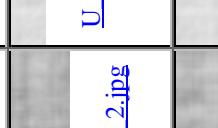
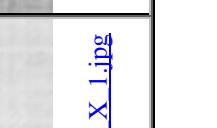
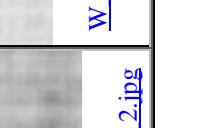
														
														
														
														
														

Appendix I: Visualisations for high similarity corpus

<u>N P 1.jpg</u>	<u>N P 2.jpg</u>	<u>N Q 1.jpg</u>	<u>N Q 2.jpg</u>	<u>N R 1.jpg</u>	<u>N R 2.jpg</u>	<u>N S 1.jpg</u>	<u>N S 2.jpg</u>	<u>N T 1.jpg</u>	<u>N T 2.jpg</u>
<u>N U 1.jpg</u>	<u>N U 2.jpg</u>	<u>N V 1.jpg</u>	<u>N V 2.jpg</u>	<u>N W 1.jpg</u>	<u>N W 2.jpg</u>	<u>N X 1.jpg</u>	<u>N X 2.jpg</u>	<u>N Y 1.jpg</u>	<u>N Y 2.jpg</u>
<u>O P 1.jpg</u>	<u>O P 2.jpg</u>	<u>O Q 1.jpg</u>	<u>O Q 2.jpg</u>	<u>O R 1.jpg</u>	<u>O R 2.jpg</u>	<u>O S 1.jpg</u>	<u>O S 2.jpg</u>	<u>O T 1.jpg</u>	<u>O T 2.jpg</u>
<u>O U 1.jpg</u>	<u>O U 2.jpg</u>	<u>O V 1.jpg</u>	<u>O V 2.jpg</u>	<u>O W 1.jpg</u>	<u>O W 2.jpg</u>	<u>O X 1.jpg</u>	<u>O X 2.jpg</u>	<u>O Y 1.jpg</u>	<u>O Y 2.jpg</u>
<u>P Q 1.jpg</u>	<u>P Q 2.jpg</u>	<u>P R 1.jpg</u>	<u>P R 2.jpg</u>	<u>P S 1.jpg</u>	<u>P S 2.jpg</u>	<u>P T 1.jpg</u>	<u>P T 2.jpg</u>	<u>P U 1.jpg</u>	<u>P U 2.jpg</u>

Appendix I: Visualisations for high similarity corpus

Appendix J: Word Fragments for Simple Metrics for Three Sample Documents

Table J.1 shows the three sample documents.

Document identifier	Contents
D1	The cat sat on the mat. It was a black cat. It was a black mat.
D2	The feline sat and purred on the black mat. It was a black cat.
D3	Looks like raining cats and dogs, purred the feline. It was a black sky outside.

Table J.1 - Similarity documents D1, D2 and D3

Table J.2 shows all the word fragments generated from these documents for the different simple metrics. Spaces are shown with the underscore character (_).

Metric	Document					
	D1		D2		D3	
Fragment	Freq	Fragment	Freq	Fragment	Freq	Fragment
char1						
-	15	-	13	-	14	
a	11	a	8	a	6	
b	2	b	2	b	1	
c	4	c	3	c	2	
e	2	d	2	d	4	
h	2	e	5	e	6	
i	2	f	1	f	1	
k	2	h	2	g	2	
l	2	i	2	h	1	
m	2	k	2	i	6	
n	1	l	3	k	4	
o	1	m	1	l	4	
s	3	n	3	n	4	
t	9	o	1	o	4	
w	2	p	1	p	1	
		r	2	r	3	
		s	2	s	6	
		t	6	t	4	
		u	1	u	2	
		w	1	w	1	
			y	y	1	
char2						
_a	2	_a	2	_a	2	
_b	2	_b	2	_b	1	
_c	2	_c	1	_c	1	
_i	2	_f	1	_d	1	
_m	2	_i	1	_f	1	
_o	1	_m	1	_i	1	
_s	1	_o	1	_l	1	
_t	1	_p	1	_o	1	
_w	2	_s	1	_p	1	

