ISSN 1744-1986

Technical Report N° 2007/ 23

# Analysis of a phonetic and rule based algorithm approach to determine rhyme categories and patterns in verse

F Kavanagh

29 September, 2007

Department of Computing
Faculty of Mathematics, Computing and Technology
The Open University

Walton Hall, Milton Keynes, MK7 6AA
United Kingdom

http://computing.open.ac.uk

# Analysis of a phonetic and rule based algorithm approach to determine rhyme categories and patterns in verse

A dissertation submitted in partial fulfilment
of the requirements for the Open University's
Master of Science Degree
in Computing for Commerce and Industry

Frank Kavanagh
(X0702320)

**26 February 2008**

Word Count: **15258**

## Preface

I would like to thank those people without whose assistance, support and guidance I would not have been able to complete this research.

Many thanks to my tutor Doug Leith for his assistance with the research approach taken and also his guidance with the documentation of my dissertation. I would also like to thank Trinity College Dublin for allowing me to use their extensive library and in particular to Dr Helen Conrad-O'Briain, School of English and Isolde Harpur librarian for their assistance in identifying and locating texts referenced in this research and also those used as background information to supplement my understanding of English verse.

My gratitude also goes to the two people who acted as reviewers for this research document, John Roche and Irene Hayden. Their comments on the technical aspects and readability of the document were invaluable.

During my own research many other researchers in this field who were kind enough to offer their support, insight and research results when requested. My thanks go to Justin Zobel, David Robey and Marc Plamondon for taking the time to assist and encourage me with my own research.

Finally I would like to my family for all their support during this period, my wife Irene and my two sons Sam and Finn. I could not have done it without them.

## Table of Contents

## List of Figures

## List of Tables

## Abstract

This dissertation analyses the use of a rule based algorithm incorporating a phonetic dictionary to identify the rhyme structure and pattern of English language poetic verse. Current methods of rhyme analysis incorporate the use of rhyming tables to identify rhyme. These rhyming tables require considerable manual effort to maintain and update and are cumbersome to use to allow for the variety of pronunciation and accentuation used in English verse. The research conducted and outlined in this paper assesses the feasibility of using the rules based algorithm to determine rhyme word pairs and hence rhyme structure and patterns.

For this research a prototype software application was developed using the JAVA programming language. Various rhyme pattern matching rules were modelled to identify particular rhyme types and words were identified and matched through these series of rules. By loading a phonetic dictionary it was possible to represent each word in multiple formats: the original ‘base’ word, the phonetic spelling representation of the word and the stressed vowel patterns of the word. A fourth representation was also possible by applying phonetic representation rules based on the Phonix/Editex method described by Zobel and Dart (1996). Using these four representations of a single word to facilitate comparison with another improved both the occurrence of a match and the confidence of the match.

The results of the research shows that it is possible to apply a rule based algorithm to the problem of rhyme structure identification in English verse. Although time is required to model the rules and rule exceptions to improve the accuracy of the rhyme identification this approach requires no manual effort once these are modelled. A single rule can correctly identify multiple rhyme types and word pairs in comparison to a look

up rhyme table that can only identify a single rhyme pairing for each entry and must be updated for each new pairing regardless of its similarity to an existing one. By using a phonetic dictionary the variance of pronunciation and accentuation in English verse may be avoided by loading a dictionary with the appropriate phonetic representations (assuming a dictionary with the appropriate pronunciation exists). This means it is very simple to alternate between the identification of rhyme patterns in eighteen century English verse and identifying rhyme patterns in twentieth century American English verse by simply loading the appropriate phonetic dictionary.

# Chapter 1 Introduction

## 1.1 Background to the research

In the survey on the state of the art in Human Language Technology by Cole et al (1997 pp387) it is recognised that, to date, data collection and dissemination efforts have been extremely successful and that efforts should now be focused on principles, procedures and tools for analyzing these data. Cole (loc. cit.) goes on to stress that there is a need for manual, semi-automatic and automatic methods that help produce linguistically motivated analyses that make it possible to derive further facts and generalisations that are useful in improving the performance of language processors.

One facet of Human Language Technology is the automated analysis of poetic verse. According to Houdek (1997) this computer analysis of English verse is currently a relatively unexplored field. Its goals are to identify a poem's metre, rhyme structure and form to enhance understanding and appreciation of the poem, to offer new insights into the poem or to act as a teaching or research aid.

A body of work already exists for the identification of the poetic metre of the verse. Plamondon (2006) uses syllable structure to identify the metre of the poem, that is, the arrangement of the vocally stressed and unstressed syllables. Another approach to the identification is the application of generative metrics. Generative metrics is a method of evaluating if a poetic line is metrical by building a tree structure of the stress of all the lines components and comparing it to the stress based on the word division of the line. Hayward (1996) applies connectionist models to generative metrics in which each syllable is weighted and assessed by its impact on, and the affect of, other syllables

within the line. For instance a change of stress in one syllable may affect the stress of others. There is, however, very little work on the identification of the rhyme structure to the verse.

Automated analysis of poetic rhyme should be able to identify all rhyme structures within the verse. This includes alliteration, internal rhyme (where the rhyme pair occur within the same line or a word within the line rhymes with a word at the end of another line) and end rhyme (where the last word on a line rhymes with the last word on another). It should also be capable of matching the various rhyme types and patterns as outlined in Table 1.1 and Table 1.2 below adapted from Fry (2005, pp168)

| **Rhyme type** | **Description** | **Example** |
|---|---|---|
| Rich Rhyme | Where the same word is used but has a different meaning or a different word is rhymed with another with an identical sound | Red rose/He rose<br>My nose/ she knows |
| Eye rhyme | Where a similar word is used that looks the same but does not rhyme phonetically | Fool/ wool<br>Heard/ beard |
| Masculine rhyme | Where the last syllable is stressed and the rhyme occurs on the last syllable | Box/ frocks<br>Spite/ tonight |
| Feminine rhyme | Where the last syllable is unstressed and the rhyme occurs on the previous syllable | Breathing/ seething<br>Relation/ nation |
| Assonance | Where the rhyme occurs on the vowel sound | Pit/ kiss<br>Mean/ dream |
| Consonance | Where the rhyme occurs on the consonant sound | Coils/ gulls<br>Coils/ cools |

***Table 1.1 – Rhyme Types***

| Rhyme Pattern | Description | Example |
|---|---|---|
| Couplet | Where the following line rhymes with the first<br>Pattern (AA) | Humpty Dumpty sat on a **wall**,<br>Humpty Dumpty had a great **fall**, |
| Triplet | Where three successive lines end with the same rhyme<br>Pattern (AAA) | What Flocks of Critiques hover here **today**,<br>As vultures wait on Armies for their **prey**,<br>All gaping for the carcass of a **Play**!<br><br>John Dryden: Prologue to 'All for love' |
| Cross Rhyme | Where the alternate lines rhyme<br>Pattern (ABAB) | What need you, being come to **sense**,<br>But fumble in a greasy *till*<br>And add the halfpence to the **pence**<br>And prayer to shivering prayer, *until*<br><br>W.B. Yeats: 'September 1913' |
| Envelope Rhyme | Where a rhyme couplet is braced between two rhyming pairs<br>Pattern (ABBA) | Much have I travell'd in the realms of **gold**<br>And many goodly states and kingdoms *seen*;<br>Round many western islands have I *been*<br>Which bards in fealty to Apollo **hold**.<br><br>John Keats: 'On first Looking into Chapman's Homer' |

***Table 1.2 – Rhyme patterns***

Matching rhyme requires knowledge of the phonetic pronunciation of the component syllables of the verse and also the semantic structure of the verse. Identification of the phonetic pronunciation of a word itself present challenges as the same word can have multiple meanings, for example, *bass* rhymes with *lass* if it is a fish but with *base* if it is a musically tone. The pronunciation may also change depending on the word usage as a noun or a verb, for example, *read* rhymes with *red* if used as a noun 'he is well read' but with *reed* if used as a verb 'he likes to read'. These challenges have been raised by Plamondon (2006, pp13) and Cole et al (1997, pp178).

Some English verse is 'blank verse' and does not rhyme (Fry, 2005 pp123). However where a rhyme is used, the rhyme structure is important as it affects the stress applied to syllables in the lines of verse and therefore has an affect on the metre. Hascall (1969) posited the rule that when a word is rhymed, the syllable, which carries the rhyme, is

considered to have full stress, whether or not it has linguistic stress. Alliteration, where successive words have the same letter i.e. ‘big black box’, also has this affect on stress (Plamondon, 2006). Alliteration is also known as ‘Head Rhyme’.

Current approaches for rhyme identification such as Plamondon (2006) and Adams and Birnbaum (1996) only identify the end rhyme to identify the rhyme pattern; no attempt is made to determine the internal rhyme structure or alliteration. Plamondon (op cit.) stresses the need for the incorporation of a phonetic dictionary as it would enhance the computer’s ability to identify rhythmic effects, such as slight pauses between words that are not signalled by punctuation and it would allow the computer to identify alliteration.

The current rhyme matching used by Plamondon is achieved by a rhyme pairing lookup table. This requires manual intervention for each new word found. These deficiencies; identification only of end rhyme and a manually maintained rhyme pairing lookup table, makes this implementation unsustainable as an approach towards a fully automated system. A more desirable approach to allow for full automation would be a rule based identification of the rhyme based on changeable phonetic and grammatical rules that would allow the application to adapt by the addition of new rules to correctly identify the rhyme pairings if they exist or to identify the poem as blank verse.

The current approaches to rhyme identification and classification therefore have two drawbacks as described above. The first is the use of rhyming tables that require manual checking and update. The second drawback is that currently only the end rhyme pattern is identified.

