A COMPARISON OF SIMILARITY METRICS FOR E-ASSESSMENT OF MS OFFICE ASSIGNMENTS

Dissertation submitted by

WILLEM STERRENBERG JACOBUS MARAIS

to the

Department of Computer Science and Informatics
Faculty of Natural and Agricultural Sciences
University of the Free State, South Africa

Submitted in full fulfilment of the requirements for the degree

Magister Commercii

17 July 2015

Study Leader: Prof P.J. Blignaut

Abstract

Computerised assessment is prevalent in various disciplines where immediate and accurate feedback with regard to students' assignments is required. It is used as an alternative to manual assessment of computer programming assignments, computer proficiency tests and free-text responses to questions. The implementation of the Office Open XML (OOXML) standard, as the default document format for Microsoft Office, instigated the development of alternative computerised assessment algorithms with the ability to assess word-processing documents of the DOCX format. Word-processing assignments are primarily assessed by comparing the final document, submitted by the student, to the ideal solution provided by the examiner. Research into the anatomy of OOXML-based documents delivered several alternative approaches with regard to the computerised assessment of DOCX document types. OOXML simplifies the evaluation process of word-processing documents by providing easily identifiable elements within the document structure. These elements can then be used to assess the content and formatting of the document to determine whether the solution, submitted by the student, matches the ideal solution provided by the examiner. By examining current OOXML-based algorithms, certain gaps within the implementation thereof were identified. An alternative algorithm, dubbed the *OOXML algorithm* that could alleviate these issues, is introduced. It improves the assessment techniques of current OOXML-based algorithms by firstly simplifying the structure of the DOCX documents to ensure that the student's document and examiner's solution conform to a homogeneous structure. It then identifies corresponding paragraphs between the student's document and the examiner's solution. Finally, the student's simplified document is assessed by comparing the content and formatting elements within the OOXML structure of the corresponding paragraphs with one another. To determine the accuracy and reliability of the proposed OOXML algorithm, it is compared with three established algorithms as well as manual assessment techniques. The three algorithms include a string comparison algorithm called fComp, the Levenshtein algorithm and a document difference algorithm, implemented by a system called Word Grader. The same group of word-processing assignments is graded by the specified algorithms and manually assessed by multiple human markers. Analysis of the results of a quasi-experimental study concluded that the proposed OOXML algorithm and its element comparison metric not only produced more reliable results than the human markers but also more accurate results than the human markers and the other selected document analysis algorithms.

Opsomming

Gerekenariseerde assessering kom algemeen voor in verskeie dissiplines waar onmiddellike en akkurate terugvoer met betrekking tot studente se werksopdragte vereis word. Dit word gebruik as ‘n alternatief tot die assessering van rekenaarprogrammeringsopdragte, rekenaarvaardigheidstoetse en vrye-teks antwoorde op vrae deur menslike nasieners. Die implementering van die Office Open XML (OOXML) standaard as die verstek dokument-formaat vir Microsoft Office, het die ontwikkeling van alternatiewe gerekenariseerde assesserings-algoritmes, met die vermoë om woordverwerkingsdokumente van die DOCX formaat te evalueer, genoodsaak. Woordverwerkingsopdragte word hoofsaaklik geassesseer deur die finale dokument, wat deur die student ingedien word, met die ideale oplossing van die eksaminator te vergelyk. Navorsing van die anatomie van OOXML-gebaseerde dokumente het verskeie alternatiewe benaderings met betrekking tot die gerekenariseerde assessering van DOCX-dokument tipes opgelewer. OOXML vereenvoudig die evalueringsproses van woordverwerkingsdokumente deur maklik-identifiseerbare elemente binne die dokumentstruktuur te verskaf wat gebruik kan word om die inhoud en formaat van die dokument te assesseer. Hierdie elemente word dan gebruik om te bepaal of die oplossing, wat deur die student ingedien is, ooreenstem met die ideale oplossing wat deur die eksaminator verskaf word. Deur huidige OOXML-gebaseerde algoritmes te ondersoek, is sekere leemtes in die implementering daarvan geïdentifiseer. 'n Alternatiewe algoritme, genaamd die *OOXML-algoritme,*wat hierdie kwessies kan verminder is bekendgestel. Dit verbeter die assesseringstegnieke van huidige OOXML-gebaseer algoritmes deur eerstens die struktuur van die DOCX-dokumente te vereenvoudig om te verseker dat die student se dokument en die eksaminator se oplossing aan 'n homogene struktuur voldoen. Hierna identifiseer dit ooreenstemmende paragrawe tussen die student se dokument en die eksaminator se oplossing. Laastens, word die student se vereenvoudigde dokument geassesseer deur die inhoud en formateringselemente binne die OOXML-struktuur van die ooreenstemmende paragrawe met mekaar te vergelyk. Om die akkuraatheid en betroubaarheid van die voorgestelde OOXML-algoritme te bepaal, word dit vergelyk met drie gevestigde algoritmes sowel as met hand-assesseringstegnieke. Die drie algoritmes sluit in 'n string-vergelykingsalgoritme genaamd fComp, die Levenshtein-algoritme en 'n algoritme wat verskille in dokumente identifiseer, geïmplementeer deur 'n stelsel genaamd *Word Grader*. Dieselfde groep woordverwerkingsopdragte is deur die gespesifiseerde algoritmes gemerk asook deur verskeie nasieners met die hand gemerk. Die analise-resultate van 'n kwasi-

eksperimentele studie het bevind dat die voorgestelde OOXML-algoritme en sy element-vergelykingsmaatstaf nie slegs meer betroubare resultate as menslike nasieners verskaf nie, maar ook akkurater resultate as menslike nasieners en die ander gekose dokumentontledings-algoritmes opgelewer het.

Acknowledgments

The author would like to thank the following for their contributions and support:

- My wife, Marilize, for her love and moral support
- Prof P.J. Blignaut for his invaluable guidance
- Mrs E.H. Dednam for providing the word-processing assignments relevant to this research project
- Ms T.S. Nkalai for providing a benchmark by re-assessing the sample word-processing assignments
- Mrs S. Opperman for proofreading and editing of the final document

Table of Contents

List of Figures

List of Tables

Chapter 1

Overview of the Research Project

This chapter focuses on the following aspects:

- **The need for the computerised assessment of computer proficiency**
- **Current word-processing assessment algorithms and their limitations**
- **The main objective, hypotheses, research methodology and scope of the project**
- **The structural layout of the dissertation**

1.1 Introduction

The assessment of skills in information technology (IT) has become increasingly important in the twenty-first century (Tuparova & Tuparov, 2010). Consequently, computerised assessment systems are prevalent in the assessment of various IT skills. This research project focuses on the implementation of alternative similarity metrics to develop an improved algorithm for the computerised assessment of word-processing skills. Similarity metrics can be defined as the measures that are implemented to determine the similarity between two objects (Li, Chen, Li, Ma, & Vitányi, 2004; Lin, 1998). The algorithm emerging from this research project is also compared with manual assessment techniques and already established computerised assessment algorithms to determine its accuracy and reliability. Accuracy refers to the degree to which a measured result conforms to a standard or true value (Menditto, Patriarca, & Magnusson, 2006). Reliability is determined by evaluating whether the assessment results are stable and consistent (Carmines & Zeller, 1979).

1.2 Background

IT has become a ubiquitous part of modern civilisation (Easton, Easton, & Addo, 2006). It is transforming the way people live, conduct business and interact with one another, increasingly forming an integral part of people's daily lives. The global job market currently requires prospective employees to possess a basic to advanced level of computer proficiency with regard to word-processing, spreadsheet and presentation applications. The employment of personnel with adequate IT skills, in current and future technologies, is imperative in today's competitive global economy (Grant, Malloy, & Murphy, 2009). Computer proficiency comprises the *"knowledge and ability to use specific computer applications (spreadsheet,*

word processors, etc.)" (Grant et al., 2009, p. 142). Unfortunately, job applicants might not be truthful regarding their computer proficiency. A survey conducted by the Chartered Institute of Educational Assessors in 2009, revealed that 33% of applicants were not completely honest when compiling their curriculum vitae (Global Achievements, 2012). To ensure that applicants are proficient with regard to relevant IT skills, the assessment of prospective employees is becoming critical (Grant et al., 2009).

In the USA, Grade twelve learners have to pass a computer skills assessment test also in order to graduate (Grant et al., 2009). This is, however, not yet a general requirement in South African schools. Nevertheless, at the University of the Free State (UFS), in South Africa, most academic programmes include introductory computer proficiency modules. The necessity for IT skills assessment and the number of students concerned instigated a need for the computerised assessment of IT skills instead of manual assessment (Dowsing, Long, & Sleep, 1998; Hunt, Hughes, & Rowe, 2002).

1.2.1 The need for computerised assessment

Several factors contributed to the need for computerised assessment: First and foremost, manual assessment generates a heavy workload (Dowsing et al., 1998; Fonte, da Cruz, Gançarski, & Henriques, 2013; Jackson & Usher, 1997; Morris, 2003). Instructors and examiners have to assess each student's submission individually, which can be very time consuming and costly since it interferes with the instructors' other responsibilities (Burstein, Kukich, Wolff, Lu, & Chodorow, 1998b; Douce, Livingstone, & Orwell, 2005; Dowsing et al., 1998; Fonte et al., 2013; Hollingsworth, 1960; Page, 1966; Pérez-Marín, Pascual-Nieto, & Rodríguez, 2009).

Secondly, manual assessment is susceptible to human error and produces unreliable results. This is due to human markers being subjective or inconsistent with the allocation of marks and lacking the ability to focus cognitively for extended periods of time (Burstein et al., 1998b; Fonte et al., 2013; Morris, 2003; Page, 1966; Pérez-Marín et al., 2009; Whittington & Hunt, 1999). The popularity of online and blended learning[1], as well as students demanding immediate and quality feedback on their assessment results, also contributed to the need for

[1] Blended learning refers to the mixture of face-to-face instruction with computer mediated instruction through a learning management system (Suleman, 2008)

computerised assessment (Alber & Debiasi, 2013; Butcher, Swithenby, & Jordan, 2009; Hull, Powell, & Klein, 2011; Jordan & Mitchell, 2009; Pellet & Chevalier, 2014; Pieterse, 2013).

Computerised assessment has therefore been adopted by various disciplines in several academic institutions (Alber & Debiasi, 2013). It has been implemented in computer programming classes to facilitate the automated grading of computer programming assignments (Fonte et al., 2013; Hollingsworth, 1960; Pieterse, 2013). Likewise, the computerised assessment of free-text, in the form of essays or short answer responses, has also realised (Butcher & Jordan, 2010; Page, 1966; Pérez-Marín et al., 2009; Wang & Brown, 2007). Furthermore, the evaluation of mathematical formulas and the verification of virtual machine[2] system administration have been automated by means of computerised assessment (Alber & Debiasi, 2013).

Most importantly (since it involves the focus of this research project), the assessment of computer proficiency levels has been automated through the implementation of computerised assessment systems (Dowsing et al., 1998; Hunt et al., 2002; Tuparova & Tuparov, 2010). This research project, however, reveals certain gaps (see Section 1.3) within the algorithms that are implemented by these assessment systems (particularly word-processing assessment systems) and proposes an alternative algorithm to alleviate the problems.

1.2.2 Computerised assessment algorithms

Numerous algorithms have been developed with regard to the computerised assessment of free-text responses (Pérez-Marín et al., 2009), computer programming skills (Alber & Debiasi, 2013; Douce et al., 2005) and computer proficiency levels (Dowsing et al., 1998; Hill, 2011; Hunt et al., 2002; Tuparova & Tuparov, 2010; Wolters, 2010). Each algorithm implements specific similarity metrics to accomplish its objective of determining whether the response or solution, provided by the student, is correct. Although the algorithms that are used to assess computer proficiency levels differ from the algorithms used to assess computer programming skills and free-text responses, they do share certain commonalities. All the algorithms implement a comparison between an ideal solution to the problem, provided by the

[2] A virtual machine comprises software that emulates a physical computer system (Baumstark & Rudolph, 2013; Dean, 2012)

instructor or examiner, and the attempt submitted by the student. The method of implementation, however, differs among the various algorithms (see Chapter 3).

Microsoft Office is a globally used, well-respected application suite (Lánskỳ, Lokočr, Váňa, Hyskỳ, & Hájková, 2013; Strauss, 2008). Currently, Office Open XML (OOXML) is the default file format for Microsoft Office's word-processing documents, spreadsheets and presentations (Van Vugt, 2007). The implementation of the OOXML standard by Microsoft Office, constituted a fresh approach in the development of computerised assessment algorithms for Microsoft Word, Excel and PowerPoint (Lánskỳ et al., 2013; Pellet & Chevalier, 2014; Wolters, 2010). OOXML simplifies the comparison of identifiable elements in Microsoft Office documents to determine whether the solution, submitted by the student, matches the ideal solution provided by the examiner (Lánskỳ et al., 2013; Pellet & Chevalier, 2014; Wolters, 2010).

Research into the anatomy of OOXML-based documents delivered several alternative approaches with regard to the computerised assessment of these document types. These approaches include the procedural assessment of word-processing tasks (Lánskỳ et al., 2013), the extraction and comparison of formal document features (Pellet & Chevalier, 2014) and the implementation of parsing technologies (Wolters, 2010). Unfortunately, the algorithms that emerged from these approaches still have certain limitations, as discussed in the following section.

1.3 Problem statement and motivation

The algorithm developed by Lánskỳ et al. (2013) combines two approaches: a procedural assessment approach that evaluates the word-processing tasks by means of programmed procedures, and a comparative approach that matches the OOXML structure of the completed tasks within the submitted document, with the OOXML structure of the correct solution. Unfortunately, the procedural approach was found to be ineffective, since each procedure is linked to one particular task that students had to perform. Therefore, specifying additional tasks would involve creating and compiling additional procedures for each new task.

Matching the OOXML structure of the completed tasks with their corresponding solutions produced a problem of its own. The text content for a particular paragraph can have multiple

OOXML representations, as portrayed in Figure 1.1; however, both OOXML representations will produce exactly the same visible output. This complicates the direct comparison between the OOXML structure of documents (Lánskỳ et al., 2013).

```
<w:p>
  <w:r>
    <w:t>Docu</w:t>
  </w:r>
  <w:r>
    <w:t>ment Text</w:t>
  </w:r>
</w:p>
```

```
<w:p>
  <w:r>
    <w:t>Document Text</w:t>
  </w:r>
</w:p>
```

Figure 1.1 Two OOXML representations of the same document text content

An alternative approach by Wolters (2010) alleviates this problem through the application of a parsing algorithm, developed as part of the Office Skills Assessment Project (OSAP). The algorithm involves converting the documents (the student's submitted document, as well as the examiner's solution) into their object-oriented representations, removing all the irrelevant data from the documents, keeping the relevant document formatting information and merging the OOXML text elements within paragraphs. The parsed structure of the documents allow for the direct comparison between the student's submitted document and the ideal solution provided by the examiner.

Another challenge is the comparison between documents that contain different paragraph counts. In an effort to resolve this issue, Wolters (2010) applies an optimisation algorithm that matches and evaluates each paragraph of the student's attempt with each paragraph of the examiner's solution. An optimal one-to-one paragraph mapping of the documents is generated, based on the cross-paragraph evaluation results. Unfortunately, this produces unnecessary comparisons between the paragraphs of the documents. As a solution, this research project proposes an alternative method for matching documents with different paragraph counts.

Pellet and Chevalier (2014) followed a similar approach to the Moodle e-learning platform. Moodle's algorithm extracts content and property elements from the student's submitted document and matches it against a regular expression[3] representation of the ideal solution. A possible problem with Pellet and Chevalier's algorithm is its implementation in the Scala

[3] Regular expressions are sequences of characters that specify flexible search patterns within text (Friedl, 2006)

programming language, which requires a running Java Virtual Machine, since not all computer platforms might adhere to this software specification setup.

1.4 Main objective

The purpose of this research project is to create a computerised assessment algorithm for word-processing skills, derived from the positive aspects of the algorithms developed by Pellet and Chevalier (2014), Lánskỳ et al. (2013) and Wolters (2010), that addresses the issues (identified in Section 1.3) associated with these algorithms.

One of these issues is the comparison of documents with different paragraph counts and, consequently, how to effectively evaluate the documents without creating unnecessary paragraph comparisons. Another matter involves the standardisation of the OOXML structure within documents to ensure that the document content is represented by a uniform OOXML structure that allows for documents to be matched directly. As explained in Section 1.3, this issue was alleviated through the application of a parsing algorithm developed by Wolters (2010). This research project aims to solve the problem in a similar manner, but by different means through the implementation of classes and methods contained in the OOXML Software Development Kit (SDK) v2.0 for Visual Studio (Lánskỳ et al., 2013).

The main objective is to determine the accuracy and reliability (defined in Section 1.1) of the similarity metrics implemented by the algorithm emerging from this research project. This is accomplished by comparing the algorithm, in relation to a benchmark, with manual assessment metrics applied by human markers, as well as the similarity metrics embedded in established algorithms. In terms of this research project accuracy can be defined as how closely the assessment results, produced by human markers and the assessment algorithms involved, match the benchmark's assessment results. In the same sense, reliability can be defined as the degree to which human markers, as well as the assessment algorithms, produce exactly the same assessment results when re-assessing the same word-processing assignments.

1.5 Research design and methodology

Similarity metrics involved in the computerised assessment of word-processing skills are compared and investigated. To achieve this, the assessment results, produced by multiple human markers as well as four different computerised assessment algorithms (that implement

the relevant similarity metrics), are analysed. The assessment results are obtained from the evaluation of word-processing assignments submitted by first year students enrolled for a computer proficiency module at the UFS. A sample of 200 assignments has been selected from a population of 1466 successfully submitted assignments (see Chapter 7).

The first of four computerised assessment algorithms is an OOXML document analysis algorithm (see Chapter 4), developed as part of this research project and henceforth referred to as the *OOXML algorithm*. The newly proposed algorithm builds on existing OOXML-based algorithms, identified in Section 1.3, and attempts to alleviate the problems associated with these algorithms. The other three algorithms are well established and embedded in current software applications. The applications include WordGrader (Hill, 2011) that implements a document comparison algorithm, fComp (Miller & Myers, 1985) that implements a file comparison algorithm and Optical Character Recognition (OCR) software that utilises the Levenshtein string matching algorithm (Haldar & Mukhopadhyay, 2011; Levenshtein, 1966).

The proposed OOXML algorithm, the Levenshtein algorithm and fComp are bundled into an assessment system called Word Assessment Manager (WAM) that was developed as part of this research project (see Chapter 5). Using WAM, the selected word-processing assignments are separately assessed by each embedded algorithm and the assessment results captured. The selected sample of word-processing assignments is also re-evaluated by an instructor of the UFS computer proficiency module. This will provide assessment results that reflect greater consistency with regard to the allocation of marks. These assessment results will be used as a benchmark (see Section 6.4.3) for determining the accuracy and reliability of the similarity metrics embedded in the proposed OOXML algorithm.

A quantitative quasi-experiment[4] is conducted (see Chapter 7) to compare and analyse the assessment results produced by the similarity metrics of the relevant assessment methods. In this way, the accuracy and reliability of the OOXML algorithm can be evaluated.

1.6 Hypotheses

The main objective of this research project, specified in Section 1.4, identifies two tasks that need to be carried out. These tasks are executed by formulating certain hypotheses with regard

[4] A quasi-experiment is an experiment where the independent variable is not randomly assigned (Ellis, 1999)

to the proposed OOXML algorithm. *"A hypothesis is a tentative assumption or explanation for an observation, phenomenon or scientific problem that can be tested by further investigation"* (Leach, 2004, p. 58). Hypothesis testing is essentially used to determine the truth status of the hypothesis (Poletiek, 2013).

To determine the accuracy and reliability of the assessment results produced by the similarity metrics of the proposed OOXML algorithm, in comparison to multiple human markers, the following null hypotheses were formulated:

- $H_{0,1}$: There is no difference between the assessment accuracy of the OOXML algorithm and multiple human markers.
- $H_{0,2}$: There is no difference between the assessment reliability of the OOXML algorithm and multiple human markers.

To establish whether the assessment results produced by the proposed OOXML algorithm are comparable to the assessment results produced by WordGrader, fComp and the Levenshtein algorithm, the following null hypotheses were formulated:

- $H_{0,3}$: There is no difference between the assessment accuracy of the OOXML algorithm and the selected computerised assessment algorithms.
- $H_{0,4}$: There is no difference between the assessment reliability of the OOXML algorithm and the selected computerised assessment algorithms.

1.7 Contribution

This research project contributes to the knowledge of computerised assessment with regard to computer proficiency levels by filling the gaps identified in current and earlier assessment algorithms. It also strengthens the credibility of computerised assessment by reinforcing the accuracy and reliability thereof.

1.8 Project scope

The research conducted in this project is focused towards word-processing assignments in Microsoft Word. However, the outcome of this project is relevant to all word-processing

applications that implement the OOXML standard as a file format for word-processing documents, such as Microsoft Office, OpenOffice and LibreOffice to name a few.

1.9 Limitations

Research with regard to the computerised assessment of different OOXML-based documents has been introduced by Pellet and Chevalier (2014) and Wolters (2010). The algorithm emerging from this research project is limited to the computerised assessment of word-processing documents that conform to the OOXML standard. However, the algorithm could be adjusted to include the computerised assessment of other document types that implement the OOXML standard, such as spreadsheets and presentations. It should also be mentioned that the actions executed by the student to produce the final document cannot be determined from the algorithm developed as part of this research project. A different assessment approach that solves this limitation is discussed in Chapter 2, but is not implemented in this project.

1.10 Outline of the dissertation

Chapter 2 provides an in depth discussion on computerised assessment. Current computerised assessment systems are introduced and different assessment approaches identified, focusing on word-processing skills. Factors that contribute to the credibility of computerised assessment, such as validity and reliability, are discussed. Aspects that influence the computerised assessment of word-processing skills and the value that assessment guidelines provide, are also included in the discussion.

The debate continues in Chapter 3, where various algorithms with regard to computerised assessment are analysed. This analysis provides valuable insight into the similarity metrics involved and identifies possible similarity metrics that could be applied in terms of the computerised assessment of word-processing assignments. The accuracy whereby the relevant algorithms and their embedded similarity metrics produce assessment results is also pointed out.

The structural composition of OOXML-based word-processing documents is explained in Chapter 4. Algorithms that focus on the assessment of OOXML-based documents are analysed comprehensively and their flaws identified. An improved assessment algorithm for OOXML-based word documents that address these flaws is proposed.

In Chapter 6, the research design and methodology used to conduct an experimental study are discussed. In this experiment, word-processing assignments are evaluated via multiple human markers as well as four computerised assessment algorithms, employed by two computerised assessment systems. The operation of the two assessment systems, one developed as part of this research project and one commercially available, is demonstrated in Chapter 5.

A quasi-experimental research study is conducted in Chapter 7 to determine the accuracy and reliability of the similarity metrics implemented by the proposed OOXML algorithm developed as part of this project. This is accomplished by comparing it with manual assessment metrics, applied by human markers, as well as the similarity metrics embedded in established algorithms. The statistical test results are analysed and interpreted.

Chapter 8 draws conclusions from the results obtained in the experimental study. The findings with regard to the accuracy and reliability of the newly developed algorithm are discussed. The limitations of the study are addressed and future developments are mentioned as well.

1.11 Chapter summary

This chapter highlighted the importance of IT skills in modern society and reiterated the resulting need for the assessment of computer proficiency. The tendency towards computerised assessment was established and several factors that contributed to the need thereof were identified. Several disciplines that employ computerised assessment techniques were revealed, with the focus on computer proficiency assessment.

Current word-processing assessment algorithms were introduced and the latest developments with regard to the computerised assessment of OOXML-based documents were discussed briefly. The problems and limitations of current OOXML-based assessment algorithms that provided the motivation for this research project were identified.

The main objective of this research project was established and its contribution to the body of knowledge determined. The research design and methodology, hypotheses, project scope and limitations of the project were also discussed. In the following chapter computerised assessment is discussed in detail with regard to its origin, progress, assessment techniques, limitations and credibility.

Chapter 2

Computerised Assessment

This chapter focuses on the following aspects:

- **The beginning of computerised assessment and how it evolved with regard to various fields of study**
- **Assessment techniques implemented by computerised assessment systems**
- **Approaches focused towards the assessment of word-processing skills**
- **Aspects influencing the computerised assessment of word-processing skills**
- **Contributions to the credibility of computerised assessment**

2.1 Introduction

In Chapter 1, the aim and purpose of this research project were discussed. Chapters 2 and 3 deal with the application of computerised assessment in various fields of study and focus on the computerised assessment of word-processing skills. Chapter 2 discusses the origin of computerised assessment, the technological advancement thereof and introduces various computerised assessment systems that have been developed to assess free-text, computer programming and word-processing skills. Certain challenges with regard to the development of computerised assessment systems are presented and the limitations of computerised assessment are addressed. Various aspects that might influence the computerised assessment of word-processing skills are discussed, as well as aspects that might contribute to the credibility of computerised assessment. In the following chapter, the assessment algorithms that are implemented by these assessment systems are described and the accuracy of the assessment results they produce is discussed.

2.2 The evolution of computerised assessment systems

Research with regard to computerised assessment started as early as the nineteen sixties at the Rensselaer Polytechnic Institute, in Troy, New York, when Jack Hollingsworth developed an automatic grader for computer programming classes (Hollingsworth, 1960), referred to as the Rensselaer grader (Forsythe & Wirth, 1965). Forsythe and Wirth (1965) contributed to this field of research by presenting two programs, namely GRADER2 and Test, with the ability to grade computer programs, developed in the ALGOL computer programming language.

Even though most research with regard to computerised assessment was conducted within the computer programming field of study (Alber & Debiasi, 2013), other fields of study soon followed suit. The computerised assessment of free-text was introduced by Ellis Batten Page at Duke University in the United States of America (USA), when he developed Project Essay Grade (PEG) (Page, 1966, 1968, 1994; Page & Petersen, 1995). Various assessment approaches led to the development of many computerised assessment systems (Alber & Debiasi, 2013; Whittington & Hunt, 1999). These computerised assessment systems can be classified into different categories according to the type of responses or solutions that each system is able to assess. As already stated, some systems are able to grade computer programs (Forsythe & Wirth, 1965; Hollingsworth, 1960) while others are used to grade essays (Page, 1966, 1968, 1994; Page & Petersen, 1995; Pérez-Marín et al., 2009). Some can evaluate short-answer responses (Pérez-Marín et al., 2009), such as multiple-choice, true-false, single-word (Alfonseca et al., 2005) and single-sentence responses (Butcher & Jordan, 2010). Additionally, specific computerised assessment systems have been developed to evaluate mathematical formulas for mathematical and engineering assessments (Alber & Debiasi, 2013; Quah, Lim, Budi, & Lua, 2009), as well as systems that have the ability to assess administrative skills with regard to the setup of virtual machines - software that emulates a physical computer system (Baumstark & Rudolph, 2013; Dean, 2012).

Another kind of computerised assessment came into existence as skills in information technology (IT), and the assessment thereof as pointed out by Tuparova and Tuparov (2010), became imperative. The Quality Assurance Agency in the United Kingdom publishes benchmark statements for subjects presented by Higher Education institutions. Transferable skills[1], such as word-processing, are included as a core component in most Higher Education programmes (Hunt et al., 2002). In 1996, Dowsing, Long and Sleep released a fully functional computerised system with the ability to assess word-processing skills (Dowsing et al., 1998). This constituted a larger project, called the Computer-aided Assessment of Transferable Skills (CATS), to develop computerised assessment systems for several IT skills. CATS and other assessment systems that were also developed to assess IT skills are discussed further in Section 2.2.3. Although this research project focuses on the computerised assessment of word-processing skills, additional fields of study are included in this discussion to demonstrate the tendency towards the implementation of computerised assessment systems in

[1] Transferable skills are skills obtained by students to be applied within their qualified professions (Fallows & Steven, 2013)

various fields of study. It also increases the credibility of computerised assessment, as well as highlights the necessity and purpose thereof, as discussed in Section 2.4. Examining the algorithms and metrics involved in the computerised assessment of other fields of study, might also provide useful insight into improving current assessment algorithms for word-processing skills. Therefore, various assessment systems that implement free-text and computer programming assessment algorithms are discussed in Sections 2.2.1 and 2.2.2.

2.2.1 Computerised assessment of free-text

The latest improvements with regard to Natural Language Processing (NLP) and Machine Learning techniques, the growing popularity of blended[2] and online learning, as well as the lack of quality feedback with regard to student assessments due to time constraints, have encouraged the computerised assessment of free-text (Garrison & Kanuka, 2004; Lim, Morris, & Kupritz, 2007; Pérez-Marín et al., 2009). To effectively measure either the conceptual accuracy of free-text responses or the technical writing quality thereof, or both according to Christie (2003), forms the key issue when developing a computerised assessment system (Pérez-Marín et al., 2009). Most approaches to the assessment of essay content compare the student's attempt to an ideal solution provided by the examiner. Analysing abstract measures, such as variety, fluency or quality, to assess the technical writing quality of essays requires a different approach. One traditional approach involves analysing direct features of the free-text, for example, word number or word length from which the abstract measures are deduced (Christie, 2003; Page, 1966).

Nevertheless, according to Butcher and Jordan (2010), various computerised assessment systems have limitations with regard to the matching of short-answer free-text responses. These systems are often only able to match short answers containing one or two words and seldom provide for negation, synonyms and alternative spelling, or word order. Therefore, questions that require more complex responses are converted to questions that require multiple-choice, true-false or single-word responses. However, computerised assessment methods for short-answer responses do not really measure the depth of a student's knowledge (Whittington & Hunt, 1999). Also, according to Foltz, Laham and Landauer (1999), "*essay-*

[2] Blended learning comprises of combining face-to-face instruction with computer mediated instruction through a learning management system (Suleman, 2008)

based testing is thought to encourage a better conceptual understanding of the material and to reflect a deeper, more useful level of knowledge and application by students" (p. 939).

This led to further computerised assessment systems like the Electronic Essay Rater (e-rater) (Burstein, Kukich, Wolff, Lu, & Chodorow, 1998a; Burstein et al., 1998b; Burstein, Kukich, Wolff, Lu, & Chodorow, 1998c; Burstein, Leacock, & Swartz, 2001), the Intelligent Essay Marking System (IEMS) (Ming, Mikhailov, & Kuan, 1999), the Intelligent Essay Assessor (IEA) (Foltz et al., 1999), IntelliMetric (Pérez-Marín et al., 2009; Wang & Brown, 2007), the Automated Text Marker (ATM) (Callear, Jerrams-Smith, & Soh, 2001) and Atenea (Pérez, Alfonseca, & Rodríguez, 2004) to name a few; all of which were developed to assess essay-based free-text responses. Research in computerised assessment of short-answer free-text responses contributed to the development of Concept-rater (C-rater) (Burstein et al., 2001), FreeText Author (Jordan & Mitchell, 2009) and OpenMark (Butcher, 2008) among others.

All the computerised assessment systems of free-text mentioned above implement specific techniques, containing algorithms, to either assess the technical writing quality of essays, including content, organisation, style, mechanics and creativity, or to assess the conceptual accuracy of short answers (Page, 1968, 1994; Pérez-Marín et al., 2009). Some systems are able to assess both the technical writing quality and the conceptual accuracy of essays (Valenti, Neri, & Cucchiarelli, 2003). The assessment techniques for free-text responses, implemented by specific computerised assessment systems, are shown in Table 2.1. The algorithms contained in these techniques are discussed in Chapter 3.

Table 2.1 Computerised assessment techniques for free-text responses (Adapted from Pérez-Marín et al., 2009)

System	Technique	Assessment object
Atenea	N-gram co-occurrence metrics	Essays
Automated Text Marker	Information extraction	Essays
Concept-rater	Natural language processing	Short-answers
Electronic Essay Rater	Natural language processing	Essays
FreeText Author	Information extraction	Short-answers
Intelligent Essay Assessor	Latent semantic analysis	Essays
Intelligent Essay Marking System	Pattern matching with clustering	Essays
IntelliMetric	Natural language processing / AI	Essays
OpenMark	Algorithmic keyword manipulation	Short-answers
Project Essay Grade	Measure surface linguistic features	Essays

With free-text responses, the text content of an answer is provided by the student in response to a question. Therefore, the purpose of a computerised assessment system would be to assess the text content of short-answer free-text responses as well as the technical writing quality of essay-based free-text responses. In word-processing skills assessment, the examiner provides the text content of the document on which the student has to perform specific tasks, as described in Section 2.2.3. It would therefore be irrelevant to assess the text content of the document. Instead, the properties of the document, such as the layout, styling and formatting of the document, should be assessed.

2.2.2 Automated assessment of computer programming skills

The assessment of computer programming skills has been computerised from the start (Douce et al., 2005; Forsythe & Wirth, 1965; Hollingsworth, 1960), since it involves the compilation and execution of program code on a computer (Edwards, 2003). However, computer programming assessment was not automated before the existence of the Rensselaer grader. Students at the Rensselaer Polytechnic Institute had to run their own programs, written in assembly language and compiled on punch cards, by taking turns on an IBM 650 computer. To run a single program took between 30 and 60 seconds per student on average. This was reduced by 87.5% with the implementation of the Rensselaer grader as higher numbers of students could now be accommodated in computer programming courses (Douce et al., 2005; Hollingsworth, 1960).

Automated assessment systems for computer program code can be classified into two categories according to the analysis technique they implement, namely dynamic and static code analysis. Dynamic code analysis is used to assess the functionality and efficiency of computer programs and involves compiling and running the program code (Fonte et al., 2013; Pieterse, 2013). Various authors agree that computer programs should be evaluated with regard to functionality to assess whether the program compiles and executes its tasks successfully (Edwards, 2003; Forsythe & Wirth, 1965; Helmick, 2007; Hext & Winings, 1969; Hollingsworth, 1960; Hull et al., 2011; Jackson & Usher, 1997; Pieterse, 2013; Suleman, 2008; Von Matt, 1994). The functionality of computer programs is assessed by submitting them to several test cases, developed by the instructor. Each test case specifies test input data of which the resulting output is known (Fonte et al., 2013; Leal, Moreira, & Moreira, 1998; Pieterse, 2013). Additionally, some assessment systems also assess the

efficiency of computer programs by evaluating the programs' utilisation of system resources, such as memory, storage space and CPU time (Forsythe & Wirth, 1965; Hext & Winings, 1969; Jackson & Usher, 1997; Pieterse, 2013). Dynamic code analysis techniques have been implemented since the conception of computer program assessment (Forsythe & Wirth, 1965; Hollingsworth, 1960) by automated assessment systems, such as the Rensselaer grader (Hollingsworth, 1960), GRADER2 (Forsythe & Wirth, 1965), Test (Forsythe & Wirth, 1965), BAGS (Hext & Winings, 1969), Kassandra (Von Matt, 1994), Automatic Marker (Suleman, 2008), Infandango (Hull et al., 2011), and Fitchfork (Pieterse, 2013).

Static code analysis involves analysing a computer program without compiling or running the program code and is used to assess the style and structure of the source code (Fonte et al., 2013). Computer programming paradigms have changed over the years as computer technology and programming languages advanced (Papajorgji & Pardalos, 2014; Samuel & Kovalan, 2014). When investigating the assessment of computer programming skills, it is important to consider the computer technology, programming languages and programming paradigms involved at the time of a particular assessment method, since these influence the criteria whereby computer programs are evaluated. Automated assessment systems that apply static code analysis techniques, in addition to dynamic code analysis, include ASSYST (Jackson & Usher, 1997), Web-CAT Grader (Edwards, 2003) and AutoGrader (Helmick, 2007). Table 2.2 portrays the code analysis techniques and relevant programming languages that are supported by the various automated assessment systems discussed in this section.

Table 2.2 Dynamic and static code analysis systems

System	Code analysis technique	Target programming language/s
ASSYST	Static and dynamic	Ada, C
AutoGrader	Static and dynamic	Java
Automatic Marker	Dynamic	Java
BAGS	Dynamic	ALGOL, MINIGOL, KFD 9 assembly code
Fitchfork	Dynamic	C++
GRADER2	Dynamic	ALGOL
Infandango	Dynamic	Java
Kassandra	Dynamic	Fortran, Maple, Matlab (Scientific computing)
Rensselaer	Dynamic	Assembly
Test	Dynamic	ALGOL
Web-CAT Grader	Static and dynamic	Java

Research on the automated assessment of computer programming skills should provide valuable knowledge that could be implemented by the computerised assessment of word-processing skills. Dynamic code analysis compares a computer program's output results with the expected output results as specified by the instructor (Fonte et al., 2013). In a similar manner, word-processing skills are evaluated by comparing the final document, produced by the student, with the memorandum provided by the examiner (Dowsing et al., 1998). Analysing the similarity metrics that are implemented by the dynamic code analysis algorithms to assess computer programming skills could provide valuable insight into their operation. This could determine whether these or similar metrics can be implemented in the assessment of word-processing skills.

The structure of a computer program's source code is analysed through static code analysis techniques (Fonte et al., 2013). The algorithms involved in the static code analysis of computer programs might be useful with regard to word-processing skills, namely in assessing the style and formatting of the document produced by the student. The algorithms within these automated assessment techniques of computer programming skills are examined further in Chapter 3.

2.2.3 Computerised assessment of word-processing skills

Before the conception of CATS in 1996, traditional assessment techniques were predominantly used to evaluate IT skills including word-processing. Traditional assessment techniques consist of examiners marking the final document submitted by a student and observing the methods the student used to produce the final document. The examiners mark the final document by comparing it with the memorandum (Dowsing et al., 1998). The process where students submit a final document to be evaluated, containing a demonstration of their IT skills, is also referred to as real-life performance-based assessment (Tuparova & Tuparov, 2010).