a_	2 _t	1 r	1
ac	2 _w	1 _s	1
as	2 a_	1 _t	1
at	5 ac	2 _w	1
bl	2 an	1 a_	1
ca	2 as	1 ac	1
ck	2 at	3 ai	1
e_	2 bl	2 an	1
he	2 ca	1 as	1
it	2 ck	2 at	1
k_	2 d_	2 bl	1
la	2 e_	3 ca	1
ma	2 ed	1 ck	1
n_	1 el	1 d_	2
on	1 fe	1 de	1
s_	2 he	2 do	1
sa	1 in	1 e_	2
t_	4 it	1 ed	1
th	2 k_	2 el	1
wa	2 la	2 fe	1
	li	1 g_	1
	ma	1 gs	1
	n_	1 he	1
	nd	1 id	1
	ne	1 ik	1
	on	1 in	3
	pu	1 it	1
	re	1 k_	1
	rr	1 ke	1
	s_	1 ks	1
	sa	1 ky	1
	t_	2 la	1
	th	2 li	2
	ur	1 lo	1
	wa	1 nd	1
		ne	1
		ng	1
		ni	1
		og	1
		ok	1
		oo	1
		ou	1
		pu	1
		ra	1
		re	1
		rr	1
		s_	3
		si	1
		sk	1
		t_	1
		th	1
		ts	2
		ur	1
		ut	1
		wa	1
		y_	1

char3					
a	2	_a_	1	_a_	1
_bl	2	_an	1	_an	1
_ca	2	_bl	2	_bl	1
_it	2	_ca	1	_ca	1
_ma	2	_fe	1	_do	1
_on	1	_it	1	_fe	1
_sa	1	_ma	1	_it	1
_th	1	_on	1	_li	1
_wa	2	_pu	1	_ou	1
a_b	2	_sa	1	_pu	1
ack	2	_th	1	_ra	1
as_	2	_wa	1	_sk	1
at_	2	a_b	1	_th	1
bla	2	ack	2	_wa	1
cat	2	and	1	a_b	1
ck_	2	as_	1	ack	1
e_c	1	at_	1	ain	1
e_m	1	bla	2	and	1
he_	2	cat	1	as_	1
it_	2	ck_	2	ats	1
k_c	1	d_o	1	bla	1
k_m	1	d_p	1	cat	1
lac	2	e_b	1	ck_	1
mat	2	e_f	1	d_d	1
n_t	1	e_s	1	d_t	1
on_	1	ed_	1	dog	1
s_a	2	eli	1	e_f	1
sat	1	fel	1	e_r	1
t_o	1	he_	2	ed_	1
t_s	1	ine	1	eli	1
t_w	2	it_	1	fel	1
the	2	k_c	1	g_c	1
was	2	k_m	1	he_	1
		lac	2	ide	1
		lin	1	ike	1
		mat	1	ine	1
		n_t	1	ing	1
		nd_	1	ini	1
		ne_	1	it_	1
		on_	1	k_s	1
		pur	1	ke_	1
		red	1	ks_	1
		rre	1	ky_	1
		s_a	1	lac	1
		sat	1	lik	1
		t_a	1	lin	1
		t_w	1	loo	1
		the	2	nd_	1
		urr	1	ng_	1
		was	1	nin	1
				ogs	1
				oks	1
				ook	1
				out	1
				pur	1
				rai	1