The use of a phonetic dictionary to address the problem of rhyme identification in verse analysis also has some obvious problems. One of these problems is that a word may not exist in the dictionary and an alternative way of identifying the words structure may

need to be found. The word may also exist in multiple forms as with the example of '*bass*' provided earlier. Even when a word is contained in the dictionary as a single entry the usage and pronunciation of the word may have altered over the centuries since the poem was written. Russell (1971) highlights the difficulty of dialect. Different authors write in different dialects either due to location or time and so there may be many variations of pronunciations for a single word. Fry (2005) also provides the example that originally '*love*' and '*prove*' may once have been true rhyme as they appear frequently in poetry of the Elizabethan era. Now however it is reduced to an eye rhyme.

The nature of poetry and rhyme itself also poses some problems. Fabb (1998) identifies issues with alliteration and states it can break over two lines and have other unstressed words between, while Hayward (1996) states that the affect of alliteration must be considered. Both Plamondon (2006) and Robey (1993) mention the problem of rhyme being a subjective nature with Robey (op cit.) maintaining that because of this a 'substantial degree' of manual intervention will always be required.

Once poetic structure is identified it is necessary to store that information in a reusable fashion. Sproat et al (1997) argue that in order to be able to make most efficient use of computers in the processing of online text, it is necessary to have mechanisms for marking those features of the text that are deemed to be salient. To facilitate reuse and storage of the forms identified it is generally accepted to save this information with the poem within a single file using a generic format to 'mark-up' the features. This marked up text version of the poem could then be used as input for other applications such as text-to-speech applications.

Taking all the issues discussed above it is obvious that an approach to rhyme identification is needed therefore to address the issues of:

- Automatically identifying all rhyme types
- Reduce manual intervention required
- Producing the output in a reusable form by incorporating it in a mark up language

To identify rhyme types automatically requires an ability to check the pronunciation for each word. Any method of identification should require as little manual intervention as possible to provide this automatic identification of rhyme. Therefore a system other than rhyme pair tables is required because this requires constant manual intervention for every new pair found. The use of a phonetic dictionary would allow the identification of words listed within it; however other methods would be required to assist in the identification of unlisted words. An approach using a phonetic dictionary and incorporating pronunciation rules would facilitate the identification of rhymes. A single rule would most likely apply to multiple instances of similar rhyme structures or pairs rather than a single pair. This would reduce the overall intervention required to just require the establishment of those rules rather than identifying every single possible rhyme pair.

### 1.2 Aims and objectives of the research project

The aim of this project is to develop and evaluate an algorithm for the determination of rhyme using phonetic dictionary and rules. This algorithm will be used to identify all rhyme types as outlined in Table 1.1 and to use a mark-up language to show the rhyme

scheme. The benefit of a rules approach is that it may be refined and improved to accommodate irregularities. As these irregularities are identified new rules could be modelled and added and therefore allow for better identification of rhyme types rather than manually inputting data as used by the current approaches. As each new rule is added it could be assessed for its suitability in identifying the irregularities.

The project will be confined in scope to English verse primarily from the late Nineteenth and early Twentieth Century. This range was chosen as it allows selection of a variety of poetry encompassing classic rhyming schemes, sprung rhyme and blank verse and is modern enough to limit problems of dialect or archaic language.

Rules will be added to the system that incorporates some basic rhyme determination approaches and basic prosody rules such as those outlined in Table 1.3 below.

| Rule | Description |
|---|---|
| Use phonetic dictionary | The core tenet of the application. This rule applies phonetic dictionary pronunciation to the poem to find rhyme matches |
| Use phonetic string matching | An augmentation rule that would allow unknown words (words contained in the poem but not listed in the dictionary) to be matched using edit distance or basic pronunciation methods such as Editex or SAMPA |
| Use scansion | End rhymes are considered ‘masculine’ rhymes when the rhyme is matched on the stressed syllable, otherwise it is a ‘feminine’ rhyme. Using the poems scansion to determine syllable stress would allow these two rhyme types to be distinguished |
| Check for eye rhyme | An ‘eye’ rhyme pair are two words that look like they should rhyme but don’t i.e. ‘food’ and ‘good’ |
| Distinguish rich rhyme | Rich rhyme is a rhyme where the same word is used. The context may mean that the usage and semantics are different but the word is the same. An example of this would be the word ‘down’ this may be the feathered ‘down’ of a bird or the direction ‘down’ or even the Irish county ‘Down’. Phonetically these would normally show as full rhyme. |
| Check internal rhyme | The poems rhyming pattern is normally confined to end rhyme i.e. the last syllables of two or more lines. This rule would check the internal words in a line to identify internal rhyme |
| Check alliteration | Alliteration would not normally be considered in a phonetic match for rhyme. This rule would allow the application to check within a line for examples of alliteration (‘head’ rhyme) |
| Partial rhyme | Partial rhyme consists of assonance, where the rhyme occurs on the vowel sound; for example ‘size’ and ‘five’, or consonance, where the rhyme occurs on the consonant sound; for example ‘one’ and ‘down’. |

***Table 1.3 - Prosody rhyme rules***

By developing a rule based algorithm and analysing the results of the output at various points I will be able to determine the benefit of rules processing in determination of the rhyme classification and identification. The aim of the project is therefore to validate the question: “Can the application of a rule based algorithm assist in the automatic determination of rhyme structure”.

I will attempt to validate this by achieving the project objectives set out below

1. Develop the algorithm for rule processing.
2. Modelling rules to assist the identification of rhyme.
3. Analyse the efficiency and accuracy of rhyme determination by comparison against a benchmark of the poem that has had its rhyme structure manually identified.
4. Analyse the effect and impact of these rules by testing a corpus of poetry with particular rules turned on or off.
5. Output the poem in a reusable format in a standardised mark up language.

The analysis of the accuracy of the rhyme determination and the effect and impact of the addition of new rules will allow an assessment to be made of the suitability of this approach to the problem of natural language processing for the determination of poetic rhyming schemes. Comparisons can be drawn from the results of this research to both the existing database approach and to the impact of using combinations of rules to identify rhyme patterns. The success of rhyme identification can be assessed by comparing the number of rhyme pairs correctly matched and identifying those rhyme pairs missed or incorrectly matched in order to assess the cause of the error.

The application may have some use in the classroom as a tool to assist scholars in analysing and criticising poetry. Plamondon (2005) and Blake (2005) both describe teaching potential for their applications that are similar in usage to this application. The application would have the dual benefit of allowing students to automatically identify the rhyming scheme of the poem and also to see the effect of using different rules has

on the identified rhyme scheme. By applying certain rules it would be possible to see how the poem is structured by its rhyme pattern and may give the student an insight into why the poet has arranged the poem in its final form.

### 1.3 Overview of the dissertation

This dissertation is divided into six sections as outlined below. Each section follows from the previous section and so should not necessarily be read in isolation.

Chapter 1 Introduction – This section.

Chapter 2 Literature Review – Describes the current body of knowledge.

Chapter 3 Research Methods – Describes the methods and approach used for this research project.

Chapter 4 Software Overview – This section gives a brief outline to the software developed for this project and the pertinent aspects for the research.

Chapter 5 Results – The results of the research.

Chapter 6 Conclusions – Describes the conclusions that have been drawn from the analysis of the results.

# Chapter 2 Literature Review

## 2.1 Introduction

Computer analysis has a long association with the field of the Humanities dating back to 1949 when Father Roberto Buso began a concordance of the works of St Thomas Aquinas (Hockey 2000 pp5). It use has long been associated with analysing texts for concordance and word counts (Hockey, 2000 pp 49; Morgan, 1991; Robey, 1990; Russell, 1971) and syllable counts (Robey, 1999). The possibilities for computer analysis however extend beyond these common areas of functionality. This section describes a review of the current body of work on the subject of the automated analysis of poetry and the description of that analysis in a described mark-up language. This section is divided into two subsections. The first section 'Verse Analysis' describes the current work in the area of computer analysis of poetry. It also deals with the current work available on the literature, highlighting the difficulties of automated analysis and some approaches to overcoming these difficulties and the linguistic rules that must be considered. The second section 'Mark-up and Reuse' outlines the review on the use of a mark-up language to describe the elements of the poem identified during the analysis as a means to store the information in a platform and application independent format.

## 2.2 Verse Analysis

Several approaches to computer analysis of poetic verse are described in the literature. Robey (1993) describes the use of the SPITBOL programming language to develop an interactive application to analyse Dante's 'Divine Comedy' for accent and syllable count in the original Italian. The purpose of this application is to allow the scanning of the text metrically. Hayward (1996) applied a neural network connectionist model to the

problems associated with generative metrics. Hayward (op. cit.) applies a parallel distributed processing model in which information is affected by its interaction with other associated units. In this case the units are syllables in a line of verse. The stress accorded any one unit affects the stress of the other units. By applying multiple iterative processing runs, during which each unit is examined in relation to the others, the application achieves a 'steady state'. Once this state is reached the line can be examined to see if it may be considered metrical according to the principles of generative metrics. Smolinsky and Sokoloff (2006) use phonetic highlighting in a user interface to describe how the phonetic and phonological structure contributes to its meaning and power. Their application, the 'Pattern-Finder', converts text to a phonetic description alphabet (an alphabet that uses phoneme pronunciation characters to spell the word phonetically). This phonetic transcription is then analysed for phonetic features, these are then presented back to the operator through the interface. Adams and Birnbaum (1996) use an application written in the C programming language to extract rhyme schemes for Russian poetry (although the aim of the paper is to evaluate the difference in perspectives to computing between a humanities scholar and a Computer Scientist). The application uses Russian pronunciation rules to identify the phonetic character for each word and compares them for rhyme pairings. Plamondon (2006) uses the Visual Basic .NET programming language to identify the scansion and basic rhyme pattern of English verse. The application's purpose is to identify the dominant metre of the poem and its end rhyme pattern.