According to Dowsing (2000), a computerised assessment system is ideal for assessing word-processing skills, since the assessment is conducted on a computer that can be used to capture the information needed for the computerised assessment. Earlier assessment methods for IT skills, implemented by computerised systems, mostly utilised multiple choice questions where a student had to choose the correct answer to a question from a list of possible answers

(Dowsing et al., 1998; Hunt et al., 2002). Earlier systems also implemented fixed function tests where a student had to perform a single word-processing action at a time. Systems like these included the PC Driving Test, developed by the National Computer Centre (NCC) in the United Kingdom, and the European Driving Test initiated by the European Community. Examination boards considered word-processing assessments of this kind, but the majority rejected it (Dowsing et al., 1998). This was due to the fact that multiple choice and structured questions do not assess the practical application of knowledge, but rather the knowledge itself (Biggs & Tang, 2011; Dowsing et al., 1998; Hunt et al., 2002). The student's word-processing ability is also not fully assessed by means of fixed function tests since it only evaluates whether the student can complete a specific word-processing task (Dowsing et al., 1998). Dowsing also states that *"the only satisfactory way to measure basic IT skills is to assess the process or result of candidates undertaking typical tasks such as word-processing assignments"* (Dowsing, 2000, p. 453).

The issues arising from earlier systems led to the development of computerised assessment systems that would address these shortcomings. Current computerised assessment systems can be classified into two categories according to the assessment technique they implement, namely assessment through event stream analysis and assessment through document analysis. Event stream analysis involves analysing the actions that a student executed to complete a specific word-processing task (Dowsing, 2000; Dowsing et al., 1998). Systems such as ActivTest (Activ Training, n.d.), CompAssess (Business Wire, 2003; Training Zone, 2000), FastPath (ISV, n.d.), MyITLab (Pearson, n.d.), Skills Assessment Manager (SAM) (Cengage Learning, n.d.), SIMnet (McGraw-Hill Education, n.d.) and Train and Assess IT (Robbins & Zhou, 2007) assess word-processing skills through event stream analysis within a simulated word-processing environment (Hill, 2011; Tuparova & Tuparov, 2010). Document analysis involves comparing the final document with the memorandum, i.e. an illustration of how the final document should look when all the word-processing tasks have been executed successfully. This technique is implemented by systems such as the CATS word-processing assessor (Dowsing et al., 1998), the MS Word Skills Assessment System (Tuparova & Tuparov, 2010), MyITLab Grader (Pearson, n.d.), SAM Projects (Cengage Learning, n.d.) and Word Grader (Hill, 2011). A system called Formative Automated Computer Testing (FACT) combines event stream analysis with document analysis. FACT's event stream analysis provides meaningful assessment by assessing word-processing skills in an actual word-processing environment, such as Microsoft Word (Hunt et al., 2002).

Newer systems, such as the OSAP assessment tool developed as part of the Office Skill Assessment Project (OSAP) (Wolters, 2010), AUTOPOT (Lánskỳ et al., 2013), as well as another tool developed by Pellet and Chevalier (2014) from the Lausanne University of Teaching Education (HEP-VD) in Switzerland, also implement document analysis techniques. The latter assessment tool is hereafter referred to as the *HEP-VD Grader*. These newer systems, however, only focus on assessing OOXML-based documents, i.e. documents that adhere to the OOXML document format standard implemented by Microsoft Office (Van Vugt, 2007).

Table 2.3 portrays computerised assessment systems that implement event stream analysis against those that implement document analysis. As illustrated, some event stream analysis and document analysis systems are related. The algorithms contained within these word-processing assessment systems are discussed in Chapter 3.

Table 2.3 Event stream analysis systems vs. document analysis systems

	Event stream analysis	**Document analysis**
Related	FACT	FACT
	MyITLab	MyITLab Grader
	Skills Assessment Manager (SAM)	SAM Projects
Unrelated	ActivTest	AUTOPOT
	CompAssess	CATS word-processing assessor
	FastPath	HEP-VD Grader
	SIMnet	MS Word Skills Assessment System
	Train and Assess IT	OSAP assessment tool
		Word Grader

This research project involves the development of an OOXML-based assessment algorithm that emulates the traditional assessment of word-processing skills through document analysis. The developed algorithm will be compared with other known algorithms, relevant to document analysis, to evaluate the efficiency and reliability of the algorithm when assessing word-processing skills. The algorithm and its implementation are described in Chapter 4.

2.3 Approaches to assessing word-processing skills

Human examiners usually evaluate word-processing skills by firstly assessing the accuracy of the final document produced by the student, i.e. whether the student knows the word

processor's fundamental editing operations and how to implement these operations to produce a particular document. This forms part of the lower levels of assessment and is relatively simple to assess, for instance, assessing whether copying and pasting of a word or phrase was implemented accurately. Secondly, the appearance of the final document is also assessed. This forms part of a higher level of assessment and involves the student's ability to apply these word-processing skills in real life situations to produce professional looking documents needed to fulfil everyday tasks. Unfortunately, it is difficult to separate these two aspects and therefore it is necessary to assess the accuracy in combination with the appearance of the final document (Dowsing et al., 1998).

As with the computerised assessment of free-text, specific techniques, containing algorithms, are implemented during the computerised assessment of word-processing skills. However, other than the assessment of free-text answers, word-processing skills are assessed by evaluating the accuracy with which a student completes a specific task or combination of tasks resulting in a final document (Hill, 2011; Tuparova & Tuparov, 2010). Tuparova and Tuparov (2010) outline two approaches for assessing IT skills, and therefore word-processing skills: real-life performance-based and simulation-based. The real-life performance-based approach consists of assessing word-processing skills through document analysis, which involves the analysis of the document with regard to its content, formatting and layout. The assessment of word-processing skills through event stream analysis, by analysing the events that were carried out to produce the final document, is referred to as the simulation-based approach. As it indicates, this approach involves the use of a simulated word-processing environment (Dowsing et al., 1998; Hill, 2011; Tuparova & Tuparov, 2010).

This research project involves the application of document analysis to assess the competency of students when using Microsoft Word. Therefore, a real-life performance-based skills assessment approach is required with regard to Microsoft Word assignments, where students have to perform specific tasks to deliver a final document. The tasks include deleting, copying and pasting, and changing the format of the text as well as altering the layout of the document. The relevant assignment tasks are specified in Appendix A and the assessment thereof is discussed in Chapter 6.

Four different algorithms, employing specific similarity metrics, will be used to compare the final document produced by the student, with the memorandum. The project focuses on

comparing the results obtained from the implementation of these similarity metrics with regard to the assessment of word-processing skills. To achieve this, it is necessary to develop a computerised assessment system that can implement these algorithms individually. The details of the proposed assessment system are discussed in Chapter 5.

2.4 The credibility of computerised assessment

Various assessment systems have proven their credibility with regard to the automation of assessing free-text responses, computer programming and word-processing skills. According to Butcher and Jordan (2010), the use of computers for evaluating short-answer questions should have the same outcome as when these questions are marked by humans. They verified this by comparing the evaluation results of free-text responses marked by humans with the evaluation results as assessed by three computerised assessment systems: FreeText Author (from Intelligent Assessment Technologies) (Jordan & Mitchell, 2009), regular expressions[3] and OpenMark (Butcher, 2008).

Contrary to this, Wang and Brown (2007) demonstrated in a casual-comparative study that a significant difference exists between the essay grades produced by an automated computerised assessment system, called IntelliMetric, and the grades assigned by human examiners. Many critics have labelled the computerised assessment of essays as being shallow, biased, inaccurate and not transparent (Deane, 2013; Haswell & Wilson, 2013).

Nevertheless, other research projects show a high level of agreement between the computerised assessment results and the results for essays assessed by human examiners (Pérez-Marín et al., 2009; Wang & Brown, 2007). Research has also demonstrated reliable (Hunt et al., 2002) and extremely accurate (Hill, 2011) assessment results produced by computerised assessment systems with regard to word-processing skills, as well as the automation of computer programming skills assessment (Suleman, 2008). Several authors have experienced that an automated assessment system has the ability to identify errors in the source code of a student's solution to a computer programming assignment that were completely missed by human markers (Jackson & Usher, 1997; Von Matt, 1994).

[3] Regular expressions are sequences of characters that specify flexible search patterns within text (Friedl, 2006)

As in Dowsing et al. (1998), the assessment algorithm developed as part of this research project obtains its assessment results from the comparison between the student's document and the memorandum. It is, however, necessary to determine the validity of the assessment results, i.e. whether the assessment results truly reflect the competency of the student's word-processing skills (Carmines & Zeller, 1979). The reliability of the similarity metrics, applied to measure the student's word-processing skills, also needs to be determined to ensure that the computerised assessment system produces stable and consistent results (Carmines & Zeller, 1979). This will contribute to the credibility of computerised assessment. The question is how to measure the validity of assessment results and the reliability of the similarity metrics embedded in the document analysis algorithms.

2.4.1 The reliability and validity of computerised assessment

The reliability of manual assessment depends on the type of responses or solutions being assessed. According to Page (1966) and Pérez-Marín et al. (2009), manual assessment of free-text responses is not very reliable. The same free-text response might obtain a different assessment result when evaluated by another examiner since there exists no single correct response to a question. An examiner might even assign a different result when re-assessing the same free-text response to a question (Page, 1966). Similarly, the manual assessment of programming assignments is prone to human error also. Fatigue, favouritism and inconsistency are the main reasons why different human markers would assign different marks to the same student's programming solution (Fonte et al., 2013; Jackson & Usher, 1997). To increase the reliability of manual free-text assessment results, free-text responses are usually double checked by other examiners, which subsequently makes the assessment process very expensive and time-consuming (Page, 1966; Pérez-Marín et al., 2009).

Other than with free-text responses and programming solutions, the final document of a word-processing assignment has only one correct outcome. However, according to Dowsing et al. (1998), the manual assessment of word-processing skills may also produce inconsistent assessment results, even though the human examiners are provided with detailed guidelines on how to allocate marks. This might be due to human examiners exercising their own discretion when assessing word-processing skills. While some examiners might allocate marks strictly according to the memorandum, others might decide that the font colour of a particular text phrase, in the document being marked, is close enough to the font colour

specified in the memorandum. This, combined with the high pass rate for word-processing skills assessments, and the high number of perfect assessment results, bring the reliability of manual assessment into question (Dowsing et al., 1998). Computerised assessment, on the other hand, enables the reliable evaluation of students' competency in word-processing (Hunt et al., 2002). It almost instantaneously delivers consistent assessment results and treats all students equally with regard to word-processing, computer programming and free-text responses (Jackson & Usher, 1997; Page, 1966; Pellet & Chevalier, 2014).

Furthermore, to ensure that manual assessment results are a true reflection of a student's actual word-processing abilities, the actions executed by the student to produce the solution also need to be assessed (Dowsing, 2000; Dowsing et al., 1998; Hunt et al., 2002; Tuparova & Tuparov, 2010). However, this is again very time-consuming and creates a very high workload on the part of the examiners (Dowsing et al., 1998; Page, 1966; Pérez-Marín et al., 2009). Computerised assessment provides a solution to this problem through event stream analysis that enables the automated assessment of the actions taken to perform a specific word-processing task, thus providing valid assessment results with regard to the competency of word-processing skills (Dowsing, 2000; Hunt et al., 2002). However, the focus of this research project is not event stream analysis, but rather document analysis and the relevant assessment techniques (described in Chapter 3) implemented by these assessment systems, as mentioned in Section 2.2.3.

2.4.2 Providing a reliable benchmark

This research project relies on providing a reliable benchmark for the comparison of the assessment results produced by four selected assessment algorithms (see Section 3.6). As stated in Section 1.5, the benchmark is provided by the manual assessment of word-processing assignments. In order to ensure reliable benchmark results, the issues mentioned in Section 2.4.1 with regard to manual assessment have to be considered. Therefore, the benchmark results should be obtained by evaluating the word-processing assignments by a single human marker to reduce any inconsistencies relating to the use of multiple markers. Furthermore, the memorandum should be strictly adhered to in an effort to eliminate any chances of inconsistent mark allocation due to own discretion or favouritism. Finally, a reasonable amount of time should be granted to the marker to evaluate the word-processing

assignments, to ensure that the assessment results are not affected by exhaustion or the inability to concentrate.

2.5 The importance of assessment guidelines

Whether essays, short-answers, programming skills or word-processing skills are assessed, guidelines for the allocation of marks must be provided. This can be complemented by standardisation discussions where examiners and markers get together in an attempt to ensure consistent assessment results (Dowsing et al., 1998). Basic assessment techniques involve providing the correct solution to the problem or question that the student had to solve. A comparison between the student's attempt and the correct solution provided by the examiner should verify whether the student's attempt is correct or not. If it matches the examiner's solution, the student's attempt has been implemented correctly (Dowsing et al., 1998; Fonte et al., 2013). The opposite, however, is not true with regard to incorrectly implemented student attempts. This conclusion is derived from complications that may arise from several causes that are discussed below, as specified by Dowsing et al. (1998).

2.5.1 Complications influencing computerised assessment

Some student attempts might differ slightly from the solution provided in the memorandum, but might still be acceptable as a viable solution to the problem. For instance, a heading that had to be centred may be slightly off centre, due to the student using an alternative method when centring the heading (Dowsing et al., 1998). This situation is explained by the following example:

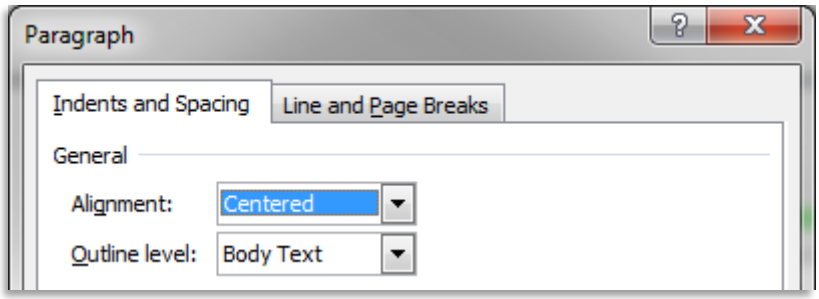


Figure 2.1 Paragraph properties dialog box

The correct method to centre a heading may be to set the alignment property of the heading's paragraph to *centered*, as illustrated in Figure 2.1 (Beskeen, 2013). This method modifies the properties of the specific paragraph in the document as indicated in Figure 2.2. The property

relevant to the centring of the heading is highlighted in Figure 2.2 and displays the modified property in OOXML format as contained within the document (Van Vugt, 2007). Chapter 4 contains a detailed discussion on the open standard of the XML format, referred to as OOXML, contained within word-processing documents.

```
<w:pPr>
  <w:spacing w:after="0" w:line="240" w:lineRule="auto"/>
  <w:jc w:val="center"/>
  <w:rPr>
    <w:rFonts w:ascii="Arial" w:hAnsi="Arial" w:cs="Arial"/>
    <w:b/>
    <w:color w:val="00B050"/>
    <w:sz w:val="48"/>
    <w:szCs w:val="48"/>
    <w:lang w:val="en-US"/>
  </w:rPr>
</w:pPr>
```

Figure 2.2 Centring attribute within the alignment property of the paragraph

The student, however, may have centred the heading by inserting a tab space in front of the heading. This is usually implemented by setting tab stop positions in the relevant paragraph as illustrated in Figure 2.3 (Beskeen, 2013). This method relies on the student's ability to accurately set a centre alignment tab stop position that aligns the centre of the heading with the centre of the page.

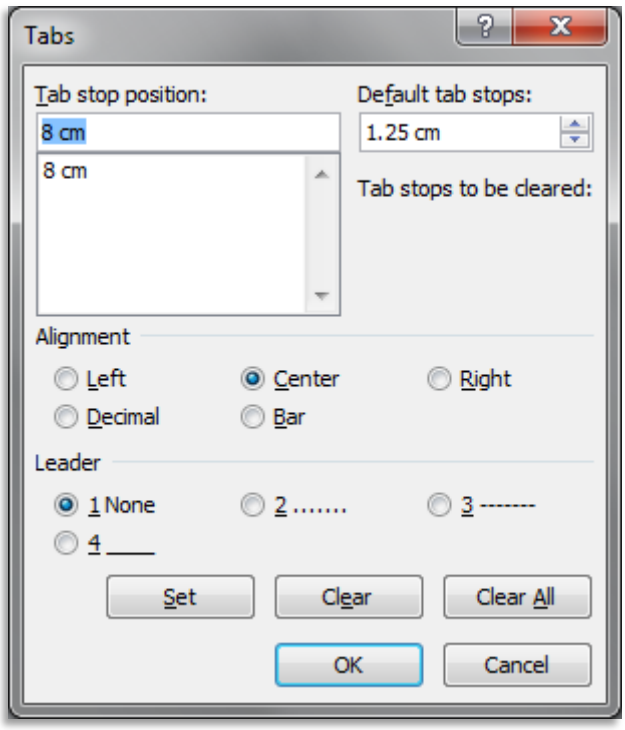


Figure 2.3 Tab stop dialog box

Implementing this method will modify the properties of the paragraph that contains the heading by adding a tab stop property with a centring attribute as indicated in Figure 2.4 (Van Vugt, 2007). In a similar manner, a left or right alignment tab stop position can be set to centre the heading on the page. Unfortunately, using tab stop positions may result in the heading being slightly off-centre, since the positions are manually specified by the student.

```
<w:pPr>
  <w:spacing w:after="0" w:line="240" w:lineRule="auto"/>
  <w:tabs>
    <w:tab w:val="center" w:pos="4536"/>
  </w:tabs>
  <w:rPr>
    <w:rFonts w:ascii="Arial" w:hAnsi="Arial" w:cs="Arial"/>
    <w:b/>
    <w:color w:val="00B050"/>
    <w:sz w:val="48"/>
    <w:szCs w:val="48"/>
    <w:lang w:val="en-US"/>
  </w:rPr>
</w:pPr>
```

Figure 2.4 Centring attribute within the tab stop property of the paragraph

The question is whether to accept the slightly off-centre heading as correct or not, or to only accept the centring of the heading by means of the alignment property. In a situation such this, where the rules defining acceptable and inacceptable solutions are often vague, human judgement is necessary to assess whether the student's attempt is acceptable or not. This might lead to the student's attempt being graded with regard to its degree of correctness (Dowsing et al., 1998). According to Dowsing et al. (1998), the implementation of assessment criteria for situations like these, within an automated computerised system, is in principle possible; however, in practice the number of possible variations to the solution might hamper the feasibility thereof.

Another situation that might arise is the possibility of multiple correct solutions to the problem. In this case, the student's attempt should be compared to all possible correct solutions presented in the memorandum (Dowsing et al., 1998). For example, the centring of a heading by setting tab stop positions may be included in the memorandum as an acceptable alternative solution, as well as other possible solutions. The student's attempt should therefore be compared to all these possible solutions. However, it might also be decided to accept only the most effective or efficient solution to the problem, increasing the stringency of the assessment, as is the case with computer programming assignments (Suleman, 2008).

A third complication that affects the computerised assessment of word-processing skills is the deduction of a student's ability to demonstrate a specific skill by assessing the outcome of the solution that the student implemented. The student's attempt might not directly indicate the skill set needed to implement the solution (Dowsing et al., 1998). For example, the student might be asked to replace all the occurrences of a specific phrase in a document with another specified phrase. According to Beskeen (2013), the fastest action by which this can be accomplished is with the *find and replace* utility of the word processor, seen in Figure 2.5.

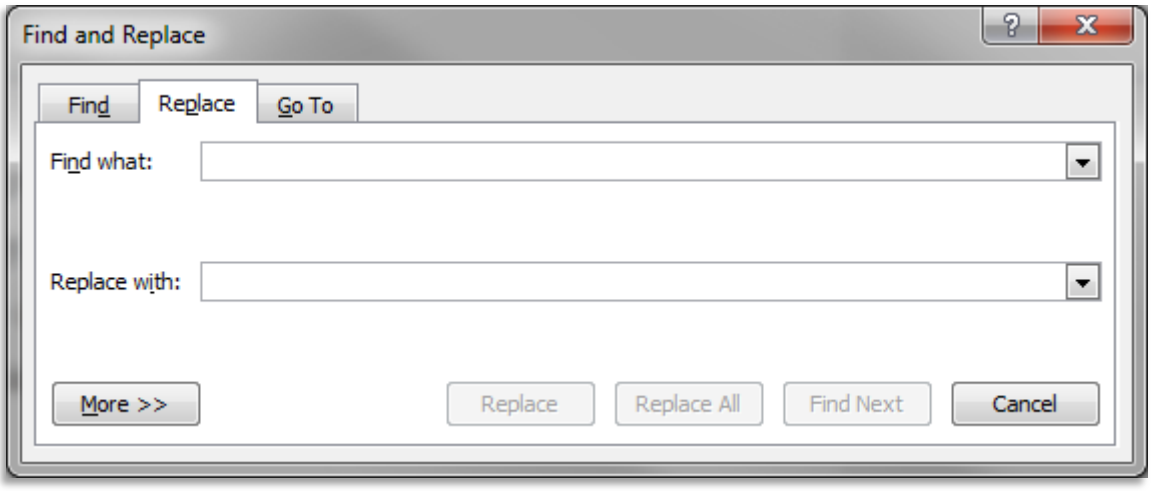


Figure 2.5 Find and Replace dialog box

Unfortunately, after all the occurrences of the specific word or phrase have been replaced, there is no evidence in the document to indicate whether the student utilised the *find and replace* utility, or searched for and replaced each occurrence manually. This is confirmed by comparing the underlying OOXML of a document before and after replacing a designated word or phrase with another word or phrase, as illustrated in Figures 2.6, 2.7 and 2.8.

```
<w:r>
  <w:rPr>
    <w:sz w:val="48"/>
    <w:szCs w:val="48"/>
    <w:lang w:val="en-US"/>
  </w:rPr>
  <w:t>
    This example illustrates a third complication that can influence the
    computerised assessment of word-processing skills.
  </w:t>
</w:r>
```

Figure 2.6 Text phrase to be replaced

Figure 2.6 displays the underlying OOXML of a document that contains the highlighted phrase, "illustrates a third complication", that needs to be replaced with the phrase, "demonstrates one of three complications". After the phrase has been replaced by means of the *find and replace* utility, the OOXML content within the document differs only with regard to the phrase that has been replaced, as illustrated in Figure 2.7.

```
<w:r>
  <w:rPr>
    <w:sz w:val="48"/>
    <w:szCs w:val="48"/>
    <w:lang w:val="en-US"/>
  </w:rPr>
  <w:t>
    This example demonstrates one of three complications that can influence the
    computerised assessment of word-processing skills.
  </w:t>
</w:r>
```

Figure 2.7 Text phrase replaced by the find and replace utility

Replacing the specified phrase manually results in the same outcome as when utilising the *find and replace* utility, as illustrated by Figure 2.8, leaving no indication as to the method implemented to obtain the final result.

```
<w:r>
  <w:rPr>
    <w:sz w:val="48"/>
    <w:szCs w:val="48"/>
    <w:lang w:val="en-US"/>
  </w:rPr>
  <w:t>
    This example demonstrates one of three complications that can influence the
    computerised assessment of word-processing skills.
  </w:t>
</w:r>
```

Figure 2.8 Text phrase replaced manually

According to Suleman (2008), the allocation of partial marks for a solution that differs slightly from the correct solution specified in the memorandum, increases the complexity of the assessment system's design significantly. Therefore, computerised assessment systems with the capability of assigning partial marks to partially correct solutions are seldom designed. Instead, most assessment systems apply more stringent assessment criteria as marks are allocated only if the student's solution is successful when compared with the solution provided in the memorandum (Fonte et al., 2013; Suleman, 2008).

2.5.2 Applicable assessment guidelines

In light of the above mentioned complications that influence the computerised assessment of word-processing skills, as stipulated by Dowsing et al. (1998), it is important to realise the necessity of guidelines with regard to the allocation of marks. From these complications it is clear that the guidelines should include the stringency of mark allocation, i.e. how severely wrong solutions should be penalised by deducting marks. It should also specify whether or not multiple solutions are applicable to a specific problem.

With regard to manual assessment, if a situation arises where the student provides an alternative viable solution not stipulated in the memorandum, the examiner could exercise intelligence to assess the alternative solution (Dowsing et al., 1998). To implement the same intelligence in a computerised assessment system would not only require provision for all possible solutions to each problem, but also the level of acceptability for all possible solutions. Since the feasibility to specify how every situation should be handled is hampered by the number of possible solutions (Dowsing et al., 1998), the assessment technique implemented by this research project only provides a single solution to each problem. This should be taken into account when conducting the study as it might have a significant effect on the stringency of the assessments and, therefore, the assessment results.

As explained in Section 2.5.1, directly assessing the skill set needed to implement a solution to a specific problem might not be possible from the solution contained in the student's final document. Mere document analysis will not be able to ascertain the actions implemented by the student to complete a specific word-processing task (Dowsing, 2000; Dowsing et al., 1998). To counter this complication another technique is required, namely the assessment of word-processing skills through event stream analysis, briefly introduced in Section 2.2.3. The student's ability to demonstrate a specific skill can be assessed directly by analysing the actions that were implemented to produce the solution to a specific problem, (Dowsing, 2000; Dowsing et al., 1998). However, the computerised assessment algorithm developed as part of this research project, will not apply event stream analysis, but will instead only be able to assess the final document submitted by the student, through document analysis.

Since only the outcome of the student's actions is visible in the final document, it will not be known whether the student applied the faster, more correct methods to complete a specific

task, or completed a specific task by means of slower, more complicated methods. The question is how to encourage the student to implement the faster, more correct methods. One solution to this complication might be to allocate less time so that the student is unable to complete the assignment if the faster, more correct methods are not applied. The effect of providing less time to complete an assignment does, however, not fall within the focus of this research project.

2.6 Chapter summary

In this chapter, the evolution of computerised assessment from its conception up to current developments was discussed. The discussion included the computerised assessment of free-text with examples (see Table 2.1) of several computerised assessment systems. A distinction was made between the computerised assessment of essays and short free-text responses. Automation of the computerised assessment of computer programming skills was also included in the discussion. Two computer program code analysis techniques, namely dynamic and static code analysis, were identified and the difference between the two techniques were explained, along with several automated assessment systems (see Table 2.2) and the relevant programming languages that they support.

The traditional assessment of word-processing skills was described, as well as the practical complications that made it necessary for the development of computerised assessment systems in this regard. The limitations of early computerised assessment systems, with regard to word-processing, were explained. Two word-processing skills assessment approaches were identified along with two analysis methods that are currently applied to the automated assessment of word-processing skills. The validity and reliability of manual and computerised assessment were questioned. Reasons for the development of computerised assessment were specified, along with limitations that needed to be addressed. Finally, certain complications with regard to the assessment of word-processing skills were illustrated, which emphasised the necessity of guidelines with regard to the computerised assessment of word-processing skills.

Chapter 3 investigates the various algorithms and similarity metrics that are employed by the computerised assessment systems, introduced in Chapter 2, used to assess free-text, computer programming and word-processing skills.

Chapter 3

Assessment Algorithms and Metrics

This chapter focuses on the following aspects:

- **Various algorithms used to assess free-text, computer programming and word-processing skills**
- **The metrics embedded in these algorithms**
- **The accuracy by which they assess students' assignments**
- **Similarity metrics included in the quasi-experimental study**

3.1 Introduction

In Chapter 2, the evolution of computerised assessment was described. Several computerised assessment systems were introduced and the reliability, validity and complications of manual and computerised assessment were debated. In this chapter, the specific algorithms that are implemented by these assessment systems are described in relation to the assessment of free-text (see Section 3.2), computer programming (see Section 3.3) and word-processing skills (see Section 3.4). The accuracy of the relevant algorithms' assessment results is investigated by studying several research projects that have been conducted by different authors. This has great significance in identifying all possible candidate algorithms that would be relevant to the assessment of word-processing skills. Finally, taking into consideration all the algorithms described, the chapter concludes by specifying which algorithms, along with their applicable similarity metrics, will be included in the quasi-experimental[1] study to determine the accuracy and reliability of the algorithm developed as part of this research project.

3.2 Free-text assessment algorithms

According to Whittington and Hunt (1999), several algorithms for assessing essays have been implemented in computerised assessment systems. Project Essay Grade (PEG) (Page, 1966, 1968, 1994; Page & Petersen, 1995) measures surface linguistic features, whereas Essay Rater (e-rater) (Burstein et al., 1998a, 1998b, 1998c) implements natural language processing (NLP) techniques when evaluating essays. They both use a statistical approach by means of

[1] In a quasi-experimental study the independent variable is not randomly assigned (Ellis, 1999)

regression techniques to predict essay grades. Concept-rater (C-rater) (Burstein et al., 2001) also implements NLP techniques to evaluate students' understanding of content material. The Intelligent Essay Marking System (IEMS) (Ming, Mikhailov, & Kuan, 2000) implements pattern-matching techniques combined with clustering, whereas the Intelligent Essay Assessor (IEA) (Foltz et al., 1999) uses Latent Semantic Analysis when assessing essays. The Automated Text Marker (ATM) (Callear et al., 2001) and FreeText Author (Jordan & Mitchell, 2009) rely on information extraction techniques to assess students' essays, while the short free-text response matching system, OpenMark (Butcher, 2008), implements algorithmic keyword manipulation to achieve its objective.

3.2.1 Measurement of surface linguistic features

When measuring surface linguistic features, as implemented by PEG (Page, 1966), *trins* (Page's term for intrinsic variables of interest within an essay) are identified within an essay. Since computers cannot directly measure *trins*, computer approximations or measures of *trins*, called *proxes* (another term introduced by Page (1966)), have to be used to simulate human grading. To calculate these *proxes*, a set of training essays are marked manually. The *proxes*, along with the human assessed grades, are then used in a standard multiple regression analysis to calculate the regression coefficients, which represent the weights of the calculated *proxes*. They are then used with *proxes* from unmarked essays to predict the expected grades of the unmarked essays (Page, 1966).

Earlier versions of PEG (Page, 1966, 1968) showed a reasonable correlation between essay grades allocated by human examiners and essay grades predicted by PEG. Unfortunately, no language processing techniques had been implemented at that time (Pérez-Marín et al., 2009). The system purely relied on a statistical approach (Whittington & Hunt, 1999) where text content and word order were disregarded. This made PEG unsuitable for assessing factual disciplines where lexical content is of high importance (Callear et al., 2001). In 1990, the system was enhanced with a grammar parser and a part-of-speech tagger (Pérez-Marín et al., 2009). In his latest experiments with PEG (Page & Petersen, 1995), Page indicated that a correlation of as high as 87% exists between expected essay grades, predicted by a computer, and essay grades that were allocated by humans. Today, the system is able to assess the content, organisation, style, mechanics and creativity of essays (Shermis, Koch, Page, Keith, & Harrington, 2002).

3.2.2 Natural language processing

E-rater and C-rater apply NLP techniques in the assessment of free-text. NLP extracts features from essays that are derived from syntactic and discourse structure information, as well as topical analysis and vocabulary content information (Burstein et al., 1998c). The weights of these features are calculated through stepwise linear regression analysis. Equivalent predictive features within the unmarked essays are then used, according to their calculated weights, to predict the final essay scores (Burstein et al., 1998b).

As with PEG, e-rater uses a set of training data to extract these features, while C-rater only requires a single correct answer, supplied by the examiner (Pérez-Marín et al., 2009). E-rater evaluates the organisation, sentence structure and content of essays (Alfonseca et al., 2005), whereas C-rater concentrates on measuring students' understanding of content material, disregarding their writing skills. Instead of merely implementing word matching, C-rater also takes care of different syntactic structures, other variations of words, similar terms, synonyms, misspelled words and the application of pronouns to identify paraphrases of the correct responses (Pérez-Marín et al., 2009). E-rater also does not consider word order, similar to PEG. However, text content is taken into account (Whittington & Hunt, 1999) but only to the extent of spotting weighted keywords. Consequently, e-rater is unsuitable to assess factual disciplines (Callear et al., 2001).

More than 750 000 Graduate Management Admissions Test (GMAT) essays have been assessed since 1999, exhibiting a correlation of more than 97% between e-rater and human graders (Burstein et al., 2001). A comparison between the assessment results assigned by human examiners and those produced by e-rater V.2, demonstrated that the results produced by e-rater V.2 were more reliable, and the true-score correlation between e-rater V.2 and the examiners was almost perfect, yielding a value of .97 (Attali & Burstein, 2006).

C-rater coincided 80% with the instructor when used in a small-scale study in conjunction with a university virtual learning program (Burstein et al., 2001; Valenti et al., 2003). In Leacock (2004), an accuracy of 85% was achieved in a large-scale assessment comprised of 170 000 short-answer responses, 19 reading comprehension and 5 algebra questions.

Another computerised assessment system, IntelliMetric, developed by Vantage Learning, implements a combination of NLP and Artificial Intelligence to assess essays (Pérez-Marín et al., 2009; Wang & Brown, 2007). In a recent correlational study, Rudner, Garcia and Welch (2006) obtained a correlation coefficient of .83 between the assessment results produced by IntelliMetric and those provided by human markers. In a study by Vantage Learning (2000), IntelliMetric achieved an agreement of 98% with human markers. In 2003, Vantage Learning (as cited in Wang & Brown, 2007) conducted a comparative study by means of a paired-sample t-test that indicated no significant difference between the assessment results produced by IntelliMetric and the results provided by two expert examiners ($t = .265$, $p > .05$).

3.2.3 Pattern matching with clustering

When it comes to qualitative material, i.e. biology, psychology, history or anatomy, the IEMS has been most useful. The IEMS implements pattern matching techniques (Alfonseca et al., 2005) based on the Indextron, a Pattern Indexing Neural Network (Ming et al., 2000). On one occasion the IEMS achieved 80% agreement with the teacher. It involved 85 third year Mechanical Engineering students who were asked to summarise a passage of 800 words, with the title “Crime in Cyberspace”, to maximum 180 words (Ming et al., 2000; Pérez-Marín et al., 2009).

3.2.4 Latent semantic analysis

Computerised assessment systems like IEA (Foltz et al., 1999), use latent sematic analysis (LSA) to determine the level of agreement between a student’s essay and references to articles or textbooks relevant to the content of the essay (Landauer, Laham, Rehder, & Schreiner, 1997). The algorithm searches for hidden relationships between the words of separate documents. The main purpose of this algorithm is to assess the knowledge conveyed within the essay without taking into account the word order; therefore disregarding its style, syntax or argument structure. It also has the ability to recognise synonyms in order to treat words with similar meaning as the same word (Pérez-Marín et al., 2009).

In a study by Streeter, Pstoka, Laham and MacCuish (as cited in Pérez-Marín et al., 2009), IEA achieved a higher inter-reliability with regard to the instructor than the inter-reliability achieved between different instructors. The inter-reliability between instructors yielded a rate of only 31%, where IEA obtained an inter-reliability of 35% with the instructor. In writings of

psychology, medicine and history, IEA agreed 80% to 90% with human markers (Pérez-Marín et al., 2009). The system also achieved an agreement of 85% to 91% with human examiners during the assessment of GMAT essays (Valenti et al., 2003). Kukich states that IEA is able to accurately determine whether a meaningful similarity exists between two documents, irrespective of their vocabulary overlap (as cited in Wang & Brown, 2007).

Klein, Kyrilov and Tokman (2011) developed an LSA-based system that is able to assess paragraph responses to exam questions, exhibiting a correlation of more than 80% with human graders. LSA is, however, not suitable for the assessment of single sentence free-text responses since it does not consider grammar, syntax or word order (Callear et al., 2001; Pérez-Marín et al., 2009).

3.2.5 Information extraction and algorithmic keyword manipulation

FreeText Author and ATM applies information extraction (IE) to match free-text responses (Callear et al., 2001; Jordan & Mitchell, 2009), while regular expressions[2] and OpenMark utilise the algorithmic manipulation of keywords to accomplish free-text matching (Butcher & Jordan, 2010). IE entails the acquisition of structured information by identifying relationships between entities of interest within text (Pérez-Marín et al., 2009). The algorithm implements syntax and semantic analysers that automatically break down the student's text response into its basic concepts and their relationships. These relations are then compared with the examiner's ideal response.

In a comparative study conducted by Jordan and Mitchell (2009), the free-text responses to seven short answer questions were separately assessed by FreeText Author, six tutors and the author of the questions. The assessment results produced by FreeText Author had an agreement rate of between 97.5% and 99.6% with the assessment results assigned by the question author. The tutors' assessment results demonstrated discrepancies with one another, indicating inconsistencies with regard to human marking.

Butcher and Jordan (2010) further extended the study. FreeText Author and OpenMark along with regular expressions, implemented in Java programming code, were used to assess the same set of free-text responses a total of three times. After each assessment, the assessment

[2] Regular expressions are sequences of characters that specify flexible search patterns within text (Friedl, 2006)

systems' answer matching algorithms were improved. FreeText Author and OpenMark produced assessment results that were similar in accuracy. The final assessment results demonstrated an agreement with the question author of 98.5% to 100% for FreeText Author and 98.7% to 100% for OpenMark. The regular expressions demonstrated the lowest agreement of 95.5% to 99.7% with the question author.

3.2.6 Outcome of the investigation

A summary of earlier research study results discussed in Section 3.2, is displayed in Table 3.1. The agreement rates depicted in the table are the highest of the average agreement rates obtained in the relevant research studies. The author could not obtain agreement rates with regard to PEG and e-rater. Instead, their relevant correlations with human graders are depicted.