			red	1
			rre	1
			s_a	2
			s_l	1
			sid	1
			sky	1
			t_w	1
			the	1
			ts_	1
			tsi	1
			urr	1
			uts	1
			was	1
			y_o	1
char4	_a_b	2	char	4
	_bla	2	_a_b	1
	_cat	2	_and	1
	it	2	_bla	1
	_mat	2	_cat	1
	on	1	_dog	1
	_sat	1	_fel	1
	_the	1	_it_	1
	_was	1	_lik	1
	a_bl	2	_mat	1
	ack_	2	_on_	1
	as_a	2	_pur	1
	at_o	1	_sat	1
	at_s	1	_the	1
	blac	1	ack_	1
	cat_	1	_was	1
	ck_c	1	a_bl	1
	ck_m	1	ack_	1
	e_ca	1	and_	1
	e_ma	1	as_a	1
	he_c	1	aini	1
	he_m	1	at_a	1
	it_w	1	and_	1
	k_ca	1	ats_	1
	k_ma	1	blac	1
	lack	1	blac	1
	n_th	1	cats	1
	on_t	1	ck_s	1
	s_a_	2	d_on	1
	sat_	1	d_pu	1
	t_on	1	e_bl	1
	t_sa	1	d_do	1
	t_wa	1	e_fe	1
	the_	2	dogs	1
	was_	2	ed_o	1
			e_fe	1
			elin	1
			elin	1
			feli	1
			fe	1
			fe	1
			g_ca	1
			he_f	1
			ike_	1
			ing_	1
			lack	1
			line	1
			linin	1
			it_w	1
			k_sk	1
			ke_r	1
			ks_l	1
			ky_o	1
			purr	1
			lack	1

		red_	1	like	1	
		rred	1	line	1	
		s_a_	1	look	1	
		sat_	1	nd_d	1	
		t_an	1	ng_c	1	
		t_wa	1	ning	1	
		the_	2	oks_	1	
		urre	1	ooks	1	
		was_	1	outs	1	
				purr	1	
				rain	1	
				red_	1	
				rred	1	
				s_a_	1	
				s_an	1	
				s_li	1	
				side	1	
				sky_	1	
				t_wa	1	
				the_	1	
				ts_a	1	
				tsid	1	
				urre	1	
				utsi	1	
				was_	1	
				y_ou	1	
char5	_a_bla	2	_a_bla	1	_a_bla	1
	_blac	2	_and_	1	_and_	1
	cat	1	_blac	2	_blac	1
	_it_w	2	_feli	1	_cats	1
	_on_t	1	_it_w	1	_dogs	1
	sat	1	_on_t	1	_feli	1
	the	1	_purr	1	_it_w	1
	was	2	_sat_	1	_like	1
	a_bla	2	_the_	1	_outs	1
	ack_c	1	_was_	1	_purr	1
	ack_m	1	a_bla	1	_rain	1
	as_a_	2	ack_c	1	_sky_	1
	at_on	1	ack_m	1	_the_	1
	at_sa	1	and_p	1	_was_	1
	black	2	as_a_	1	a_bla	1
	cat_s	1	at_an	1	ack_s	1
	ck_ca	1	black	2	ainin	1
	ck_ma	1	ck_ca	1	and_d	1
	e_cat	1	ck_ma	1	as_a_	1
	e_mat	1	d_on_	1	ats_a	1
	he_ca	1	d_pur	1	black	1
	he_ma	1	e_bla	1	cats_	1
	it_wa	2	e_fel	1	ck_sk	1
	k_cat	1	e_sat	1	d_dog	1
	k_mat	1	ed_on	1	d_the	1
	lack_	2	eline	1	e_fel	1
	n_the	1	felin	1	e_rai	1
	on_th	1	he_bl	1	ed_th	1
	s_a_b	2	he_fe	1	eline	1
	sat_o	1	ine_s	1	felin	1