Although not all the research mentioned above approach the problem of rhyme identification these approaches all contain some aspect of the work as described in this paper, such as use of phonemes or application of rules. Of the works that do offer some analysis of the poems rhyme structure the analysis is only partially complete.

### 2.2.1 Scansion and rhyme

A poem's scansion and rhyme are interconnected in two ways. The scansion of a poem denotes where the rhyme may fall as full rhyme occurs only on the stressed syllables of a word or word group (Plamondon, 2006; Adams and Birnbaum, 1996). While partial rhyme may occur on unstressed syllables the rhyme pattern of a poem is more normally associated or identified by the full rhymes. Secondly, the rhyme itself may affect the scansion of the poem. As mentioned previously, Hascall (1969) proposes that when a word rhymes the syllable should be considered as carrying full stress even where it has no linguistic stress. Therefore it is important that the stress associated with syllables is considered when identifying the rhyme structure.

As the aim of this research is primarily centred on the identification of rhyme I will not be identifying the scansion automatically. The identification of the scansion has already been documented in the respective papers of Plamondon (2006), Adams and Birnbaum (1996) and Robey (1993). Many previous commentators have used pre-inputted information to identify or to assist in the determination of the scansion. Plamondon uses a database of words whose syllable structure is identified while Hayward assumes the poem to be in iambic pentameter. Adams and Birnbaum required that the stress be added manually.

Rhyme is identified by both Plamondon and by Adams and Birnbaum. However in both cases only the end rhyme pattern is identified. Alliteration is not identified in any of these works but yet due to its affect on stress it is required as manual input by Hayward. Rhyme's impact on the stress given to the vocalisation of the poem is vital to its reading

but yet is only partially identified in any of these approaches. The approach taken by Plamondon is to identify rhyme pairs and store these pairings in a database table. As the number identified rhyme pairs grows the efficacy of the application increases. The rhyme pattern is identified by testing a few options with the known rhyme pairs and then deciding which is most probable. Plamondon notes that this approach has some trouble distinguishing ABCB quatrains from ABAB quatrains (please refer to Table 1.2 for a description of rhyme quatrains).

The approach to be used for this research will be to identify the rhyme by applying a series of rules. The aim of this is to determine if the application of rules can identify rhyme pairs so that a single rule or combination of rules can automatically detect the rhyme of multiple combinations of words. A rules based approach has been used before however it has been primarily used for the identification of metrical stress (Robey, 1993; Hayward, 1996). Some approaches have however used rules for the identification of rhyme and other word structures. Adams and Birnbaum (1996) use a set of pronunciation rules to determine the phonetic pronunciation of the word from its orthographic representation and then apply a series of rhyming rules to assess (or weight) the potential rhyme.

The rules used in the rhyme identification will be of two particular types. The first will be processing rules; an example of which is “use the word’s stressed syllables to assist in rhyme determination”. The second type will be rhyme type search instruction rules; “search for internal rhyme” or “identify alliteration”. Russell (1971) describes some basic pronunciation rules but does not incorporate them into a computer application. Similarly Fabb (1999) lists some rules of alliteration and its affect on verse but only from a linguistic rather than an algorithmic viewpoint, while Bauschatz (2003) outlines

rules of rhyme and consonants. These rules will have to be modelled as an algorithm in the software application.

The use of rules will allow the identification of the vast majority of rhyme. It should be noted that, despite his use of rules, Robey (1993) warns that although it is possible to generate rules and exceptions to those rules, a substantial degree of manual intervention may still be required. Robey's rules however were for the automatic identification of stress. The identification of metric stress can be more subjective than the rules for rhyme.

Ideally any application should be capable of identifying rhyme patterns without any manual intervention and, where this intervention is unavoidable, it should be minimised and non repetitive. The only way to do this is to 'teach' the application so that it may identify different patterns and word matches rather than requiring manual input for each new occurrence. Hayward's work requires the manual identification of alliteration and also its weighting input required for each processing run. In fact each syllable requires six types of weighting input (Hayward, 1991). This is a sizeable constraint to analysing a large corpus of poetry. A rudimentary attempt at teaching the application is made by Plamondon by prompting the user to identify unknown rhyme pairs or confirm 'guessed' pairs and adding this to a database table. While this allows the application to correctly identify this pairing in the future it is limited to this one particular word pair. Thus the application has been 'taught' in only the most limited sense. Adams and Birnbaum approach is to use certain rhyming rules to identify possible rhymes. This increases the teach-ability of the application as a single rule may correctly identify numerous rhyming pairs and so minimises the need for human interaction. Their paper is, however, aimed at the identification of rhyme in Russian verse and so would not

necessarily apply equally to English verse and no indication of the effect of these rules is given.

### *2.2.2* Use of phonetics

The identification of rhyme depends on the identification of the phonetic structure of the word. A primary source of reference for this information would be the use of a phonetic dictionary. None of the approaches mentioned above use a phonetic dictionary to assist in the identification of syllables or rhyme. Instead Hayward avoids the use of phonetics completely and uses manual input to determine the syllable structure, Similarly Plamondon avoids phonetics and uses a database look up table with each word's syllabic count identified (using an alternative syllabic division table where more than one pronunciation exists). Smolinsky and Sokoloff use initial input with a look up and transcription to SAMPA (Speech Assessment Methods Phonetic Alphabet), a machine-readable phonetic alphabet. Adams and Birnbaum use an algorithm for determining the word accent from basic rules of Russian pronunciation.

Although stating that the use of accent-marked electronic dictionaries would assist the processing Robey argues that there are 'powerful reasons' against the use of wordlists or electronic dictionaries. He states that in Italian and other languages the word function and position affects the stress applied and also the accent may be affected by the constraints of metrical conventions. Plamondon on the other hand says that the use of a phonetic dictionary would enhance the computers ability to identify rhythmic stress and alliteration and Adams and Birnbaum stress that certain idiosyncrasies could not be accommodated in their pronunciation algorithm and list three issues identified during the course of their work that would require a lexical look up dictionary to resolve.

This application will use a phonetic dictionary as the primary source for the detection of rhyme matches. In many instances the phonemes in the dictionary use two characters to denote the sound. For instance the Carnegie Mellon University (CMU, 1998) dictionary uses the double character ‘UH’ to identify the sound of the double ‘o’ vowel sound in ‘hood’. During processing it is easier for any application to match using single characters. The phoneme characters used in the dictionary to identify the sound structure will be mapped to individual alphanumerical characters using a mapping table. This will be done so as to facilitate easier processing of the phonetically spelt words.

The obvious and unavoidable drawback to using a phonetic dictionary is the fact that it may not contain the required word. When the word is not available in the dictionary it will be necessary to determine its phonetic structure using alternative methods. A process of mapping words using the SAMPA (2005) phonetic alphabet is described by Smolinsky and Sokoloff (2006). This process however is not suited for automation because it requires some knowledge of the word sound before the appropriate symbol can be assigned to that phoneme. Other methods exist utilising direct character-to-character translation to match sounds. One such method is the ‘Editex’ method described by Zobel and Dart (1996). In this method the word is first processed by substituting known substrings at the start, end and middle of the word to produce a more phonetic spelling of the word. Then the first character is retained as a character while the following consonant and vowels are converted to numbers. Adjacent repeating numbers are then eliminated, vowel numbers are then eliminated and the first four characters returned. This four character (one alpha and three numeric characters) representation is then used to represent the word. The translation table is described in Figure 2.1 below:

| Code: | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| Letters: | aeiouy | b p | c k q | d t | l r | m n | g j | f p v | s x z | c s z |

***Figure 2.1 - Editex Phonetic Code (Zoble and Dart 1996)***

This process is designed for the identification for possible word matches allowing for communication, typing and transcription issues, and has one serious impact for the identification of rhyme and that is the removal of the vowel sounds during the processing. Vowel sounds are crucial in the identification of rhyme and so any implementation of a process such as the 'Editex' process would have to be altered to account for this.

### 2.2.3 Phonetic differences

Once the phonetic translation of the word has been produced it is necessary to assess the differences between the phonetic pronunciations of the two words being compared for possible rhyme match. The first step in this process would be to apply regular expression pattern matching as described by Hockey (2000, pp 56-57). Using regular expressions words could be found whose phonemes have a basic match, or in some basic rhyme structures even if the same phonemes exist within the word. Once identified the difference between those words could be assessed using an edit distance algorithm such as the one described by Levenshtein (1966). This algorithm works out the minimum number of edits; insertions, deletions or updates required to change one word to another. The result of this algorithm is the minimum edit cost for this translation. The Levenshtein edit distance algorithm has been used widely since its introduction. Some researchers have developed improvements to this edit distance such as Ristad and Yianilos (1998) 'Learning String-Edit Distance'. Indeed Archer et al

(2006) records an error rate 2.6 times lower with Ristad and Yianilos' 'Learning String-Edit Distance' than with Levenshtein's original algorithm. The minimum edits cost, as a ratio to the total string length would give an indication to the phonetic similarity of the two words. In practise however the algorithm outlined by Ristad and Yianilos does not provide much more benefit when used as single occurrence, 'once only' pass over small items of text such as a poem. The algorithm is also more complex than Levenshtein's. Due to this added complexity without the added benefit the application will implement Levenshtein's original algorithm.

### 2.2.4 Word classes

Another issue with word pronunciation is that some words may have multiple variations on spelling depending on the words usage. For example the phrases, 'He read the book' and 'he likes to read' use different pronunciations of the word 'read'. For detailed analysis of text in the study of Humanities the words are often normalised to their root state (dictionary headings) by a process called lemmatisation (Hockey, 2000 pp 94; Plisson et al, (N/D). This is impractical for the determination of rhyme however as the lemmatised version of the word may not have the same phonetic pronunciation. A better approach would be 'word class tagging' as described by Hockey (2000 pp 94). The word class tagging uses a mark-up language to tag each word with its word class type. This allows for easy identification of an individual word as a noun, verb, pronoun, adverb, preposition etc and its tense (past, present, future, conditional etc). This would require the input to be tagged in advance and a phonetic dictionary that has multiple pronunciations for a word based on its usage and context.