Table 3.1 Agreement rates and correlations for free-text assessment algorithms

Assessment algorithms	Assessment system	Highest average agreement rate	Correlation
Algorithmic keyword manipulation	OpenMark	99%	
	Regular expressions	98%	
Information extraction	FreeText Author	99%	
Latent semantic analysis	IEA	88%	
	Klein's system		.80
Natural language processing	C-rater	85%	
	e-rater		.97
Natural language processing with artificial intelligence	IntelliMetric	98%	.83
Pattern matching with clustering	IEMS	80%	
Surface linguistic features	PEG		.87

From the agreement rates in Table 3.1, it is evident that free-text assessment techniques involving the algorithmic manipulation of keywords, as well as information extraction, produce assessment results that have a high agreement with the assessment results of human markers. High agreement rates are also achieved when NLP is combined with artificial intelligence.

The investigation into various free-text assessment algorithms revealed two similarities with regard to the document analysis approach of word-processing skills assessment. First, the

student's attempt is compared with an ideal response to a question (with regard to IE), a set of similar essays (with regard to NLP) or with articles and textbooks that contain content relevant to the question topic (with regard to LSA). Secondly, features identified within the content of the essays are used to grade them. Table 3.2 displays the relevant comparative elements within students' essays that are used to determine the agreement of their free-text responses with that of the examiner's ideal response.

Table 3.2 Comparative elements within free-text assessment algorithms

Assessment algorithms	Comparative elements
Algorithmic keyword manipulation	Patterns within the essays' content
Information extraction	Entity relationships within the text content
Latent semantic analysis	Hidden relationships within the essay content
Natural language processing	Features of interest within the essays
Pattern matching with clustering	Patterns within the essays content
Surface linguistic features	Approximations of intrinsic variables

Unfortunately, as demonstrated by the investigation, most free-text assessment algorithms disregard word order and none of them take the font and style format of the response into consideration, which is really important with regard to word-processing. It would, therefore, be unfitting to compare the similarity metrics embedded in these free-text assessment algorithms with the similarity metrics applied by word-processing assessment algorithms. Because of these reasons, no free-text assessment algorithms were included in the quasi-experimental study of this project. Nevertheless, the results obtained from the research studies with regard to their accuracy, provided meaningful support to the credibility of computerised assessment.

3.3 Computer programming assessment algorithms

The algorithms embedded in computerised assessment systems for computer programming assignments apply two main assessment approaches (Douce et al., 2005; Fonte et al., 2013). The first and most often applied approach is dynamic code analysis and is sometimes complemented by the second approach, namely static code analysis (Fonte et al., 2013). Some computerised assessment systems have implemented certain algorithmic enhancements with regard to code analysis techniques, although the principle remains the same. This section investigates these enhancements and categorises them accordingly.

3.3.1 Standard dynamic code analysis

As specified in Chapter 2, the Rensselaer grader (Hollingsworth, 1960), GRADER2 (Forsythe & Wirth, 1965), Infandango (Hull et al., 2011), Kassandra (Von Matt, 1994), Automatic Marker (Suleman, 2008) and Fitchfork (Pieterse, 2013) all apply standard dynamic code analysis techniques within their assessment algorithms. The core concept behind standard dynamic code analysis is compiling and running the program code (Fonte et al., 2013). The purpose of dynamic code analysis is to assess the functionality of the computer program submitted by the student, to determine whether it executes its tasks successfully (Pieterse, 2013). The compiled program is executed with several cases of test input data of which the expected output is known. The output provided by the program and the expected output are then compared (Douce et al., 2005), usually by means of pattern matching (Helmick, 2007) or a simple string comparison algorithm (Fonte et al., 2013), to verify the program correctness (Fonte et al., 2013; Leal et al., 1998).

3.3.2 Enhanced dynamic code analysis

Assessment systems such as BAGS (Hext & Winings, 1969), Test (Forsythe & Wirth, 1965) and ASSYST (Jackson & Usher, 1997) also apply standard dynamic code analysis techniques to assess the functionality of computer program code (see Chapter 2). In addition, they also assess the quality of the computer program by evaluating the conservation of system resources during the program execution (Forsythe & Wirth, 1965). BAGS and ASSYST evaluate the central processor (CPU) time that was occupied during the program execution (Hext & Winings, 1969; Jackson & Usher, 1997) by initiating calls to the system timer. The execution time of the student's attempt is then compared to the execution time of the ideal solution, provided by the instructor. Furthermore, ASSYST and Test calculate the number of statements that were executed by the program to fulfil its tasks (Forsythe & Wirth, 1965; Jackson & Usher, 1997), while Test also calculates the amount of storage space used during the execution of the program code (Forsythe & Wirth, 1965); all of which is also compared to that of the ideal solution.

3.3.3 Method-based dynamic code analysis

Helmick (2007) argues that standard dynamic code analysis has a drawback. Since it only compares the final output of the program with the expected output data, it is unable to allocate

marks to a program that is only partially correct. To alleviate this issue, a system that allows for the evaluation of partially correct programs, namely AutoGrader (Helmick, 2007), was introduced. This approach assesses the functionality of the computer program methods separately, by comparing each particular method's output with its expected output data, disregarding other methods.

3.3.4 Test-driven dynamic code analysis

Another dynamic code analysis approach is applied by Edwards (2003), Jackson and Usher (1997). Instead of focusing on the output that the student's program provides, systems such as WebCAT Grader (Edwards, 2003) and ASSYST (Jackson & Usher, 1997) apply a test-driven approach that evaluates the validity of the test case data used by the student during the development of the program. This approach executes the instructor's correct version of the program by using the student's test case input data and checks whether the instructor's program provides output data that corresponds with the student's test case output data. If the student's program was implemented correctly, the student's test case output data should coincide with the output provided by the instructor's correct version of the program.

3.3.5 Static code analysis

WebCAT Grader, ASSYST and AutoGrader apply static code analysis techniques to assess the style of a program's source code (Edwards, 2003; Helmick, 2007; Jackson & Usher, 1997). ASSYST, additionally, also evaluates the complexity of the program code, which, according to McCabe (1976), depends solely on the decision structure of the program. WebCAT Grader and AutoGrader make use of PMD (Sourceforge, n.d.), a source code analyser that identifies programming flaws, such as empty catch blocks, empty if statements, unused variables and dead code.

3.3.6 Outcome of the investigation

Most research on computer programming assessment is focused towards increasing the assessment throughput - the number of programs evaluated within a given timeframe (Forsythe & Wirth, 1965; Helmick, 2007; Hext & Winings, 1969; Hollingsworth, 1960; Hull et al., 2011). However, as mentioned in Section 2.2.2, the evaluation of computer programming skills is influenced by the computer technology of the day. Therefore, it would

be improper to compare the assessment throughput of the different assessment systems with each other.

Very little research could be found regarding the accuracy of computer programming assessment. A study by Suleman (2008) compares students' grades from two consecutive years for a particular computer programming module with. The average grade obtained in 2007 (without the use of the Automatic Marker) was 79.43%, compared to 82.91% in 2008 (using the Automatic Marker). This confirms the argument that the Automatic Marker provides assessment results that are comparable with manual assessment results. Table 3.3 displays a summary of the computer programming assessment approaches that were discussed in Section 3.3. It portrays the assessment algorithms that are implemented as well as the particular elements that are compared to determine whether the student has provided a viable solution to the computer programming assignment.

Table 3.3 Assessment approaches for computer programming assignments

Assessment approach	Algorithm	Comparative elements
Standard dynamic	String comparison	Instructor's test case output
Enhanced dynamic	String comparison	Instructor's test case output & CPU-time, statement count and storage space
Method-based dynamic	String comparison	Instructor's test case output for each method
Test-driven dynamic	String comparison	Student's test case output
Static	PMD	Checks for empty catch blocks, empty if statements, unused variables and dead code

String comparison plays a major role in the assessment of computer program code. This is evident from the investigation into the various code analysis techniques. However, the algorithm as it is implemented here does not assess the computer program's output with regard to its font and formatting. Nevertheless, the string comparison algorithm implemented by the dynamic code analysis of computer programming assignments is relevant to the assessment of word-processing skills, as discussed in Section 3.4.1.

3.4 Word-processing skills assessment algorithms

Unfortunately, research on the computerised assessment of word-processing skills is relatively new compared to research on computer programming and essay assessment. Some of the earliest published research on the assessment of word-processing skills, was conducted by

Dowsing et al. (1998). This research study demonstrated that the assessment results of word-processing skills, produced by a computerised system called CATS (Dowsing et al., 1998), were approximately similar to the assessment results provided by human markers. Since then, several other algorithms that are implemented by word-processing skills assessment systems have materialised and are discussed below.

3.4.1 String comparison

At the core of the CATS word-processing assessor lies a string comparison algorithm with the ability to determine the similarities between the final document, submitted by the student, and the memorandum, compiled by the examiner (Dowsing et al., 1998). The algorithm is based on a file comparison program called fComp, developed by Miller and Myers (1985). It calculates the shortest sequence of insertions and deletions necessary to convert the student's final document into the memorandum. It is important to note that the comparison between the documents takes place at byte level, which means that the entire file content is compared, including content that specifies the documents' format, such as font, style, layout and orientation.

Dowsing et al. (1998) conducted a research study where a group of 17 participants from various backgrounds had to submit a document on which they had to perform specific word-processing tasks. Using the same assessment criteria, human markers and CATS graded the submitted documents by comparing the documents with the memorandum. CATS implemented standard difference algorithms to compare the submitted documents with the memorandum. In 65% of the cases the human markers' assessment results were identical to CATS' assessment results, which is very low when compared to the agreement rates of 80% and more between human markers and free-text assessment algorithms (see Table 3.1). CATS, however, identified all the errors that were made by the participants while the human markers overlooked certain errors. This might indicate that CATS is more stringent with regard to the application of the assessment criteria than human markers.

3.4.2 Document difference comparison

A computerised assessment system called Word Grader (Hill, 2011) implements an algorithm that utilises the *Compare and Combine* function of Microsoft Word to determine the differences between the student's final document and the memorandum. The *Compare*

method is directly obtainable from the *Microsoft.Office.Interop.Word* library within the *.Net* development framework (Microsoft Inc., n.d.). As input, the method receives the target document that needs to be compared against, as well as parameters specifying what document elements to include in the comparison. The method provides output in the form of a document in which the differences are highlighted (Lánskỳ et al., 2013).

Word Grader distinguishes between typographical, formatting and miscellaneous errors. The detection of typographical errors is conducted similar to that of the string comparison algorithm, by determining the number of deletions and insertions needed to convert the text content of the student's document into that contained in the memorandum. Formatting errors are determined by the type or description of the format changes that need to be implemented to convert the student's document into the memorandum. Miscellaneous errors refer to any error that is not detected as an insertion/deletion or formatting change (Hill, 2011).

Word Grader forms part of a larger suite called Office Grader that includes PowerPoint Grader, Excel Grader and Access Grader. A survey has been conducted with regard to Excel Grader and Access Grader (Hill, 2011). Results obtained from the survey indicated that Excel Grader and Access Grader were extremely accurate in assessing Microsoft Excel and Microsoft Access assignments. Unfortunately, research with regard to Word Graders' performance could not be found by the author of this research project. Hill (2011), however, mentions that certain upgrades were made to improve the accuracy of Word Grader in 2010. The frequently asked questions with regard to Word Grader, which can be downloaded from Wiley's website (Wiley, n.d.), report that certain attributes, such as text boxes and images within a word-processing document, cannot be graded by Word Grader 2013.

3.4.3 Component Object Model Automation

Microsoft's Component Object Model (COM) (Gray, Hotchkiss, LaForge, Shalit, & Weinberg, 1998) enables communication between different Windows applications. An assessment system called FACT, developed by Hunt (2002), uses COM Automation to launch and control Microsoft Word from the outside, through a set of automation classes provided by Microsoft Word. These automation classes enable FACT to examine almost any Word document property with regard to font, style, layout, orientation and so on. From this it is able to determine whether or not the student completed the word-processing assignment

successfully. Research has proven that FACT provides reliable and valid results regarding the assessment of word-processing competency (Hunt et al., 2002). However, no research results indicating the accuracy level of FACT could be found.

3.4.4 Proprietary assessment algorithms

Various other assessment systems, such as SAM (Cengage Learning, n.d.), SIMnet (McGraw-Hill Education, n.d.), MyITLab (Pearson, n.d.) and Train and Assess IT (TAIT) (Robbins & Zhou, 2007) are commercially available for the purpose of assessing word-processing skills. According to the knowledge of the author, little research on the accuracy of these proprietary systems has been published. Also, due to their proprietary nature, the specifics of the similarity metrics that they implement are not freely accessible for further investigation.

These systems have, however, been included in several research studies to either determine the word-processing skills of computer users or to improve current methods of word-processing skills assessment (Easton et al., 2006; Grant et al., 2009; Tuparova & Tuparov, 2010). However, Robbins and Zhou (2007) state that these proprietary systems might evaluate a viable solution presented by a student as incorrect, due to the system not being able to recognise the student's method of solving the specified problem.

Therefore, Robbins and Zhou (2007) conducted a study that compared the assessment results of the Computer Skills Placement (CSP) Test and the assessment results produced by TAIT. A correlation coefficient was calculated, which indicated that a linear relationship existed between the assessment results produced by the CSP Test and TAIT ($r(130) = .724$, $p < .001$). The study also indicated a significant difference between the mean testing scores of the CSP Test and TAIT. This creates reason for concern in terms of the accuracy of commercially available systems, since their accuracy is usually assumed valid.

3.4.5 Element / Object comparison

As already discussed in Chapter 1, Microsoft Office currently implements the OOXML standard as its default file format (Van Vugt, 2007). This led to the development of OOXML-based computerised assessment algorithms that could support the assessment of word-processing documents of this kind. These algorithms identify elements within the OOXML structure of word-processing documents that specify the font, style, layout, orientation and

several other format characteristics of the document. These elements are identified within the student's final document and compared to the same elements of the examiner's memorandum to determine the similarities between the documents (Lánskỳ et al., 2013; Pellet & Chevalier, 2014; Wolters, 2010). Chapter 4 provides an in depth explanation of the OOXML structure of a Microsoft Word document.

Current algorithms, developed by Pellet and Chevalier (2014), Lánskỳ et al. (2013) and Wolters (2010), all implement the above technique, of matching the identified elements within the student's document with that of the memorandum. However, certain gaps in their algorithmic implementation need to be addressed. Lánskỳ et al. (2013) developed a system called AUTOPOT that is able to assess individual word-processing tasks set out in the assignment. This is accomplished by extracting the relevant section where the task had to be performed, from the document in OOXML format. To extract the section, a software development kit (SDK) for Microsoft Visual Studio, called OOXML SDK v2.0, is used. It contains libraries that make it possible to create an object of the particular document. The properties of the object correspond to the different sections within the document body. The OOXML content of the section that is extracted is then compared to the OOXML content of the solution.

AUTOPOT (Lánskỳ et al., 2013) also applies a procedural assessment approach for tasks that were impossible to assess with only the OOXML content. This approach again extracts the relevant section from the document and verifies whether it contains all the required elements and attributes by means of a written procedure. The procedural approach was necessary since the OOXML approach is unable to compare different representations of the same document content, as already explained in Chapter 1.

Wolters (2010) implements a different strategy within the Office Skills Assessment Approach (OSAP). By applying a parsing algorithm, Wolters' assessment system is able to convert an OOXML-based document into its object-oriented representation. From here it is parsed again by removing all the irrelevant elements and properties and converting it into a new format, called the DocumentElements format, which is a tree-like structure of the elements contained within the document. A final parser, the DocumentSweeper that combines multiple occurrences of the same document element into one, ensuring a uniform representation of the document content, is then applied. After both the student's document and the memorandum

have been parsed, they are compared. Chapter 4 also discusses a method included in the OOXML SDK v2.0 that parses the content of word-processing documents.

Another issue with the assessment of OOXML-based documents, as mentioned in Chapter 1, deals with the comparison between documents with a dissimilar number of paragraphs. The approach by Wolters (2010) apply an optimisation algorithm. Each paragraph from the student's document is compared and evaluated with each paragraph of the memorandum. The final assessment result is obtained from the evaluation results of the cross-paragraph comparisons that deliver the best match between the two documents. As stated in Chapter 1, this causes unnecessary paragraph comparisons that are addressed by the algorithm developed in this research project.

This research project, therefore, proposes an algorithm for the computerised assessment of word-processing skills that first parses the documents to ensure that the student's document and examiner's solution conform to a homogeneous structure. It then identifies which paragraph in the student's document corresponds with which paragraph in the examiner's solution. Thereafter, it assesses the student's submitted document by comparing the parsed OOXML structure of the corresponding document paragraphs only, thereby minimising paragraph comparisons. The details of this algorithm are discussed in Chapter 4.

3.4.6 Outcome of the investigation

Table 3.4 displays a summary of the word-processing assessment algorithms discussed in Section 3.4. It specifies the relevant elements that are compared to determine whether or not the student has successfully completed a particular word-processing assignment.

Table 3.4 Assessment algorithms for word-processing assignments

Assessment algorithm	Systems	Comparative elements
COM automation	FACT	Document properties and text content
Document difference comparison	Word Grader	Insertion, deletion and formatting changes
Element / Object comparison	AUTOPOT, OSAP	Individual document elements
String comparison	CATS (fComp)	Bytes within the relevant files

The investigation into word-processing assessment algorithms produced very few research studies portraying the accuracy or agreement rates of the computerised assessment of word-

processing skills. This research project, however, contributes to this by determining the accuracy and reliability of the proposed OOXML algorithm as well as the relevant established algorithms (see Section 3.6) that are included in the research study.

3.5 The Levenshtein algorithm

In Section 3.3.6 it was stated that pattern matching and string comparisons form a major part of computer program assessment. For this reason, a well-known string comparison algorithm, the Levenshtein algorithm (Levenshtein, 1966), was investigated to determine whether it would be applicable as a comparative candidate in determining the accuracy and reliability of the proposed OOXML algorithm, developed as part of this research project.

The Levenshtein algorithm has been implemented in several software applications, such as the dictionary lookup methods within Optical Character Recognition (OCR) software (Haldar & Mukhopadhyay, 2011). The algorithm is very similar to that of fComp, implemented by CATS. The main difference between the Levenshtein algorithm and fComp is that the latter receives two files as input, whereas the Levenshtein algorithm receives two strings. The Levenshtein algorithm determines the smallest number of deletions, insertions or substitutions required to convert one string to another. This is known as the Levenshtein difference metric (Haldar & Mukhopadhyay, 2011; Levenshtein, 1966).

An experimental research study by Haldar and Mukhopadhyay (2011) improved the recognition ability of OCR by decreasing the unrecognised word count from 93 out of 500 to 66 out of 500, through the application of the standard Levenshtein distance metric within the dictionary lookup phase of OCR. The investigation into the Levenshtein algorithm demonstrates that it would be an ideal candidate to determine the accuracy and reliability of the proposed OOXML algorithm.

3.6 Algorithms included in the quasi-experiment

From the discussions in the previous sections, it is apparent that only the algorithms implemented in the assessment of word-processing skills are ideal candidates to be included in the quasi-experimental study. The algorithms applicable to essay and computer programming assessment were not included in the research study due to reasons stated in Sections 3.2.6 and 3.3.6. From the available word-processing assessment algorithms, only

those implemented by CATS and Word Grader are selected, since the author could not obtain the detailed algorithms that are implemented by FACT and proprietary assessment systems. The Levenshtein distance metric is also selected for the quasi-experimental study since it shows similarities with fComp, a string comparison algorithm implemented by CATS. Table 3.5 contains the assessment algorithms that are included in the quasi-experimental study of this research project.

Table 3.5 Assessment algorithms included in the research study

Assessment algorithm	Comparative elements
Levenshtein (String comparison)	Levenshtein distance
Document difference comparison (within Word Grader)	Insertion, deletion and formatting changes
Element / Object comparison (OOXML algorithm)	Individual document elements
String comparison (fComp)	Bytes within the relevant files

3.7 Chapter summary

Chapter 3 provided an in depth discussion of various computerised assessment algorithms and the similarity metrics that they employ, which is summarised in Tables 3.2, 3.3 and 3.4. The discussion included different fields of study and focused on algorithms of free-text, computer programming and word-processing skills assessment. The accuracy of computerised assessment was discussed at the hand of studies that have been conducted between assessment results delivered by human markers and assessment results produced by computerised assessment systems.

Finally, the assessment algorithms that would be included in the quasi-experiment of this research project were identified (see Table 3.5). In the next chapter, the OOXML standard with regard to word-processing documents will be described. An in depth discussion of the proposed OOXML algorithm will also be presented.

Chapter 4

The Structure of a Word Document

This chapter focuses on the following aspects:

- **Office Open XML**
- **The word-processing document structure**
- **Most significant parts of the structure**
- **The proposed OOXML algorithm**

4.1 Introduction

In this chapter, the OOXML (Office Open XML) document format, used by Microsoft Word and similar word processors, is described. The discussion provides an illustration of the structural layout of a DOCX word-processing document. It explains how OOXML-based algorithms are implemented to identify measurable features within a document that can be used to evaluate the text content and format of a word-processing document. The chapter concludes with a detailed explanation of the OOXML assessment algorithm that was developed as part of this research project.

4.2 Office Open XML

The OOXML standard specifies a format for saving word-processing documents (DOCX), spreadsheets (XLSX) and presentations (PPTX), each with its own representative file extension (Ditch, 2007; Van Vugt, 2007). The standard is supported by various suites, such as Microsoft Office, LibreOffice and OpenOffice (not linked in any way to the name of the standard).

Three major mark-up languages are included in the OOXML standard: WordprocessingML for documents, SpreadsheetML for spreadsheets and PresentationML for presentations. A universal mark-up language, DrawingML, is also defined for saving graphics, charts, tables and diagrams (Van Vugt, 2007). This chapter focuses on the WordprocessingML mark-up language that specifies the basic structure of a word-processing document and the underlying elements contained in a document. Only the essential document elements pertaining to this research project are discussed in full.

4.3 The document package

All documents conforming to the OOXML standard consist of a combination of directories, WordprocessingML files and relationships between these files contained in a DOCX archive, similar to a ZIP archive (Van Vugt, 2007). The files contained in such a word-processing archive are displayed in the form of a directory tree in Figure 4.1.

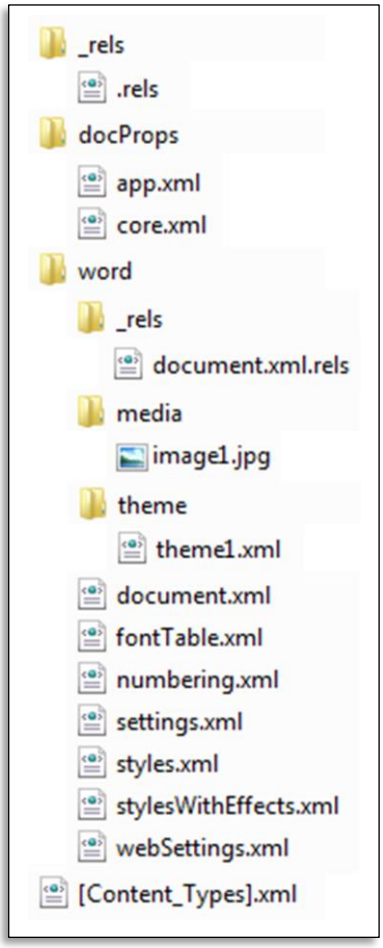


Figure 4.1 DOCX archive

4.4 The main document part

Most of the content pertaining to the document, such as the document layout properties, the style properties and the text content itself are contained in the main document part, *document.xml*, of the archive. It is used by the word-processing application to render and display the document. Spelling and grammar errors, identified by the word processor's proofing tool (also known as the spell checker), are also contained in the main document part (ECMA, 2012).

4.4.1 Spelling and grammar errors

Spelling and grammar errors in a document are indicated in the main document part by means of the WordprocessingML tag, *w:proofErr*, hereafter referred to as proofing errors (ECMA, 2012). This tag can therefore be used to identify spelling or grammar errors in a document. The extract in Figure 4.2 displays WordprocessingML mark-up indicating spelling and grammar errors in a paragraph of a document.

As indicated by the extract, two types of proofing errors exist. They are specified by the attribute, *w:type*, which can hold one of four values:

- *gramStart* indicates the start of a grammar error
- *gramEnd* indicates the end of a grammar error
- *spellStart* indicates the start of a spelling error
- *spellEnd* indicates the end of a spelling error

```
<w:p w:rsidR="00D45FD5" w:rsidRDefault="00960502">
  <w:pPr>
    <w:rPr>
      <w:lang w:val="en-GB"/>
    </w:rPr>
  </w:pPr>
  <w:proofErr w:type="gramStart"/>
  <w:r>
    <w:rPr>
      <w:lang w:val="en-GB"/>
    </w:rPr>
    <w:t>This document contain</w:t>
  </w:r>
  <w:proofErr w:type="gramEnd"/>
  <w:r>
    <w:rPr>
      <w:lang w:val="en-GB"/>
    </w:rPr>
    <w:t xml:space="preserve"> </w:t>
  </w:r>
  <w:proofErr w:type="spellStart"/>
  <w:r>
    <w:rPr>
      <w:lang w:val="en-GB"/>
    </w:rPr>
    <w:t>speling</w:t>
  </w:r>
  <w:proofErr w:type="spellEnd"/>
  <w:r>
    <w:rPr>
      <w:lang w:val="en-GB"/>
    </w:rPr>
    <w:t xml:space="preserve"> and grammar errors.</w:t>
  </w:r>
</w:p>
```

Figure 4.2 WordprocessingML mark-up with proofing errors

These attribute values are used in conjunction with each other for every proofing error occurrence (ECMA, 2012). The attribute values *gramStart* and *gramEnd* are used to encapsulate grammar errors, while spelling mistakes are encapsulated by the attribute values *spellStart* and *spellEnd*. It is therefore only necessary to locate one of each pair of attribute values to discover the type of proofing error that occurred. This can then be used to evaluate whether the student's submitted document contains any spelling or grammar errors.

4.4.2 Document layout properties

The layout of a student's document can be compared to the memorandum's layout by matching their layout properties contained in the main document part of the DOCX archive (ECMA, 2012; Lánský et al., 2013). The extract in Figure 4.3 illustrates the layout properties of a DOCX document in WordprocessingML mark-up.

```
<w:sectPr>
  <w:pgSz w:w="11906" w:h="16838" />
  <w:pgMar w:top="1417" w:right="1417" w:bottom="1417" w:left="1417" w:header="708"
           w:footer="708" w:gutter="0" />
  <w:pgBorders w:offsetFrom="page">
    <w:top w:val="single" w:sz="4" w:space="24" w:color="auto" />
    <w:left w:val="single" w:sz="4" w:space="24" w:color="auto" />
    <w:bottom w:val="single" w:sz="4" w:space="24" w:color="auto" />
    <w:right w:val="single" w:sz="4" w:space="24" w:color="auto" />
  </w:pgBorders>
  ...
  ...
</w:sectPr>
```

Figure 4.3 Document layout properties in WordprocessingML mark-up

As indicated by Figure 4.3, the layout properties include the page size, which has two attributes that specifies the width and height of the page. Additionally, the orientation attribute of the page may also be included (Van Vugt, 2007). The properties also include the page margins with attributes for the top, left, bottom and right margins of the document. Furthermore, the header, footer and gutter attributes of the page margins are also specified (ECMA, 2012). Table 4.1 summarises the WordprocessingML tags of the document layout properties and its attributes.

Table 4.1 OOXML document layout properties and attributes

Property tag	Description	Attribute tag	Description
w:pgSz	Page size	w:w	Page width
		w:h	Page height
w:pgMar	Page margins	w:top	Top margin
		w:left	Left margin
		w:bottom	Bottom margin
		w:right	Right margin
		w:header	Header margin
		w:footer	Footer margin
		w:gutter	Gutter margin
w:pgBorders	Page borders	w:offsetFrom	Border space calculation method
w:top, w:left, w:bottom, w:right	Top, left, bottom and right borders	w:val	Border style
		w:sz	Border thickness
		w:space	Border space
		w:color	Border colour

The page borders are specified in the same way. However, the top, left, bottom and right border properties are indicated separately by individual WordprocessingML tags. Each border property has its own independent set of attributes that include the style, thickness and colour of the border. The border space attribute specifies the distance between the border and either the edge of the page or the text margins. Whether the border space is calculated by the distance between the border and the edge of the page or between the border and the text margins, is indicated by the offset attribute of the page borders property (ECMA, 2012).

Figure 4.3 does not contain a comprehensive set of document layout properties. Different documents may contain different or additional layout properties. Therefore, no set list of properties exists to evaluate the layout of the student's document. On the other hand, the memorandum's layout properties section contains all the relevant layout properties with regard to a particular word-processing assignment. To match the layout of the memorandum, the student's document must contain exactly the same layout properties. Thus, a comparison between the layout properties as specified in the memorandum and the corresponding layout properties of the student's document will result in the necessary assessment outcome.

If, however, the student's document contains more layout properties than specified in the memorandum, the student's document layout will not match the layout of the memorandum. The opposite is also true. If the student's document contains fewer layout properties than specified in the memorandum, the documents' layout properties will not match. In this case it does not matter, since the student will only receive marks for the layout properties that matched. When the student's document contains additional layout properties, only those contained in the memorandum can be compared. The additional layout properties will actually alter the layout of the student's document in such a way that it does not conform to the memorandum's layout. Therefore, marks will have to be deducted for unnecessary document layout properties in the student's document.

4.4.3 Document content

The main document part can contain several paragraphs (ECMA, 2012). The paragraph formatting is specified separately for each paragraph. The extract in Figure 4.4 displays the WordprocessingML mark-up of a single paragraph within the main part of a word-processing document.

```
<w:p w:rsidR="00D45FD5" w:rsidRDefault="00960502">
  <w:pPr>
    <w:spacing w:after="0" w:line="240" w:lineRule="auto" />
    <w:jc w:val="center" />
    <w:rPr>
      <w:rFonts w:ascii="Arial" w:hAnsi="Arial" w:cs="Arial" />
      <w:b />
      <w:color w:val="00B050" />
      <w:sz w:val="48" />
      <w:szCs w:val="48" />
      <w:lang w:val="en-US" />
    </w:rPr>
  </w:pPr>
  <w:r>
    <w:rPr>
      <w:rFonts w:ascii="Arial" w:hAnsi="Arial" w:cs="Arial" />
      <w:b />
      <w:color w:val="00B050" />
      <w:sz w:val="48" />
      <w:szCs w:val="48" />
      <w:lang w:val="en-US" />
    </w:rPr>
    <w:t>A Brief History of Arbor Day</w:t>
  </w:r>
</w:p>
```

Figure 4.4 WordprocessingML mark-up of a document paragraph

As illustrated by the extract, two main sections reside within the first tier of a paragraph. The first section contains the format properties of the paragraph and the second section contains the paragraph run. A paragraph run is nothing other than the sentences within the paragraph and may consist of multiple runs (Van Vugt, 2007) as discussed in Section 4.4.4. The WordprocessingML tags of the two main paragraph elements are displayed in Table 4.2.

Table 4.2 OOXML main paragraph elements (Tier 1)

Element tag	Description
w:p	Paragraph
w:pPr	Paragraph format properties
w:r	Paragraph run

The line spacing and paragraph alignment are specified in the paragraph format properties. This appears on the second tier of the paragraph. Furthermore, format properties that are relevant to the entire collection of paragraph runs within a particular paragraph (hereafter referred to as global run properties), are specified within this tier (ECMA, 2012). Table 4.3 contains the paragraph format properties and their relevant attributes.

Table 4.3 OOXML paragraph format properties and attributes (Tier 2)

Property tag	Description	Attribute tag	Description
w:spacing	Paragraph spacing	w:after	Spacing added after paragraph
		w:line	Vertical spacing between lines
		w:lineRule	How line spacing is calculated
w:jc	Paragraph alignment	w:val	Alignment position
w:rPr	Global run properties		

As pointed out in Table 4.3, the paragraph spacing property contains attributes whereby the spacing at the end of the paragraph as well as the spacing between lines of text are specified. A third attribute identifies the method by which the line spacing is calculated. Other possible attributes of the paragraph spacing property might include *w:before*, *w:beforeAutospacing*, *w:afterAutospacing*, *w:beforeLines* and *w:afterLines*, if the creator of the document chooses to specify values for these attributes. The alignment or justification of a paragraph is indicated by the value of the paragraph alignment position, which could be *left*, *right*, *center* or *both*.

Each run within a paragraph also contains two sections (Van Vugt, 2007). The first section contains the format properties relevant to that particular paragraph run (hereafter referred to as individual run properties). The other section contains the text within the particular paragraph run. Table 4.4 displays the WordprocessingML tags of these two sections within a paragraph run.

Table 4.4 OOXML paragraph run elements (Tier 2)

Element tag	Description
w:rPr	Individual run properties
w:t	Paragraph alignment

The run format properties specify properties that manipulate the text format of the particular paragraph run (by the individual run properties) or collection of paragraph runs (by the global run properties) within the same paragraph (ECMA, 2012). This includes properties such as the fonts that are used to display the text content and can contain up to four types of content, namely ASCII, Unicode, Complex Script and East Asian characters. Each content type can use a different font in the same run, specified by means of separate attributes. Table 4.5 lists the paragraph run format properties and their relevant attributes.

Table 4.5 OOXML paragraph run format properties and attributes (Tier 3)

Property tag	Description	Attribute tag	Description
w:rFonts	Run font	w:ascii	ASCII characters font
		w:hAnsi	Unicode characters font
		w:cs	Complex script characters font
		w:eastAsia	East Asian characters font
w:b, w:u, w:i	Font style (Bold, Underlined, Italics)		
w:color	Font colour	w:val	Hexadecimal colour code
w:sz	Non-complex script font size	w:val	Font size
w:szCs	Complex script font size	w:val	Font size
w:lang	Proofing language	w:val	Language code

Other format properties include the colour of the font, the size of non-complex script font, the size of complex script font and the languages used to check spelling and grammar errors in the text content. Whether the font should be bold, underlined or in italics, is specified by the WordprocessingML tags *w:b*, *w:u* and *w:i* respectively as separate properties.

The format properties have a hierarchical structure (ECMA, 2012). The global run properties lie at the top of the format properties hierarchy. The individual run properties within each paragraph run lie at the bottom of the hierarchy. In certain circumstances, the individual run properties may differ from the global run properties by way of dissimilar attribute values or additional properties. In this case the individual run properties will override the global run properties (ECMA, 2012).

As in Figure 4.3, the list of properties depicted in Figure 4.4 does not contain all the possible paragraph format properties (ECMA, 2012). It is therefore important to note that the format properties specified in the memorandum should serve as a guideline to indicate which format properties are relevant to the assessment. Any additional or missing properties within the student's document will cause nonconformity with the format of the memorandum. Thus, marks should be subtracted for additional format properties, contained in the student's document, that are not specified in the memorandum, while missing format properties or incorrect attributes should not be allocated any marks.

4.4.4 Multiple paragraph runs

As discussed in the previous section, the sentences that form each paragraph might consist of multiple paragraph runs. This is because different runs within a paragraph might contain different format properties (Van Vugt, 2007). These sections are therefore divided into separate runs, with each run specifying its own style properties as illustrated in Figure 4.5.

```
<w:p w:rsidR="00D45FD5" w:rsidRDefault="00960502">
  <w:pPr>...Global Run Properties...</w:pPr>
  <w:r>
    <w:rPr>
      <w:sz w:val="26" />
      <w:szCs w:val="26" />
      <w:lang w:val="en-US" />
    </w:rPr>
    <w:t xml:space="preserve">This example contains two runs </w:t>
  </w:r>
  <w:r>
    <w:rPr>
      <w:b />
      <w:i />
      <w:color w:val="00B050" />
      <w:sz w:val="26" />
      <w:szCs w:val="26" />
      <w:lang w:val="en-US" />
    </w:rPr>
    <w:t>with different properties.</w:t>
  </w:r>
</w:p>
```

Figure 4.5 Paragraph runs containing dissimilar format properties

Figure 4.5 displays a paragraph containing two runs. Each run represents a section within the paragraph with properties that differ from each other. The text content in the second paragraph run is displayed in bold and italics, while the first paragraph run does not specify these properties. The font colour of the second paragraph run also differs from the first paragraph run. The first paragraph implements the *automatic* font colour (usually black), while the font colour of the second paragraph run is *green*, specified in terms of its hexadecimal code. The *automatic* font colour is determined by the Windows Text colour, specified within the Window Colour and Appearance settings of Windows (Leonhard, 2009).

The text content of a paragraph might also be divided into separate paragraph runs if a particular section of the text content was modified or added during a different session. The WordprocessingML extract in Figure 4.6 illustrates a paragraph containing two runs that were created during separate sessions.

```
<w:p w:rsidR="00D45FD5" w:rsidRDefault="00960502">
  <w:pPr>...Global Run Properties...</w:pPr>
  <w:r w:rsidR="00774561">
    <w:rPr>
      <w:sz w:val="26" />
      <w:szCs w:val="26" />
      <w:lang w:val="en-US" />
    </w:rPr>
    <w:t xml:space="preserve">This example contains two runs </w:t>
  </w:r>
  <w:r>
    <w:rPr>
      <w:sz w:val="26" />
      <w:szCs w:val="26" />
      <w:lang w:val="en-US" />
    </w:rPr>
    <w:t>that were created during separate sessions.</w:t>
  </w:r>
</w:p>
```

Figure 4.6 Runs created during separate sessions

The paragraph in Figure 4.6 is separated into two runs, because the first section was modified during a different session, after the original paragraph had been constructed (ECMA, 2012). This feature allows different versions of the same document to be merged by the word processor. The merged document will then only contain the most recent modifications and additions to the document, as well as the original content that remained unchanged.