	t_on_	1 it_wa	1 g_cat	1
	t_sat	1 k_cat	1 he_fe	1
	t_was	2 k_mat	1 ike_r	1
	the_c	1 lack_	2 ing_c	1
	the_m	1 line_	1 ining	1
	was_a	2 n_the	1 it_wa	1
		nd_pu	1 k_sky	1
		ne_sa	1 ke_ra	1
		on_th	1 ks_li	1
		purre	1 ky_ou	1
		red_o	1 lack_	1
		rred_	1 like_	1
		s_a_b	1 looks	1
		sat_a	1 nd_do	1
		t_and	1 ng_ca	1
		t_was	1 ning_	1
		the_b	1 oks_1	1
		the_f	1 ooks_	1
		urred	1 outsi	1
		was_a	1 purre	1
			raini	1
			red_t	1
			rred_	1
			s_a_b	1
			s_and	1
			s_li	1
			sky_o	1
			t_was	1
			the_f	1
			ts_an	1
			tside	1
			urred	1
			utsid	1
			was_a	1
			y_out	1
char6	_a_bla	2 _a_bla	1 _a_bla	1
	_black	2 _and_p	1 _and_d	1
	_cat_s	1 _black	2 _black	1
	_it_wa	2 _felin	1 _cats_	1
	_on_th	1 _it_wa	1 _felin	1
	_sat_o	1 _on_th	1 _it_wa	1
	_the_m	1 _purre	1 _like_	1
	_was_a	2 _sat_a	1 _outsi	1
	a_blac	2 _the_b	1 _purre	1
	ack_ca	1 _was_a	1 _raini	1
	ack_ma	1 a_blac	1 _sky_o	1
	as_a_b	2 ack_ca	1 _the_f	1
	at_on_	1 ack_ma	1 _was_a	1
	at_sat	1 and_pu	1 a_blac	1
	black_	2 as_a_b	1 ack_sk	1
	cat_sa	1 at_and	1 aiming	1
	ck_cat	1 black_	2 and_do	1
	ck_mat	1 ck_cat	1 as_a_b	1
	e_cat_	1 ck_mat	1 ats_an	1
	he_cat	1 d_on_t	1 black_	1
	he_mat	1 d_purr	1 cats_a	1

	it_was	2 e_blac	1 ck_sky	1
	lack_c	1 e_feli	1 d_dogs	1
	lack_m	1 e_sat_	1 d_the_	1
	n_the_	1 ed_on_	1 e_feli	1
	on_the	1 eline_	1 e_rain	1
	s_a_bl	2 feline	1 ed_the	1
	sat_on	1 he_bla	1 feline	1
	t_on_t	1 he_fel	1 g_cats	1
	t_sat_	1 ine_sa	1 he_fel	1
	t_was_	2 it_was	1 ike_ra	1
	the_ca	1 lack_c	1 ing_ca	1
	the_ma	1 lack_m	1 ining_	1
	was_a_	2 line_s	1 it_was	1
		n_the_	1 k_sky_	1
		nd_pur	1 ke_rai	1
		ne_sat	1 ks_lik	1
		on_the	1 ky_out	1
		purred	1 lack_s	1
		red_on	1 like_r	1
		rred_o	1 looks_	1
		s_a_bl	1 nd_dog	1
		sat_an	1 ng_cat	1
		t_and_	1 ning_c	1
		t_was_	1 oks_li	1
		the_bl	1 ooks_l	1
		the_fe	1 outsid	1
		urred_	1 purred	1
		was_a_	1 rainin	1
			red_th	1
			rred_t	1
			s_a_bl	1
			s_and_	1
			s_like	1
			sky_ou	1
			t_was_	1
			the_fe	1
			ts_and	1
			urred_	1
			utside	1
			was_a_	1
			y_outs	1
char7	_a_blac	2 _a_blac	1 _a_blac	1
	black	2 _and_pu	1 _and_do	1
	_cat_sa	1 _black_	2 _black_	1
	_it_was	2 _feline	1 _cats_a	1
	_on_the	1 _it_was	1 _feline	1
	_sat_on	1 _on_the	1 _it_was	1
	_the_ma	1 _purred	1 _like_r	1
	_was_a_	2 _sat_an	1 _outsid	1
	a_black	2 _the_bl	1 _purred	1
	ack_cat	1 _was_a_	1 _rainin	1
	ack_mat	1 a_black	1 _sky_ou	1
	as_a_bl	2 ack_cat	1 _the_fe	1
	at_on_t	1 ack_mat	1 _was_a_	1
	at_sat_	1 and_pur	1 a_black	1
	black_c	1 as_a_bl	1 ack_sky	1