### 2.2.5 Mark-up and Reuse

The identification of rhyme patterns in English verse is not sufficient as the information is not reusable. Of the current approaches to verse analysis Robey (1993), Smolinsky and Sokoloff (2006) and Hayward (1996) do not output the resource in a mark up language. Adams and Birnbaum (1996) produce a report from the application that, although not specified, has the appearance of a HTML mark-up document. Plamondon (2006) stores the output in a database using Microsoft Access however he does state that, if required, the output could be produced as XML. Regardless of the output type none of these papers describe the output format and so the likelihood of reuse with other applications or the use as a searchable resource is exceedingly limited because, for reuse, it must be accompanied by documentation (Giordano, 1995). Once described this output could be used in text to speech applications (Hockey, 2000 pp166) however unless a general standard is applied it will still remain unworkable as the output from every disparate application or manual mark-up would need to be separately handled by the processing application.

What is required is a process to mark up the output of the application so that the results can be used as input for another application or be saved by the user in a permanent method. Warwick et al (2006, pp 228) specify the need for a non-platform specific method of documenting the resource to facilitate reuse where the user may have no knowledge of the original application that created the resource.

The TEI (Text Encoding Initiative) was established by the Association for Computers and the Humanities, the Association for Computational Linguistics, and the Association for Literary and Linguistic Computing to look at issues and solutions to this problem (Hockey, 2000 pp36). It recommendations was to apply Standard Generalised Mark-up

Language (SGML) to encode text verse. The TEI outlined guidelines for the application of SGML and a subset of SGML called eXtensible Mark-up Language (XML). These guidelines as they apply to encoding of verse text were outlined by Chisholm and Robey (1995). Chisholm and Robey (op cit.) however warned that the guideline were 'exceedingly daunting' for all but dedicated experts. A smaller subset of the TEI guidelines entitled 'TEI Lite' was proposed and documented by Burnard and Sperberg-McQueen (2002). This 'lite' version of the XML mark-up guidelines is less cumbersome and unwieldy to use. Unfortunately the 'lite' version of the TEI guidelines does not contain the mark-up specifications required for rhyme. It does not have guidelines for identifying the stress applied to syllables or for the identification of rhyme patterns or the matching of rhyme pairs.

The full guidelines (Text Encoding Initiative Consortium, 2007) are therefore required in order to produce a standardised mark-up for verse. Several approaches to the identification of rhyme are documented in the guidelines and this may be applied as considered appropriate for the verse being analyzed. Using the methods for tagging words, lines and groups of lines as described in the guidelines allow most of the rhyme patterns and structures to be identified in the mark-up text of the verse. The guidelines do not however allow for the identification of alliteration and so new attributes may have to be added to the tags to denote this.

### 2.3 Research question

The objective of the project is to answer the research question: "Can the application of a rule based algorithm assist in the automatic determination of rhyme structure". The existing literature, in particular the work of Adams and Birnbaum (1996), show that it is possible to implement software rules for phonetics. Plamondon (2006) also indicates

that, in his opinion, the use of phonetic dictionaries will assist in the determination of rhyme. The research also allows an indirect comparison to be drawn between existing approaches such as the database rhyme table lookup approach used by Plamondon (2006).

### 2.4 Summary

Of the existing approaches it was encouraging that rules had been previously implemented and used for a similar problem, that of scansion identification, by Adams and Birnbaum (1996). The use of rules allows an alternative approach to database look-up tables. Rhyme identification by phonetic comparison has a lot of advantages over the base word comparison so both the use of the CMU (1998) dictionary and the work in string comparisons based on phonetics by Zobel and Dart (1996) have offered interesting possibilities when applied to this particular issue. This research uses both of these approaches.

# Chapter 3 Research Methods

## 3.1 Introduction

The research method has had two main facets, the first was the data gathering stage during which poems were selected for analysis and a software prototype was developed. The application was built using the results of the literature research for existing approaches as a guide. The literature research determined the programming approach and algorithm as well as helping identify any pitfalls and known issues in relation to parsing poetry and the identification of rhyme. The second facet is the analysis of the results. This has involved the application of analysis procedures and methods to the gathered data to document the results and allow conclusions to be drawn from those results.

The approach used was that first taken by Adams and Birnbaum (1996) and further adapted by Dublin and Birnbaum (2005) for the parsing of Russian poetry but has been extended to include alliteration (head rhyme), internal rhyme as well as end rhyme for English verse. This has involved the creation of a software application prototype to evaluate and test. The actual rhyme matching has been achieved by using string comparisons. Each word may have up to four string representations depending on its existence in the phonetic dictionary, an example of these representations is shown in Table 4.2. If the word exists in the phonetic dictionary then the comparison is based on single character mappings for all phonemes for each phonetic representation of the word in the phonetic dictionary. These phoneme characters, used to identify the pronunciation and syllables, are mapped to the single characters using

an external look-up table held in a plain text file as described in ‘Appendix A – CMU Phoneme translation table’. The use of this character mapping allows for different Phonetic dictionaries to be used. For instance the CMU (1998) phonetic dictionary uses the characters UH to represent the double ‘o’ vowel sound in ‘hood’ whereas Mitton (1986) Phonetic dictionary uses the character U to represent this sound. The different dictionaries can be used interchangeably provided the correct external mapping file is used. This approach is chosen as it is easier to compare words using single characters rather than trying to match where two or more characters must be taken as a single unit.

The application uses the single characters to compare the mapped phonemes using direct string comparison and edit distance algorithms such as the one developed by Levenshtein (1966). If the word is not found in the phonetic dictionary then only the original word and the RhymeX representation (as described in the section ‘4.3 RhymeX phonetic assist’) are used in the algorithm.

The application applies a series of rules as outlined in Table 3.1 and Table 3.2 to assist in the identification of the rhyme patterns and to provide analysis of the effectiveness of applying those rules.

## 3.2 Research Techniques

### 3.2.1 Data sample selection

Sharp et al. (2002, p143) advise that, in choosing sample data, a target population should be chosen that permits ‘interesting conclusions’ to be drawn. To achieve this it

was necessary to select a range of poetry and rhyme styles. For the input to the application a corpus of poetry was chosen. This is primarily from the Nineteenth and early Twentieth Centuries so that the language usage is modern enough to be unaffected by changes in pronunciation and yet not still under copyright. To assess the application over a broad range of styles some of the poetry was specifically selected for their rhyme types.

- Blank verse (non rhyming) poetry of William Carlos Williams.
- Off rhyme of Emily Dickinson.
- Internal Rhyme usage of Edgar Allan Poe.
- Sprung rhyme method of Gerard Manley Hopkins.
- Rhyme patterns of William Wilfred Campbell and A E Housman.

To assess possible impacts of accent and dialect I use the poetry of Robert Burns. For an example of eye rhyme I have also used an excerpt from ‘As You Like It’ (Act 2, scene VII) by William Shakespeare. This is the oldest of the poems I use as it dates from the sixteenth century. The scope of the project is limited to English language verse.

Verification of the poems text has been achieved by comparison from a number of sources. The selected poem texts will be cross referenced against the internet poetry reference sites; Every poet, American Poems, Representative Poetry Online and Elite Skills. This has allowed verification that no transcription errors have occurred. Where any discrepancy between the texts within the references cited is found then the text represented by the majority of sources has been used.

There may be some serious doubts concerning the quality and possible bias of data found on the World Wide Web (Sharp et al. 2002, p164). To offset this I have chosen the four sites listed above rather than limiting the text reference to two corroborating sites to minimise possible corruption of quality of the data caused by the sites obtaining the text from the same reference. I have also chosen the above sites because they are of good repute and therefore less likely to have biased data.

### 3.2.2 Data pre-processing

The corpus of poetry used as input data for this project has been manually parsed to identify the rhyme structures (please refer to 'Appendix B – Rhyme identified for poems used in data input' for these identified rhyme structures). This will provide the benchmark data in order to test the applications efficiency in identifying the patterns and structures.

### 3.2.3 Data processing

The method of collection of the data will be achieved by direct comparison between the benchmarked poems and the output of the application. This manual comparison method is the only suitable method for this type of prototype research project and has been previously used by Plamondon (2006) and Dublin and Birnbaum (2005). Other methods of data gathering such as field observation, archival data or survey/ interview responses (Sharp et al, 2002, pp153-158) cannot be used to collect and compare the data. The application outputs the verse in a mark-up language with the rhymes identified by tagging, this facilitates the identification of missed rhymes. The

application also produces a report in plain text of the number and types of rhymes it has identified. It also reports the overall rhyme structure pattern and provides a confidence rating for this match. The confidence rating is based on the number of lines it has identified the rhyme for, against the total number of lines in the poem.

As well as the data reported through the application for the identified rhymes it was also important to establish what rhymes within the overall rhyme pattern the application failed to identify and to record these failures. Accordingly the applications output was manually compared against the benchmark sample for these omissions. This data and the data from the application report have been collated in a spreadsheet for analysis.

Many observers such as both Plamondon (2006) and Robey (1993) describe that data of this nature is often subjective as words may only rhyme if pronounced with certain accents or dialects or indeed, as Russell (1971) points out, may once have been pure rhyme when originally written but now have different pronunciation. Sharp et al (2002, p 152) take the view that while data may be deemed to be subjective, what is important is that it be demonstrated that the data has been gathered in a fashion that could be repeated by others at the same time and lead to the same results within limits of measurable error. For this project I have adopted this view of Sharp et al (loc. cit.) regarding subjective data. The repeatable nature of these experiments may be demonstrated by the applications of the rules where it is shown how the application of those rules and a phonetic dictionary can identify the rhyme patterns.