Another reason why a paragraph might be divided into multiple runs is when the text content contains spelling or grammar errors (ECMA, 2012). This was illustrated clearly in Figure 4.2. For every spelling and grammar error a separate paragraph run is inserted to contain the particular spelling or grammar error.

The comparison between paragraphs of two documents can create unnecessary complications if the number of paragraph runs, within the corresponding paragraphs, differs. It would be a lot simpler if redundant paragraph runs could be merged into single runs, so that the only multiple paragraph runs that remain are due to different format properties, specified within the separate runs. The OOXML SDK v2.0 for Microsoft Visual Studio provides classes and methods, similar to the parsing algorithms of Wolters (2010), to achieve this result by simplifying the WordprocessingML contained within the documents.

4.4.5 Drawings within paragraph runs

Paragraph runs may also contain graphical content, such as images, instead of text content. This is indicated by the DrawingML tag, *w:drawing*, as Figure 4.7 illustrates.

```
<w:p>
  <w:pPr>...Global Run Properties...</w:pPr>
  <w:r>
    <w:rPr>...Individual Run Properties...</w:rPr>
    <w:drawing>
      ...
      DrawingML mark-up
      ...
    </w:drawing>
  </w:r>
</w:p>
```

Figure 4.7 DrawingML tag within a paragraph run

As already discussed, the text content's format properties are specified prior to the text content within the individual run properties. In the presence of graphical content, the properties are specified within the DrawingML mark-up of the graphical content section itself (ECMA, 2012). This, however, falls outside the scope of this research project.

4.5 The core properties part

The core properties of a word-processing document are contained within the *core.xml* file of the DOCX archive, portrayed in Figure 4.1. These properties include the title of the document, the subject of the content in the document and the author of the document (ECMA, 2012), as illustrated in the Figure 4.8. A word-processing task might request students to specify these properties as part of the assignment. It would, therefore, also have to be compared to the memorandum's core properties for assessment purposes.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<cp:coreProperties ...>
  <dc:title>Arbor day</dc:title>
  <dc:subject>History</dc:subject>
  <dc:creator>WSJ Marais</dc:creator>
  <cp:lastModifiedBy>Jaco</cp:lastModifiedBy>
  <cp:revision>16</cp:revision>
  ...
  ...
</cp:coreProperties>
```

Figure 4.8 Core document properties

A comparison between the title and subject of the student's document and memorandum is straightforward, as the student's document title and subject should coincide with the memorandum if the student performed the task successfully. However, the creators of the documents will almost certainly differ, since the student's name might differ from the examiner. To alleviate this situation, a regular expression can be applied to assess whether the creator property contains a valid human name.

For the purpose of this research project, a regular expression is applied within the proposed OOXML algorithm to assess whether the student supplied valid initials followed by a surname. It might also be applied to determine whether the student saved the document under the correct file name.

4.6 The extended properties part

Some other properties that might be of interest with regard to the assessment of word-processing skills are the extended document properties (ECMA, 2012) displayed in Figure 4.9. These properties specify information relevant to the native application of the document. For assessment purposes the number of pages, paragraph count, word count, character count and number of whitespaces are of interest.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Properties ...>
  <Template>Normal.dotm</Template>
  <TotalTime>199</TotalTime>
  <Pages>3</Pages>
  <Words>257</Words>
  <Characters>1466</Characters>
  <Application>Microsoft Office Word</Application>
  <DocSecurity>0</DocSecurity>
  <Lines>12</Lines>
  <Paragraphs>3</Paragraphs>
  <ScaleCrop>false</ScaleCrop>
  ...
  <Company>Univeristy of the Free State</Company>
  <LinksUpToDate>false</LinksUpToDate>
  <CharactersWithSpaces>1720</CharactersWithSpaces>
  <SharedDoc>false</SharedDoc>
  <HyperlinksChanged>false</HyperlinksChanged>
  <AppVersion>14.0000</AppVersion>
</Properties>
```

Figure 4.9 Extended document properties

However, during the development of the proposed OOXML algorithm it was found that these properties are not regularly updated by the application and are therefore unreliable. Instead, the real-time extended properties of the student's and the examiner's documents are calculated and compared with one another.

4.7 The OOXML algorithm

The OOXML algorithm was developed as part of the research project in an effort to improve the OOXML-based algorithms of Pellet and Chevalier (2014), Lánskỳ et al. (2013) and Wolters (2010) as discussed in Chapter 1. The core concept of the OOXML algorithm is to inspect the individual paragraphs of the memorandum of the particular assignment, locate the corresponding paragraph within the student's document and compare them with one another. This is illustrated in Figure 4.10.

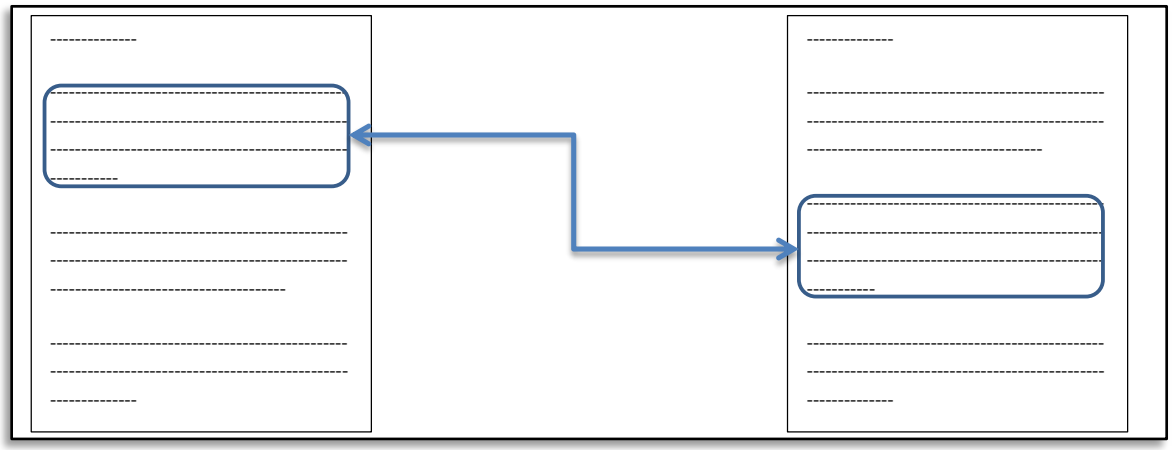


Figure 4.10 Locating corresponding paragraphs between two documents

4.7.1 The assignment memoranda

Before the assessment of the students' assignments commences, the memoranda of all the documents that had to be submitted as part of the assignment are parsed by means of the classes and methods provided by the OOXML SDK v2.0 (Lánskỳ et al., 2013). The parsing operation simplifies the WordprocessingML mark-up of the specified documents by removing all the information that is irrelevant to the assessment process, as well as information that can cause unnecessary nonconformity between the student's document and the relevant memorandum. This includes revision information, bookmarks and proofing errors, as well as the merging of multiple runs within the same paragraphs. Figure 4.11 provides the pseudo code for this operation.

```
For each memo of the assignment {
   Call the SimplifyDoc method, passing the memo;
}

SimplifyDoc(document) {
   Set the simplification settings of the SimplifyMarkupSettings class; //OOXML SDK
   Call the SimplifyMarkup method, passing the document and settings; //OOXML SDK
}
```

Figure 4.11 Pseudo code for parsing the memoranda

4.7.2 The essential document parts

To assess all the students' assignments, the OOXML algorithm has to evaluate all the documents in each student's submitted assignment. To accomplish this, the algorithm steps through all the submitted assignments, assignment by assignment, document by document. First, it parses the document that is currently being evaluated, similarly to parsing the memorandum. A copy of the original student document is retained to assess the document for grammar and spelling mistakes, since parsing the document removes all proofing information.

Thereafter, the simplified documents and memoranda are used to obtain the three essential parts from their DOCX archives (see Section 4.3), contained in the *document.xml*, *core.xml* and *app.xml* files. Utilising the classes and methods from the *.Net* framework, the OOXML algorithm converts these files into their object-oriented representations. The pseudo code for this operation is displayed in Figure 4.12.

```
For each student's assignment {
   For each document { // An assignment submission may contain multiple documents
      Call the SimplifyDoc method, passing the document;

      Create an object of the core document properties;
      Create an object of the core memo properties;
      Create an object of the extended document properties;
      Create an object of the extended memo properties;
      Create an object of the main document part of the original document;
      Create an object of the main document part of the simplified document;
      Create an object of the main memo part of the simplified memo;

      Call the AssessDocuments method, passing the core document properties, core
         memo properties, extended document properties, extended memo properties,
         main document part, main memo part and main original document part;
   }
}
```

Figure 4.12 Pseudo code for creating objects of the three essential document parts

4.7.3 Assessing the document parts

The essential document parts are assessed individually. The algorithm first assesses the core and extended properties of the student's document by comparing their objects with that of the memorandum's core and extended property objects. This is then followed by the assessment of the main document parts also by means of an object-oriented comparison. Figure 4.13 illustrates the method calls with regard to these operations in pseudo code.

```
AssessDocuments(coreDocProp, coreMemoProp, extDocProp, extMemoProp, mainDoc,
                mainMemo, mainOrigDoc) {
   Call the AssessCoreProperties method, passing the core document properties and
      core memo properties;
   Call the AssessExtendedProperties method, passing the extended document
      properties and extended memo properties;
   Call the AssessExtendedProperties method, passing the main document part and
      main memo part;
   Call the AssessMainDocument method, passing the main original document part,
      main document part and main memo part;
}
```

Figure 4.13 Pseudo code for assessing the document parts

4.7.4 Assessing the core properties

The core properties are assessed by matching the student's document core properties with the memorandum's core properties. This applies to the document title and subject. However, since the examiner's name might differ from the student's name, the author of the document is evaluated by means of a regular expression to determine if the relevant information is contained in the document's author property. The marks that are to be allocated are specified in the assignment. Figure 4.14 contains the pseudo code for the assessment of the core properties.

```
AssessCoreProperties(coreDocProp, coreMemoProp) {
   If the document and memo titles match {
      Add specified mark;
   }
   If the document and memo subjects match {
      Add specified mark;
   }
   If the document author matches the regular expression {
      Add specified mark;
   }
}
```

Figure 4.14 Pseudo code for assessing the core properties

4.7.5 Assessing the extended properties

Furthermore, the extended properties of the student's document are compared to that of the memorandum. Only the properties relevant to the assessment are evaluated. Irrelevant properties include the file creation date, the file modification date, the application used to create the document, the version of the application and more (see Section 4.6). The pseudo code for the comparison of the extended document properties are illustrated in Figure 4.15. The first method compares the number of pages that each document consists of.

```
AssessExtendedProperties(extDocProp, extMemoProp) {
   If the document and memo page count matches {
      Add specified mark;
   }
}

AssessExtendedProperties(mainDoc, mainMemo) {
   Determine the document and memo paragraph count;
   If the document and memo paragraph count matches {
      Add specified mark;
   }
   Determine the document and memo word count;
   If the document and memo word count matches {
      Add specified mark;
   }
   Determine the document and memo character count;
   If the document and memo character count matches {
      Add specified mark;
   }
   Determine the document and memo character and whitespace count;
   If the document and memo whitespace count matches {
      Add specified mark;
   }
   Check the document for redundant whitespaces;
   If the assignment requires maximum one mark deducted for redundant whitespaces {
      For one redundant whitespace found {
         Deduct one mark;
      }
   }
   Else {
      For each redundant whitespace found {
         Deduct one mark;
      }
   }
}
```

Figure 4.15 Pseudo code for assessing the extended properties

However, as mentioned in Section 4.6, some extended property values are unreliable since they are not regularly updated by the application. Therefore, the actual values are calculated and compared. This is illustrated by the second pseudo code method in Figure 4.15. The number of marks deducted for redundant whitespaces depend on whether the word-processing

assignment requires a deduction of one mark per redundant whitespace or a single mark deduction for all redundant whitespaces in the document.

4.7.6 Assessing the main document part

Assessing the main document part comprises checking for spelling and grammar errors (see Section 4.4.1), evaluating the document layout properties (see Section 4.4.2) and evaluating properties and content of every individual paragraph. The pseudo code of these operations is displayed in Figure 4.16.

```
AssessMainDocument(mainOrigDoc, mainDoc, mainMemo) {
   Call the CheckSpellingAndGrammar method, passing the main original document part
      and the SpellStart proofing error value;
   Call the CheckSpellingAndGrammar method, passing the main original document part
      and the GrammarStart proofing error value;
   Call the CheckLayoutProperties method, passing the main document part and main
      memo part;
   Call the CheckParagraphs method, passing the main document part and main memo
      part;
}
```

Figure 4.16 Pseudo code for assessing the main document part

4.7.7 Assessing the grammar and spelling

First, the grammar of the document is assessed by searching for grammar proofing errors. Thereafter, the spelling errors are penalised. Marks are deducted, depending on whether the word-processing assignment requires a deduction of one mark per proofing error or a single mark deduction for all the proofing errors in the document. This assessment approach is illustrated by the pseudo code in Figure 4.17.

```
CheckSpellingAndGrammar(mainOrigDoc, proofValue) {
   If the assignment requires maximum one mark deducted for all proofing errors {
      For one proofing error found {
         Deduct one mark;
      }
   }
   Else {
      For each proofing error found {
         Deduct one mark;
      }
   }
}
```

Figure 4.17 Pseudo code for assessing the grammar and spelling

4.7.8 Assessing the document layout properties

The properties that determine the layout of the document, as specified in Section 4.4.2, are evaluated by comparing each property that is specified in the memorandum to its corresponding property in the document, if it exists. If the particular layout property is missing from the student's document it is simply not compared and no marks are obtained. This approach is illustrated in Figure 4.18 by the relevant pseudo code.

```
CheckLayoutProperties(mainDoc, mainMemo) {
   For each layout property specified within the memo {
      If the particular document and memo layout property matches {
         Add specified mark;
      }
   }
}
```

Figure 4.18 Pseudo code for assessing the layout properties

4.7.9 Assessing the paragraphs

The assessment of the individual document paragraphs is achieved by first locating and then matching the specified paragraph with its corresponding memorandum paragraph. To locate a particular paragraph from the student's document that corresponds with a paragraph from the memorandum, the Levenshtein algorithm (Levenshtein, 1966) is applied. This operation is demonstrated by the pseudo code in Figure 4.19.

```
CheckParagraphs(mainDoc, mainMemo) {
   For each paragraph of the memo {
      For each paragraph of the document {
         Calculate the Levenshtein distance with regard to their text content;
      }
      Determine which document paragraph produced the smallest distance;
      Call the CheckParagraphProperties method, passing the document and memo
         paragraph that corresponds;
      Call the CheckParagraphRun method, passing the document and memo paragraph
         that corresponds;
   }
}
```

Figure 4.19 Pseudo code for assessing the paragraphs

Every paragraph from the student's document is paired with every paragraph from the memorandum and the Levenshtein distance between the paragraph pairs are calculated. The paragraph pair that produces the smallest distance is recognised as the closest corresponding pair and is then matched to determine if the student's paragraph contains the relevant content

and formatting of the memorandum's corresponding paragraph. The assessment is conducted by comparing each paragraph property and paragraph run that are specified in the memorandum to its corresponding property or run in the document. If, however, the particular element is missing from the student's document it is not compared and no marks are allocated, as indicated in Figure 4.20.

```
CheckParagraphProperties(paraDoc, paraMemo) {
   For each paragraph property specified within the memo {
      If the particular document and memo paragraph property matches {
         Add specified mark;
      }
   }
}

CheckParagraphRun(paraDoc, paraMemo) {
   For each paragraph run specified within the memo {
      For each paragraph run property  specified within the memo {
         If the particular document and memo paragraph run property matches {
            Add specified mark;
         }
      }
      If the particular document and memo paragraph run content matches {
         Add specified mark;
      }
   }
}
```

Figure 4.20 Pseudo code for assessing the paragraph properties and runs

After every document of a particular assignment has been assessed, the assignment's maximum mark is recalculated by adding the document mark that could have been obtained had the document contained no errors. The assessment score that the student received for the particular document is also added to the total score already obtained. The recalculation operation is illustrated by the pseudo code in Figure 4.21. If the document has been saved according to the filename structure provided in the assignment specifications, another mark is allocated to the assessment result.

```
For each student's assignment {
   For each document {
      Add the document's mark to the assignment's maximum mark;
      Add the document's assessment score to the student's total score;

      If the document filename matches the regular expression {
         Add specified mark;
      }
   }
}
```

Figure 4.21 Pseudo code for recalculating the assessment score totals

The final assessment result is calculated by dividing the student's total score by the possible assignment total. This produces a value that indicates how much of the word-processing assignment was completed successfully.

4.8 Chapter summary

In this chapter, the OOXML standard, as defined by ECMA International (ECMA, n.d.), was described and investigated. The discussion included the structure and content of an OOXML-based word-processing document. The three main parts of an OOXML-based document were discussed with regard to their properties, attributes and content applicable to this research project.

The chapter concluded with a detailed explanation of the inner workings of the OOXML assessment algorithm, developed as part of this research project. The explanation included the pseudo code of the algorithm as it is implemented. As part of the research study, this algorithm and three other established algorithms, namely fComp, the Levenshtein algorithm and Word Graders' assessment algorithm, will be used to assess a group of selected word-processing assignments and the assessment results compared. Chapter 5 discusses the operating procedures of two computerised assessment systems, namely Word Assessment Manager and Word Grader that implement the specified algorithms.

Chapter 5

A Purpose Built Assessment System

This chapter focuses on the following aspects:

- **Word Assessment Manager**
- **Word Grader**

5.1 Introduction

The previous chapter provided an overview of the OOXML document format. It discussed the structural layout of a DOCX word-processing document and explained the operation of the proposed OOXML algorithm that was designed as part of this research project. In this chapter, the operating procedures of two computerised assessment systems are discussed. These systems embed the algorithms that were selected for the quasi-experimental study (see Section 3.6). The proposed OOXML algorithm, Levenshtein algorithm and fComp are embedded in the first system, called Word Assessment Manager (WAM), while Word Grader, a commercially available assessment system, embeds a Compare and Combine algorithm, described in Section 3.4.2. The technical specifications and components of each system are described in the following sections.

5.2 Word Assessment Manager

WAM is a purpose built computerised assessment system, developed as part of this research project, to assess students' word-processing assignments through the individual implementation of selected assessment algorithms, mentioned in Section 5.1. WAM was developed through the use of the OOXML SDK v2.0 for Microsoft Visual Studio in the C# programming language. The following sections describe the basic functionality of WAM. A detailed discussion of the development of WAM falls outside the scope of this research project.

5.2.1 Technical specifications

WAM is compatible with all Windows operating systems that implement .*Net* Framework 4.5. It was tested on various Windows versions that included, Windows 7, 8 and 8.1. During the

assessment process several temporary directories are created, files are extracted and documents are copied. Therefore, a reasonable amount of hard drive space is required, depending on the number of documents that are evaluated. The following hardware specifications are recommended:

- 1 GHz or faster 32-bit (x86) or 64-bit (x64) processor
- 1 GB RAM for 32-bit systems or 2 GB RAM for 64-bit systems
- 5 GB of available hard drive space
- VGA graphics card and a 1024 × 576 or higher resolution monitor

5.2.2 The application components

WAM consists of three main components: An archive extraction component, an assessment component and an export results component. Figure 5.1 displays the main application window of the system. The following sections describe the basic functionality of WAM.

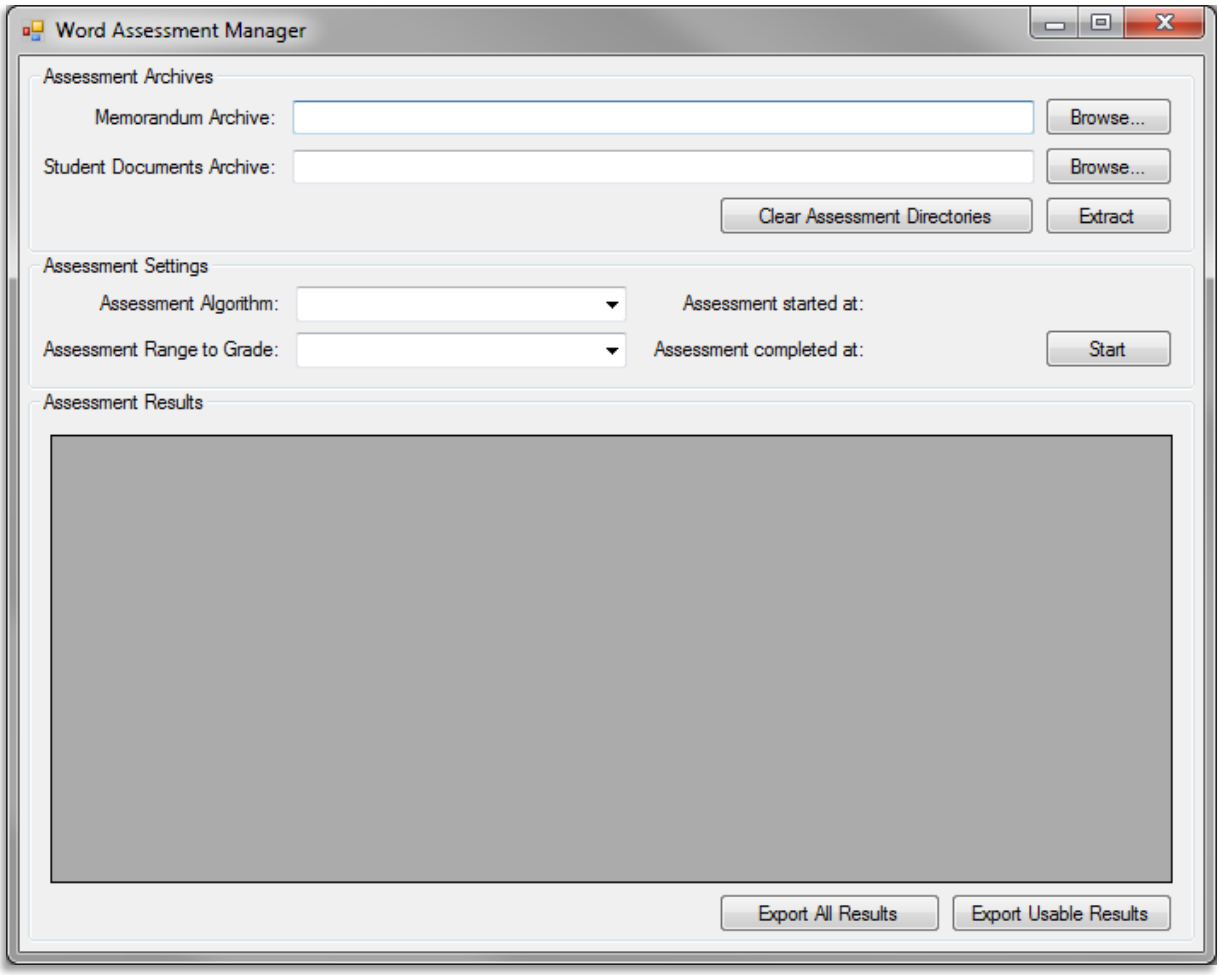


Figure 5.1 Word Assessment Manager

5.2.3 Extracting the assignments and memorandums

The students are required to compress their final documents into a single ZIP file[1] before submitting them to a learning management system (see Section 6.4.1), called Blackboard (Blackboard Inc., n.d.), used by the University of the Free State. Thus, before the assignments can be assessed, the content of these ZIP files have to be extracted. The memoranda of the documents that the students had to submit are also contained in a ZIP file that has to be extracted. This is accomplished by the extraction component of WAM. Figure 5.2 illustrates the user interface of the extraction component. To extract the ZIP files, the user has to browse and locate them on any storage device connected to the computer. The extraction sequence is initiated by clicking the Extract button.

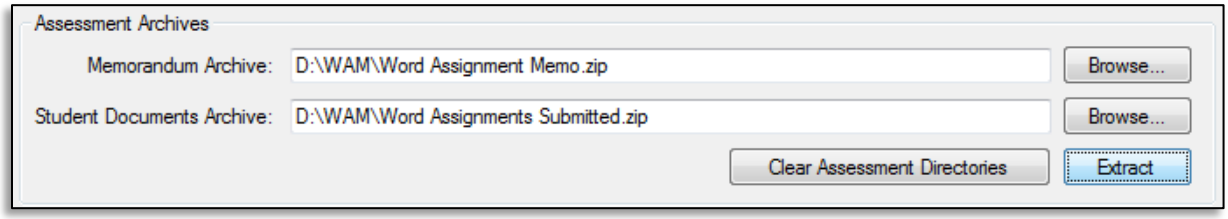


Figure 5.2 Extraction component of WAM

Contained in these ZIP files are the relevant word-processing documents that the students had to submit, as well as the assessment results that were manually assigned to the students' assignments by multiple human markers. During the extraction process, temporary directories are created, containing each individual student's documents that need to be assessed by WAM. The memorandum for each document that the students had to submit is also extracted to a temporary directory. WAM, therefore, provides the ability to automatically assess multiple documents within multiple assignments in a single run.

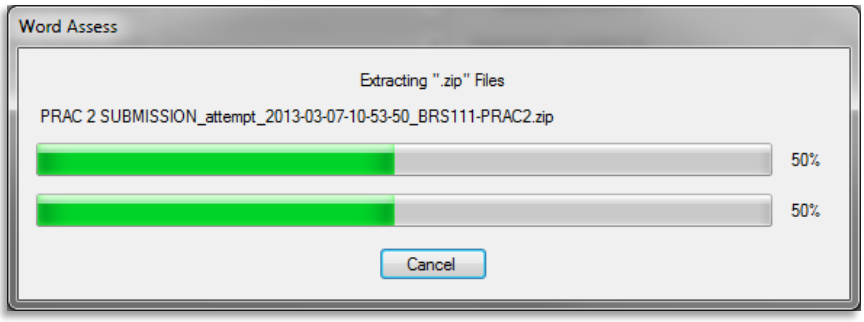


Figure 5.3 Extracting the ZIP files

[1] A ZIP file is a container that comprises of one or multiple files that have been compressed into a single file called an archive and usually has a *.zip* file extension (U.S. Patent No. US 6,879,988 B2, 2005)

5.2.4 Assessing the assignments

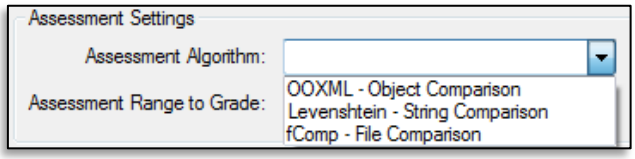


Figure 5.4 Selecting assessment algorithm

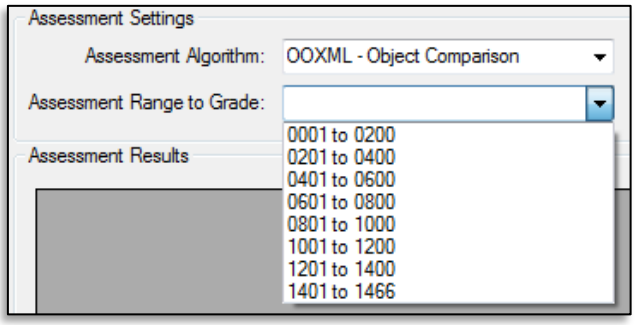
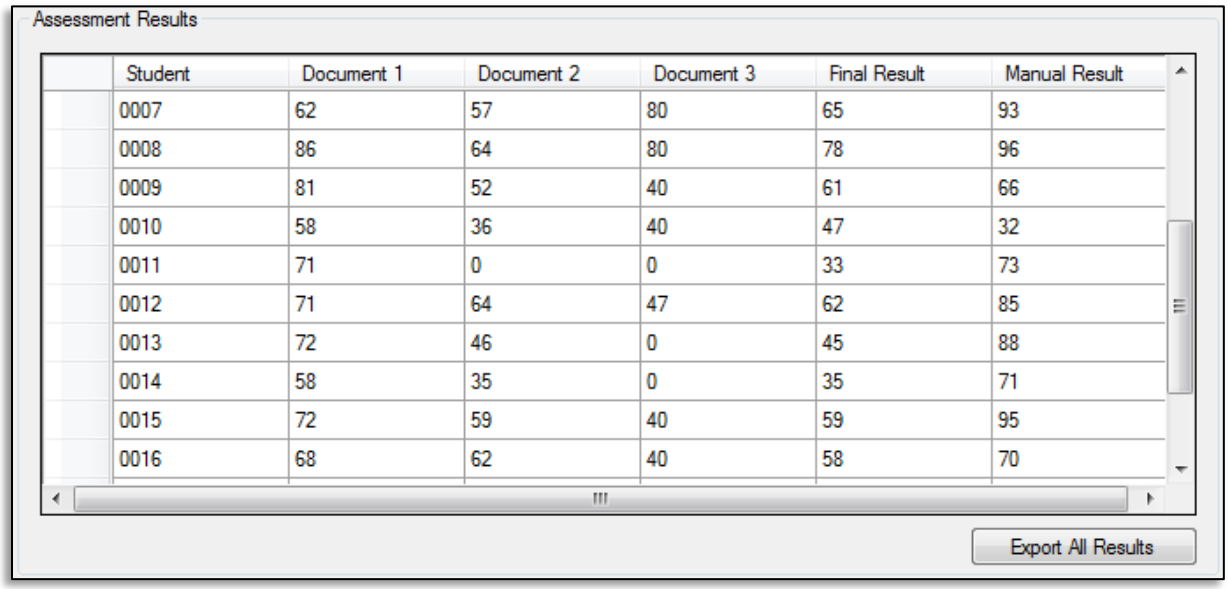


Figure 5.5 Selecting assessment range

After the relevant documents have been extracted, the assessment component of WAM is used to evaluate all the student assignments and assign a mark to each student's assignment. Before the assessment can commence, the desired assessment algorithm has to be selected from three available algorithms, illustrated in Figure 5.4. These algorithms were selected to be implemented by WAM in Section 3.6. The desired number of assignments to be graded has to be selected also, as illustrated in Figure 5.5. Thereafter, the assessment sequence is initiated by clicking the Start button. The objective is to assess the same word-processing assignments with each of the chosen algorithms. In this way the computerised assessment results needed for the quasi-experimental study are obtained.

5.2.5 Exporting the assessment results

After the assignments have been evaluated, their assessment results are displayed in the Assessment Results section of the application's user interface, illustrated in Figure 5.6.



Student	Document 1	Document 2	Document 3	Final Result	Manual Result
0007	62	57	80	65	93
0008	86	64	80	78	96
0009	81	52	40	61	66
0010	58	36	40	47	32
0011	71	0	0	33	73
0012	71	64	47	62	85
0013	72	46	0	45	88
0014	58	35	0	35	71
0015	72	59	40	59	95
0016	68	62	40	58	70

Figure 5.6 Assessment results produced by WAM

The results include the individual assessment results for each document the students had to submit, the final assessment result, which is a weighted average of the individual documents' assessment results and the manual assessment results assigned to each student's word-processing assignment. For the purpose of the research study, the students' word-processing assignments are anonymised (see Section 6.7) by assigning an integer value to each student as indicated in Figure 5.6. The assessment results can be exported in the form of a CSV file[2] by clicking the Export All Results button. For the purpose of this research project the assessment results were exported to be statistically analysed in the discussion in Chapter 7.

5.3 Word Grader

Word Grader (Hill, 2011) is commercially available from: http://www.officegrader.com. The trial version (see Figure 5.7) can also be downloaded from the official website (Hill, 2014).

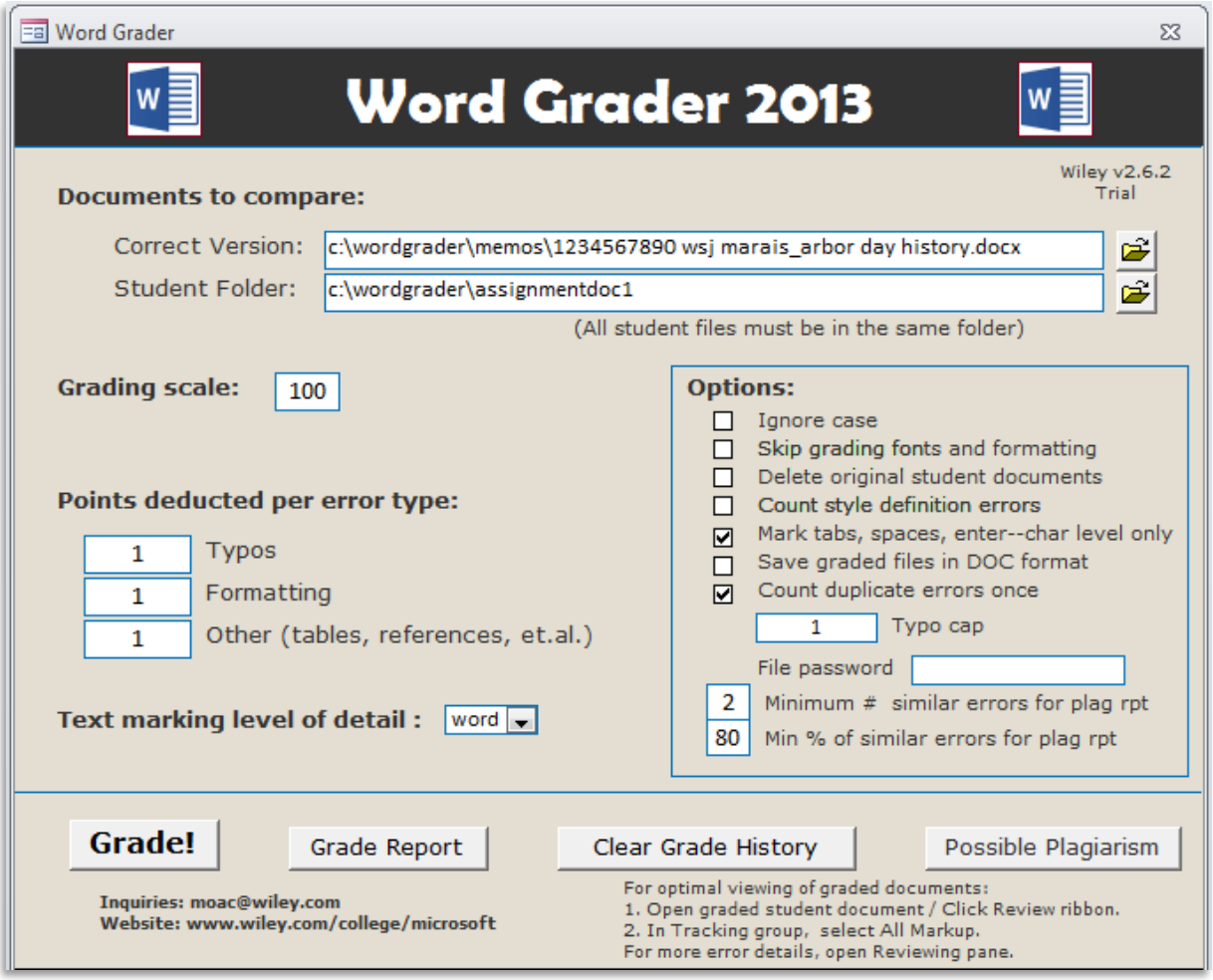


Figure 5.7 Word Grader

[2] A CSV file is a comma delimited file (Repici, 2010)

The trial version provides full functionality. However, it can only be used for a maximum of ten assessment sessions, whereupon the trial period expires. The trial version is, therefore, acceptable for the purpose of this research project, since only three assessment sessions are necessary to evaluate the three documents that formed part of the word-processing assignment that the students had to complete (see Section 6.4.1).

5.3.1 Technical specifications

According to the answers provided in a frequently asked questions document distributed by John Wiley and Sons Inc. (see Appendix C), Word Grader 2013 is compatible with Microsoft Office 2007 and later versions, including Microsoft Office 2013. The following system specifications are recommended for Microsoft Office 2013 (Microsoft Inc., 2015):

- 1 GHz or faster 32-bit (x86) or 64-bit (x64) processor
- 1 GB RAM for 32-bit systems or 2 GB RAM for 64-bit systems
- 3 GB of available hard drive space
- DirectX10 graphics card and a 1024 × 576 or higher resolution monitor
- .*Net* Framework 3.5, 4.0 or 4.5.
- Microsoft Windows 7 or later

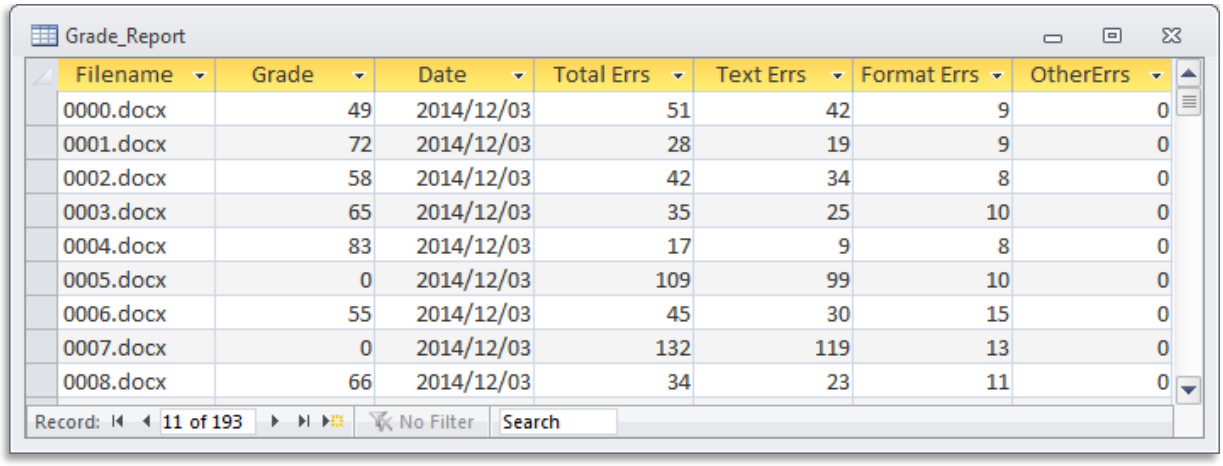
5.3.2 Assessing the assignments

Before Word Grader is able to assess the students' word-processing assignments, all the relevant documents need to be copied to the same directory. Thereafter, the user needs to set the relevant assessment options visible on the main application windows. For the purpose of this research project the recommended assessment options were set. These options are available in the answers to frequently asked questions with regard to Word Grader (see Appendix C). The assessment is initiated by browsing for the relevant word-processing document directory as well as the memorandum for the particular document and clicking the Grade button.