	black_m	1 at_and_	1 aining_	1
	cat_sat	1 black_c	1 and_dog	1
	e_cat_s	1 black_m	1 as_a_bl	1
	he_cat_	1 d_on_th	1 ats_and	1
	it_was_	2 d_purre	1 black_s	1
	lack_ca	1 e_black	1 cats_an	1
	lack_ma	1 e_felin	1 ck_sky_	1
	n_the_m	1 e_sat_a	1 d_the_f	1
	on_the_	1 ed_on_t	1 e_felin	1
	s_a_bla	2 eline_s	1 e_raini	1
	sat_on_	1 feline_	1 ed_the_	1
	t_on_th	1 he_blac	1 g_cats_	1
	t_sat_o	1 he_feli	1 he_feli	1
	t_was_a	2 ine_sat	1 ike_rai	1
	the_cat	1 it_was_	1 ing_cat	1
	the_mat	1 lack_ca	1 ining_c	1
	was_a_b	2 lack_ma	1 it_was_	1
		line_sa	1 k_sky_o	1
		n_the_b	1 ke_rain	1
		nd_purr	1 ks_like	1
		ne_sat_	1 ky_outs	1
		on_the_	1 lack_sk	1
		purred_	1 like_ra	1
		red_on_	1 looks_l	1
		rred_on	1 nd_dogs	1
		s_a_bla	1 ng_cats	1
		sat_and	1 ning_ca	1
		t_and_p	1 oks lik	1
		t_was_a	1 ooks_li	1
		the_bla	1 outside	1
		the_fel	1 purred_	1
		urred_o	1 raining	1
		was_a_b	1 red_the	1
			rred_th	1
			s_a_bla	1
			s_and_d	1
			s_like_	1
			sky_out	1
			t_was_a	1
			the_fel	1
			ts_and_	1
			urred_t	1
			was_a_b	1
			y_outsi	1
char8	_a_black	2 _a_black	1 _a_black	1
	_black_c	1 _and_pur	1 _and_dog	1
	_black_m	1 _black_c	1 _black_s	1
	_cat_sat	1 _black_m	1 _cats_an	1
	_it_was_	2 _feline_	1 _it_was_	1
	_on_the_	1 _it_was_	1 _like_ra	1
	_sat_on_	1 _on_the_	1 _outside	1
	_the_mat	1 _purred_	1 _purred_	1
	_was_a_b	2 _sat_and	1 raining	1
	a_black_	2 _the_bla	1 _sky_out	1
	as_a_bla	2 _was_a_b	1 _the_fel	1
	at_on_th	1 a_black_	1 _was_a_b	1