For the purposes of this project a number of rules have been implemented to assist in the recognition of rhyme patterns. These rules are listed below in Table 3.1.

| Rule | Description | Specific inclusion or exclusion rules |
|---|---|---|
| A | Check to see if the words are identical | None |
| B | Check if one word is identical to the ending of the other | None |
| C | Check if the words are substrings of each other | None |
| D | Check for identical endings | None |
| E | Check for similarity using Edit Distance | • Both words must have a minimum of four characters |
| F | Check for similarity using a Pattern Match | • Return value must be greater than the length of the smallest string or greater than 3. This implies three consonants or a special character and a consonant |
| G | Check for alliteration by comparing first characters | • Must be a minimum of three characters in both words being compared.<br>• Only compares the first three characters |
| H | Check the words match on stressed phoneme | None |
| I | Check the words on stressed phoneme matches to see if a masculine or feminine rhyme can be determined | • Both must exist in the phonetic dictionary<br>• If last matching phoneme is flagged as having primary or secondary stress then set as masculine<br>• If last matching phoneme is flagged as having no stress then set as feminine |
| J | Check the words for possible assonance/ consonance match. | • Only run test if no other rhyme was found or the rhyme confidence was very low (less than 40%)<br>• If there is one or more vowel match and there are more vowel matches than consonant matches then flag as assonance.<br>• If there are two or more consonant matches and there are more consonant matches than vowel matches then flag as consonance. |

***Table 3.1 - Implemented rules***

As can be seen there are very few exclusion rules implemented. Exclusion rules are predominantly the result of exception identification and would require more research time than that available. The application can be instructed to search for particular rhyme types or to implement certain rules. For each user selected search rule type several implemented rule models may be applied. These are listed in Table 3.2.

| Rule | Description | Uses software rules | Optional software rules |
|---|---|---|---|
| 1 | Try match on basic words | a, b, c, d | e, f |
| 2 | Try match on mapped phonemes | a, b, c, d | e, f |
| 3 | Try match on stressed phonemes | h | |
| 4 | Try RhymeX match | a, b, c, d | e, f |
| 5 | Check for Eye Rhyme (end rhyme only) | a, b, c, d | |
| 6 | Check for masculine and feminine rhyme (end rhyme only) | i | |
| 7 | Check for Internal rhyme | a, b, c, d, g, h | |
| 8 | Check for alliteration | g | |
| 9 | Check for assonance/ consonance (end rhyme only) | j | |

***Table 3.2 - Implemented rules used by user selection***

The application reports these rules based on the user selected criteria being used and the implemented rule used in the match. For instance if the test is using only the method implemented by rule '1' (match on the basic word only) and a rhyme match is found using the implemented rule 'c' (Check if the words are substrings of each other ) then the application will report rule type '1c' found a possible rhyme match. This allows the actual criteria used in the match to be identified when multiple user selected search rules are being used and so a reported rhyme match of '4a' shows the match to be identical strings using the 'RhymeX' representation of the word whereas a reported rhyme match of '2a' shows the match to be identical strings using the phonetic representation.

### 3.2.4 Data analysis technique

Analysis of data is crucial to the project. As this project evaluates the effectiveness of the application of a rule-based algorithm to the determination of the rhyme structures within the verse it is essential that the application produces reports on its processing. The application outputs the confidence level for both the identified overall rhyme and for each possible rhyme type match identified. The algorithm applies multiple rules

for each possible rhyme pair and compares the confidence rating of several rules to select the highest confidence match. This highest confidence match is then used to determine the rhyme pattern and is reported separately in the confidence report. Crucially the reporting may be done for certain rules to determine their effectiveness on the overall confidence level of a rhyme structure match. For each rhyme structure that is identified for a poem therefore the application produces a confidence rating.

The processing of a poem will produce the following data outputs

- The types of rhyme identified and the confidence rating for each possible match are included in the report.
- A separate listing of the highest confidence match found for each rhyme word pair.
- The overall rhyme pattern and its confidence level are documented in the report output of the poem
- A marked up XML formatted version of the poem that is used in the manual validation process comparing the original benchmarked poem with the output version.

This data is stored for analysis in a spreadsheet. The identification of the rhyme types and the corresponding counts are stored in the format displayed below in Table 3.3.

| | Total End Rhyme | Parity | Ratio | Overall Accuracy |
|---|---|---|---|---|
| **Total Rhyme *** | **20** | **0** | | |
| **Found** | 20 | | **01:01** | **1** |
| **False positives** | 18 | | **09:10** | **0.9** |
| **Missed** | 0 | | **-** | **-** |
| | | | | |

***Table 3.3 - Data storage + analysis for identified rhyme types***

All rhyme pairs will be recorded as in Table 3.3 above. This chart defines the matches for the particular rhyme criteria tested for. In the example above the test has been to match the end rhyme (in Cell B1) only. Data for particular rhyme types;

alliteration, masculine, feminine, rich, eye, internal assonance or consonance would be stored in a similar table. The first row shows the number of rhyme pairs in the particular poem (in the example above this is the value ’20’). The number of correct matches found, those missed and the false positives identified are recorded in the following rows. Accuracy is measured as the total divided by the number of matches found having a maximum of 1 if all rhymes are matched and a fractional otherwise. Parity is measured as the number found available minus the total available. Hence a parity of -4 records where four rhyme words were missed (two rhyme pairs). A perfect parity score therefore will be a zero and the worst score will be a negative number equal in absolute value to the total number rhymes. The totals are checked against the totals stated in the application output to verify the count of the individual data items matches.

The data on individual rhyme matches (couplets, triplets etc) is stored and assessed based on the accuracy and the parity rating. Table 3.3 shows both the accuracy and parity ratings for a sample poem for each rhyme type and for the poem overall. This method of recording allows for easy identification of the number of matches found, missed and incorrectly identified and also the comparative performance against the total number of rhymes actually to be found in the poem by means of the accuracy and parity figures.

The application also produces the overall rhyme pattern and a related confidence rating. This is compared against the benchmarked sample. This is recorded in the following format outlined in Table 3.4.

| | **RHYME PATTERN*** <br> * The table uses the standard verse mark-up practise of labelling a rhyme pair with a letter, every new rhyme pair that do not rhyme with a previously identified pair is labelled with a new character. | | | | | | | | | | | | | | | | | | | | **Total lines** | **Total missed** | **Total wrong** | **Report confi-dence** | **Actual rate** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Benchmark | A | A | B | B | C | C | D | D | E | E | F | F | G | G | H | H | I | I | J | J | **20** | **n/a** | **n/a** | **n/a** | **n/a** |
| Analysed Poem | *A* | *A* | *B* | *B* | *C* | *C* | *D* | *D* | *E* | *E* | *F* | *x* | *F* | *F* | *A* | *A* | *B* | *B* | *G* | *G* | **20** | **0** | **n/a** | **93%** | **65%** |
| Adjusted output | A | A | B | B | C | C | D | D | E | E | F | x | F | F | A | A | B | B | J | J | **20** | **0** | **7** | **93%** | **65%** |
| Correct | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | | | | | |
| Note 1: 'x' denotes an rhyme not identified by the application <br> Note 2: '\' is used to denote a line that is 'blank' verse i.e. does not contain a rhyme | | | | | | | | | | | | | | | | | | | | | | | | | |

***Table 3.4 - Data storage + analysis for identified rhyme pattern***

The above table records the actual manually identified rhyme pattern of the poem and the pattern matched by the application. The first row labelled 'Benchmark' shows the manually identified pattern and the row 'Analysed poem' shows the output of the application. The 'Adjusted output' line is necessary to compare the result against the benchmark by allowing for rhymes missed or incorrectly identified by the application. If, for instance, the application misses the 'BB' couplet within a 'ABBACC' envelope rhyme then it would start the 'CC' match as 'BB' and appear as 'AxxABB'. This gives the impression that four rhyme words (two rhyme pairs) are incorrect. For example in Table 3.4 above the 'GG' rhyme pair have been falsely identified as matching the 'F' rhyme. This means that the final pair which should be labelled 'JJ' is labelled 'GG' by the application as it is simply using the next letter for this new pair. The 'Adjusted output' row allows for this and re-labels the pair 'JJ'. This makes comparison much easier. These characters are labels purely to identify the pairing and as such there is no need for them to correspond however it does facilitate easier comparison if these omissions are allowed for and the rhyme label output adjusted.