5.3.3 The assignment results

During the assessment process a grade report is generated in a Microsoft Access database table, illustrated in Figure 5.8. The grade report can be viewed by clicking the Grade Report

button on the main application window. The assessment results can also be exported as a Microsoft Excel Workbook. Again, this provides a convenient method for retrieving the assessment results for the statistical analysis conducted in the quasi-experiment in Chapter 7.



Grade_Report

Filename	Grade	Date	Total Errs	Text Errs	Format Errs	OtherErrs
0000.docx	49	2014/12/03	51	42	9	0
0001.docx	72	2014/12/03	28	19	9	0
0002.docx	58	2014/12/03	42	34	8	0
0003.docx	65	2014/12/03	35	25	10	0
0004.docx	83	2014/12/03	17	9	8	0
0005.docx	0	2014/12/03	109	99	10	0
0006.docx	55	2014/12/03	45	30	15	0
0007.docx	0	2014/12/03	132	119	13	0
0008.docx	66	2014/12/03	34	23	11	0

Record: 11 of 193 No Filter Search

Figure 5.8 Word Grader assessment results

5.4 Chapter summary

Chapter 5 demonstrated how two computerised assessment systems, namely WAM and Word Grader, assess students' word-processing assignments. The technical specifications of each system were specified and the different components and operating procedures were discussed, including methods to export the assessment results. Chapter 6 discusses the research design and methodology of the research study that was implemented to perform comparisons with regard to the above mentioned assessment results.

Chapter 6

Using Repetitive Measures to Compare Similarity Metrics

This chapter focuses on the following aspects:

- **The methodology of the research study**
- **The population size and sample selection**
- **The assessment procedure**
- **Ethical considerations with regard to the study**
- **The statistical analysis methods implemented in the research study**

6.1 Introduction

The previous chapter provided an overview of Word Assessment Manager (WAM) that was developed to provide a single platform from where different assessment algorithms, specified in Section 3.6, could be implemented to assess word-processing assignments. It also provided an overview of Word Grader and discussed the operating procedures and technical specifications of both systems. In this chapter, the methodology implemented in the research study is described. The research design, collection of assessment material, population and sample selection, as well as the environment in which the study was conducted, are described. The assessment procedures implemented on the word-processing assignments, the data analysis techniques and the ethical issues with regard to the study are also discussed.

6.2 Research approach and design

A quantitative research approach is applied. According to Creswell (2003), quantitative research involves testing a theory or hypothesis. Grove, Gray and Burns (2014, p. 32) define quantitative research as *"a formal, objective, rigorous, systematic process for generating numerical information"*. The objective of quantitative research is to determine whether a relationship exists between an independent and dependent variable within a population and what that relationship might be (Hopkins, 2008). The research study aims to determine whether or not the assessment results (dependent variable) of word-processing assignments (independent variable) differ when assessed by alternate similarity metrics.

The study implements a single-factor repeated-measures quasi-experimental design. Repeated measures are used when the outcome variable is measured on multiple occasions or under multiple experimental conditions (Ellis, 1999; Lawal, 2014). The students' word-processing assignment submissions are assessed on multiple occasions, but by a different similarity metric at each occasion. In this case the outcome variable is the assessment results and the experimental conditions are the application of different similarity metrics. In essence, the repeated measures research design entails that each student's submitted assignment produces multiple sets of data.

According to Cook, Campbell and Day (1979), repetitive measures may be applied in a quasi-experimental design. A quasi-experimental design involves at least one independent variable, where the independent variable is not assigned randomly. It differs from an experimental research design, where the independent variable is assigned at random (Ellis, 1999). The experiment in this research study uses only one independent variable, namely the word-processing assignments of the students. This is referred to as a single-factor experiment and measures the effect of a single independent variable on a dependent variable (Gliner, Morgan, & Leech, 2011).

6.3 Research environment

The study was conducted in the Department of Computer Science and Informatics on the main campus of the University of the Free State in Bloemfontein, South Africa. The University enrols about 2000 students for a first year, first semester computer proficiency module each year. As part of the module curricula, the students have to complete several computer proficiency assignments, which include a word-processing assignment. The completed assignments have to be submitted on the university's learning management system (University of the Free State, n.d.), called Blackboard.

In 2013, after the students had already completed the particular word-processing assignment and it had already been manually assessed, it was decided to use the submitted assignments in the research study. The students, therefore, completed the word-processing assignment under normal conditions. The results of the research study would, therefore, not be affected by any outside factors, such as students being nervous due to knowing that they were participating in a research study.

6.4 The research population and sample

Gliner et al. (2011) define the research population as the entire group of elements (individuals, objects or events) that are relevant to the study, from which the sample is selected. A total of 1578 word-processing assignments were submitted. Unfortunately, as explained in Section 6.4.2, only 1466 submitted assignments could be included in the research population for the purpose of being graded by WAM, a computerised assessment system described in Chapter 5.

6.4.1 An overview of the word-processing assignment

The students had to complete a word-processing assignment (see Appendix A for the detailed assignment tasks) and submit their assignments by means of a learning management system, called Blackboard (Blackboard Inc., n.d.; University of the Free State, n.d.). The assignment contained six sections.

The first section dealt with the procedure that the students had to follow to login on the computers. The second section explained the procedure for downloading the assignment from Blackboard. In the following three sections, the students had to create three separate documents; one for each section. These three sections specified the word-processing tasks that the students had to perform, such as copying and pasting text phrases, inserting images into the document, inserting, deleting and replacing document elements, changing the style and format of document elements and altering the layout and attributes of the document. The final section of the assignment specified the assignment submission procedures.

Although the word-processing assignment itself does not form part of the research project, it was still necessary to investigate it in order to determine what was expected of the students, specifically regarding the submission procedures. For submission purposes the students had to create a single ZIP file containing the three documents, and submit the ZIP file via Blackboard. This method of submission is taught to the students at the start of the semester and was included on the Blackboard submission interface. The information was vital to the development of WAM, since it influenced the decision to include an automated extraction algorithm within the system (see Chapter 5).

6.4.2 Identifying the population

Unfortunately, not all the assignments were submitted according to the submission guidelines. This might have been due to the students not noticing the submission procedure on Blackboard or to the fact that it was not fully stipulated in the assignment (see Appendix A). Thus, from 1578 submissions retrieved from Blackboard, only 1490 were comprised of ZIP files and could therefore be extracted successfully. The submission figures of the assignments are displayed in Table 6.1.

Table 6.1 Assignment submission figures

Submitted assignments	Submission count	
Corrupted ZIP files	1	
Empty ZIP files	5	
Duplicate ZIP files	18	
ZIP files excluded from population	24	
ZIP files included in population	1466	
Total ZIP files		1490
Non-ZIP files		88
Total submissions		1578

One of the submitted ZIP files was corrupted and could not be extracted successfully. There were also 5 empty ZIP files that contained no documents and 18 of the 1490 ZIP files had been submitted twice. Therefore, a total of 24 submitted ZIP files (1 corrupted, 5 empty and 18 duplicates) could not be included in the population. This is of no consequence to the research project, since it still provides an adequate population size of 1466 assignments to select a sample from, as Figure 6.1 illustrates.

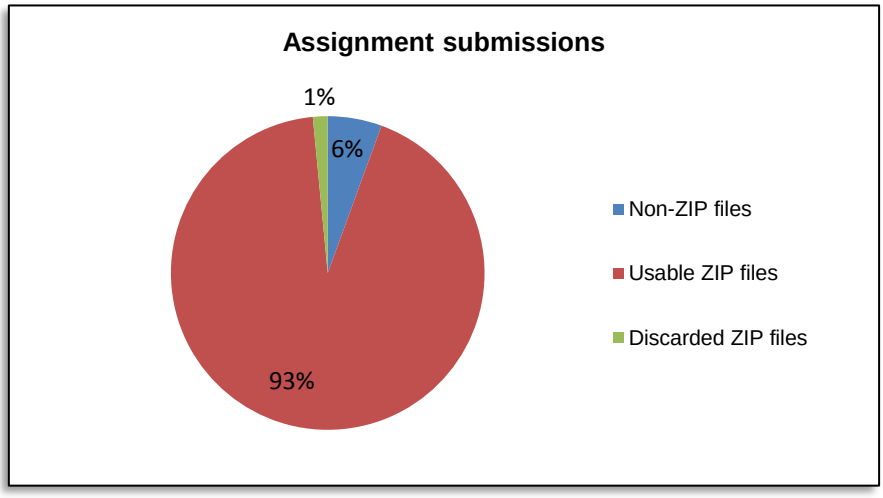


Figure 6.1 ZIP file ratio of submitted assignments

6.4.3 Creating a benchmark

One of the objectives of this research project, stated in Chapter 1, is to determine the accuracy (see Section 1.4) and reliability (see Section 1.4) of similarity metrics involved in the computerised assessment of word-processing skills. This could be accomplished by comparing the computerised assessment results with the manual assessment results of human markers. The manual assessment of the assignments was therefore conducted by multiple human markers that acted as student assistants for the relevant computer proficiency module. About 80 student assistants each marked an equal share of the submitted word-processing assignments. The student assistants are annually appointed based on their academic achievement for the same computer proficiency module. However, as explained in Section 2.4.1, the manual assessment of word-processing skills can be inconsistent and questionable (Dowsing et al., 1998). It is thus necessary to provide reliable assessment results as a benchmark, not just for computerised assessment results, but also for human markers' manual assessment results.

6.4.4 Selecting and re-assessing the sample

The need to provide a benchmark for the assessment results required the re-assessment of the submitted assignments. In an effort to provide reliable benchmark results, the re-assessment was carried out by the instructor of the computer proficiency module to reduce the inconsistent allocation of marks. Figure 6.2 illustrates the portion of the submitted assignments that was selected to be re-assessed.

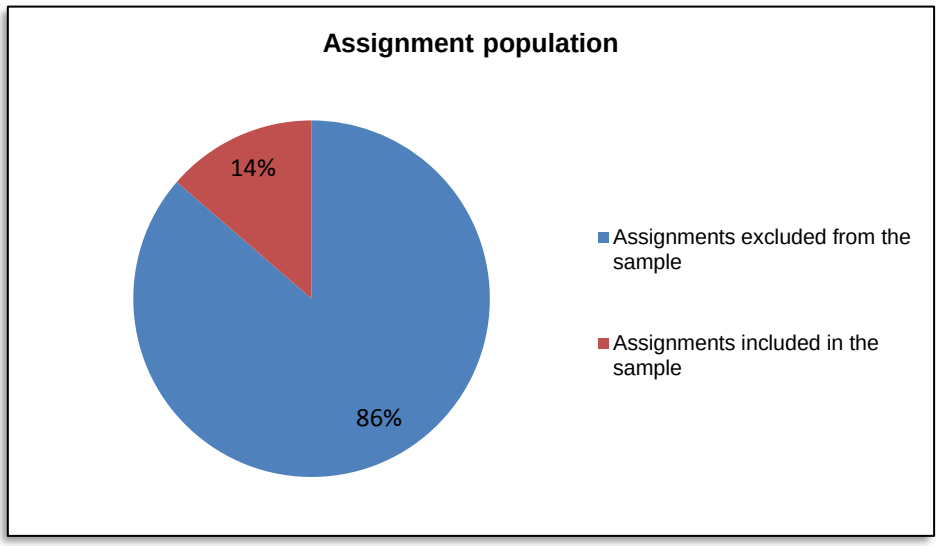


Figure 6.2 Assignment population and sample ratio

A purposive sample of 200 assignments was non-randomly selected from the population of 1466 assignments. According to Fraenkel, Wallen, and Hyun (1993), a purposive sample is a group of elements that are selected from a population because they possess the necessary characteristics required for the study.

The selection was conducted by ranking the assignments according to the student numbers of the students that submitted the assignments. The first 200 assignments, in ranking order, were then selected to be re-assessed for the purpose of creating a benchmark. This allowed the instructor to complete the assessment within a reasonable time frame, yet still provide sufficient assessment results for statistical analysis that are representative of the population.

6.5 Verification of assessment parameters

To ensure that the assessment results produced by WAM and Word Grader are only influenced by the algorithms that they implement and not because of any flaws within the word-processing assignment or memorandum, the memorandum was verified for accuracy. This was achieved by evaluating the memorandum through manual verification as well as by means of the relevant assessment algorithms.

It is also necessary to provide Word Grader with the correct assessment options that can be set on its main application window, as discussed in Section 5.3.2. Therefore, the recommended options specified by John Wiley and Sons Inc. (see Appendix C) were obtained. Using this as a reference, the following options were set:

- The option to ignore the letter case was left unchecked, since the other algorithms were critical towards this.
- The Skip Grading Fonts and Formatting option was also left uncheck, since this is the purpose of the assessment.
- Counting style definition errors produced very poor assessment results and it was therefore decided to adhere to the recommended setting of unchecking this option.
- The Mark White Space Characters option was selected.
- The option to override the character count specifies the amount of characters added to the starting document. These values were therefore specified for each document in the assignment.

6.6 The computerised assessment procedure

The assessment algorithms chosen in Section 3.6 that include the proposed OOXML algorithm, Levenshtein algorithm, fComp and Compare and Combine algorithm implemented by Word Grader, were separately applied during the assessment process.

6.6.1 First assessment

First, the proposed OOXML algorithm (see Section 4.7) was implemented by WAM to assess the 200 selected word-processing assignments with its object comparison technique. Secondly, the Levenshtein algorithm applied the Levenshtein Distance metric to evaluate the same sample of assignments. This was followed by the byte comparison algorithm, applied by fComp, and finally, Word Grader was used to assess the assignments by means of its document difference comparison algorithm. All the assessment results were recorded and exported according to the operational procedures described in Chapter 5.

6.6.2 Second assessment

As specified in Chapter 1, the research study intends to compare the reliability and accuracy of computerised assessment with that of human markers. Accuracy and reliability as it pertains to this research project was defined in Section 1.4. The benchmark, discussed in Section 6.4.4, provides assessment results for the purpose of analysing the various algorithms' accuracy (see Section 7.6). In order to determine the reliability of the algorithms another set of computerised assessment results is required. For this reason the same 200 assignments were assessed a second time. The same procedure, as in Section 6.6.1., was followed.

6.7 Ethical considerations

During the assessment procedure certain ethical considerations were taken into account. To render the study ethical, the students' right to anonymity and confidentiality were conserved. Pfitzmann and Köhntopp (2001) define anonymity as the inability to link individuals to information relevant to them. To accomplish this, the assessment system replaced the individuals' student numbers with an integer value that could not be linked them. The research study was interested only in the students' assessment results and not their identities. Confidentiality was also maintained by not revealing the identities of the creators of the word-

processing documents during the reporting or publishing of this research project. When individuals are assured that personal information will not be disclosed, it is referred to as being confidential (O'Brien & Yasnoff, 1999).

Verbal permission was obtained from the coordinator of the computer literacy module and the head of the Department of Computer Science and Informatics, at the University of the Free State, to use the students' word-processing assignments in this research project.

6.8 Statistical analysis

After exporting the assessment results from the assessment systems, they were imported to a computer application called Statistical Package for Social Sciences (SPSS). The assessment results were statistically analysed through descriptive statistics. Paired-sample comparative tests were conducted between the relevant assessment results to determine if there existed a difference between the results obtained from the different algorithms. The correlation between the various algorithms' assessment results was also determined through the calculation of the correlation coefficient.

6.9 Chapter summary

This chapter provided an overview of the methodology implemented by the research study. The research design was described and the research environment identified. A sample of 200 word-processing assignments was selected from the population of successfully submitted assignments. These assignments were re-graded by a single marker, the instructor of the computer proficiency module, to provide a benchmark for the results of the other assessment methods.

The assessment parameters for the assessment process were double checked. Thereafter the selected assignments were consecutively assessed by each of the chosen assessment algorithms and the assessment results were exported to SPSS. In the following chapter, the statistical analyses of the assessment results are described.

Chapter 7

The Quasi-Experimental Study

This chapter focuses on the following aspects:

- **Comparing the assessment results of human markers and computerised assessment algorithms with the benchmark assessment results**
- **Comparing the assessment results provided by human markers with those produced by the assessment algorithms**
- **Comparing the object comparison algorithm with other specified algorithms**

7.1 Introduction

In Chapter 6, the research design and methodology of the study were discussed. The research sample of word-processing assignments was selected from the population and assessed according to the procedure specified in Section 6.6.1. The objective of Chapter 7 is to determine how the assessment algorithms' results compare with the manual assessment results of multiple human markers. This should provide a reasonable indication of the accuracy[1] and reliability[2] of the similarity metrics applied by the algorithms, especially the document object comparison implemented by the OOXML algorithm (see Section 4.7) developed as part of this research project. Another objective is to evaluate whether the proposed OOXML algorithm yields assessment results that are comparable with other established algorithms.

In order to achieve the specified objectives, separate paired-sample comparative tests are conducted and statistically analysed: First, the assessment results assigned by multiple human markers and those produced by each assessment algorithm (see Section 6.6.1) are individually paired with the benchmark assessment results (discussed in Section 6.4.4) for the same student assignments. Secondly, the assessment results of the human markers are separately paired with the assessment algorithms' results. Finally, the assessment results produced by the object comparison algorithm (see Section 6.6.1) are separately paired with the remaining algorithms' assessment results. To complement this, the correlation coefficients between the

[1] Accuracy refers to the degree to which a measured result conforms to a standard or true value (Menditto, Patriarca, & Magnusson, 2006)

[2] Reliability is determined by evaluating whether the assessment results are stable and consistent (Carmines & Zeller, 1979)

distributions of the assessment results are calculated to determine the existence of a linear relationship between the allocations of marks by each assessment method. The purpose for each statistical test is explained and the analysis results are interpreted and discussed.

7.2 Statistical analysis of the assessment results

A selected sample of 200 assignments, specified in Section 6.4.4, was graded by WAM and Word Grader through the implementation of four different document analysis algorithms that each applies a specific similarity metric (see Section 6.6.1). The results produced by the computerised assessment algorithms and multiple human markers were statistically analysed to determine how they compare to the benchmark (see Section 6.4.4) assessment results. A descriptive analysis of the entire sample's assessment results is displayed in Table 7.1.

Table 7.1 Descriptive analysis report of the assessment results distributions

	Benchmark	Markers	OOXML	Levenshtein	fComp	Word Grader
N	200	200	200	200	200	200
Mean	55.37	68.77	53.89	59.11	63.38	46.05
Std. Error of Mean	1.366	1.817	1.274	1.117	1.236	1.654
Std. Deviation	19.318	25.702	18.020	15.797	17.478	23.391
Skewness	-0.955	-1.213	-1.102	-1.176	-1.130	-0.208
Std. Error of Skewness	.172	.172	.172	.172	.172	.172
Median	60.50	75.50	59.00	63.00	67.00	49.00
Minimum	0	0	0	0	0	0
Maximum	86	100	82	81	86	86

The means of all the assessment results differ from that of the benchmark. The assessment results of Word Grader produced the lowest mean value ($\bar{x}$ = 46.05, s = 23.391). As mentioned in Section 6.5, Word Grader contains option settings that have an influence on the assessment results it produces. Word Grader was, however, employed with its recommended settings. The mean of the OOXML assessment results ($\bar{x}$ = 53.89, s = 18.020) differs the least from that of the benchmark's mean value ($\bar{x}$ = 55.37, s = 19.318). Its value is also lower than the benchmark's mean, which might indicate that the proposed OOXML algorithm implements more stringent assessment criteria. On the other hand, the mean of the human markers' results ($\bar{x}$ = 68.77, s = 25.702) differ the most from that of the benchmark. It also has a higher value than the benchmark's mean. Figure 7.1 visually illustrates the differences between the means.

The maximum value of the assessment results produced by human markers is 100, which is the highest of all the assessment methods. This, combined with the human markers' results having the highest mean value, confirms the statement by Dowsing et al. (1998) that human markers produce high pass rates for word-processing skills assessments and that many students receive full marks for their assessments.

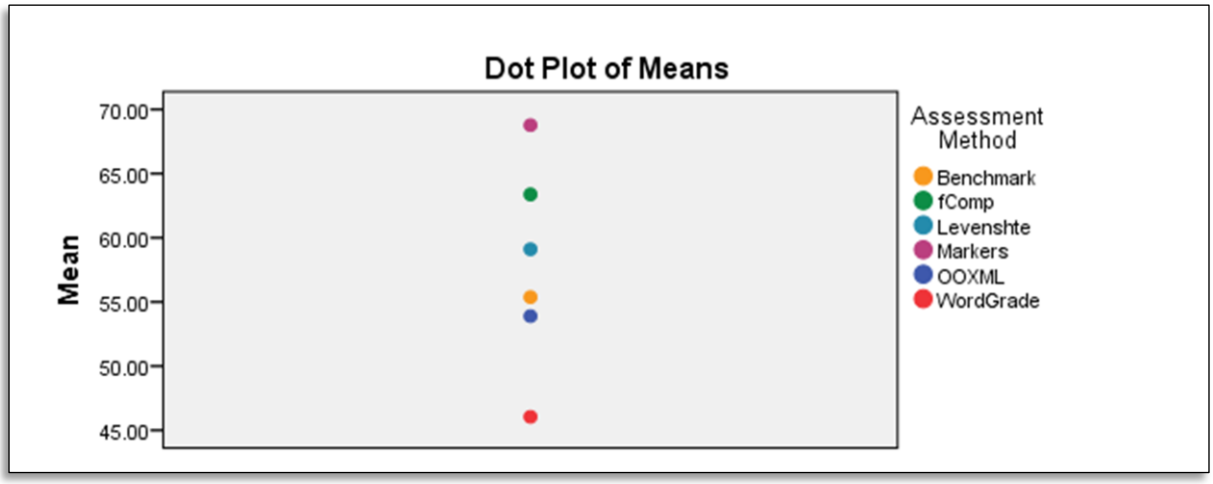


Figure 7.1 Dot plot of the assessment methods' means

By examining the median and skewness of the distributions in Table 7.1, it can be seen that the assessment results for each distribution, except for Word Grader, are skewed to the right. This is indicated by each relevant distribution's negative skewness factor in combination with its median being noticeably higher than the value of the distribution's mean. Therefore, it might be necessary to perform a logarithmic transformation on the reflected assessment results, if parametric statistical analysis, such as Pearson's correlation, is to be performed on the data.

Further statistical analysis was conducted on the assessment results to determine whether the computerised assessment of word-processing skills is accurate and reliable, in comparison to the assessment by human markers. The main purpose was to determine the accuracy and reliability of the proposed OOXML algorithm.

7.3 Assessment results versus the benchmark results

Separate paired-sample t-tests were conducted to examine how the assessment results, produced by each of the assessment algorithms and the human markers, compare with the benchmark results. Each assessment method's results, including those produced by human

markers, were paired with the benchmark assessment results and subjected to a paired-sample t-test. To ensure reliable test results, a test for normality was first performed on each group of paired variables.

An indication of whether the paired-variable distributions conform to the normal distribution is already visible by examining the skewness of the distributions, displayed in Table 7.2. A skewness of zero indicates perfect symmetry and conformity to the normal distribution (Rice, 2007), while a positive or negative skewness indicates that the relevant distribution is either positively skewed to the left or negatively to the right.

Table 7.2 Descriptive analysis of the benchmark and assessment methods' paired variables

	Benchmark & Markers	**Benchmark & OOXML**	**Benchmark & Levenshtein**	**Benchmark & fComp**	**Benchmark & Word Grader**
N	200	200	200	200	200
Mean	13.4000	-1.4850	3.7350	8.0050	-9.3150
Median	15.0000	-1.0000	4.0000	7.5000	-5.5000
Std. Deviation	19.26332	7.26870	8.39387	7.63850	15.57171
Skewness	-1.116	-.759	-.014	-.343	-.401

7.3.1 Tests for normality

Tests for normality (see Table 7.3) were performed on the distributions of the differences between the paired variables of the benchmark and assessment methods. The Shapiro-Wilk test was selected as it is the most powerful test for normality (Razali & Wah, 2011). The following hypothesis was formulated:

- H_0 = The differences between the paired variables are normally distributed.

The Shapiro-Wilk test for normality, with regard to the distribution of differences between the Benchmark and Markers paired variables, indicates a significant deviation from the normal distribution (D = .854, p < .01), as indicated in Table 7.3. The accompanying Q-Q plot in Figure 7.2 also illustrates a lack of linearity between the observed values and that of a normal distribution. Fortunately, the histogram indicates that the distribution is approximately symmetrical around the sample median ($\tilde{x}$ = 15.0), which differs little from the sample mean ($\bar{x}$ = 13.4). Therefore, instead of the paired-sample t-test, it would be more appropriate to conduct a statistical analysis by means of the non-parametric Wilcoxon signed-rank test.

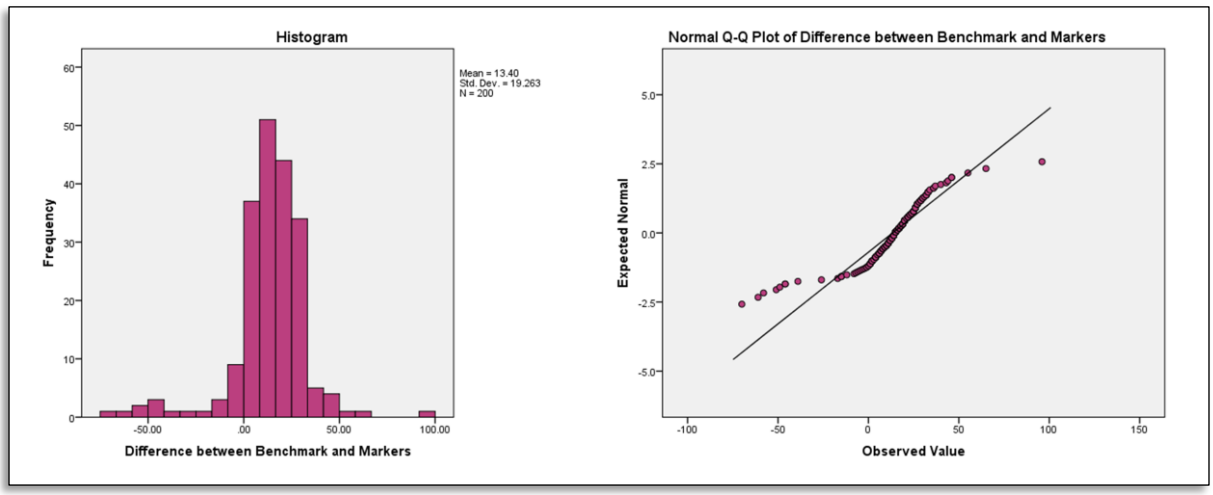


Figure 7.2 Differences between Benchmark and Markers paired variables

When assessing the normality of the differences between the Benchmark and OOXML paired variables, it is evident from the graphs in Figure 7.3 that the differences between the paired variables are approximately normally distributed. This is, however, not confirmed by the Shapiro-Wilk test for normality in Table 7.3, which rejects the null hypothesis (D = .948, p < .01). This might be due to the outliers depicted in the Q-Q plot. Nevertheless, due to the large sample size, the paired-sample t-test should be robust against non-normality (Rice, 2007).

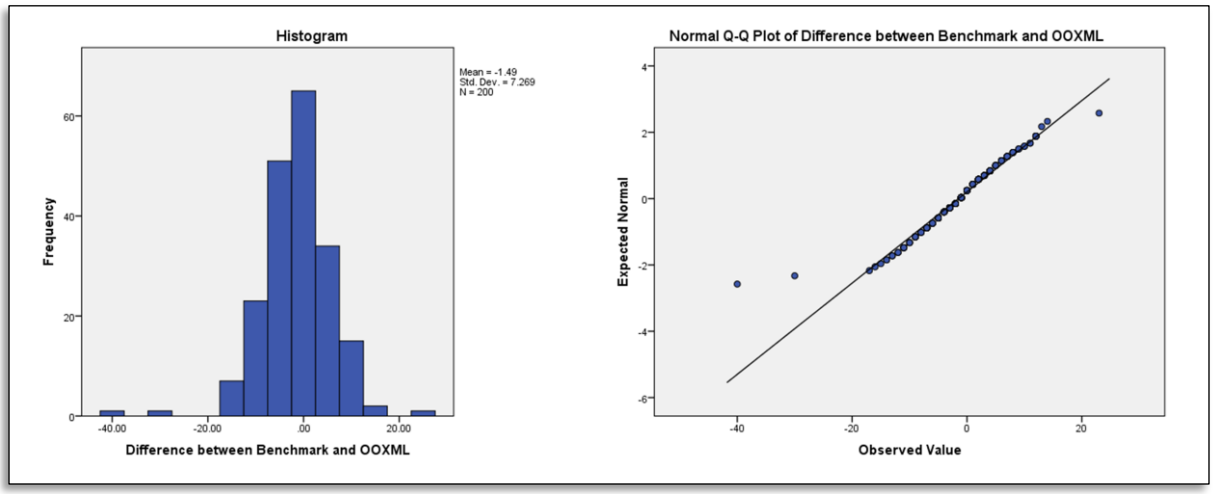


Figure 7.3 Differences between Benchmark and OOXML paired variables

The Shapiro-Wilk test for normality indicates that the distribution of differences between the Benchmark and Levenshtein paired variables do not deviate significantly from a normal distribution (D = .986, p > .01). Figure 7.4 confirms that the null hypothesis cannot be rejected, due to the normal distribution being resembled by the shape of the distribution's histogram and the linear relationship displayed in the Q-Q plot.

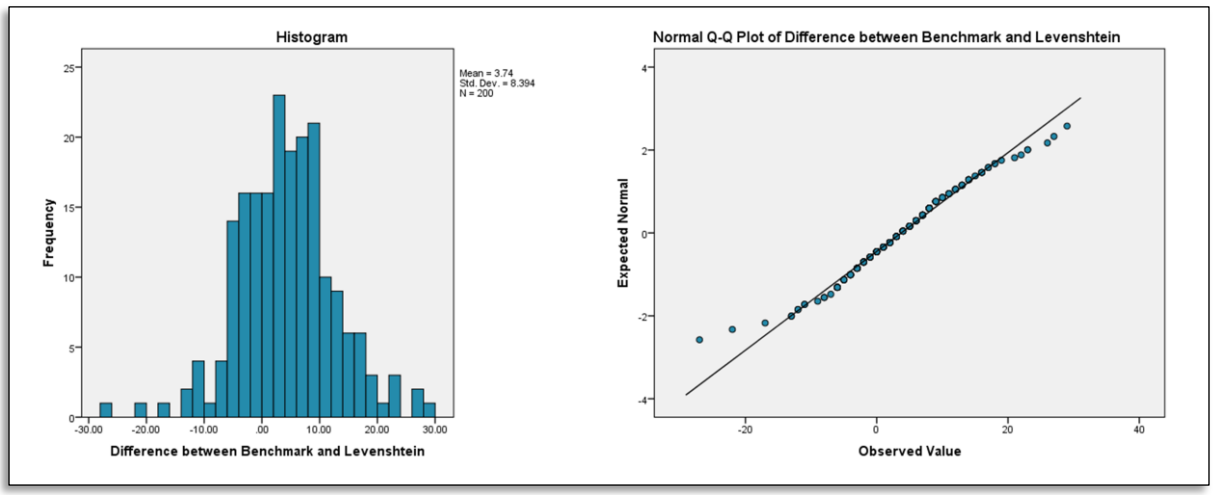


Figure 7.4 Differences between Benchmark and Levenshtein paired variables

The Shapiro-Wilk test for normality, with regard to the distribution of the differences between the Benchmark and fComp paired variables, produces a *p*-value that indicates a significant deviation from the normal distribution ($D = .964$, $p < .01$). However, when the histogram and Q-Q plot are examined (see Figure 7.5), they show an approximate agreement with the normal distribution. This, in combination with the large sample size, should provide the paired-sample t-test with the necessary robustness against non-normality (Rice, 2007).

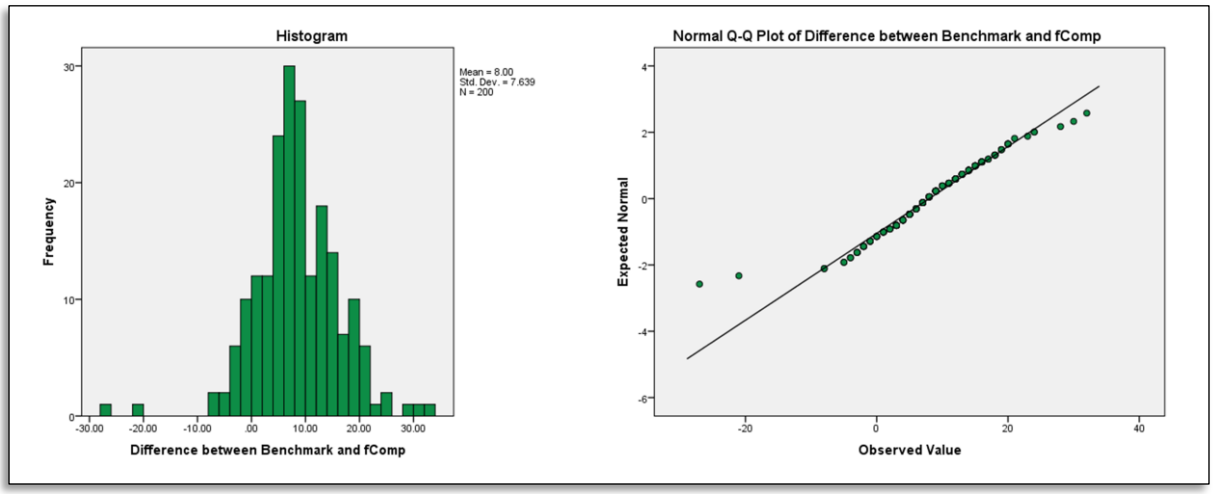


Figure 7.5 Differences between Benchmark and fComp paired variables

The differences between the paired variables, with regard to the Benchmark and Word Grader, produce a distribution that, when visually examined, is approximately normally distributed (see Figure 7.6). Conversely, the Shapiro-Wilk test for normality generates a *p*-value that causes the null hypothesis to be rejected ($D = .972$, $p < .01$). The distribution should, however, be robust against non-normality, due to the sample size (Rice, 2007).

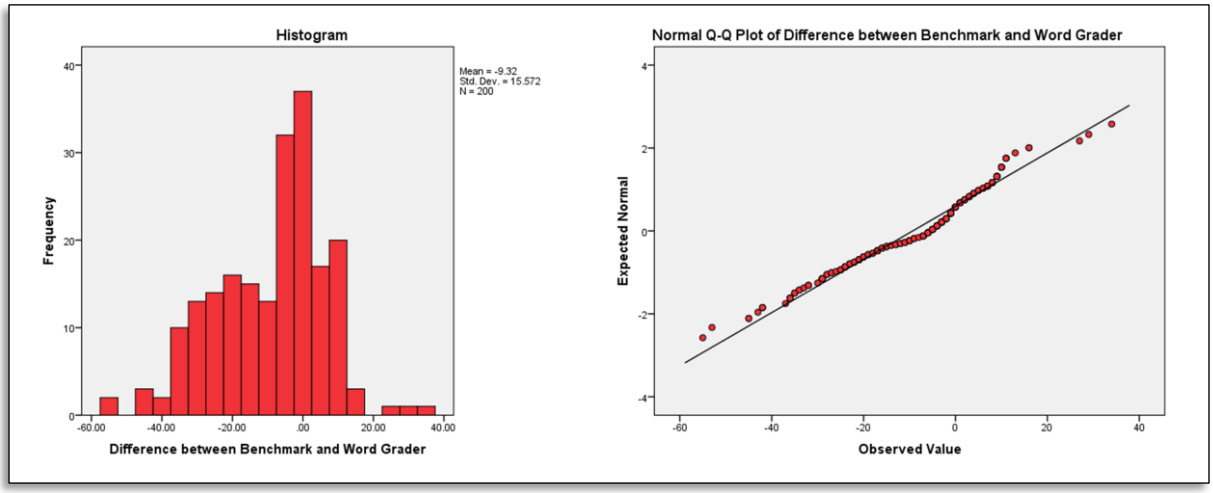


Figure 7.6 Differences between Benchmark and Word Grader paired variables

Table 7.3 contains the results of the tests for normality performed on the distributions of the differences between the benchmark and assessment methods' paired variables.