	at_sat_o	1	and_purr	1	a_black_	1
	black_ca	1	as_a_bla	1	ack_sky_	1
	black_ma	1	at_and_p	1	aining_c	1
	cat_sat_	1	black_ca	1	and_dogs	1
	e_cat_sa	1	black_ma	1	as_a_bla	1
	he_cat_s	1	d_on_the	1	ats_and_	1
	it_was_a	2	d_purred	1	black_sk	1
	lack_cat	1	e_black_	1	cats_and	1
	lack_mat	1	e_feline	1	ck_sky_o	1
	n_the_ma	1	e_sat_an	1	d_the_fe	1
	on_the_m	1	ed_on_th	1	e_feline	1
	s_a_blac	2	eline_sa	1	e_rainin	1
	sat_on_t	1	feline_s	1	ed_the_f	1
	t_on_the	1	he_black	1	g_cats_a	1
	t_sat_on	1	he_felin	1	he_felin	1
	t_was_a_	2	ine_sat_	1	ike_rain	1
	the_cat_	1	it_was_a	1	ing_cats	1
	was_a_bl	2	lack_cat	1	ining_ca	1
			lack_mat	1	it_was_a	1
			line_sat	1	k_sky_ou	1
			n_the_bl	1	ke_raini	1
			nd_purre	1	ks_like_	1
			ne_sat_a	1	ky_outsi	1
			on_the_b	1	lack_sky	1
			purred_o	1	like_rai	1
			red_on_t	1	looks_li	1
			rred_on_	1	ng_cats_	1
			s_a_blac	1	ning_cat	1
			sat_and_	1	oks_like	1
			t_and_pu	1	ooks lik	1
			t_was_a_	1	purred_t	1
			the_blac	1	raining_	1
			the_feli	1	red_the_	1
			urred_on	1	rred_the	1
			was_a_bl	1	s_a_blac	1
					s_and_do	1
					s_like_r	1
					sky_outs	1
					t_was_a_	1
					the_feli	1
					ts_and_d	1
					urred_th	1
					was_a_bl	1
					y_outsid	1
char9	_a_black_	2	_a_black_	1	_a_black_	1
	_black_ca	1	_and_purr	1	_and_dogs	1
	_black_ma	1	_black_ca	1	_black_sk	1
	_cat_sat_	1	_black_ma	1	_cats_and	1
	_it_was_a	2	_feline_s	1	_it_was_a	1
	_on_the_m	1	_it_was_a	1	_like_rai	1
	_sat_on_t	1	_on_the_b	1	_purred_t	1
	_was_a_bl	2	_purred_o	1	_raining_	1
	a_black_c	1	_sat_and_	1	_sky_outs	1
	a_black_m	1	_the_blac	1	_the_feli	1
	as_a_blac	2	_was_a_bl	1	_was_a_bl	1
	at_on_the	1	a_black_c	1	a_black_s	1

	at_sat_on	1 and_purre	1 ack_sky_o	1
	black_cat	1 as_a_blac	1 aining_ca	1
	black_mat	1 at_and_pu	1 as_a_blac	1
	cat_sat_o	1 black_cat	1 ats_and_d	1
	e_cat_sat	1 black_mat	1 black_sky	1
	he_cat_sa	1 d_on_the_	1 cats_and_	1
	it_was_a_	2 d_purred_	1 ck_sky_ou	1
	n_the_mat	1 e_black_m	1 d_the_fel	1
	on_the_ma	1 e_feline_	1 e_raining	1
	s_a_black	2 e_sat_and	1 ed_the_fe	1
	sat_on_th	1 ed_on_the	1 g_cats_an	1
	t_on_the_	1 eline_sat	1 he_feline	1
	t_sat_on_	1 feline_sa	1 ike_raini	1
	t_was_a_b	2 he_black_	1 ing_cats_	1
	the_cat_s	1 he_feline	1 ining_cat	1
	was_a_bla	2 ine_sat_a	1 it_was_a_	1
		it_was_a_	1 k_sky_out	1
		line_sat_	1 ke_rainin	1
		n_the_bla	1 ks_like_r	1
		nd_purred	1 ky_outsid	1
		ne_sat_an	1 lack_sky_	1
		on_the_bl	1 like_rain	1
		purred_on	1 looks_li	1
		red_on_th	1 ng_cats_a	1
		rred_on_t	1 ning_cats	1
		s_a_black	1 ooks_like_	1
		sat_and_p	1 ooks_like	1
		t_and_pur	1 purred_th	1
		t_was_a_b	1 raining_c	1
		the_black	1 red_the_f	1
		the_felin	1 rred_the_	1
		urred_on_	1 s_a_black	1
		was_a_bla	1 s_and_dog	1
			s_like_ra	1
			sky_outsi	1
			t_was_a_b	1
			the_felin	1
			ts_and_do	1
			urred_the	1
			was_a_bla	1
			y_outside	1
word1	a	2 a	1 a	1
	black	2 and	1 and	1
	cat	2 black	2 black	1
	it	2 cat	1 cats	1
	mat	2 feline	1 dogs	1
	on	1 it	1 feline	1
	sat	1 mat	1 it	1
	the	2 on	1 like	1
	was	2 purred	1 looks	1
		sat	1 outside	1
		the	2 purred	1
		was	1 raining	1
			sky	1
			the	1
			was	1