For each failed or misidentified rhyme pairing the rhyme words is also be recorded to analyse the possible causes of the omission. The cause is expected to be one of the following factors:

- Word is not identified in the phonetic dictionary
- Pronunciation of word cannot be accurately modelled using the string match algorithms RhymeX
- The rhyme was originally wrenched to rhyme and does not naturally rhyme (this will not be deemed a failure)
- The particular word class was not identified for this usage of the word (i.e. ‘he is well read’, ‘I like to read’ use different word classes for ‘read’ with different pronunciation)
- There is no rule modelled to identify the type of rhyme
- The modelled rule failed to identify the rhyme

Due to the subjective nature of rhyme structures there will be a certain amount of quantitative data to be analysed to ensure there is cause for comparison and to rule out rhyme structures that should not be matched. These include rhyme pairings identified manually which must be wrenched in order to rhyme and rhymes identified by localised accentuation that is not reflected in the phonetic dictionary. Every rhyme pair found and those missed are listed for analysis in a spreadsheet. An example of this is shown in Table 3.5 below.

| Correct rhyme pair (line numbers in brackets) | Missed/ incorrect rhyme pair (line numbers in brackets) | Identifying Rule | Correct | Incorrrect | Comments |
|---|---|---|---|---|---|
| Doon (1) Doon (13) | | 1a | 1a | | |
| | Doon (1) me (20) | 4b | | 4b | Phonix removes the trailing 'e' character. M and N belong in the same character group |
| fair (2) care (4) | | 3h | 3h | | |
| birds (3) bird (5) | | 3h | 3h | | |
| birds (3) bird (9) | | 3h | 3h | | |
| bird (5) bird (9) | | 1a | 1a | | |
| | Bough (6) true (8) | MISSED | | MISSED | Accentuated rhyme |

***Table 3.5 - Rhyme pairing analysis***

The rule used to match the word, or its omission if no rule found the match, is summarised in the following format outlined in Table 3.6. This summary table allows for easy analysis of the rules reporting the highest confidence match for the rhyme pair and also those rules providing the false positives.

| | RULE MATCHING | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *1a* | *1b* | *1c* | *1d* | *1e* | *1f* | *2a* | *2b* | *2c* | *2d* | *2e* | *2f* | *3h* | *4a* | *4b* | *4c* | *4d* | *4e* | *4f* | *5a* | *5b* | *5c* | *5d* | *6i* | *7a* | *7b* | *7c* | *7d* | *7g* | *7h* | *8g* | *9j* | *MISSED* |
| **Correct** | 3 | | | | | | | | | | | | 6 | | | | | | | | | | | | | | | | | | | | | |
| **Incorrect** | | | | | | | | | | | | | | | 4 | 2 | | | | | | | | | | | | | | | | | | 1 |

***Table 3.6 - Rule matching summary***

This list provides a breakdown of the rules that were used to match the rhyme pairs. For correct matches the maximum amount for any one rule is the number of rhyme pairs in the poem. The total number of all correct matches by rule cannot exceed the total number of rhyme pairs. For incorrect matches there is no upper limit however it is useful to identify the rules that create the most incorrect results to truly evaluate the rules usefulness. The most important aspect of the analysis is to determine the accuracy of the match. This will be used in order to determine the answer to the research question. There is no pass or fail rating for this match. Any figure applied to draw such conclusion would be arbitrary and existing approaches do not use a pass or

fail rating. The aim of the project is the analysis of the rule based approach to confirm if it is a valid approach to the problem. It is not intended to develop an application that applies all possible rules for a hundred percent match and therefore a less than perfect match should be expected.

The poems selected for this research are small and so the analysis of the output is manageable. Longer works of poetry could be tested by using a selected portion of the poem or by using sequential sampling methods such as those proposed by Barnard (1946) and described by the Open University (2000, pp 19-20) on a subset of the results.

The configurable nature of the application facilitates the assessment of the impact, if any, of the application of specific rules and their effectiveness in rhyme identification. Direct comparison may be drawn using the output from two runs of the application applying different rules to a poem or by successively enabling new rules to be used in conjunction with those used in the previous test. This quantitative impact can be evaluated by means of the ratio scale as outlined by Sharp et al (2002) with zero being no rhyme pattern identified.

### 3.3 Summary

The sample poems output have been analysed to check the implementation of the rules and their ability to identify the various rhyme types. Perfect matching is not expected, the purpose of this research is to determine if it is possible and feasible to use a rules based algorithm to determine rhyme. The purpose is not to create a perfect rhyme identifier. Analysis consists of both a comparison against the benchmarked

poem and an analysis of the impact of including more rules to assist in the determination of rhyme. This analysis may also allow comparison to be drawn against other approaches to this issue such as the database rhyme match table approach used by Plamondon (2006).

# Chapter 4 Software Overview

## 4.1 Introduction

This section describes the key aspects of the software developed for this research project. No technical descriptions of the coding implementation are described here. The actual code used for this research may be found in Appendix F – Application software.

## 4.2 Processing

The application allows the user to select the rhyme rules to be checked for during processing and to enable or disable the checks using edit distance calculations. Each of the nine rhyme rules to check for can be turned on or off by setting the appropriate configuration setting in the application properties file ‘Analysepoems.properties’. The property to allow the pattern matching (edit distance based checks) rules E and F can also be turned on or off by setting the value to true or false as shown in Figure 4.1 below:

```
#[Rules]
# Apply the following rhyme test rules
# Rule 1 - End + Internal Rhyme - Try match on words
# Rule 2 - End + Internal Rhyme - Try match on mapped phonemes
# Rule 3 - End + Internal Rhyme - Try match on stressed phonemes
# Rule 4 - End + Internal Rhyme - Try RhymeX match
# Rule 5 - Check for Eye Rhyme (End Rhyme Only)
# Rule 6 - Check Masculine and feminine rhyme (End Rhyme Only)
# Rule 7 - Check for Internal rhyme
# Rule 8 - Check for alliteration
# Rule 9 - Check for assonance/ consonance (End Rhyme Only)

apply.default.rule_1   = true
apply.default.rule_2   = true
apply.default.rule_3   = true
apply.default.rule_4   = true
apply.default.rule_5   = false
apply.default.rule_6   = false
apply.default.rule_7   = false
apply.default.rule_8   = false
apply.default.rule_9   = false

# Apply pattern matching rules E (Edit Distance) + F (Vowel Pattern Match)
apply.pattern.matching = true
```

***Figure 4.1 - Property file setting to enable or disable rules***

This is particularly useful during testing to allow for tests to be run for specific matching. The rules implemented are outlined in Table 3.1 and Table 3.2.

The application converts the poem file to XML if it is not already in XML format. Once the poem is in XML format the application processes the file line by line and stores it in collections of object instances. The poem is stored in a hierarchy of objects. The highest level being the ‘Poem’ object containing a collection of ‘Stanza’ objects. The ‘Stanza’ objects contain a collection of ‘Line’ objects that, in turn, contain a collection of ‘Word’ objects. The object hierarchy is shown in Figure 4.2. This is probably more complex than was necessary for this project but allows for far greater flexibility.

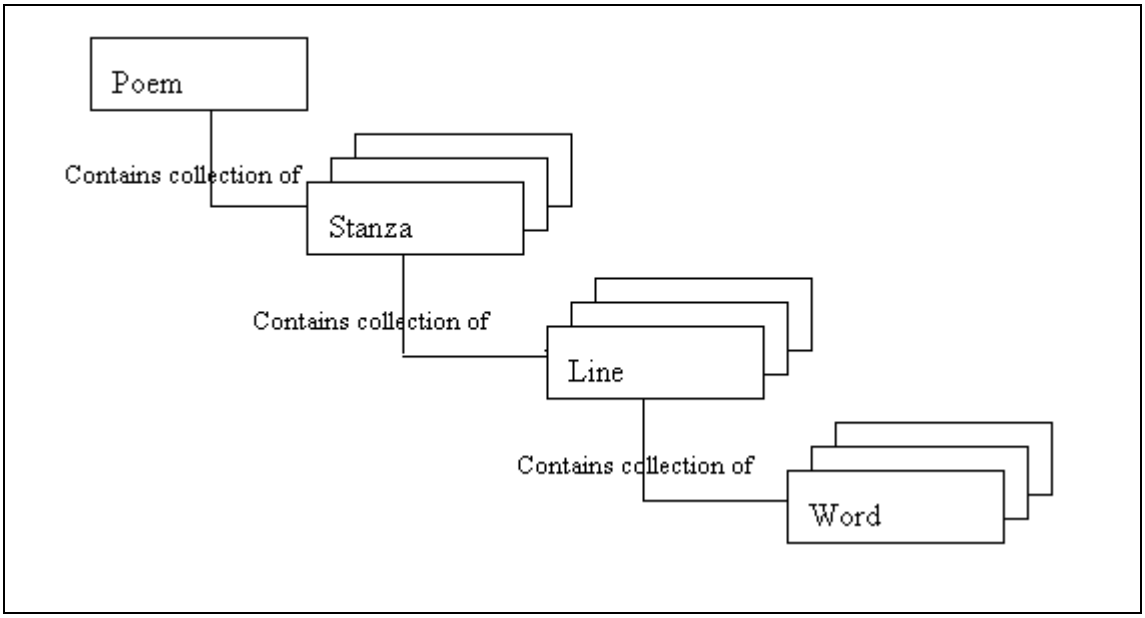


***Figure 4.2 – Poem objects***

When designing any computer application there are some considerations that must be taken. The Association for Computing Machinery (ACM, 1997) lists eight ethical points. These points have been assessed using the scale provided by the Open University (2002 pp 14). Of relevance to this project is the avoidance of 'harm' to others where harm is described as 'injury or negative consequences, such as undesirable loss of information, loss of property, property damage or unwanted environmental impacts' (ACM op cit.). While the software produced for this project has adhered to this it should be noted that, for every test, run two XML files are created. These files are never deleted by the application and, unless manually deleted, could take up disk space. Also no consideration has been given to the length of processing time. The test files used ran in seconds on a 1Ghz PC with 512 MB of RAM. Larger poems on lower specification machines may take considerable more time to process. The veracity of the data, within the limits of the subjective nature of rhyme determination, has also been ensured to avoid negative consequences to the user but it should be remembered that this is subjective data.

The impact of the user interface as outlined by Pressman (2000, Book 3, pp394) has been minimised by running the file from the command line and outputting the resulting poem in XML that may be viewed in any web browser and a report in plain ASCII that is viewable in most text viewers (Note: This report is formatted by using spaces and therefore should be viewed in a text viewer that allows for proportional fonts. MS Notepad should be avoided unless the text is first saved in MS DOS format).