Table 7.3 Tests for normality of the differences between the paired variables

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
Paired variables	**Statistic**	**df**	**Sig.**	**Statistic**	**df**	**Sig.**
Benchmark & Markers	.143	200	.000	.854	200	.000
Benchmark & OOXML	.074	200	.009	.948	200	.000
Benchmark & Levenshtein	.066	200	.035	.986	200	.042
Benchmark & fComp	.083	200	.002	.964	200	.000
Benchmark & Word Grader	.119	200	.000	.972	200	.001

a. Lilliefors Significance Correction

7.3.2 Wilcoxon signed-rank test

Since the assumption of normality failed significantly with regard to the distribution of differences between the Benchmark and Markers paired variables, it would have been inappropriate to compare the benchmark results with the human markers' assessment results by means of a paired-sample t-test. Instead, a non-parametric Wilcoxon signed-rank test was conducted to compare the assessment results. To determine whether the benchmark's assessment results differ significantly from the human markers' assessment results, the following hypothesis was formulated:

- H_0 = There is no difference between the benchmark results and the human markers' assessment results.

According to the signed ranks (see Table 7.4), 22 of the assessment results assigned by the human markers were lower than the benchmark's assessment results, while 176 of the human markers' assessment results were higher than those of the benchmark. Only 2 of the samples' assessment results were equal.

Table 7.4 Signed ranks of the differences between the paired variables

	N	Mean Rank	Sum of Ranks
Negative Ranks *(Markers < Benchmark)*	22	100.45	2210.00
Positive Ranks *(Markers > Benchmark)*	176	99.38	17491.00
Ties *(Markers = Benchmark)*	2		
Total	200		

The Wilcoxon signed-rank test (see Table 7.5) shows that there is a significant difference in the assessment results provided by the benchmark and those allocated by the human markers ($Z = -9.465$, $p < .01$).

Table 7.5 Wilcoxon signed-rank test statistics

	Markers - Benchmark
Z	-9.465[a]
Asymp. Sig. (2-tailed)	.000

a. Based on negative ranks.

7.3.3 Paired-sample t-tests

Paired-sample t-tests were conducted to determine whether there is a significant difference between the benchmark results and the computerised assessment results produced by the assessment algorithms involved. The following hypothesis was formulated with regard to the paired-sample t-tests:

- H_0 = There is no difference between the benchmark results and the assessment algorithms' results.

Table 7.6 displays the results of the individual paired-sample t-tests that were conducted. The test results demonstrate a significant difference between the benchmark results and Word Grader's assessment results ($t(199) = -8.460$, $p < .01$). The direction of the *t*-value and the difference in the means ($\Delta = -9.315$) of the two sample groups lead to the conclusion that Word Grader's assessment results are noticeably lower than the benchmark results. Likewise,

the assessment results produced by fComp differ significantly from the benchmark results ($t(199) = 14.821$, $p < .01$).

Table 7.6 Paired-sample t-tests

	Paired Differences							
				95% Confidence Interval				
Paired samples	Mean Diff.	Std. Dev.	Std. Error Mean	Lower	Upper	t	df	Sig. (2-tailed)
Benchmark & OOXML	-1.485	7.269	.514	-2.499	-.471	-2.889	199	.004
Benchmark & Levenshtein	3.735	8.394	.594	2.565	4.905	6.293	199	.000
Benchmark & fComp	8.005	7.639	.540	6.940	9.070	14.821	199	.000
Benchmark & Word Grader	-9.315	15.572	1.101	-11.486	-7.144	-8.460	199	.000

It can also be concluded, due to the mean difference ($\Delta = 8.005$) of the two sample groups involved and the direction of the *t*-value, that fComp produces assessment results that are measurably higher than the benchmark results. A significant difference also exists between the benchmark and Levenshtein algorithm's assessment results ($t(199) = 6.293$, $p < .01$). This evidently concludes that the Levensthtein assessment results are higher than the benchmark results, due to the mean difference ($\Delta = 3.735$) and the sign of the *t*-value.

A comparison between the benchmark results and the assessment results produced by the OOXML algorithm, resulted in a *t*-value which indicates that the assessment results differ from each other ($t(199) = -2.889$, $p < .01$). However, the difference is not as significant as with the other document analysis algorithms. Therefore, taking into account the sign of the *t*-value and the difference between the means ($\Delta = -1.485$) of the sample groups involved, it can be concluded that the OOXML algorithm produces assessment results that are not noticeably lower than the benchmark results.

7.4 Human markers versus document analysis algorithms

The assessment results assigned by the human markers were also paired with each of the document analysis algorithms' assessment results. The distributions of the differences between the paired variables were analysed, producing the results displayed in Table 7.7.

Table 7.7 Descriptive analysis of the markers' and assessment algorithms' paired variables

	Markers & OOXML	Markers & Levenshtein	Markers & fComp	Markers & Word Grader
N	200	200	200	200
Mean	-14.8850	-9.6650	-5.3950	-22.7150
Median	-18.0000	-11.0000	-8.0000	-22.0000
Std. Deviation	19.18573	19.81022	19.32086	23.13113
Skewness	1.196	1.139	.945	.603

7.4.1 Tests for normality

The distributions of the differences between the paired variables of the human markers and the respective document analysis algorithms were tested for normality to determine whether they demonstrated any resemblance towards the normal distribution (see Table 7.8). The following hypothesis was formalised to accomplish this:

- H_0 = The differences between the paired variables are normally distributed.

The Shapiro-Wilk test for normality, with regard to the distribution of differences between the Markers and OOXML paired variables, indicates a significant deviation from the normal distribution (D = .868, p < .01). The Q-Q plot in Figure 7.7 supports this indication as it illustrates a lack of linearity between the observed values and that of the normal distribution. When the sample mean ($\bar{x}$ = -14.89) and median ($\tilde{x}$ = -18.0) of the distribution are examined, it is evident that the distribution is also not symmetrical around the median (see Table 7.7). The histogram of the distribution, displayed in Figure 7.7, also illustrates this statistic.

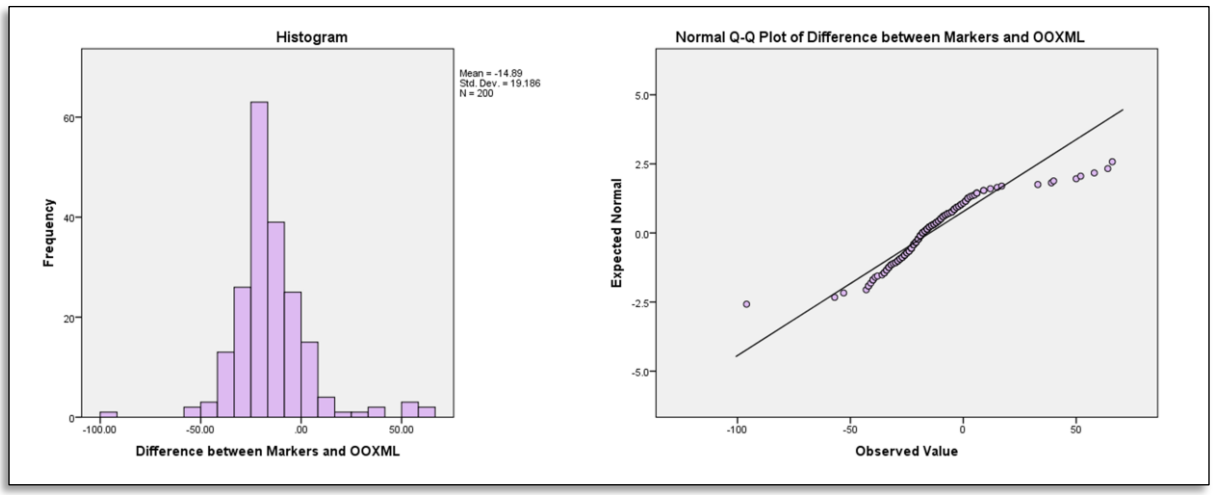


Figure 7.7 Differences between Markers and OOXML paired variables

The Shapiro-Wilk test for normality indicates that the distribution of differences between the Markers and the Levenshtein paired variables deviates significantly from a normal distribution (D = .864, p = < .01). By examining the Q-Q plot, displayed in Figure 7.8, it is evident that there also exists no linearity between this distribution and the normal distribution. Yet, the histogram shows that the distribution is approximately symmetrical, which is verified by the small difference between the distribution's mean ($\bar{x}$ = -9.67) and median ($\tilde{x}$ = -11.0), previously displayed in Table 7.7.

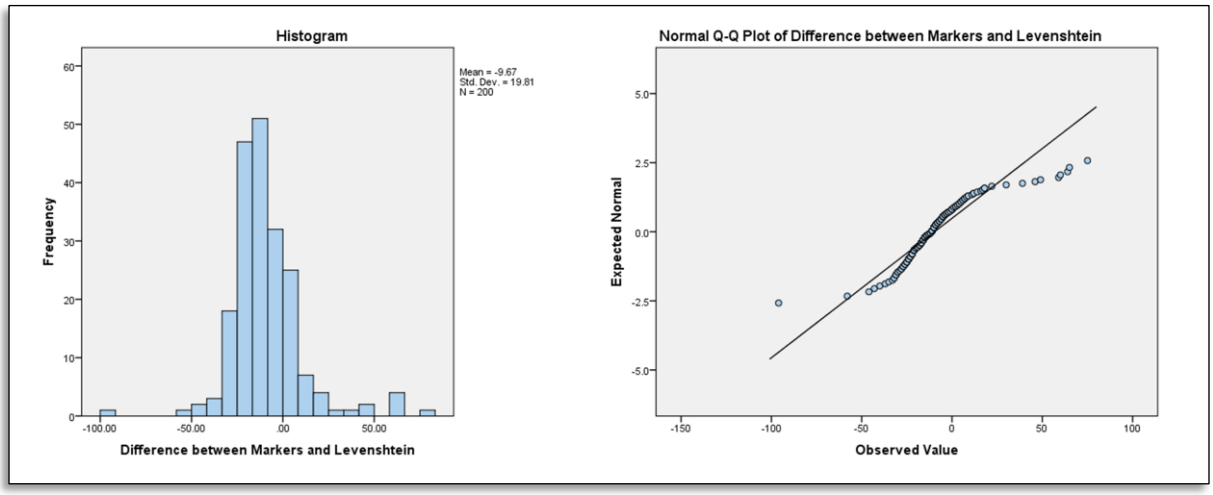


Figure 7.8 Differences between Markers and Levenshtein paired variables

The distribution of the differences between the paired variables of Markers and fComp is not symmetrical, as indicated by the histogram of the distribution in Figure 7.9. There also exists no linearity between this distribution and the normal distribution as depicted by the Q-Q plot.

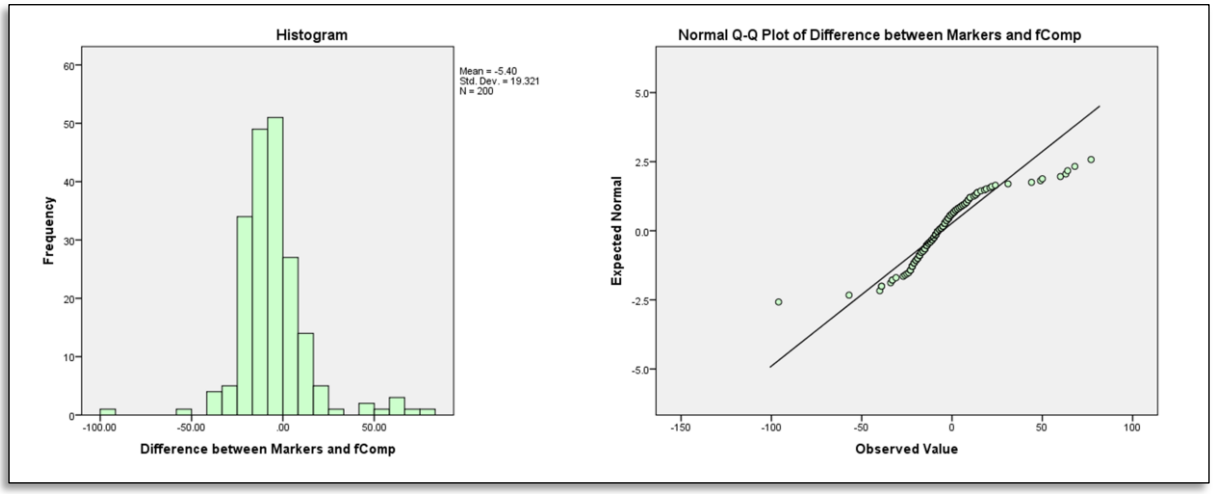


Figure 7.9 Differences between Markers and fComp paired variables

The particular distribution also demonstrates a significant deviation from the normal distribution (D = .857, p < .01), according to the Shapiro-Wilk test. The mean ($\bar{x}$ = -5.40) and median ($\tilde{x}$ = -8.0) of the distribution also significantly differ from one another (see Table 7.7).

The histogram of the paired variables (see Figure 7.10), with regard to the differences between the Markers and Word Grader, displays a distribution that conforms to the normal distribution. The Q-Q plot strengthens this notion by illustrating signs of linearity between this distribution and the normal distribution. Therefore, even though the Shapiro-Wilk test for normality rejects the null hypothesis (D = .961, p < .01), it would still be appropriate to conduct a paired-sample t-test between the human markers' assessment results and the assessment results produced by Word Grader.

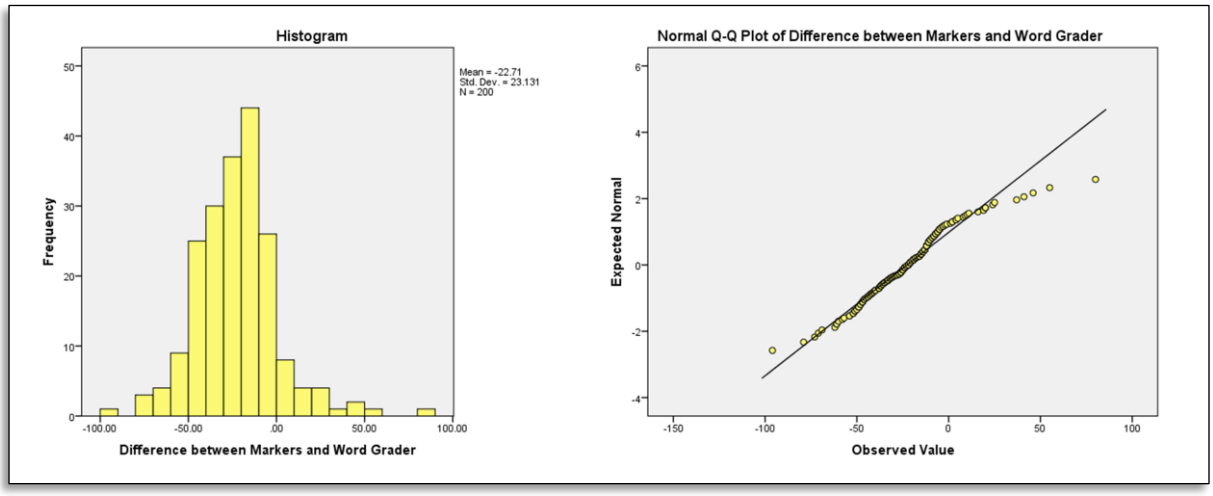


Figure 7.10 Differences between Markers and Word Grader paired variables

The results of the tests for normality that were performed on the distributions of the differences between the paired variables of the markers and assessment algorithms are contained in Table 7.8.

Table 7.8 Tests for normality of the differences between paired variables

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
Paired variables	**Statistic**	**df**	**Sig.**	**Statistic**	**df**	**Sig.**
Markers and OOXML	.115	200	.000	.868	200	.000
Markers and Levenshtein	.132	200	.000	.864	200	.000
Markers and fComp	.135	200	.000	.857	200	.000
Markers and Word Grader	.092	200	.000	.961	200	.000

a. Lilliefors Significance Correction

7.4.2 Paired-sample sign test

As discussed in Section 7.4.1, the distribution with regard to the differences between the Markers and OOXML paired variables, deviates from the normal distribution. The paired variables of the distribution with regard to the differences between the Markers and fComp also deviate from the normal distribution. The deviations from normality, combined with the lack of symmetry around the median and lack of linearity towards the normal distribution, necessitate the use of paired-sample sign tests for the comparative analysis of the paired variables within these two distributions. To determine whether the human markers' assessment results differ significantly from the assessment results produced by the OOXML and fComp algorithms, the following hypothesis was formulated:

- H_0 = There is no difference between the human markers' assessment results and the assessment algorithms' (OOXML and fComp) results.

The frequencies by which the assessment results produced by the OOXML and fComp assessment algorithms differ from the assessment results assigned by human markers are displayed in Table 7.9. The frequencies indicate that 172 of the assessment results assigned by the human markers were higher than the assessment results produced by the OOXML algorithm. Likewise, 145 of the human markers' assessment results were higher than the assessment results produced by fComp. Twenty-seven (27) of the human markers' assessment results were lower than the OOXML algorithm's assessment results, while 53 of the human markers' assessment results were lower than fComp's assessment results. Only 1 of the OOXML algorithm's results and 2 of fComp's assessment results were equal to the assessment results assigned by the human markers.

Table 7.9 Frequencies of the differences between the paired variables

OOXML - Markers	**N**	**fComp - Markers**	**N**
Negative Differences *(OOXML < Markers)*	172	Negative Differences *(fComp < Markers)*	145
Positive Differences *(OOXML > Markers)*	27	Positive Differences *(fComp > Markers)*	53
Ties *(OOXML = Markers)*	1	Ties *(fComp = Markers)*	2
Total	200	Total	200

Therefore, the sign test (see Table 7.10) shows that there is a significant difference between the assessment results produced by the OOXML assessment algorithm and those assigned by the human markers (Z = -10.208, p < .01). Correspondingly, there is also a significant

difference between the assessment results produced by fComp and the human markers' assessment results (Z = -6.467, p < .01).

Table 7.10 Sign test statistics

	OOXML - Markers	fComp - Markers
Z	-10.208	-6.467
Asymp. Sig. (2-tailed)	.000	.000

7.4.3 Wilcoxon signed-rank test

As explained in Section 7.4.1, the test for normality failed with regard to the distribution of differences between the Markers and the Levenshtein paired variables. The distribution also displayed no linearity towards the normal distribution; however, the distribution was symmetrical around the median. Therefore, a non-parametric Wilcoxon signed-rank test was conducted to compare the human markers' results with the assessment results produced by the Levenshtein algorithm. To determine whether the human markers' assessment results differ significantly from the Levenshtein algorithm's assessment results, the following hypothesis was formulated:

- H_0 = There is no difference between the human markers' assessment results and the Levenshtein algorithm's assessment results.

The signed ranks displayed in Table 7.11 indicate that 155 of the assessment results assigned by the human markers were higher than the assessment results produced by the Levenshtein assessment algorithm. Only 38 of the human markers' assessment results were lower than those produced by the Levenshtein algorithm and 7 of the paired variables' assessment results were equal.

Table 7.11 Signed ranks of the differences between the paired variables

	N	Mean Rank	Sum of Ranks
Negative Ranks *(Levenshtein < Markers)*	155	100.77	15620.00
Positive Ranks *(Levenshtein > Markers)*	38	81.61	3101.00
Ties *(Levenshtein = Markers)*	7		
Total	200		

The Wilcoxon signed-rank test (see Table 7.12) result was calculated on the positive ranks, since the sum of the positive ranks is lower than the sum of the negative ranks (Ott &

Longnecker, 2008). The test result indicates a significant difference between the assessment results provided by the Levenshtein algorithm and those assigned by the human markers (Z = -8.057, $p < .01$).

Table 7.12 Wilcoxon signed-rank test statistics

	Levenshtein - Markers
Z	-8.057[a]
Asymp. Sig. (2-tailed)	.000

a. Based on positive ranks.

7.4.4 Paired-sample t-test

A paired-sample t-test was conducted to determine whether the assessment results assigned by the human markers differ significantly from the results produced by Word Grader. The comparison by means of a paired-sample t-test is appropriate, since the distribution of the differences between the Markers and Word Grader paired variables is approximately normally distributed (see Section 7.4.1). The following hypothesis was formulated with regard to the paired-sample t-test:

- H_0 = There is no difference between the human markers' assessment results and the assessment results produced by Word Grader.

The results of the paired-sample t-test are displayed in Table 7.13. The test results demonstrate a significant difference between the human markers' and Word Grader's assessment results ($t(199)$ = -13.888, $p < .01$).

Table 7.13 Paired-sample t-test between the markers' and Word Grader's assessment results

	Paired Differences							
				95% Confidence Interval				
Paired samples	Mean Diff.	Std. Dev.	Std. Error Mean	Lower	Upper	t	df	Sig. (2-tailed)
Markers & Word Grader	-22.715	23.131	1.636	-25.940	-19.490	-13.888	199	.000

Therefore, it can be concluded from the direction of the t-value and the difference between the means of the two sample groups (Δ = -22.715) that the assessment results produced by Word Grader's document difference comparison algorithm is significantly lower than the markers'

assessment results. The differences between these assessment results were already visible from the dot plot of the assessment results' means (see Figure 7.1) provided in Section 7.2.

7.5 Assessing the performance of the proposed OOXML algorithm

The performance of the proposed OOXML algorithm (see Section 4.7) and the similarity metric that it implements were evaluated by comparing its assessment results with the assessment results produced by the other assessment algorithms involved. The assessment results produced by the OOXML algorithm were paired with each of the remaining algorithms' assessment results. The distributions of the differences between the paired variables were calculated. Table 7.14 contains the mean, median and skewness of each of the paired variable distributions, which already indicates a strong tendency of normality.

Table 7.14 Descriptive analysis of the OOXML and assessment algorithms' paired variables

	OOXML & Levenshtein	**OOXML & fComp**	**OOXML & Word Grader**
N	200	200	200
Mean	5.2200	9.4900	-7.8300
Median	5.0000	9.0000	-7.0000
Std. Deviation	6.57936	6.15870	15.23851
Skewness	.285	.260	-.432

7.5.1 Tests for normality

The distributions of the differences between the paired variables were tested for normality to determine whether there was any deviation from the normal distribution (see Table 7.15). The following hypothesis was formulated with regard to the tests for normality:

- H_0 = The differences between the paired variables are normally distributed.

As illustrated by the histogram in Figure 7.11, the distribution of the differences between the OOXML and Levenshtein paired variables is approximately normally distributed. This is confirmed by the Shapiro-Wilk test for normality (see Table 7.15), which does not reject the null hypothesis (D = .991, p > .01), and also by the distribution's skewness factor, presented in Table 7.14. The Q-Q plot of the distribution's observed values also indicates a linear relationship with the normal distribution. Therefore, a paired-sample t-test should be appropriate for a comparative analysis.

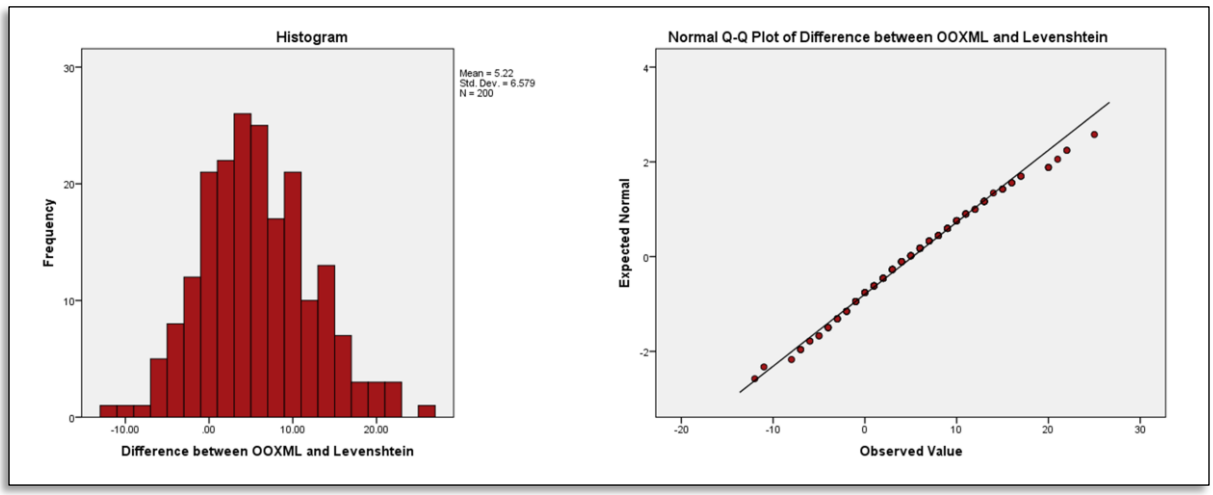


Figure 7.11 Differences between OOXML and Levenshtein paired variables

The null hypothesis is also not rejected by the Shapiro-Wilk test (see Table 7.15) for normality (D = .985, p > .01), with regard to the distribution of the differences between the OOXML and fComp paired variables. The strong linear relationship with the normal distribution, demonstrated by the Q-Q plot in Figure 7.12, confirms the result of the Shapiro-Wilk test for normality. A comparative analysis through a paired-sample t-test would therefore be appropriate.

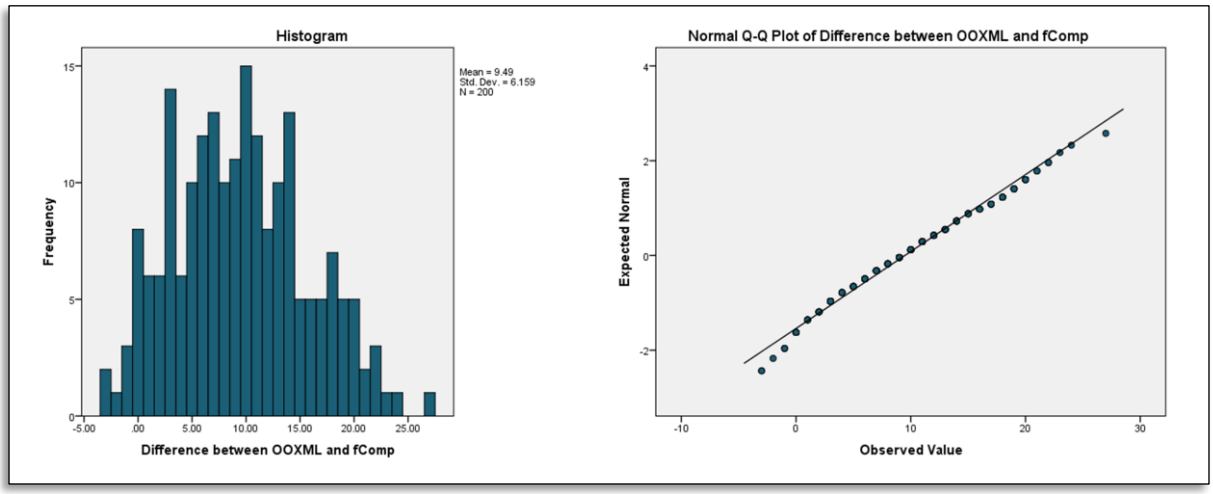


Figure 7.12 Differences between OOXML and fComp paired variables

The Shapiro-Wilk test for normality, with regard to the distribution of the differences between the paired variables of OOXML and Word Grader, produces a p-value (see Table 7.15) that demonstrates a significant deviation from the normal distribution (D = .981, p < .01) by only a small margin. However, when the histogram and Q-Q plot are examined (see Figure 7.13), they show an approximate agreement with the normal distribution. This, in combination with

the large sample size, should provide the paired-sample t-test with the necessary robustness against non-normality (Rice, 2007).

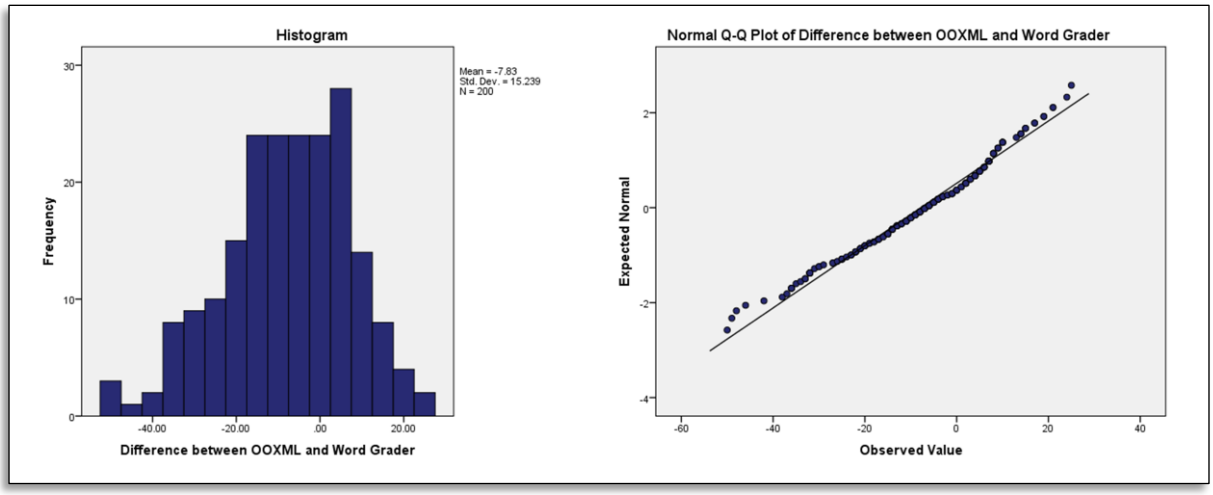


Figure 7.13 Differences between OOXML and Word Grader paired variables

The results of the tests for normality that were performed on the distributions of the differences between the paired variables of the OOXML algorithm and the other assessment algorithms are contained in Table 7.15.

Table 7.15 Tests for normality of the differences between paired variables

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
Paired variables	**Statistic**	**df**	**Sig.**	**Statistic**	**df**	**Sig.**
OOXML and Levenshtein	.063	200	.053	.991	200	.242
OOXML and fComp	.062	200	.059	.985	200	.035
OOXML and Word Grader	.071	200	.015	.981	200	.009

a. Lilliefors Significance Correction

7.5.2 Paired-sample t-tests

To determine how the assessment results produced by the OOXML algorithm compare with the assessment results produced by the Levenshtein, fComp and Word Grader assessment algorithms, separate paired-sample t-tests were conducted. To achieve the specified objective, the following hypothesis was formulated and tested:

- H_0 = There is no difference between the OOXML algorithm's assessment results and the assessment results produced by the Levenshtein, fComp and Word Grader assessment algorithms.

Table 7.16 displays the results of the individual paired-sample t-tests that were conducted. The paired-sample t-test that compares the assessment results produced by the OOXML algorithm with the assessment results of the Levenshtein algorithm demonstrates a significant difference between the assessment results ($t(199) = 11.22$, $p < .01$).

Similarly, the paired-sample t-test results demonstrate a significant difference between the assessment results produced by the OOXML algorithm and the assessment results produced by fComp ($t(199) = 21.792$, $p < .01$). The test results also demonstrate a significant difference between the assessment results produced by the OOXML algorithm and Word Grader's assessment results ($t(199) = -7.267$, $p < .01$).

Table 7.16 Paired-sample t-tests

	Paired Differences							
				95% Confidence Interval				
Paired samples	Mean Diff.	Std. Dev.	Std. Error Mean	Lower	Upper	t	df	Sig. (2-tailed)
OOXML & Levenshtein	5.220	6.579	.465	4.303	6.137	11.220	199	.000
OOXML & fComp	9.490	6.159	.435	8.631	10.349	21.792	199	.000
OOXML & Word Grader	-7.830	15.239	1.078	-9.955	-5.705	-7.267	199	.000

By examining the direction of the *t*-values of the paired-sample t-tests and the mean difference of the paired samples' distributions, certain conclusions can be drawn. With regard to the OOXML and Levenshtein paired variables, the direction of the *t*-value and the difference between the means ($\Delta = 5.22$) of the sample groups involved, demonstrate that the assessment results produced by the OOXML algorithm are generally lower than the Levenshtein assessment results. The direction of the *t*-value, with regard to the OOXML and fComp paired variables, as well as the mean difference ($\Delta = 9.49$) of the relevant sample groups indicate that the OOXML algorithm's assessment results are on average lower than fComp's assessment results.

In contrast, the direction of the *t*-value, with regard to the OOXML and Word Grader paired variables, in addition to the mean difference ($\Delta = -7.83$) of the relevant sample groups, indicates that the assessment results produced by the OOXML algorithm are higher than Word Grader's assessment results.

7.6 Outcome with regard to accuracy and reliability

As stated in Chapter 1, accuracy is determined by the degree to which a measured result conforms to a standard or true value (Menditto et al., 2006), whereas reliability provides an indication of the stability and consistency of assessment results (Carmines & Zeller, 1979). The comparative analyses in Sections 7.3, 7.4 and 7.5 demonstrated statistically significant differences between the assessment methods' results. The parametric comparison tests also demonstrated which assessment results agreed most with the benchmark, human markers and OOXML assessment results. Table 7.17 contains a summary of the mean differences between the distributions of the assessment results, including the distributions that were involved in non-parametric comparison tests since they did not make use of the distribution means. The mean differences between the distributions are interpreted and discussed in the following sections.

Table 7.17 Mean differences of paired-sample comparisons

	Benchmark	Markers	OOXML
Markers	13.400		
OOXML	-1.485	-14.885	
Levenshtein	3.735	-9.665	5.220
fComp	8.005	-5.395	9.490
Word Grader	-9.315	-22.715	-7.830

The proposed OOXML algorithm demonstrates a lower level of agreement with the other assessment algorithms (Levenshtein, fComp and Word Grader) than with the benchmark. The mean differences between the markers' and assessment algorithms' results indicate that the assessment algorithms have a higher level of agreement with the benchmark than with the markers. Also, the analysis of the mean differences between the benchmark and the relevant assessment methods' results suggests that the human markers provided less accurate assessment results than the assessment algorithms. The accuracy by which the human markers graded the word-processing assignments is therefore questionable, as also stated by Dowsing et al. (1998). It can thus be concluded that the proposed OOXML algorithm, produced more accurate results than the human markers and the other document analysis algorithms.

To determine the reliability of the similarity metrics applied by the OOXML algorithm, the Levenshtein algorithm, fComp and Word Grader's document difference comparison

algorithm, a reliability analysis was conducted. To achieve this, the selected sample of 200 word-processing assignments was re-assessed by WAM and Word Grader (see Section 6.6.2). The re-assessment results were compared with the original assessment results to determine whether the re-assessment yielded the same outcome. A reliability analysis was also conducted with regard to the human markers. Since the benchmark results are the re-assessed results of human markers' results, the human markers assessment results were compared with the benchmark results to determine the level of inconsistency with regard to the allocation of marks among multiple markers compared to one marker (the instructor).

All four document analysis algorithms involved in this research project produced exactly the same assessment results as during their original assessment, as would be expected of computerised assessment algorithms. The reliability analysis of the human markers yielded an alpha coefficient (Cronbach's Alpha) of .781. This suggests that the allocation of marks was not very consistent with regard to the original and re-assessed results. From the reliability analysis, it is clear that the assessment results produced by the document analysis algorithms were more consistent than those delivered by the human markers. Therefore, it can be concluded that the computerised document analysis algorithms and the similarity metrics they apply produced more reliable results than the human markers.

7.7 Correlations

To determine whether there existed any similarity with regard to the allocation of marks, all the assessment methods' results, including the human markers' results, were correlated with the benchmark results. The document analysis algorithms' assessment results were also correlated with the human markers' assessment results. Similarly, the results produced by the proposed OOXML algorithm were correlated with the remaining algorithms' assessment results.

The skewness of the distributions (see Table 7.1) indicates that tests for normality would have to be conducted to determine whether the distributions of the assessment results are approximately normally distributed. This is necessary, since Pearson's correlation test assumes that the distributions involved are approximately normally distributed. If not, Spearman's correlation test would be more appropriate in calculating correlation coefficients (Rubin, 2009).

7.7.1 Testing for normality and transforming the data

Tests for normality were conducted (see Table 7.18) on the distributions of all the assessment results, i.e. the distributions of the benchmark results, the human markers' assessment results and all the document analysis algorithms' assessment results. The following hypothesis was formulated with regard to the tests for normality:

- H_0 = There is no difference between the distributions of the assessment results and the normal distribution.

Table 7.18 Tests for normality of the assessment results' distributions

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
Distributions	**Statistic**	**df**	**Sig.**	**Statistic**	**df**	**Sig.**
Benchmark	.115	200	.000	.930	200	.000
Markers	.119	200	.000	.873	200	.000
OOXML	.150	200	.000	.905	200	.000
Levenshtein	.127	200	.000	.905	200	.000
fComp	.122	200	.000	.906	200	.000
Word Grader	.062	200	.059	.967	200	.000

a. Lilliefors Significance Correction

As the results demonstrate, all of the distributions failed the Shapiro-Wilk test for normality since they deviate significantly from the normal distribution with $p < .01$. This is also confirmed by the distributions' histogram plots, displayed in Figures 7.14 to 7.19. The skewness factors (see Table 7.1) of the distributions indicate that they are skewed to the right, except for Word Grader's assessment results. Therefore, as mentioned in Section 7.2, the skewed distributions' data need to be transformed by means of a logarithmic transformation. Thereafter, the transformed values were subjected to tests for normality, which yielded the results displayed in Table 7.19. The following hypothesis was formulated with regard to the tests for normality:

- H_0 = There is no difference between the transformed distributions of the assessment results and the normal distribution.

According to the Shapiro-Wilk test for normality, the original distribution of the benchmark results, indicated in Table 7.18, deviated significantly from the normal distribution ($D = .930$, $p < .01$), which is also demonstrated by the skewed histogram in Figure 7.14. However, the

transformed distribution of the benchmark results, displayed in Table 7.19, is not significantly different from the normal distribution, as indicated by the Shapiro-Wilk test for normality (D = .991, p > .01). Therefore, the null hypothesis is not rejected with regard to the transformed distribution. The Q-Q plot also demonstrates strong linearity between the transformed distribution and the normal distribution.