word2	a_black black_cat black_mat cat_it cat_sat it_was mat_it on_the sat_on the_cat the_mat was_a	2 a_black 1 and_purred 1 black_cat 1 black_mat 1 feline_sat 2 it_was 1 mat_it 1 on_the 1 purred_on 1 sat_and 1 the_black 2 the_feline was_a	1 a_black 1 and_dogs 1 black_sky 1 cats_and 1 dogs_purred 1 feline_it 1 it_was 1 like_raining 1 looks_like 1 purred_the 1 raining_cats 1 sky_outside 1 the_feline 1 was_a	1
word3	a_black_cat a_black_mat black_cat_it cat_it_was cat_sat_on it_was_a mat_it_was on_the_mat sat_on_the the_cat_sat the_mat_it was_a_black	1 a_black_cat 1 and_purred_on 1 black_mat_it 1 feline_sat_and 1 it_was_a 2 mat_it_was 1 on_the_black 1 purred_on_the 1 sat_and_purred 1 the_black_mat 1 the_feline_sat 2 was_a_black	1 a_black_sky 1 and_dogs_purred 1 black_sky_outside 1 cats_and_dogs 1 dogs_purred_the 1 feline_it_was 1 it_was_a 1 like_raining_cats 1 looks_like_raining 1 purred_the_feline 1 raining_cats_and 1 the_feline_it 1 was_a_black	1
word4	a_black_cat_it black_cat_it_was cat_it_was_a cat_sat_on_the it_was_a_black mat_it_was_a on_the_mat_it sat_on_the_mat the_cat_sat_on the_mat_it_was was_a_black_cat was_a_black_mat	1 and_purred_on_the 1 black_mat_it_was 1 feline_sat_and_purred 1 it_was_a_black 2 mat_it_was_a 1 on_the_black_mat 1 purred_on_the_black 1 sat_and_purred_on 1 the_black_mat_it 1 the_feline_sat_and 1 was_a_black_cat 1	1 a_black_sky_outside 1 and_dogs_purred_the 1 cats_and_dogs_purred 1 dogs_purred_the_feline 1 feline_it_was_a 1 it_was_a_black 1 like_raining_cats_and 1 looks_like_raining_cats 1 purred_the_feline_it 1 raining_cats_and_dogs 1 the_feline_it_was 1 was_a_black_sky	1
word5	a_black_cat_it_was black_cat_it_was_a cat_it_was_a_black cat_sat_on_the_mat it_was_a_black_cat it_was_a_black_mat mat_it_was_a_black on_the_mat_it_was sat_on_the_mat_it the_cat_sat_on_the the_mat_it_was_a was_a_black_cat_it	1 and_purred_on_the_black 1 black_mat_it_was_a 1 feline_sat_and_purred_on 1 it_was_a_black_cat 1 mat_it_was_a_black 1 on_the_black_mat_it 1 purred_on_the_black_mat 1 sat_and_purred_on_the 1 the_black_mat_it_was 1 the_feline_sat_and_purred 1	1 and_dogs_purred_the_felin e 1 cats_and_dogs_purred_the 1 dogs_purred_the_feline_it 1 feline_it_was_a_black 1 it_was_a_black_sky 1 like_raining_cats_and_dog s 1 looks_like_raining_cats_an d 1 purred_the_feline_it_was 1 raining_cats_and_dogs_pu rred 1 the_feline_it_was_a 1 was_a_black_sky_outside	1
word6	a_black_cat_it_was_a	1 and_purred_on_the_black_	1 and_dogs_purred_the_felin e_it	1