### 4.3 RhymeX phonetic assist

For this project a string matching algorithm has been developed adapting the 'Editex' algorithm developed by Zobel and Dart (1996). The purpose of the 'Editex' algorithm is to identify possible matches from data that may have been corrupted through transcription or typing errors. In order to match the strings the word is first altered for known phoneme constructs. A table of start, middle and end character substrings is consulted and these are mapped to a more phonetic representation, please refer to 'Appendix C – Phonix replacement table' for a full list of these strings. 'Editex' follows the 'Phonix' method of leaving the first letter, removing vowels, removing duplicates and then matching the first three consonants to their letter group. This results in a four character representation of the word (one alpha and three numeric). For the purposes of this project it was necessary to amend this as vowels are crucial to the determination of rhyme and the word ending cannot be truncated as it is the most likely position of the rhyme sound match. I have therefore amended the rules used by 'Editex' and so given it the new name 'RhymeX'. This representation uses the Phonix replacement table and the Editex letter grouping for consonants outlined in Figure 2.1 but retains the vowels and full number of characters in the alphanumeric

representation of the word. The first letter is also converted in RhymeX. Vowels are retained as they are after the Phonix replacement have been applied and the characters H, W and Y are all treated as vowels.

An example of the conversion is given in Figure 4.3 below. The consonants have been matched to groups and the vowels to individual letters.

| | **RhymeX character mapping** | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Code | A | E | I | O | U | H | W | Y | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| Letter | a | e | i | o | u | h | w | y | b, p | c, k, q | d, t | l, r | m, n | g, j | f, v | s, x, z |

***Figure 4.3 - RhymeX character mapping***

**4.4 Phonetic Dictionary and phoneme mapping**

The application allows for a phonetic dictionary to be loaded to determine the actual phonetic representation of the word. If the phonetic dictionary also supplies the vocal stress applied to the phonemes then this may also be used to match the words. The application compares words based on the various string representations on a character-by-character basis. It is therefore difficult to implement this string comparison using the double character representations of phonemes used in phonetic dictionaries. A mapping table is used to convert these double character representations down to single character representations. For instance the phoneme characters UH are used to represent the double ‘O’ sound found in the word ‘hood’. This phoneme representation is matched to the single character ‘u’. A full list of these phoneme mappings may be found in Appendix A – CMU Phoneme translation table.

The application describes a ‘Dictionary’ interface to allow the loading of a phonetic dictionary and its mapping table. For the purposes of this research I used the CMU (Carnegie Mellon University) (1998) Phonetic Dictionary. Other dictionaries could be used by developing a new ‘Dictionary’ class specific to the particular dictionary used.

### 4.5 Confidence

For each rhyme match found the application gives a confidence rating depending upon the rule used to identify the rhyme pair and the extent of the matching criteria. The following table (Table 4.1 – Rhyme confidence rating calculations) describes the confidence calculations for each of the implemented rules.

| Rule | Description | Confidence calculation |
|---|---|---|
| A | First check to see if the words are identical | if they are then confidence = 100% |
| B | Check if one words is identical to the ending of the other | if they are then confidence = 100% |
| C | Check if the words are substrings of each other | if they are then<br>confidence = 100*(length smaller word / length larger word)% |
| D | Check for identical endings | if 2 or more chars match then<br>confidence = 100*(number of matching chars / length smaller word)% |
| E | Check for similarity using Edit Distance | if<br>cost-diff in length <= minlength / 4<br>then<br>confidence = (70*(cost-diff))/minlength%<br><br>cost = Edit distance calculated cost<br>diff = difference in length of the two words<br>minlength = length of smallest words<br><br>**Note**: No measure for how many edits prevent the possiblility of rhyme<br>A minor change i.e Through, trough can completely change the pronunciation. Therefore we must use a low threshold and allow for small word size.<br>Cost should allow for insertion/ deletion required due to difference in lengths and should be scaled against the size of the minimum word<br>As this is inexact with regards to a possible rhyme match the default confidence rate is low (i.e. max 70%) |
| F | Check for similarity using a Pattern Match | if<br>(sim>=minlength \|\| sim >=4)<br>then<br>confidence =<br>if sim>=minlength then<br>70<br>else<br>70*(sim)/minlength;<br><br>sim = similarity index returned from a procedure to check patterns of vowels and consonants<br>minlength = length of smallest word<br><br>**Note**: This checks the strings for special characters (in this case vowel chars) and looks for patterns where vowels and at least one other character match. It weights the result giving a value of 2 for each vowel. Therefore the similarity index returned could |

| | | |
|---|---|---|
| | | be >= length of the word<br>As matching patterns does not equate to matching rhymes the base confidence rate is low (70%) |
| G | Check for alliteration - Check there is a min of 3 chars in the words being compared and then see how many first chars match | if the first chars match then<br>confidence = 25 + (25* number of matches) |
| H | Check the words match on stressed phoneme | if they do then<br>confidence =<br>if stress = primary then 90 else 80 |
| I | Check the words on stressed phoneme matches to see if a masculine or feminine rhyme can be determined | if they do then<br>confidence = original confidence + 1<br><br>**Note:** For this check the words would already have been matched as a rhyme pair |
| J | Check the words for possible assonance/ consonance match. | if they do then<br>confidence = 70*(abs(sim))/maxlength<br><br>sim = similarity index returned from a procedure to check vowel and consonant similarity<br>abs = absolute value (index is negative if assonance positive if consonance)<br>maxlength = length of largest word |

***Table 4.1 – Rhyme confidence rating calculations***

The confidence is used in the final reporting and mark up. A single rhyme pair could be matched on multiple rules with different levels of confidence. The highest confidence match for the pair defines the rhyme type denoted for each word pair.

In addition to each rhyme pair match being assigned a confidence rating, the entire poem is also given a confidence rating based on the overall confidence of the successful end rhyme matches. This calculation is set out in Figure 4.4.

```
If (Number of words found as part of a rhyme pair > zero) then
   Confidence = Sum of all end rhyme confidence / number of words found as part of
                a rhyme pair
Else
   Confidence = 0
```

***Figure 4.4 - Overall confidence rating calculation***

This overall rating is used in the final reporting.

### 4.6 Reporting

In order to evaluate the rules the application gives a full report on the rhyme matching identified and the rules used to do so. The report begins by listing the rhyme rules to test for as selected by the user and the actual software modelled rules this will implement.

For every rhyme match identified a record will be written to the report showing the line for the first word and the word itself (including the RhymeX or Phonetic representation used) and the line number and word matched. The rule used, the level of confidence of the match and the identified rhyme type are also written out. Each word may be matched with another by more than one rule as there will be up to four different representations of the word (the basic word, its RhymeX equivalent, its phonetic representation and its stressed phoneme representation) as shown in Figure 4.5 below. The figure shows the words ‘stone’ and ‘alone’ and their various text representations as outlined in Table 4.2.

```
| Line|              Word| Line|               Word|Rule| Conf.|           Type|
-------------------------------------------------------------------------------
|    3|             stone|    4|              alone|  1d|    60|     Full Rhyme|
|    3|       stone(sTON)|    4|         alone(3LON)|  2d|    50|     Full Rhyme|
|    3|  stone(S T OW1 N)|    4|  alone(AH0 L OW1 N)|  3h|    90|     Full Rhyme|
|    3|       stone(83O5)|    4|         alone(A4O5)|  4d|    50|     Full Rhyme|
```

***Figure 4.5 - Rhyme match reporting***

| Word | RhymeX | Mapped Phoneme | Stressed Phoneme |
|---|---|---|---|
| Stone | 83O5 | sTON | S T OW1 N |
| Alone | A4O5 | 3LON | AH0 L OW1 N |

***Table 4.2 - Sample word and different representations***

For each match found the highest confidence match for that particular word pair is stored. When the application has finished processing the poem, and has reported all matches, it then writes out a table of the highest confidence match for each word pair.

Finally the stanza rhyme pattern is written to the report and an overall confidence given based on the relative confidence of all the identified matched rhymes (lines that did not have a match are excluded from this calculation) the poem is also reported in full with the rhyme pattern included.

### 4.7 Formatted poem output

One of the project objectives is to write the poem back out in a reusable standardised format. For this the application writes out the poem in TEI compliant XML format as specified by the Text Encoding Initiative Consortium (2007). The root node of <TEI> is used to encase the entire poem and then information about the poem itself is written out in a header tag <teiHeader> with the mandatory child nodes as outlined in the example in Figure 4.6 below.

```
<teiHeader>
  <fileDesc>
    <titleStmt>
      <title />
      <author />
      <respStmt>
        <resp>Auto generated from manual text input by</resp>
        <name>Analyse Poems (copyright Frank Kavanagh)</name>
      </respStmt>
    </titleStmt>
    <publicationStmt>Unknown  Auto generated from manual text input</publicationStmt>
    <sourceDesc>Manual input</sourceDesc>
  </fileDesc>
</teiHeader>
```

***Figure 4.6- TEI header block for describing verse***

The poem text is stored within the <text> and <body> child node of the root <TEI> no. The poem is defined in stanzas using line group tags '<lg>'. Each stanza tag contains a number of child nodes using the tag <l>. Each of these child nodes is a single line of verse.

The guidelines currently allow for rhyme patterns to be identified by using the tags described in Table 4.3 below).

| XML Tag | Attribute | Format | Note |
|---|---|---|---|
| <lg> | type | **<lg type="custom-stanza"… <lg>** | The line group tag may use the 'type' attribute to denote custom values or verse constructs. For instance it may contain the word 'couplet' and have two associated child <l> nodes or 'quatrain' and have four child <l> nodes. |
| | rhyme | **<lg type="quatrain" rhyme="abab">…<lg>** | The line group tag may use the 'rhyme' attribute to store the rhyme pattern for the associated groups of lines |
|  | type | **<div type="custom-stanza"… ** | The division tag may use the 'type' attribute to denote custom values or verse constructs. For instance it may contain the word 'couplet' and have two associated child <l> nodes or 'quatrain' and have four child <l> nodes. |
| | rhyme | ** …<lg>** | The division group tag may use the 'rhyme' attribute to store the rhyme pattern for the associated groups of lines |

***Table 4.3 - TEI guidelines for mark-up of rhyme patterns***

For this project the rhyme pattern is shown using the '<lg>' tag. The ' ' tag is not used. The stanza type is also not explicitly identified.