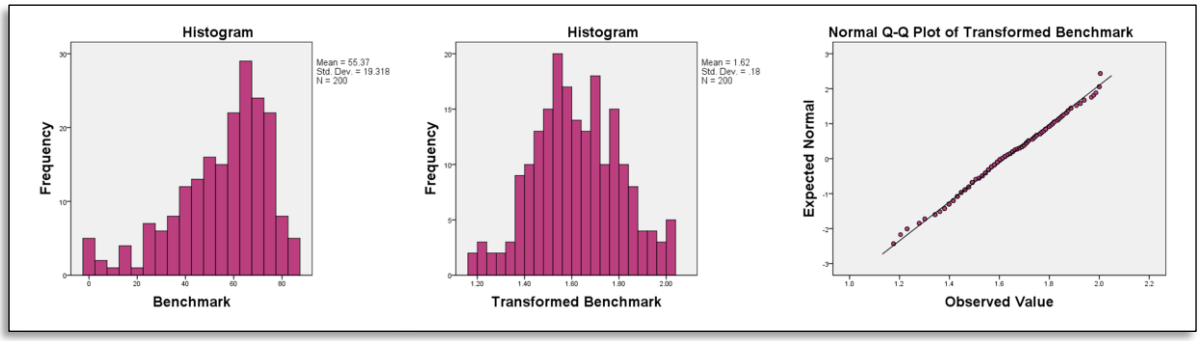


Figure 7.14 Original and transformed distributions of the benchmark results

The Shapiro-Wilk test for normality rejected the null hypothesis with regard to the original distribution (D = .873, p < .01), as well as the transformed distribution of the human markers' assessment results (D = .963, p < .01). This is confirmed by the histograms in Figure 7.15. The Q-Q plot also displays a lack of linearity between the transformed distribution and the normal distribution. Therefore, it would not be appropriate to use the Pearson's correlation test with regard to either of the distributions of the human markers' assessment results. Instead, the non-parametric Spearman's correlation test would be more appropriate in providing a valid correlation coefficient.

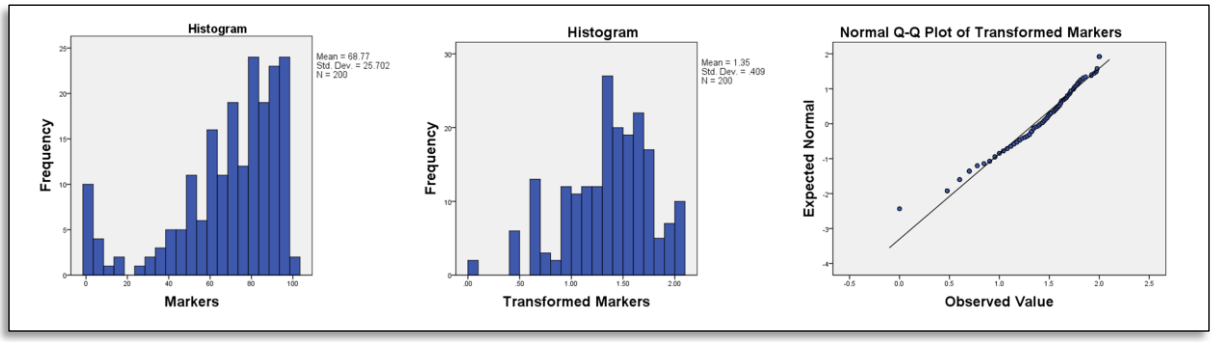


Figure 7.15 Original and transformed distributions of the markers' assessment results

With regard to the original distribution of the of the OOXML assessment results, the null hypothesis was rejected by the Shapiro-Wilk test for normality (D = .905, p < .01). It is also demonstrated by the original distribution's histogram (see Figure 7.16), which is skewed to

the right. The transformed distribution, however, demonstrates an approximate resemblance towards the normal distribution, displayed by the transformed distribution's histogram in Figure 7.16. The Q-Q plot also indicates a linear relationship with regard to the normal distribution. Thus, even though the Shapiro-Wilk test for normality rejects the null hypothesis with regard to the transformed OOXML distribution (D = .980, p < .01), the Pearson's correlation test would still provide a valid correlation coefficient.

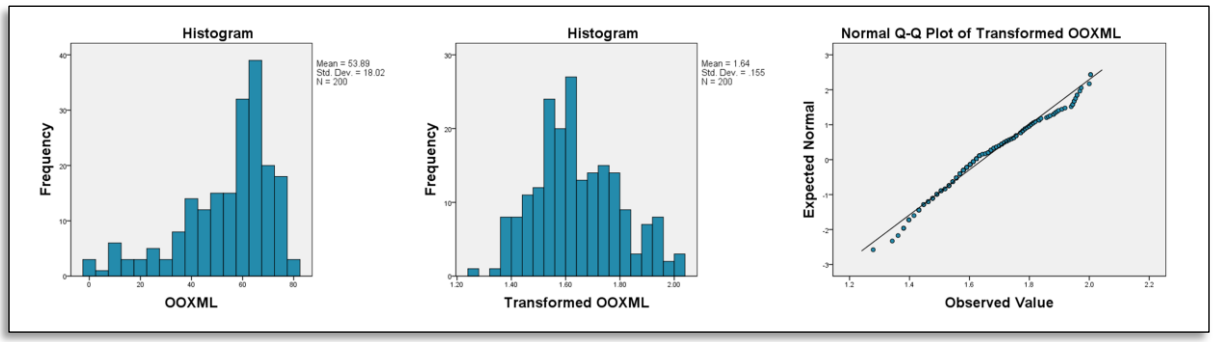


Figure 7.16 Original and transformed distributions of the OOXML assessment results

It is evident from the histogram of the Levenshtein assessment results' original distribution that the distribution was skewed to the right and could therefore not conform to the normal distribution. This is indicated by the Shapiro-Wilk test for normality as well, which rejected the null hypothesis (D = .905, p < .01). The logarithmic transformation of the data did also not produce a distribution that agreed with the normal distribution, as illustrated by the histogram of the transformed Levenshtein assessment results in Figure 7.17. This is confirmed by the Shapiro-Wilk test for normality, which also rejects the null hypothesis with regard to the transformed distribution (D = .975, p < .01).

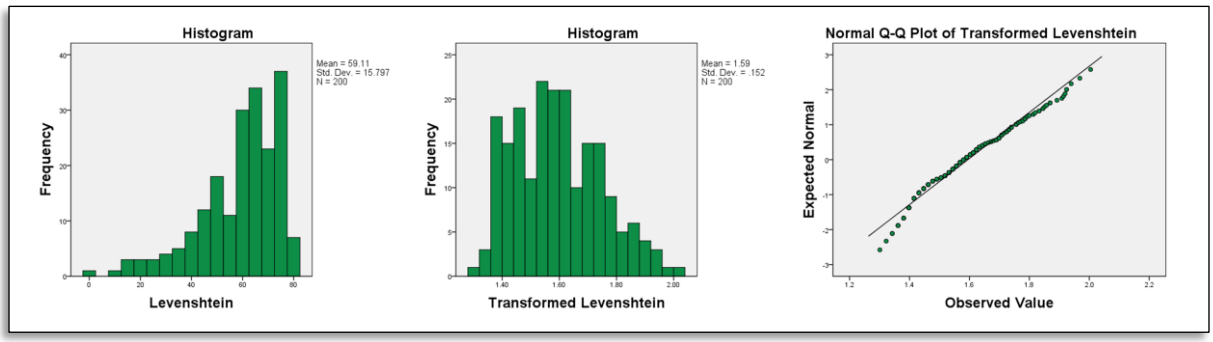


Figure 7.17 Original and transformed distributions of the Levenshtein assessment results

The original distribution of fComp's assessment results was also skewed to the right (see Figure 7.18). Thus, it deviated significantly from the normal distribution, as the Shapiro-Wilk

test for normality indicates in Table 7.18 (D = .906, p < .01). The logarithmic transformation, however, produced a distribution that is approximately normally distributed. This is demonstrated by the histogram of the transformed assessment results, even though the Shapiro-Wilk test for normality rejects the null hypothesis (D = .980, p < .01). The linear relationship, depicted in the Q-Q plot, between the transformed data and the normal distribution also supports the use of Pearson's correlation test, with regard to the transformed fComp distribution.

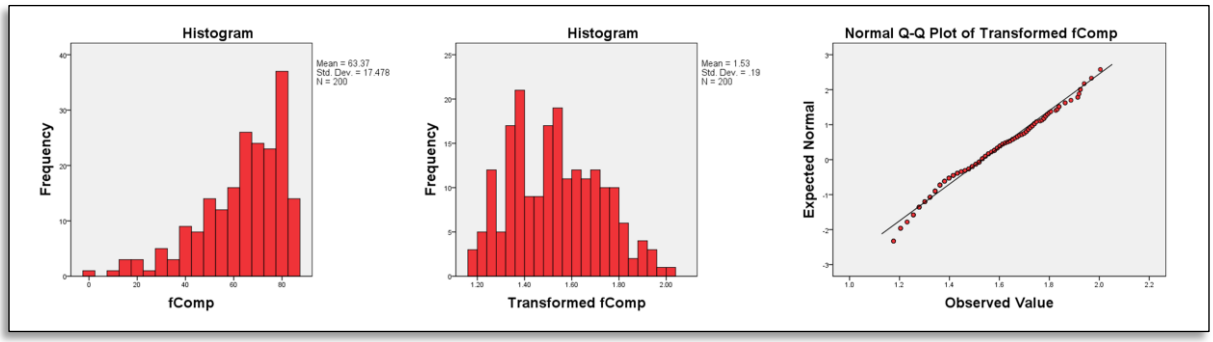


Figure 7.18 Original and transformed distributions of the fComp assessment results

Word Grader's original assessment results were not normally distributed. This was demonstrated by the result of the Shapiro-Wilk test for normality (D = .967, p < .01), as well as the histogram plot of the original data (see Figure 7.19). The transformation of the data was also unsuccessful, since the original distribution was not skewed. The logarithmic transformation is only applicable to skewed distributions. Therefore, the test for normality with regard to the transformed distribution also rejects the null hypothesis (D = .951, p < .01). In fact, the Shapiro-Wilk test statistics (see Tables 7.18 and 7.19) indicate that the original Word Grader assessment results distribution deviated less from the normal distribution than the transformed distribution.

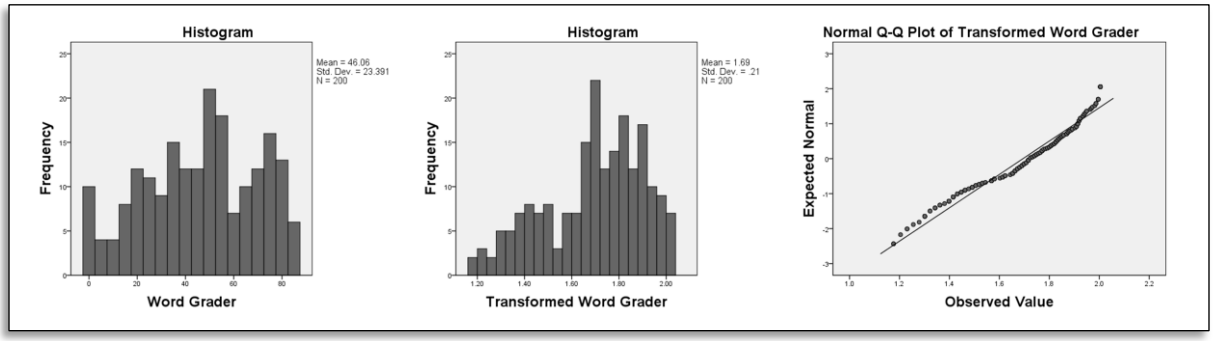


Figure 7.19 Original and transformed distributions of Word Grader's assessment results

Table 7.19 contains the results of the tests for normality that were performed on the logarithmic transformations of the assessment results' distributions.

Table 7.19 Tests for normality of the assessment results' transformed distributions

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
Transformed distributions	**Statistic**	**df**	**Sig.**	**Statistic**	**df**	**Sig.**
Benchmark	.049	200	.200*	.991	200	.271
Markers	.086	200	.001	.963	200	.000
OOXML	.089	200	.001	.980	200	.006
Levenshtein	.064	200	.047	.975	200	.001
fComp	.076	200	.007	.980	200	.006
Word Grader	.097	200	.000	.951	200	.000

a. Lilliefors Significance Correction *. This is a lower bound of the true significance.

7.7.2 Correlation between the benchmark and assessment methods' results

The correlation between the benchmark results and the results assigned by the human markers, as well as the correlation between the benchmark results and the document analysis algorithms' assessment results, were examined. The following methods were used to calculate the correlation coefficients of the distributions. Pearson's correlation coefficient was used with regard to the transformed distributions that were approximately normally distributed (see Table 7.19). The non-parametric Spearman's correlation coefficient was used with regard to the original distributions that deviated from the normal distribution (see Table 7.18) and could not be transformed to conform to the normal distribution. The following hypothesis was formulated to determine whether a positive relationship exists between the benchmark results and the assessment results produced by the human markers and document analysis algorithms:

- H_0 = There is no correlation between the benchmark results and the assessment results, produced by the human markers and document analysis algorithms.

Table 7.20 indicates that all the assessment methods' results have a significant positive correlation with the benchmark results. Therefore, the null hypothesis is rejected by all the assessment methods. According to Rice (2007), the correlation coefficient is an indication of the strength of the linear relationship between two variables. Therefore, a deeper analysis into the correlation coefficients was conducted to determine the strength of the linear relationships between the benchmark results and the relevant assessment methods.

Table 7.20 Correlation between benchmark and assessment methods' results

		Pearson's[a]		Spearman's[b]	
Distributions	**N**	**Correlation (r)**	**Sig.**	**Correlation (r_S)**	**Sig.**
Markers	200			.714	.000
OOXML	200	.910	.000	.903	.000
Levenshtein	200			.870	.000
fComp	200	.888	.000	.892	.000
Word Grader	200			.759	.000

a. Calculated with regard to transformed distributions.

b. Calculated with regard to original distributions.

Even though all the document analysis algorithms rejected the null hypothesis with regard to the paired-sample t-tests (see Section 7.3.3), they all demonstrate a significant positive correlation with the benchmark results. The human markers' assessment results also show a significant correlation with the benchmark results, despite the null hypothesis of the Wilcoxon signed-rank test being rejected (see Section 7.3.2). As portrayed in Table 7.20, the human markers' assessment results display the weakest correlation with regard to the benchmark results (r_s = .714, p < .01), despite the correlation being significant. Likewise, the correlation between Word Grader's assessment results and the benchmark results is significant but still weak (r_s = .75, p < .01), when compared to the other document analysis algorithms. Even though the assessment results produced by fComp are significantly different from the benchmark results, as concluded by the paired-sample t-test (see Section 7.3.3), they still display a strong correlation with regard to the benchmark results (r = .888, p < .01). The correlation between the Levenshtein algorithm's assessment results and the benchmark results is also significantly strong (r_s = .870, p < .01). However, the assessment results produced by the OOXML algorithm demonstrate the strongest positive correlation with the benchmark results (r = .910, p < .01).

7.7.3 Correlation between human markers and document analysis algorithms

The correlation between the human markers' assessment results and the results produced by the document analysis algorithms was examined. The following hypothesis was formulated to determine whether a positive relationship exists between the human markers' results and the assessment results produced by the document analysis algorithms:

- H_0 = There is no correlation between the human markers' assessment results and the assessment results produced by the document analysis algorithms.

The assessment results produced by the assessment algorithms differ significantly from the human markers' assessment results (see Section 7.4). However, as indicated in Table 7.21, there is a significant positive correlation with regard to the assessment results assigned by the markers and those produced by the assessment algorithms. fComp's assessment results and the results assigned by the markers have the strongest positive correlation (r_s = .723, p < .01), whereas, despite being significant, the weakest correlation exists between Word Grader's results and the results assigned by the human markers (r_s = .608, p < .01).

Table 7.21 Correlation between markers' and assessment algorithms' results

		Spearman's[a]	
Distributions	**N**	**Correlation (r_s)**	**Sig.**
OOXML	200	.690	.000
Levenshtein	200	.692	.000
fComp	200	.723	.000
Word Grader	200	.608	.000

a. Calculated with regard to original distributions.

The results produced by the OOXML and Levenshtein assessment algorithms correlate significantly with the human markers' assessment results, with almost identical correlation coefficients (OOXML: r_s = .690, p < .01; Levenshtein: r_s = .692, p < .01). The document analysis algorithms' assessment results do, however, have a stronger correlation with the benchmark results than with the human markers' assessment results.

7.7.4 Correlation between the OOXML and other document analysis algorithms

Correlation coefficients were calculated to determine the existence of a relationship between the similarity metrics, implemented by the proposed OOXML algorithm, and the other similarity metrics involved in this research project. With regard to correlating the distributions of the assessment results, produced by the assessment algorithms, the following hypothesis was formulated:

- H_0 = There is no correlation between the assessment results produced by the document analysis algorithms.

Although some correlations are stronger than others there is a significant positive correlation between the OOXML assessment results and the results produced by the Levenshtein algorithm, fComp and Word Grader. Word Grader's assessment results have the weakest

correlation with the assessment results produced by the OOXML algorithm (r_s = .785, p < .01), compared to the remaining correlation coefficients (see Table 7.22).

The results produced by fComp and the Levenshtein's algorithm correlate significantly with the OOXML assessment results. Their correlation coefficients are almost identical (fComp: r_s = .897, p < .01; Levenshtein: r_s = .894, p < .01), with fComp having a slightly stronger correlation. The Levenshtein, fComp and Word Grader assessment results also have a stronger correlation with the OOXML assessment results than with the human markers' results.

Table 7.22 Correlation between OOXML and assessment algorithms' results

		Pearson's[a]		Spearman's[b]	
Distributions	**N**	**Correlation (r)**	**Sig.**	**Correlation (r_s)**	**Sig.**
Levenshtein	200			.894	.000
fComp	200	.906	.000	.897	.000
Word Grader	200			.785	.000

a. Calculated with regard to transformed distributions

b. Calculated with regard to original distributions.

7.8 Chapter summary

The word-processing assessment results produced by human markers and four document analysis algorithms were statistically analysed to determine how they compare with one another. Comparative analysis, such as the paired-sample sign test, the paired-sample t-test and Wilcoxon ranked-sign test were conducted to test various hypotheses. Shapiro-Wilk tests for normality were performed on the relevant distributions in order to determine the most appropriate test statistics to apply for a particular statistical analysis.

It was concluded that the selected assessment methods differed from one another according to the assessment results they delivered. This was determined by the rejection of the hypotheses (see Sections 7.3.2 and 7.3.3) with regard to a comparison between the benchmark and other analysis methods, illustrated in Table 7.23.

Table 7.23 Hypotheses of the comparison between the benchmark and other analysis methods

Null Hypotheses (H_0)	Result	Reference
$H_{0,1}$: There is no difference between the benchmark results and the human markers' assessment results.	Rejected	Table 7.5
$H_{0,2}$: There is no difference between the benchmark results and the assessment algorithms' results.	Rejected	Table 7.6

Table 7.24 contains the hypotheses test results regarding a comparison between the human markers and assessment algorithms, conducted in Sections 7.4.2, 7.4.3 and 7.4.4. As indicated, all the hypotheses were rejected, which yields the conclusion that there is a difference between the assessments of word-processing assignments when conducted by document analysis algorithms versus multiple human markers.

Table 7.24 Hypotheses of the comparison between human markers and assessment algorithms

Null Hypotheses (H_0)	Result	Reference
$H_{0,1}$: There is no difference between the human markers' assessment results and the assessment algorithms' (OOXML and fComp) results.	Rejected	Table 7.10
$H_{0,2}$: There is no difference between the human markers' assessment results and the Levenshtein algorithm's assessment results.	Rejected	Table 7.12
$H_{0,3}$: There is no difference between the human markers' assessment results and the assessment results produced by Word Grader.	Rejected	Table 7.13

Furthermore, it was also concluded that the assessment of word-processing assignments by the proposed OOXML algorithm produced different results than when assessed by the Levenshtein algorithm, fComp and Word Grader (see Section 7.5.2). This is illustrated by the hypothesis test result in Table 7.25.

Table 7.25 Hypothesis of the comparison between the OOXML algorithm and other assessment algorithms

Null Hypothesis (H_0)	Result	Reference
$H_{0,1}$: There is no difference between the OOXML algorithm's assessment results and the assessment results produced by the Levenshtein, fComp and Word Grader assessment algorithms.	Rejected	Table 7.16

It was also determined that, even though there was a significant difference between the assessment methods' results, the assessment results displayed a strong correlation (see Sections 7.7.2, 7.7.3 and 7.7.4) with one another as demonstrated by the hypotheses tests in Table 7.26.

Table 7.26 Hypotheses of the correlation between the assessment methods

Null Hypotheses (H_0)	Result	Reference
$H_{0,1}$: There is no correlation between the benchmark results and the assessment results, produced by the human markers and document analysis algorithms.	Rejected	Table 7.20
$H_{0,2}$: There is no correlation between the human markers' assessment results and the assessment results produced by document analysis algorithms.	Rejected	Table 7.21
$H_{0,3}$: There is no correlation between the assessment results produced by the document analysis algorithms.	Rejected	Table 7.22

An accuracy analysis suggested that the OOXML assessment results deviated the least from the benchmark results. It was thus concluded that the proposed OOXML algorithm delivered more accurate assessment results than the other assessment methods involved in this research project. It was also determined that the computerised document analysis algorithms and the similarity metrics they implement, provided consistent assessment results, unlike the less consistent results delivered by human markers.

The final chapter presents an overview of the results obtained in the quasi-experimental study. The results are interpreted and conclusions are made with regard to the reliability and accuracy of computerised assessment in comparison with manual assessment.

Chapter 8

Conclusion

This chapter focuses on the following aspects:

- **Re-evaluating the research purpose and objective**
- **Discussing and interpreting the research analysis and findings**
- **Future research and development**

8.1 Introduction

This research project was conducted to develop a computerised assessment algorithm with the ability to assess word-processing skills that would improve on the assessment paradigms of current OOXML-based document analysis algorithms. The emerging algorithm, dubbed the OOXML algorithm (see Section 4.7), applies an object comparison assessment technique derived from research conducted by Pellet and Chevalier (2014), Lánskỳ et al. (2013) and Wolters (2010). It combines current effective OOXML-based methods of comparing document features between the student's document and the memorandum. It also improves thereon by determining which document paragraphs correspond with each other before commencing with the comparison. An investigation into the accuracy[1] and reliability[2] of the OOXML algorithm involved a quasi-experimental study that compared the assessment results of word-processing assignments, produced by the proposed OOXML algorithm, the Levenshtein algorithm, a file comparison algorithm, Word Grader and multiple human markers. This chapter re-examines and interprets the results that were obtained in the quasi-experimental study. It re-evaluates the research purpose and objective of this project, specified in Chapter 1, and determines whether the objective has been achieved.

8.2 Current OOXML-based algorithms

An investigation into document analysis algorithms revealed certain gaps within current OOXML-based algorithms. Algorithms of this sort emerged as a result of the recently

[1] Accuracy refers to the degree to which a measured result conforms to a standard or true value (Menditto et al., 2006)

[2] Reliability is determined by evaluating whether the assessment results are stable and consistent (Carmines & Zeller, 1979)

introduced OOXML document format standard. A procedural assessment approach, where word-processing tasks are assessed by means of programmed procedures (Lánskỳ et al., 2013), proved to be ineffective due to new procedures having to be created for each new word-processing task. The document object comparison technique, implemented by the Office Skills Assessment Project (OSAP), is reasonably effective and delivers acceptable results, according to Wolters (2010). It does, however, have a problem with regard to the comparison of documents that contain different numbers of paragraphs. The OSAP algorithm generates an optimal one-to-one paragraph mapping of the documents by comparing and marking each paragraph of the student's document with that of the memorandum. This produces a great deal of unnecessary paragraph object comparisons.

8.3 The proposed OOXML algorithm

This research project delivered a document analysis algorithm that addresses the limitations of the current OOXML-based algorithms. The proposed OOXML algorithm has the ability to assess word-processing skills by means of object comparison, i.e. a similarity metric that compares the objects within a word-processing document with that of the memorandum. First, it parses the documents that are to be compared, to ensure that the student's document and the memorandum conform to a homogeneous structure, by removing all the irrelevant document objects such as revision information, bookmarks and proofing errors.

It differs from the OSAP algorithm (Wolters, 2010) in that it identifies which paragraphs in the student's document correspond with which paragraphs in the examiner's solution before applying its object comparison metric. Corresponding paragraphs are identified by means of the smallest calculated Levenshtein distance (Levenshtein, 1966) between the paragraphs of the two relevant documents. This method helps to avoid unnecessary paragraph object comparisons, making the OOXML algorithm more efficient[3] than the OSAP algorithm.

Thereafter, it assesses the essential document parts (main document part, core properties and extended properties) of the student's submitted assignment as described in Chapter 4. In order to measure the effectiveness[4] of the proposed OOXML algorithm, it was necessary to determine its accuracy and reliability with regard to the assessment results it produced.

[3] Efficiency refers to achieving a desired outcome with minimum effort (Kumar & Gulati, 2009)

[4] Effectiveness is a measure of the ability to achieve a pre-defined goal or objective (Kumar & Gulati, 2009)

8.4 Determining the accuracy and reliability of the proposed OOXML algorithm

A comparison between the OOXML algorithm's object comparison metric and the manual assessment metrics applied by human markers, as well as the similarity metrics embedded in established algorithms, were conducted to determine the accuracy and reliability of the proposed OOXML algorithm. The assessment results of 200 word-processing assignments, produced by these assessment methods, were compared in a quasi-experimental[5] study. The differences and correlations between the assessment results were determined. Furthermore, the accuracy and reliability of the assessment methods were also established.

To identify candidate comparative algorithms, an investigation was conducted into various assessment algorithms used to evaluate free-text responses and computer programming skills, as well as non-OOXML-based document analysis algorithms. The investigation provided insight with regard to the broader context of computerised assessment and identified three document analysis algorithms that could be used to determine the accuracy and reliability of the proposed OOXML algorithm. The three algorithms included the Levenshtein algorithm (Levenshtein, 1966) that applies the Levenshtein distance metric, a file comparison program called fComp (Dowsing et al., 1998) that applies a byte comparison metric and Word Grader (Hill, 2011), that applies a document difference comparison metric to determine whether the student's document coincides with the memorandum.

A benchmark for determining the accuracy and reliability of the assessment results was also established by the instructor of a computer proficiency module re-assessing the 200 selected word-processing assignments.

8.5 Research analysis and findings

The accuracy and reliability of the proposed OOXML algorithm were compared to that of multiple human markers by using the benchmark results as a reference. In order to determine whether a difference existed between the accuracy and reliability of their assessment results the following hypotheses (formulated in Section 1.6) had to be tested:

- $H_{0,1}$: There is no difference between the assessment accuracy of the OOXML algorithm and multiple human markers.

[5] In a quasi-experimental study the independent variable is not randomly assigned (Ellis, 1999)

- $H_{0,2}$: There is no difference between the assessment reliability of the OOXML algorithms and multiple human markers.

The analysis results of the research study proved that there is a significant difference between the accuracy of the assessment results produced by the OOXML algorithm and those allocated by multiple human markers. A reliability test also indicated a significant difference between the reliability of the OOXML algorithm and multiple human markers. The test demonstrated that multiple human markers provided inconsistent assessment results, whereas the OOXML algorithm provided consistent assessment results. Both null hypotheses were thus rejected.

Furthermore, the statistical analysis demonstrated better agreement between the OOXML algorithm and the benchmark assessment results than between the human markers and benchmark results. Therefore, it was concluded that the object comparison metric, applied by the OOXML algorithm, produced assessment results that were more accurate than those produced by multiple human markers.

It was also important to determine whether the assessment results of the OOXML algorithm were comparable to the assessment results produced by WordGrader, fComp and the Levenshtein algorithm as implemented in Word Assessment Manager (WAM). To accomplish this, the following hypotheses had to be tested:

- $H_{0,3}$: There is no difference between the assessment accuracy of the OOXML algorithm and the selected computerised assessment algorithms.
- $H_{0,4}$: There is no difference between the assessment reliability of the OOXML algorithm and the selected computerised assessment algorithms.

Again, the assessment results produced by the relevant algorithms were compared to the benchmark results and analysed. The statistical analysis caused the first hypothesis to be rejected (see Section 7.6), while the second hypothesis could not be rejected (see Section 7.6).

It was therefore determined that a significant difference existed between the accuracy of the assessment results produced by the OOXML algorithm and those produced by the Levenshtein algorithm, fComp and Word Grader's document difference comparison algorithm. The analysis of the descriptive statistics (see Section 7.6) provided evidence that

the OOXML algorithm produced assessment results that were in better agreement with the benchmark than the assessment results produced by the other relevant algorithms. According to the analysis of a reliability test, no difference exists between the reliability of any of the document analysis algorithms; since they all produced consistent assessment results (see Section 7.6).

The research project has therefore produced an OOXML-based document analysis algorithm called the OOXML algorithm. Its application of object comparison techniques provides it with the ability to assess word-processing assignments in an accurate and reliable way.

8.6 Recommendations

This research project confirmed the results of Dowsing et al. (1998), namely that the manual assessment of word-processing skills tends to produce unreliable results. The implementation of computerised assessment with regard to the assessment of word-processing assignments is therefore an accurate and reliable option. However, to ensure that students execute the most relevant actions for a specific word-processing task it is necessary to apply event stream analysis in conjunction with document analysis techniques.

8.7 Future research and development

The computerised assessment algorithm that was developed as part of this research project focused mainly on the assessment of word-processing skills. However, it provided groundwork for the development of an assessment algorithm for the evaluation of spreadsheets and presentations that implement the OOXML standard. Furthermore, the development of an assessment system that implements the OOXML algorithm, with the ability to setup word-processing assignments, could also be conducted in future.

As with all document analysis algorithms, the OOXML algorithm is also incapable of determining the actions that the student performed to complete a specific word-processing task (see Section 2.4.1). It might thus be of great relevance to determine the feasibility of combining the OOXML algorithm with an event stream analysis algorithm to address this issue.

References

Activ Training. (n.d.). Online learning solutions for training providers. Retrieved 5 November 2014, from http://www.activ-training.com/solutions/training-providers

Alber, S., & Debiasi, L. (2013). Automated assessment in massive open online courses. Retrieved from http://www.uni-salzburg.at/fileadmin/multimedia/SRC/docs/teaching/SS13/SaI/Paper_Alber_Debiasi.pdf

Alfonseca, E., Carro, R.M., Freire, M., Ortigosa, A., Pérez, D., & Rodríguez, P. (2005). Authoring of adaptive computer assisted assessment of free-text answers. *Educational Technology & Society*, *8*(3), 53–65.

Attali, Y., & Burstein, J. (2006). Automated essay scoring with e-rater® V. 2. *The Journal of Technology, Learning and Assessment*, *4*(3).

Basin, Y., Beirne, M.J., Peterson, J.C., & Peterson, K.L. (2005). System and method for manipulating and managing computer archive files. Wisconsin. Retrieved from http://www.google.com/patents/US6879988

Baumstark, L., & Rudolph, E. (2013). Automated online grading for virtual machine-based systems administration courses. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education* (pp. 477–482). ACM.

Beskeen, D. (2013). *Microsoft Office 2013: Illustrated introductory, First course*. Cengage Learning.

Biggs, J., & Tang, C. (2011). *Teaching for quality learning at university*. McGraw-Hill International.

Blackboard Inc. (n.d.). Blackboard Learn. Retrieved 11 December 2014, from http://uki.blackboard.com/sites/international/globalmaster/Platforms/Blackboard-Learn.html

Burstein, J., Kukich, K., Wolff, S., Lu, C., & Chodorow, M. (1998a). Automated scoring using a hybrid feature identification technique. In *Proceedings of the 17th International Conference on Computational Linguistics-Volume 1* (pp. 206–210). Association for Computational Linguistics.

Burstein, J., Kukich, K., Wolff, S., Lu, C., & Chodorow, M. (1998b). *Computer analysis of essays*. Educational Testing Service.

Burstein, J., Kukich, K., Wolff, S., Lu, C., & Chodorow, M. (1998c). *Enriching automated essay scoring using discourse marking*. Educational Testing Service.

Burstein, J., Leacock, C., & Swartz, R. (2001). Automated evaluation of essays and short answers. In *Proceedings of the 5th CAA Conference*, Loughborough: Loughborough University

Business Wire. (2003, March 26). SVI introduces computer competency for our schools program. Retrieved 20 November 2014, from http://www.businesswire.com/news/home/20030326005053/en/SVI-Introduces-Computer-Competency-Schools-Program#.VG3bIMmM-So

Butcher, P.G. (2008). Online assessment at the Open University using open source software: Moodle, OpenMark and more. In *Proceedings of the CAA Conference*, Loughborough

Butcher, P.G., & Jordan, S.E. (2010). A comparison of human and computer marking of short free-text student responses. *Computers & Education*, *55*(2), 489–499.

Butcher, P.G., Swithenby, S.J., & Jordan, S.E. (2009). e-Assessment and the independent learner. In *ICDE World Conference on Open Learning and Distance Education* (pp. 1A8). Maastricht, The Netherlands

Callear, D., Jerrams-Smith, J., & Soh, V. (2001). Bridging gaps in computerised assessment of texts. In *Proceeding of IEEE International Conference on Advanced Learning Technologies, 2001*. (pp. 139–140). IEEE.

Carmines, E.G., & Zeller, R.A. (1979). *Reliability and validity assessment*. SAGE Publications.

Cengage Learning. (n.d.). SAM. Retrieved 20 November 2014, from http://www.cengage.com/samoffice2013/aboutsam.html

Christie, J. (2003). Automated essay marking for content - does it work? Presented at the 3rd Computer Assisted Assessment International Conference, Loughborough, UK.

Cook, T.D., Campbell, D.T., & Day, A. (1979). *Quasi-experimentation: Design & analysis issues for field settings* (Vol. 351). Houghton Mifflin Boston.

Creswell, J.W. (2003). *Research design: qualitative, quantitative, and mixed method approaches* (2nd Ed). Thousand Oaks, Calif: Sage Publications.

Deane, P. (2013). On the relation between automated essay scoring and modern views of the writing construct. *Assessing Writing*, *18*(1), 7–24.

Dean, T. (2012). *Network+ guide to networks* (6th Ed). Clifton Park, NY: Course Technology, Cengage Learning.

Ditch, W. (2007). XML-based office document standards. *JISC Technology & Standards Watch*, *1*(08).

Douce, C., Livingstone, D., & Orwell, J. (2005). Automatic test-based assessment of programming: A review. *Journal on Educational Resources in Computing (JERIC)*, *5*(3), 4.

Dowsing, R.D. (2000). Assessing word-processing skills by event stream analysis. *International Journal of Human-Computer Studies*, *52*(3), 453–469.

Dowsing, R.D., Long, S., & Sleep, M.R. (1998). Assessing word processing skills by computer. *Information Services and Use*, *18*(1), 15–24.

Easton, A.C., Easton, G., & Addo, T. (2006). But I am computer literate: I passed the test. *Journal of College Teaching & Learning (TLC)*, *3*(2).

ECMA. (2012, December). ECMA-376, 4th edition Office Open XML file formats — Fundamentals and Markup Language Reference. ECMA International. Retrieved from http://www.ecma-international.org/publications/standards/Ecma-376.htm

ECMA. (n.d.). Standard ECMA-376. Retrieved 13 November 2014, from http://www.ecma-international.org/publications/standards/Ecma-376.htm

Edwards, S.H. (2003). Teaching software testing: automatic grading meets test-first coding. In *Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications* (pp. 318–319). ACM.

Ellis, M.V. (1999). Repeated measures designs. *The Counseling Psychologist*, *27*(4), 552–578.

Fallows, S., & Steven, C. (2013). *Integrating key skills in higher education: Employability, transferable skills and learning for life*. Routledge.

Foltz, P.W., Laham, D., & Landauer, T.K. (1999). Automated essay scoring: Applications to educational technology. In B. Collis & R. Oliver (Eds.), *Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 1999* (pp. 939–944). Chesapeake, VA: AACE.

Fonte, D., da Cruz, D.C., Gançarski, A.L., & Henriques, P.R. (2013). A flexible dynamic system for automatic grading of programming exercises. *SLATE*, *13*, 129–144.

Forsythe, G.E., & Wirth, N. (1965). Automatic grading programs. *Communications of the ACM*, *8*(5), 275–278.

Fraenkel, J.R., Wallen, N.E., & Hyun, H.H. (1993). *How to design and evaluate research in education* (Vol. 7). McGraw-Hill New York.

Friedl, J. (2006). *Mastering regular expressions*. O'Reilly Media, Inc.

Garrison, D.R., & Kanuka, H. (2004). Blended learning: Uncovering its transformative potential in higher education. *The Internet and Higher Education*, *7*(2), 95–105.

Gliner, J.A., Morgan, G.A., & Leech, N.L. (2011). *Research methods in applied settings: An integrated approach to design and analysis*. Routledge.

Global Achievements. (2012, April). How many of us lie on CVs? Retrieved from http://www.galdac.com/Pages/FastPathOnline.aspx

Grant, D., Malloy, A., & Murphy, M. (2009). A comparison of student perceptions of their computer skills to their actual abilities. *Journal of Information Technology Education: Research*, *8*(1), 141–160.