	black_cat_it_was_a_black	1	black_mat_it_was_a_black	1	cats_and_dogs_purred_the	1
	cat_it_was_a_black_mat	1	feline_sat_and_purred_on_	1	the_feline	
	cat_sat_on_the_mat_it	1	the		dogs_purred_the_feline_it	1
	it_was_a_black_cat_it	1	mat_it_was_a_black_cat	1	_was	
	mat_it_was_a_black_cat	1	on_the_black_mat_it_was	1	feline_it_was_a_black_sky	1
	on_the_mat_it_was_a	1	purred_on_the_black_mat_	1	it_was_a_black_sky_outsi	1
	sat_on_the_mat_it_was	1	it		de	
	the_cat_sat_on_the_mat	1	sat_and_purred_on_the_bla	1	like_raining_cats_and_dog	1
	the_mat_it_was_a_black	1	ck		s_purred	
	was_a_black_cat_it_was	1	the_black_mat_it_was_a	1	looks_like_raining_cats_an	1
word7	a_black_cat_it_was_a_bla	1	the_feline_sat_and_purred	1	d_dogs	
	k	1	_on	1	purred_the_feline_it_was	1
	black_cat_it_was_a_black	1	1	1	rainning_cats_and_dogs_pu	1
	_mat	1	red_the		rred_the	
	cat_sat_on_the_mat_it_wa	1	the_feline_it_was_a_black	1	the_feline_it_was_a_black	1
	s	1	mat_it_was_a_black_cat_it	1	cat_it_was_a	
	it_was_a_black_cat_it_was	1	on		and_dogs_purred_the_felin	1
	mat_it_was_a_black_cat_it	1	black_mat_it_was_a_black	1	e_it_was	
	on_the_mat_it_was_a_bla	1	cat	1	cats_and_dogs_purred_the	1
	k	1	feline_it_was		_feline_it	
word8	sat_on_the_mat_it_was_a	1	on_the_black_mat_it_was_a	1	dogs_purred_the_feline_it	1
	black	1	b		_was_a	
	the_cat_sat_on_the_mat_it	1	ack		feline_it_was_a_black_sky	1
	_was	1	mat_it_was_a_black_cat_it	1	outside	
	on_the_mat_it_was_a_bla	1	it_was_a	1	like_raining_cats_and_dog	1
	k_cat	1	1		s_purred_the_feline	
	sat_on_the_mat_it_was_a_	1	sat_and_purred_on_the_bla	1	looks_like_raining_cats_an	1
	black	1	ck_mat_it		d_dogs_purred_the	
	the_cat_sat_on_the_mat_it	1	the_black_mat_it_was_a_b	1	purred_the_feline_it_was	1
	_was	1	lack_cat		a_black_sky	
word9	the_mat_it_was_a_black_c	1	the_feline_sat_and_purred	1	rainning_cats_and_dogs_pu	1
	at_it	1	_on_the_black		rred_the_feline_it	
	was_a_black_cat_it_was_a	1			the_feline_it_was_a_black	1
	black	1			_sky_outside	

	on_the_mat_it_was_a_black_cat_it sat_on_the_mat_it_was_a_black_cat the_cat_sat_on_the_mat_it_was_a the_mat_it_was_a_black_cat_it_was_a was_a_black_cat_it_was_a_black_mat	1 purred_on_the_black_mat_it_was_a_black 1 sat_and_purred_on_the_black_mat_it_was 1 the_feline_sat_and_purred_on_the_black_mat 1	1 like_raining_cats_and_dogs_purred_the_feline_it 1 looks_like_raining_cats_and_dogs_purred_the_feline 1 purred_the_feline_it_was_a_black_sky_outside 1 raining_cats_and_dogs_purred_the_feline_it_was	1
sent	it_was_a_black_cat	1 it_was_a_black_cat	1 it_was_a_black_sky_outside	1
	it_was_a_black_mat	1 the_feline_sat_and_purred_on_the_black_mat	1 looks_like_raining_cats_and_dogs_purred_the_feline	1
	the_cat_sat_on_the_mat	1		

Table J.2 - All possible fragments for simple metrics on documents D1, D2 and D3