Gray, D.N., Hotchkiss, J., LaForge, S., Shalit, A., & Weinberg, T. (1998). Modern languages and Microsoft's component object model. *Communications of the ACM*, *41*(5), 55–65.

Grove, S.K., Gray, J.R., & Burns, N. (2014). *Understanding nursing research: Building an evidence-based practice*. Elsevier Health Sciences.

Haldar, R., & Mukhopadhyay, D. (2011). Levenshtein distance technique in dictionary lookup methods: An improved approach. *eprint arXiv:1101.1232*. Retrieved from http://arxiv.org/abs/1101.1232

Haswell, R., & Wilson, M. (2013). Professionals against machine scoring of student essays in highstakes assessment. Retrieved 14 June 2015, from http://humanreaders.org

Helmick, M.T. (2007). Interface-based programming assignments and automatic grading of Java programs. *ACM SIGCSE Bulletin*, *39*(3), 63–67.

Hext, J.B., & Winings, J.W. (1969). An automatic grading scheme for simple programming exercises. *Communications of the ACM*, *12*(5), 272–275.

Hill, T.G. (2011). Word grader and powerpoint grader. *ACM Inroads*, *2*(2), 34–36.

Hill, T.G. (2014). Office Grader. Retrieved 4 January 2015, from http://www.officegrader.com/

Hollingsworth, J. (1960). Automatic graders for programming classes. *Communications of the ACM*, *3*(10), 528–529.

Hopkins, W. G. (2000). Quantitative research design. *Sportscience*, *4*(1). Retrieved from http://sportsci.org/jour/0001/wghdesign.html

Hull, M.J., Powell, D., & Klein, E. (2011). Infandango: Automated grading for student programming. In *Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education* (pp. 330–330). New York, NY, USA: ACM.

Hunt, N., Hughes, J., & Rowe, G. (2002). Formative automated computer testing (FACT). *British Journal of Educational Technology*, *33*(5), 525–535.

ISV. (n.d.). FastPath online. Retrieved 1 July 2015, from https://fastpath.isvinternet.com/

Jackson, D., & Usher, M. (1997). Grading student programs using ASSYST. In *ACM SIGCSE Bulletin* (Vol. 29, pp. 335–339). ACM.

Jordan, S., & Mitchell, T. (2009). e-Assessment for learning? The potential of short-answer free-text questions with tailored feedback. *British Journal of Educational Technology*, *40*(2), 371–385.

Klein, R., Kyrilov, A., & Tokman, M. (2011). Automated assessment of short free-text responses in computer science using latent semantic analysis. In *Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education* (pp. 158–162). New York, NY, USA: ACM. http://doi.org/DOI=10.1145/1999747.1999793

Kumar, S., & Gulati, R. (2009). Measuring efficiency, effectiveness and performance of Indian public sector banks. *International Journal of Productivity and Performance Management, 59*(1), 51–74.

Landauer, T.K., Laham, D., Rehder, B., & Schreiner, M.E. (1997). How well can passage meaning be derived without using word order? A comparison of Latent Semantic Analysis and humans. In *Proceedings of the 19th annual meeting of the Cognitive Science Society* (pp. 412–417).

Lánskỳ, J., Lokočr, J., Váňa, P., Hyskỳ, D., & Hájková, R. (2013). *AUTOPOT: A tool for automatic MS Word testing*. Oedm-serm.

Lawal, B. (2014). Repeated measures design. In *Applied Statistical Methods in Agriculture, Health and Life Sciences* (pp. 697–718). Springer International Publishing.

Leach, R.A. (2004). *The chiropractic theories: a textbook of scientific research*. Lippincott Williams & Wilkins.

Leacock, C. (2004). Scoring free-responses automatically: A case study of a large-scale assessment. *Examens, 1*(3).

Leal, J.P., Moreira, N., & Moreira, L.N. (1998). Automatic grading of programming exercises. In *Kurnia, A.; Cheang, B and Lim, A 'Online Judge', Computers and Education*.

Leonhard, W. (2009). *Windows 7 all-in-one for dummies*. John Wiley & Sons.

Levenshtein, V.I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady* (Vol. 10, pp. 707–710).

Li, M., Chen, X., Li, X., Ma, B., & Vitányi, P.M. (2004). The similarity metric. *Information Theory, IEEE Transactions on, 50*(12), 3250–3264.

Lim, D.H., Morris, M.L., & Kupritz, V.W. (2007). Online vs. blended learning: Differences in instructional outcomes and learner satisfaction. *Journal of Asynchronous Learning Networks*, *11*(2), 27-42

Lin, D. (1998). An information-theoretic definition of similarity. In *ICML* (Vol. 98, pp. 296–304).

McCabe, T.J. (1976). A complexity measure. *IEEE Transactions on Software Engineering*, *SE-2*(4), 308–320.

McGraw-Hill Education. (n.d.). SIMnet. Retrieved 13 November 2014, from https://mheducation.simnetonline.com/sp/

Menditto, A., Patriarca, M., & Magnusson, B. (2006). Understanding the meaning of accuracy, trueness and precision. *Accreditation and Quality Assurance*, *12*(1), 45–47.

Microsoft Inc. (2015). System requirements for Office 2013. Retrieved 24 June 2015, from https://technet.microsoft.com/en-us/library/ee624351.aspx

Microsoft Inc. (n.d.). DocumentClass.Compare method. Retrieved 19 June 2015, from https://msdn.microsoft.com/en-us/library/microsoft.office.interop.word.documentclass.compare.aspx

Miller, W., & Myers, E.W. (1985). A file comparison program. *Software: Practice and Experience*, *15*(11), 1025–1040.

Ming, Y., Mikhailov, A., & Kuan, T.L. (1999). Intelligent essay marking system. In *Educational Technology Conference*.

Ming, Y., Mikhailov, A., & Kuan, T.L. (2000). Intelligent essay marking system. *Learners Together*.

Morris, D.S. (2003). Automatic grading of student's programming assignments: an interactive process and suite of programs. In *Frontiers in Education, 2003. FIE 2003 33rd Annual* (Vol. 3, p. S3F–1). IEEE.

O'Brien, D.G., & Yasnoff, W.A. (1999). Privacy, confidentiality, and security in information systems of state health agencies. *American Journal of Preventive Medicine*, *16*(4), 351–358.

Ott, R., & Longnecker, M. (2008). *An Introduction to Statistical Methods and Data Analysis*. Cengage Learning.

Page, E.B. (1966). The imminence of grading essays by computer. *Phi Delta Kappan*, 238–243.

Page, E.B. (1968). The use of the computer in analyzing student essays. *International Review of Education*, *14*(2), 210–225.

Page, E.B. (1994). Computer grading of student prose, using modern concepts and software. *The Journal of Experimental Education*, *62*(2), 127–142.

Page, E.B., & Petersen, N. S. (1995). The computer moves into essay grading: updating the ancient test. *Phi Delta Kappan*, *76*(7), 561–65.

Papajorgji, P., & Pardalos, P. (2014). Programming paradigms. In *Software Engineering Techniques Applied to Agricultural Systems* (Vol. 93, pp. 3–8). Springer US.

Pearson. (n.d.). MyITLab. Retrieved 20 November 2014, from http://www.pearsonmylabandmastering.com/northamerica/myitlab/educators/features/index.html

Pellet, J.P., & Chevalier, M. (2014). Automatic extraction of formal features from Word, Excel, and PowerPoint productions in a diagnostic-assessment perspective. In *The 2014 International Conference on Education Technologies and Computers (ICETC)*, (pp. 1–6). IEEE.

Pérez, D., Alfonseca, E., & Rodríguez, P. (2004). Application of the BLEU method for evaluating free-text answers in an e-learning environment. In *Proceedings of the Language Resources and Evaluation Conference (LREC)* (pp. 1351–1354).

Pérez-Marín, D., Pascual-Nieto, I., & Rodríguez, P. (2009). Computer-assisted assessment of free-text answers. *The Knowledge Engineering Review*, *24*(04), 353.

Pfitzmann, A., & Köhntopp, M. (2001). Anonymity, unobservability, and pseudonymity—a proposal for terminology. In *Designing privacy enhancing technologies* (pp. 1–9). Springer.

Pieterse, V. (2013). Automated assessment of programming assignments. In *Proceedings of the 3rd Computer Science Education Research Conference on Computer Science Education Research* (pp. 4:45–4:56). Open Universiteit, Heerlen.

Poletiek, F.H. (2013). *Hypothesis-testing behaviour*. Psychology Press.

Quah, J.T.S., Lim, L.R., Budi, H., & Lua, K.T. (2009). Towards automated assessment of engineering assignments. In *Neural Networks, 2009. IJCNN 2009. International Joint Conference on* (pp. 2588–2595).

Razali, N.M., & Wah, Y.B. (2011). Power comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-darling tests. *Journal of Statistical Modeling and Analytics*, *2*(1), 21–33.

Repici, D. J. (2010). How to: The comma separated value (csv) file format. *Creativyst Inc.* Retrieved from http://www.creativyst.com/Doc/Arti cles/CSV/CSV01.htm

Rice, J.A. (2007). *Mathematical statistics and data analysis* (3rd Ed). Belmont, CA: Thomson/Brooks/Cole.

Robbins, R., & Zhou, Z. (2007). A comparison of two computer literacy testing approaches. *Issues in Information Systems*, *8*(1), 185–191.

Rubin, A. (2009). *Statistics for evidence-based practice and evaluation*. Cengage Learning.

Rudner, L.M., Garcia, V., & Welch, C. (2006). An evaluation of IntelliMetric^{TM} essay scoring system. *The Journal of Technology, Learning and Assessment*, *4*(4).

Samuel, S., & Kovalan, A. (2014). A critical evaluation on programming paradigms to achieve optimal resource utilization of mobile softwares in mobile devices. *International Journal of Innovation in the Digital Economy (IJIDE)*, *5*(1), 50–59.

Shermis, M.D., Koch, C.M., Page, E.B., Keith, T.Z., & Harrington, S. (2002). Trait ratings for automated essay grading. *Educational and Psychological Measurement*, *62*(1), 5–18.

Sourceforge. (n.d.). PMD. Retrieved 17 June 2015, from http://pmd.sourceforge.net/

Strauss, H.J. (2008). *A comparative study to determine the optimum e-assessment paradigm for testing users' word processing skills*. University of the Free State.

Suleman, H. (2008). Automatic marking with Sakai. In *Proceedings of the 2008 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on IT Research in Developing Countries: Riding the Wave of Technology* (pp. 229–236). ACM.

Training Zone. (2000, April 25). SVi launch two new software programmes for trainers. Retrieved 20 November 2014, from http://www.trainingzone.co.uk/topic/svi-launch-two-new-software-programmes-trainers

Tuparova, D., & Tuparov, G. (2010). Automated real-live performance-based assessment of ICT skills. *Procedia - Social and Behavioral Sciences*, *2*(2), 4747–4751.

University of the Free State. (n.d.). Blackboard Learn. Retrieved 11 December 2014, from https://learn.ufs.ac.za/webapps/login/

Valenti, S., Neri, F., & Cucchiarelli, A. (2003). An overview of current research on automated essay grading. *Journal of Information Technology Education: Research*, *2*(1), 319–330.

Vantage Learning. (2000). A study of expert scoring and IntelliMetric scoring accuracy for dimensional scoring of Grade 11 student writing responses (RB-397). *Newtown, PA: Vantage Learning*.

Van Vugt, W. (2007). Open XML: The markup explained. Retrieved from http://educationdisha.com/upload/Open%20XML%20Explained.pdf

Von Matt, U. (1994). Kassandra: The automatic grading system. *SIGCUE Outlook*, *22*, 26-40

Wang, J., & Brown, M.S. (2007). Automated essay scoring versus human scoring: A comparative study. *The Journal of Technology, Learning and Assessment, 6*(2).

Whittington, D., & Hunt, H. (1999). Approaches to the computerized assessment of free text responses (pp. 207–219). Presented at the Proceedings of the Third Annual Computer Assisted Assessment Conference, Loughborough, England: Loughborough University.

Wiley. (n.d.). Microsoft Official Academic Course. Retrieved 4 January 2015, from http://eu.wiley.com/WileyCDA/Section/id-302887.html

Wolters, A. (2010). Office Skill Assessment Project. Retrieved from http://repository.tudelft.nl/assets/uuid:32f1b1bb-0c94-44dc-8130-e363e15cebbf/OSAP_thesis__3_1.pdf

Appendix A

The Word-Processing Assignment

PRAC 2

MS WORD CH 1 + 2

A. SWITCH ON THE COMPUTER AND LOG IN

- **Switch on** the computer.
- **Type** "**student"** in the **Username** text box and click the arrow-button.

B. CREATE A FOLDER AND DOWNLOAD FILES FROM BLACKBOARD

a. **Create** a folder as follows on the **T:** drive:

Right-click on START

Click OPEN WINDOWS EXPLORER

Click on T: drive

Click the NEW FOLDER tab,

Type: Your student number, Initials and Surname_BRS111-PRAC2

e.g. 2011345678 SP MOLEFI_BRS111-PRAC2

b. **Download** the images **WD01_TREE** and **WD01_ARBOR_DAY** as follows from Blackboard and **save** it on the T: drive:

Double-**click** on the Blackboard icon on the desktop (or open Internet Explorer, Click on STUDENTS, eLEARN BLACKBOARD)

Type your student number in the USERNAME text box and your password in the PASSWORD text box.

Click LOGIN.

Click the MY COURSES tab.

Click BRS111_ON.

Click on the words PRAC 2 at the left-hand side.

Click on the attached file **WD01_TREE** at the right-hand side. A dialog box will open.

Click on SAVE AS. The SAVE AS dialog box will open.

If you can't see the T: drive under the COMPUTER folder at the left-hand side, **click** on the triangle before the word COMPUTER to expand the folder.

Click on T: drive.

Double-click your PRAC2 folder to open it.

Click the SAVE button to save the image in the PRAC2 folder.

Use the same method to **download WD01_ARBOR_DAY** to the T: drive.

Minimize the Blackboard window.

C. HISTORY OF ARBOR DAY

You are the program coordinator for the Free State Environmental Affairs Department. During a public gathering, you plan to give a short speech on the history of arbour day.

a. **Type** the following information in a new MS Word document.
Press the ENTER button twice at the end of each paragraph to create a blank line.

A Brief History of Arbor Day
Arbor Day originated in 1870 in the United States territory of Nebraska. Mr. J. Sterling Morton, a newcomer to the treeless plains of Nebraska, was a keen proponent of the beauty and benefit of having trees.

He persuaded the local agricultural board to set aside a day for planting trees and through his position as editor of Nebraska's first newspaper, encouraged participation in the event by publishing articles on the value of trees.

Mr. Morton's home, known as Arbor Lodge, was a testament to his love for trees and so inspired the name of the holiday; Arbor Day. Within two decades Arbor Day was celebrated in every US State and territory, and eventually spread around the world. The tradition continues annually in the second week of August, in global acknowledgment of Mr. Morton's slogan, "other holidays repose upon the past; Arbor Day proposes for the future."

In South Africa, Arbor Day was first celebrated in 1983. The event captured the imagination of people who recognized the need for raising awareness of the value of trees in our society. As sources of building material, food, medicine, and simple scenic beauty, trees play a vital role in the health and well-being of our communities. Collective enthusiasm for the importance of this issue in South Africa inspired the national government, in 1999, to extend the celebration of Arbor Day to National Arbor Week. From 1 to 7 September every year, schools, businesses and organizations are encouraged to participate in community "greening" events to improve the health and beauty of the local environment and propose a green future for South Africa.

b. **Correct** all the spelling and grammar errors.
Save the document as Your student number, Initials and Surname_ARBOR DAY HISTORY in your BRS111-PRAC2 folder,
e.g. 2011345678 SP MOLEFI_ARBOR DAY HISTORY.

c. **Turn on** the display of formatting marks.
Remove any extra blank spaces between words or at the end of lines.
Turn off the display of formatting marks.
Change the line and paragraph spacing as follows:
Line spacing: 1.0, Paragraph spacing: Remove Space after Paragraph.

d. **Change** the view to Print Layout if necessary.
Change the margins as follows:
Top: 2.5 cm, Bottom: 2.5 cm, Left: 1.5 cm, and Right: 1.5 cm.
(Page Layout, Margins, Custom Margins)
Change the formatting of the title to center, Calibri, 24 pt, bold, and green font colour.

e. **Delete** the word "having" in the first paragraph and **change** the date 1870 to "1872". (Do not type the quotation marks. It only indicates the applicable text.)
Type the text "for soil protection, fruit, shade and building." at the end of the second paragraph.

f. **End** the sentence in paragraph 2 after the word "trees".
Change the rest of the sentence ("and through this") to "Using his position as editor…".

g. **Delete** the phrase "Within two decades" in the second sentence of the third paragraph.
Replace the word "holiday" throughout the document with the word "celebration".

h. **Cut** the paragraph beginning with "He persuaded the local" and
paste it before the fourth paragraph that begins with the phrase "In South Africa, Arbor Day…".
Save the document again.

i. **Change** the formatting of the date in the first paragraph to Italics, Bold, and a green font colour. Use the Format Painter to add this formatting to all the dates in the fourth paragraph.

j. **Justify** the paragraphs.

k. **Use** the Thesaurus to **replace** the word "plains" in the first paragraph with the synonym "grasslands".
(Select the word "plains". Review tab, Thesaurus, click on the down-arrow next to the word "grasslands" in the Research panel at the right-hand side of the window. Click on INSERT.)

l. **Increase** the font size of the paragraphs to 13 pt.
Move the sentences of the last paragraph to separate lines and change their format to a bulleted list.

m. **Insert** a blank line below the title.

n. **Insert** the image **WD01_TREE** (on the T: drive) on the blank line and **center** it.
Change the picture style to SOFT EDGE, OVAL.
(Picture Tools tab, Format tab, Picture Styles, More, [click on the down-arrow next to the Picture Styles])

o. **Type** your student number, initials and surname several lines below the last line.
Type the current date on the next line.
Left-align both lines.

p. **Insert** a page break before the paragraph starting with the words "In South Africa …".
Add a border around the pages.
(Page layout, Page Borders)
Change the zoom of the page to see the whole page.

q. **Add** the following document properties:
(FILE, Click on the down-arrow next to PROPERTIES, click SHOW DOCUMENT PANEL)

Author: your student number, initials, and surname
Title: Arbor day
Subject: History

r. **Save** the document again.
Print Preview the document.
Close the document, but not the Word program.

D. ARBOR DAY POSTER

You want to create a flyer to inform learners about the Arbor Day poster competition that your department is hosting.
You will also create the fax coversheet using a Word template to send to all the schools. You will then give the fax coversheet and flyer to John, one of your colleagues to review.

a. **Type** the following information in a new MS Word document.
Press the ENTER button where indicated.

Arbor Day poster contest! ENTER

Free State ENTER (twice)

The Free State Department of Environmental affairs are calling on you to showcase your students' artistic talents by participating in the 2013 Free State Arbor Day Poster Contest.
This year's theme is "Where Do I Belong?" ENTER

The deadline for all contest submissions is April 1, 2013. Only posters that meet the contest rules will be eligible for prizes. Please read the poster contest rules in the activity guide carefully to make sure your students meet eligibility requirements. ENTER

Thank you in advance for participating in the 2013 Free State Arbor Day Poster Contest. Don't hesitate to contact me for more information. ENTER

Sincerely Yours, ENTER

Name Surname (Your own name and surname) ENTER

Office hours: Monday – Friday 8:00 a.m. to 17:00 p.m. Saturday 8:00 a.m. to 12:00 a.m.
Contact number: 051 401 5454 ENTER

2314 Telegraph Avenue

b. **Correct** all the spelling and grammar errors that are identified.

c. **Save** the document as Your student number, Initials and Surname_ POSTER in your BRS111-PRAC2 folder, e.g. 2011345678 SP MOLEFI_POSTER.

d. **Center** the entire document.

e. **Capitalize** the first letter of each word of the first line ("Capitalise Each Word").
Change the format of the text "Free State" to uppercase ("UPPERCASE").

f. **Move** the street address, including the following paragraph mark, to below your Name and Surname.
Turn off the display of formatting marks.

g. **Type** the following three lines of text between the second and third paragraphs:
April 23, Grade 3 Winner announced
April 24, Grade 4 Winner announced
April 25, Grade 5 Winner announced

h. **Change** the format of the first line of the document to a font colour of dark blue, font type of Tahoma and 26 pt font size.
Change the format of the text "Free State" as follows: font colour: Orange, font type: Arial Narrow and size: 24 pt.
Select the remaining text in the document and **increase** the font size to 14 pt.

i. **Change** the format of the three date lines below the second paragraph to a font colour of red and a font size of 16 pt.
Change the formatting of the last two lines (hours and contact number) to purple and bold.

j. **Insert** the picture WD01_ARBOR (on T: drive) on the blank line below the title "Free State" at the top of the document.
Size the graphic to be approximately 6 by 6 centimetres **using** the ruler as a guide.

k. **Add** the following document properties:

Author: your student number, initials, and surname
Title: Arbor Day Poster
Subject: Flyer to advertise poster competition

l. **Use** QUICK PARTS, AUTHOR to add your student number, initials and surname below the last line.
Add a **date field** on the line below the author. The date must update automatically.
Right align both lines.

m. **Save**, **print preview**, and then **close** the flyer document.
Do not close the MS Word program.

E. TREE OF THE YEAR

In addition to the poster competition you want to create a flyer informing people about the tree of the year as well as those of previous years.

a. **Download** the document **WD02_TREE OF THE YEAR** from Blackboard and save it in your PRAC 2 folder as Your student number, Initials and Surname_ TREE OF THE YEAR,
e.g. 2011345678 SP MOLEFI_TREE OF THE YEAR.

Download the document **WD02_TREE HISTORY** from Blackboard and save it in your PRAC 2 folder as Your student number, Initials and Surname_TREE HISTORY,
e.g. 2011345678 SP MOLEFI _TREE HISTORY

b. **Open** your **TREE OF THE YEAR** document.
On line 6, **insert** a left tab stop at 1.5 cm and center tabs at 8.0 cm and 14.5 cm.

c. **Type** the word "Common Name" at the first tab stop, "Scientific Name" at the second tab stop, and "Year" at the third tab stop.

d. **Type** the following information by **using** the tab stops:

Fever tree	Acacia xanthophloea	2010
Jacket-plum	Pappea capensis	2011
Water berry	Syzygium cordatum	2012
Blossom tree	Virgilia oroboides	2013

e. **Double-underline** the table headings.
(Click on the Font dialog box arrow next to the Font group's name / Underline style)

f. **Right-align** the title of the document.

g. **Center-align** the text "Tree of the year 2010 - 2013".

h. **Save** the document.

i. **Open** your **TREE HISTORY** document.
Display the open MS Word windows side by side. (View, Side by Side)
Copy the title and first two paragraphs of TREE HISTORY and **insert** it above "Tree of the year 2010 – 2013" in the TREE OF THE YEAR document.
Right-align the words "Tree of the year" below the title of the document.

j. **Copy** the remaining paragraphs from the TREE HISTORY document and insert it at the bottom of the TREE OF THE YEAR document.
Include two blank lines between the table and the paragraph.
Close the TREE HISTORY document.

k. **Insert** the "Horizontal Scroll" shape, from the Stars and Banners group, at the bottom of the document.
Type the text "Arbor Day" in the shape.
Change the formatting of the text to bold, 18pt.
Add green fill colour to the shape.
Move the shape to the top of the document, left of the document title, and **size** it appropriately.

l. **Add** the following document properties:

Author: your student number, initials, and surname
Title: Tree of The Year History
Subject: Tree of The Year 2010 - 2013

m. **Spell-check** the document.
Ensure that the document fits on one page.
Save the document, **Print Preview** and **close** it.

F. PROCEDURE TO END THE PRACTICUM SESSION

a. **Call** the demmie to be your **witness** when you upload your folder to Blackboard.

b. **Upload** your BRS111-PRAC2 folder, which is on the T: drive, to Blackboard as follows:

- In **Blackboard, click** on the **PRAC 2 SUBMISSION**-button (in the center)
- Under ASSIGNMENT MATERIALS: **click** on the **BROWSE MY COMPUTER** button (scroll the page down to see this button).
- Click on **COMPUTER** at the left and then click on the **T:** drive.
- **Click** on your **BRS111-PRAC2** folder.
- **Click** the **OPEN**-button.

c. **Log out** of Blackboard.

Appendix B

The Word-Processing Memoranda

A Brief History of Arbour Day

Arbour Day originated in ***1872*** in the United States territory of Nebraska. Mr J. Sterling Morton, a newcomer to the treeless grasslands of Nebraska, was a keen proponent of the beauty and benefit of trees.

Mr Morton's home, known as Arbour Lodge, was a testament to his love for trees and so inspired the name of the celebration; Arbour Day. Arbour Day was celebrated in every US State and territory, and eventually spread around the world. The tradition continues annually in the second week of August, in global acknowledgment of Mr Morton's slogan, "other celebrations repose upon the past; Arbour Day proposes for the future."

He persuaded the local agricultural board to set aside a day for planting trees. Using his position as editor of Nebraska's first newspaper, encouraged participation in the event by publishing articles on the value of trees for soil protection, fruit, shade and building.

- In South Africa, Arbour Day was first celebrated in ***1983***.
- The event captured the imagination of people who recognized the need for raising awareness of the value of trees in our society.
- As sources of building material, food, medicine, and simple scenic beauty, trees play a vital role in the health and well-being of our communities.
- Collective enthusiasm for the importance of this issue in South Africa inspired the national government, in ***1999***, to extend the celebration of Arbour Day to National Arbour Week.
- From ***1 to 7 September*** every year, schools, businesses and organizations are encouraged to participate in community "greening" events to improve the health and beauty of the local environment and propose a green future for South Africa.

1234567890, WSJ, Marais
4 December 2013

Arbour Day Poster Contest!

FREE STATE

The Free State Department of Environmental affairs are calling on you to showcase your students' artistic talents by participating in the 2013 Free State Arbour Day Poster Contest. This year's theme is "Where Do I Belong?"

The deadline for all contest submissions is April 1, 2013. Only posters that meet the contest rules will be eligible for prizes. Please read the poster contest rules in the activity guide carefully to make sure your students meet eligibility requirements.

April 23, Grade 3 Winner announced
April 24, Grade 4 Winner announced
April 25, Grade 5 Winner announced

Thank you in advance for participating in the 2013 Free State Arbour Day Poster Contest. Don't hesitate to contact me for more information.

Sincerely Yours,
Jaco Marais

2314 Telegraph Avenue
Office hours: Monday – Friday 8:00 a.m. to 17:00 p.m. Saturday 8:00 a.m. to 12:00 a.m. Contact number: 051 401 5454

Jaco
19/11/2015

Arbor Day

Free state Department of Environmental Affairs

Tree of the Year

The Tree of the Year was an initiative of the Department of Water Affairs & Forestry (now the Department of Environmental Affairs and Tourism) to raise awareness in nature. In 1996 they introduced a category of the "Rare Tree of the Year". This has now been changed to an "Alternate Tree of the Year" so that hopefully one or the other will suit your habitat.

The selection of the Tree of the Year is now a joint effort by the Department of Environmental Affairs and Tourism, The Botanical Society and the South African Bio-diversity Institute (SANBI)

Tree of the year 2010 - 2013

Common Name	Scientific Name	Year
Fever tree	Acacia xanthophloea	2010
Jacket-plum	Pappea capensis	2011
Water berry	Syzygium cordatum	2012
Blossom tree	Virgilia oroboides	2013

Arbour Day is a day in which individuals and groups are encouraged to plant and care for trees. The National Government of South Africa has extended it to National Arbour Week, which lasts from the 1st – 7th September. Two trees, one common and one rare, are highlighted to increase public awareness of indigenous trees, while various ‘greening’ activities are undertaken by schools, businesses and organizations.

Appendix C

OfficeGrader Frequently Asked Questions

Wiley OfficeGrader 2013 Trial Version FAQ – Frequently Asked Questions (distributed by John Wiley & Sons Inc.) www.wiley.com/college/microsoft

Thank you for your interest in Wiley OfficeGrader from John Wiley & Sons. As a trial, we've created separate versions of our OfficeGrader applications for Word, Excel, PowerPoint and Access. You will need to install each version separately (you do not need to install all of them – just the grader(s) you're interested in). The trial version works the same as the full version, except you are limited to 10 uses. Once you've clicked the "Grade" button ten times, the software will lock out.

A full version of Wiley OfficeGrader is available for sale on Wiley.com. Ask your MOAC sales rep about a discounted version of OfficeGrader for adopting MOAC courseware.

1. Who developed Wiley OfficeGrader?

   Tom Hill developed Excel Grader and Access Grader while working on his Ph.D. in computer science.

2. How did the idea for Wiley OfficeGrader come about?

   Dr. Hill was a full-time computer science instructor as he worked part-time on his graduate degree. The majority of the courses he taught were computer applications courses. As he searched for a dissertation topic, he had the idea of creating grading programs to mark his homework more accurately than he could with paper printouts.

3. When was Wiley OfficeGrader first developed?

   The first Office Grader paper was published in 2003. Dr. Hill received his Ph.D. the same year.

4. How does the Plagiarism detection work?

   This is an "after the fact" method of plagiarism detection. In short, all files that have been graded in the previous batch (only) are compared for common errors. The resulting report shows groups of students with the same errors. You may specify the threshold for being included in the report (100% of errors in common, 90%, etc.)

5. Why does PowerPoint Grader return a Word document and not a PowerPoint document?

We experimented with a PowerPoint document. The comments in PowerPoint are limited in length and we were unhappy with the look of the marked-up PowerPoint document. By porting it to a Word doc and marking up that file we are able to provide better feedback.

6. Explain the point deduction/percentage box in the lower left corner.

With the latest updates of the Wiley OfficeGrader programs (July 2013), all of the graders have moved to points deducted per error. Previous versions of Excel Grader used percentage allocations for errors. The other graders have always used points deducted per error. We moved to a numeric point deduction for consistency and to increase accuracy. Any numeric value can be entered in the fields. We have updated the default values to reflect the MOAC Office textbook assignments, although you may manually change any of these fields.

7. Does Wiley OfficeGrader 2013 work with Microsoft Office 2007 or Office 2010?

You should be able to use Wiley OfficeGrader 2013 products with Microsoft Office 2013, Microsoft Office 365, Microsoft Office 2010 and Microsoft Office 2007, but not Microsoft Office 2003.

8. Can you summarize what is NOT GRADED in Excel Grader?

Chart components (like object colors and labels)
Embedded objects (like pictures, clip art, org charts, word art)
Conditional format details (relational expression and format)
Drawing objects (shapes, lines, drop shadows, et.al)
Controls (buttons, list boxes, text boxes)

9. Can you summarize what is NOT GRADED in Word Grader?

Pictures
Picture/shape/illustration formatting
Text Boxes

10. Can you summarize what is NOT GRADED in PowerPoint Grader?

Picture contents
Text boxes (whose contents are not in outline view)
Animation settings
Picture/shape/illustration formatting

11. Can you summarize what is NOT GRADED in Access Grader?

Field Comments
Form and Report Formatting
Datasheet formats

12. Is there a limit to the number of files I can grade simultaneously?

This may depend on the document being graded, but in theory, no. More files increase the possibility one of the files is corrupted or so erroneous the grader cannot process it.

13. What are the recommended option settings for Word Grader?

The options include:
Ignore Case (instructor's option; We count incorrect case as a typo)
Skip Grading Fonts and Formatting: Off
Delete Original Student Documents: instructor's choice
Count Style Definition Errors: Off (sometimes generates multiple confusing errors; sometimes none)
Mark White Space Characters: On (mark: spaces, tabs, enter)
Override Character Count: If an assignment uses a starting document with 1000 characters, and only adds 250 characters, type 250 here
File Password: Required for assignments where key or student files are password protected

14. What are the recommended option settings for Excel Grader?

The options include:
Skip Grading Fonts and Formatting: Off
Delete Original Student Documents: instructor's choice
Don't Grade Protected Sheet/Locked Cells: Off
Deduct for Extraneous Cells: instructor's choice (counts off for cells with values not in the key)
Minimum number of similar errors for plagiarism report: 2--How many common errors between files cause suspicion of plagiarism?
Minimum percent of similar errors for plagiarism report: >90%--Doubt of plagiarism rises quickly with only a few errors not in common
Cap for repeated format errors: 10; To avoid excessive marking for the same error
Sheet Password: Required for assignments where key or student files are password protected

15. What are the recommended option settings for Access Grader?

The options include:
Delete Original Student Databases: instructor's choice
Only Grade Tables and Table Data: Off
Deduct for Extraneous Objects: instructor's choice (counts off for extra rows, fields, tables, forms, reports, queries
Minimum number of similar errors for plagiarism report: 2--How many common errors between files cause suspicion of plagiarism?
Minimum percent of similar errors for plagiarism report: >90%--Doubt of plagiarism rises quickly with only a few errors not in common

16. What are the recommended option settings for PowerPoint Grader?

The options include:
Skip Grading Fonts and Formatting: Off
Delete Original Student Documents: instructor's choice
Mark White Space Characters: On (mark: spaces, tabs, enter)
Margin of Error for Location/Size of pics, shapes, and media: 72 pixels

17. Do you have any tips for instructors?

a. Always do a test grading of a new assignment. Test your key against a submission that you suspect should be correct.
b. Backup documents if there is a chance of needing to re-grade them.
c. Do not assume the grader is correct: always check graded documents until you are confident the graders work correctly for your assignment.
d. Once you are confident the grader is working for your assignments, continue to check graded files that have low scores.
e. Keep Excel docs closed while using ExcelGrader.

18. How does ExcelGrader mark formulas?

Formulae are first graded by comparing their text strings. If the strings are not equal, Jens-Uwe Dolinsky's *Symbolic* computer algebra system is employed. In brief, the student's formula is subtracted from the correct formula and the expression is reduced. If the result is zero, the formula is accepted as correct.

19. What is the cell comment "Formula equivalent, but not identical"?

Points are not deducted when this comment is in a student's workbook. It means the student's formula was not precisely the same as the teacher's formula, but was algebraically equivalent. One example is the correct formula: =B9+(10%*B9) and the student formula =B9*(1+10%).

20. Excel Grader is too strict.

Since Excel Grader accepts the instructor's worksheet attributes as correct, non-equivalent student values are marked as incorrect. Extra spaces in student labels are marked. Excel Grader can differ between number formats and fonts that look almost identical (for example, Currency with no dollar sign versus Accounting with no dollar sign). Excel Grader grades negative number formatting for cells with Currency and Accounting formats.

Some ways to alleviate error overload:
- Turn off format grading
- Reduce the number of points deducted for certain error types
- Verify that the correct version is really accurate
- (it is usually a good practice to test the correct worksheet against a good student's)

21. Will the grader count off if the student puts his/her name in the workbook?

No. By default, Excel Grader does not count off for any extraneous values in fields. NOTE: in the correct version of the workbook, the corresponding cell must be empty

22. What is the Unattended batch file?

Since workbook grading can take a long time, scheduling grading to run overnight may be helpful.
You can edit the Unattended batch file by

- Clicking Start -> All Programs -> Excel Grader
- Right click on the Unattended batch file and select edit.

The form of statements in the batch file is
call "c:*path*\ExcelGrader.bat" "c:*path*\CorrectFile" "c:*path*\StudentFolder"

- Edit the last two parameters only. You may add a line for each assignment to be graded.

To schedule Unattended for overnight grading click:

- Start -> All Programs -> Accessories -> System Tools -> Scheduled Tasks
- Double click "Add scheduled task"
- Click Next, then browse to and select C:\Program Files\Excel Grader\Unattended.bat,
- Click Open
- Select Daily (or when you wish)
- Enter the time (for example 11:00 pm)
- Enter your user name and password, Click Next, then Finish

To check overnight grading, when you come in each morning, run Excel Grader and click Grade Report.

23. Why does the program skip files?

When a grader encounters a program exception, a corrupted file, or an unrecognizable error in a student document, it skips the document and continues grading the remainder of the files. Skipped files are moved to a “Not Graded” subfolder. The instructor must grade these files manually.

Occasionally an exception when processing a student document may cause the remainder of student documents not to be graded. Try to grade the folder a second time. If the folder still is not graded, remove the offending file from the student folder and re-grade the folder.

24. How does one gather and return assignments?

Instructors may have access to a Learning Management System. Some of these systems allow uploading of student assignments. If assignments are submitted as email attachments, the following programs may be useful:
Attachment Executive is a commercially available Outlook add-in for processing attachments and moving them to directories. Attachment Executive allows users to build rules for processing email attachments. Blat is a freeware command line tool for mailing graded Excel files as attachments. If you school uses Blackboard, it has tools for collecting and downloading student assignments.

25. Has this research been published?

Hill, T. “Word Grader and PowerPoint Grader.” ACM Inroads, pp. 34-36. June 2011.
T. HILL. "Excel Grader and Access Grader," inroads-The SIGCSE Bulletin, Vol. 6, No. 2, pp. 101-105, ACM, June 2004.
T. G. HILL. "MEAGER: Microsoft Excel Automated Grader," Journal of Computing Sciences in Colleges, Vol. 18, No. 6, pp. 151-164, June 2003.
Hill, Thomas G., MEAGER & MADBAGS: Automated Graders for Microsoft Excel and Access Assignments, Doctoral Dissertation, University of Mississippi , Oxford , 2003.

26. How does one handle a hung Grader?

Occasionally a corrupted file or other situation can hang a grader. If so, find the file causing the condition. Restart your computer, and then open the file that caused the error. Grade it manually and remove it from the student folder before re-grading the rest of the files.

Wiley OfficeGrader for Microsoft Office 2013 Trial Version FAQ v1.1, Updated 9/05/2013