The rhyme pairing or multiple rhyme matches may also be identified using these guidelines (see Table 4.4 below).

| XML Tag | Attribute | Format | Note |
|---|---|---|---|
| <rhyme> | None | **<l>**… did **<rhyme>**growl**</rhyme>** **</l>** **<l>**… to **<rhyme>**howl**</rhyme>** **</l>** | The 'rhyme' tag may be used simply to identify words that have a rhyme within the verse |
| | label | **<l>**… did **<rhyme label="A">**growl**</rhyme>** **</l>** **<l>**… to **<rhyme label="A">**howl**</rhyme>** **</l>** | The 'label' attribute may be used to identify pairs or more of words that rhyme. Each rhyme pair is given the same identifier. This allows a single word to be matched to one or more words with the same identifier |

***Table 4.4 - TEI guidelines for mark-up of rhyme matches***

Using the above methods and combinations all the rhyme patterns and structures can be identified in the mark-up text of the verse.

To denote words that are part of a rhyme pair the TEI tag <rhyme> is used. Currently the TEI convention only contains a ‘label’ attribute for the rhyme tag. To denote the rhyme type and to allow for alliteration I have added two other attributes to the <rhyme> tag. The two attributes added are the ‘type’ attribute to store the rhyme type and the ‘alliteration’ attribute to store the characters that denote the word as alliterating with another. An example of the full output for a line that contains a rhyme pair is shown in Figure 4.7. In the example the word ‘stile’ is marked as alliteration and that the alliteration is based on the characters ‘st’ that match with those of the word ‘stone’. The final word stone is shown as both a rhyme and alliteration, the alliteration is with ‘stile’ as mentioned previously while its rhyme is a full rhyme with the label ‘B’. The same label would be used to indicate the other rhyme in this pair on a different line.

```
<l n="3" real=""> The  aspen over  <rhyme label="" type="" alliteration="st">stile</rhyme>  and
<rhyme label="B" type="Full Rhyme" alliteration="st">stone</rhyme></l>
```

***Figure 4.7 - Amended TEI XML denoting line and rhyme***

An example of the full output for a single poem is provided in Appendix E – Sample XML output for poem.

# Chapter 5 Results

## 5.1 Introduction

To evaluate the approach a series of tests was defined. These tests have a twofold purpose. The first is to assess the rules ability to identify the particular rhyme types and the second is to assess any improvements made by using multiple rules to identify the rhyme. Various poems have been selected for each of the tests so that the identification of a particular rhyme type may be tested. A summary of the tests is outlined in Table 5.1. The full output of all tests for this research may be found in Appendix G –Test outlines, results and analysis.

| Test | Test Description | Rhyme type matching | Poems used in test |
|---|---|---|---|
| 1 | Test the applications ability to match rhyme patterns based on the raw word value only. This test will evaluate the algorithms performance in identifying the end rhyme pattern without the assistance of a phonetic dictionary or string matching computer readable translated text such as the editex variant 'RhymeX' | End Rhyme | Out of Pompeii (by William Wilfred Campbell)<br>Along the field as we came by (by A E Housman)<br>Ye Flowery Banks (Bonie Doon) (by Robert Burns) |
| 2 | Test the applications ability to match rhyme patterns based on the raw word value and the RhymeX assist. This test will evaluate the algorithms performance in identifying the end rhyme pattern without the assistance of a phonetic dictionary but with the string matching computer readable translated text editex variant 'RhymeX'. | End Rhyme | Out of Pompeii (by William Wilfred Campbell)<br>Along the field as we came by (by A E Housman)<br>Ye Flowery Banks (Bonie Doon) (by Robert Burns) |
| 3 | Test the applications ability to match rhyme patterns based on the raw word value, RhymeX assist and phonetic dictionary lookup. This test will evaluate the algorithms performance in identifying the end rhyme pattern with the assistance of both the phonetic dictionary and string matching computer readable translated text such as the editex variant 'RhymeX' | End Rhyme | Out of Pompeii (by William Wilfred Campbell)<br>Along the field as we came by (by A E Housman)<br>Ye Flowery Banks (Bonie Doon) (by Robert Burns) |
| 4 | Test the effect of using edit distance and pattern matching checks for the identification of rhyme | End Rhyme | Out of Pompeii (by William Wilfred Campbell) |

| | | | |
|---|---|---|---|
| | | | Along the field as we came by<br>(by A E Housman) |
| 5 | Test the performance of the application on a blank or off rhyme verse | End Rhyme | It was not death, for I stood up (510)<br>(by Emily Dickinson)<br>Sympathetic Portrait of a Child<br>(by William Carlos Williams) |
| 6 | Test the performance of internal rhyme identification | Internal | No Worst, There is None<br>(by Gerard Manley Hopkins)<br>The Raven (selected verses)<br>(by Edgar Allan Poe) |
| 7 | Test the performance of masculine and feminine rhyme identification | Masculine/ Feminine | No Worst, There is None<br>(by Gerard Manley Hopkins)<br>The Raven (selected verses)<br>(by Edgar Allan Poe) |
| 8 | Test the performance of eye rhyme identification | Eye Rhyme | Miners (selected verses)<br>(by Wilfred Owen)<br>As You Like It (excerpt from Act 2 Scene V11)<br>(by William Shakespeare) |
| 9 | Test the performance of alliteration identification | Alliteration | No Worst, There is None<br>(by Gerard Manley Hopkins)<br>The Raven (selected verses)<br>(by Edgar Allan Poe) |
| 10 | Test the performance of slant rhyme (assonance/ consonance) identification | Assonance/ Consonance | It was not death, for I stood up (510)<br>(by Emily Dickinson)<br>Miners (selected verses)<br>(by Wilfred Owen) |

***Table 5.1 - Test summary***

The tests are sorted into two groups, the first five tests are comparison tests used to evaluate the application of rules to the identification of basic rhyme patterns. The second set of tests, numbered 6 to 10 classify particular rhyme types. The comparison tests numbered 1 to 4 in Table 5.1 above will demonstrate the ability of the application to identify the basic rhyme pattern. At each successive test, between tests 1 and 4, new rules are enabled. This allows the assessment of any impact on rhyme identification of including new search rules. Test 5 is a control test using blank and off rhyme with the same rules enabled as in test 3 to see how the application handles non rhyming verse.

Using the results of these tests it has been possible to assess the rule based algorithms ability to identify all the rhyme types as outlined in Table 1.1. It has also been possible to assess the impact of using combinations of rules to assist the rhyme determination by comparing the results of tests 1 to 4. This has allowed some comparisons to be made with existing approach of using a database lookup.

The test results are documented in the two groupings, comparison tests 1 to 5 and then the classification tests 6 to 10. The poems used in the tests are indicated in Table 5.2, the remainder of this chapter references the poems by their index rather than the full poem title.

| Index | Poem | Author |
|---|---|---|
| Poem 1 | Out of Pompeii | William Wilfred Campbell |
| Poem 2 | A Shropshire Lad XXVI Along the field as we came by | A E Housman |
| Poem 3 | No Worst, There is None | Gerard Manley Hopkins |
| Poem 4 | It was not death, for I stood up (510) | Emily Dickinson |
| Poem 5 | Sympathetic Portrait of a Child | William Carlos Williams |
| Poem 6 | Ye Flowery Banks (Bonie Doon) | Robert Burns |
| Poem 7 | The Raven (selected verses) | Edgar Allan Poe |
| Poem 8 | Miners (selected verses) | Wilfred Owen |
| Poem 9 | As You Like It (excerpt from Act 2 Scene V11) | William Shakespeare |

***Table 5.2 - Poems used as test input***

## 5.2 Results

### 5.2.1 Comparison tests

The purpose of the comparison tests was to test and evaluate the identification of basic rhyme patterns. Tests 1 to 4 gradually increased the number of rules implemented to test for rhyme matches. The tests first used the basic word comparison and then as the test progressed added the ‘RhymeX’ phonetic string matching and the phonetic dictionary matching rules. The list of the rules

implemented by the application is provided in Table 3.1. These rules are selected by configuring the application to use the user-selected rules as outlined in Table 3.2. When the user selects rhyme matching criteria rules to apply, a range of rules actually implemented by the software are used to match that criteria. Often the same implemented rules based on string comparison are used for different match criteria. In these cases the same rule is used but the criteria is to match on different string representations of the word as shown in Table 4.2. The rules selected and the software rules they implement to check for possible rhyme matches are listed in Table 5.3.

| | **User match criteria rules** | **Software rules implemented by user selection** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Match criteria** | **a** | **b** | **c** | **d** | **e** | **f** | **g** | **h** | **i** | **j** |
| Test 1 | Base word only | Y | Y | Y | Y | - | - | - | - | - | - |
| Test 2 | Base word and RhymeX representation | Y | Y | Y | Y | - | - | - | - | - | - |
| Test 3 | Base word, RhymeX, Phoneme and stressed Phoneme representation | Y | Y | Y | Y | - | - | - | Y | - | - |
| Test 4 | As per Test 3 including edit distance pattern matching rules | Y | Y | Y | Y | Y | Y | - | Y | - | - |
| Test 5 | As per Test 3 | Y | Y | Y | Y | - | - | - | Y | - | - |

***Table 5.3 - Comparison tests – rules used***

The results of the first three tests summarised in Figure 5.1 show a steady increase in the number of correctly identified rhyme pairs as new rules were included in the matching criteria. As the tests progress the number of missed rhyme pairs diminish showing the effectiveness of applying more rules to the rhyme match algorithm.

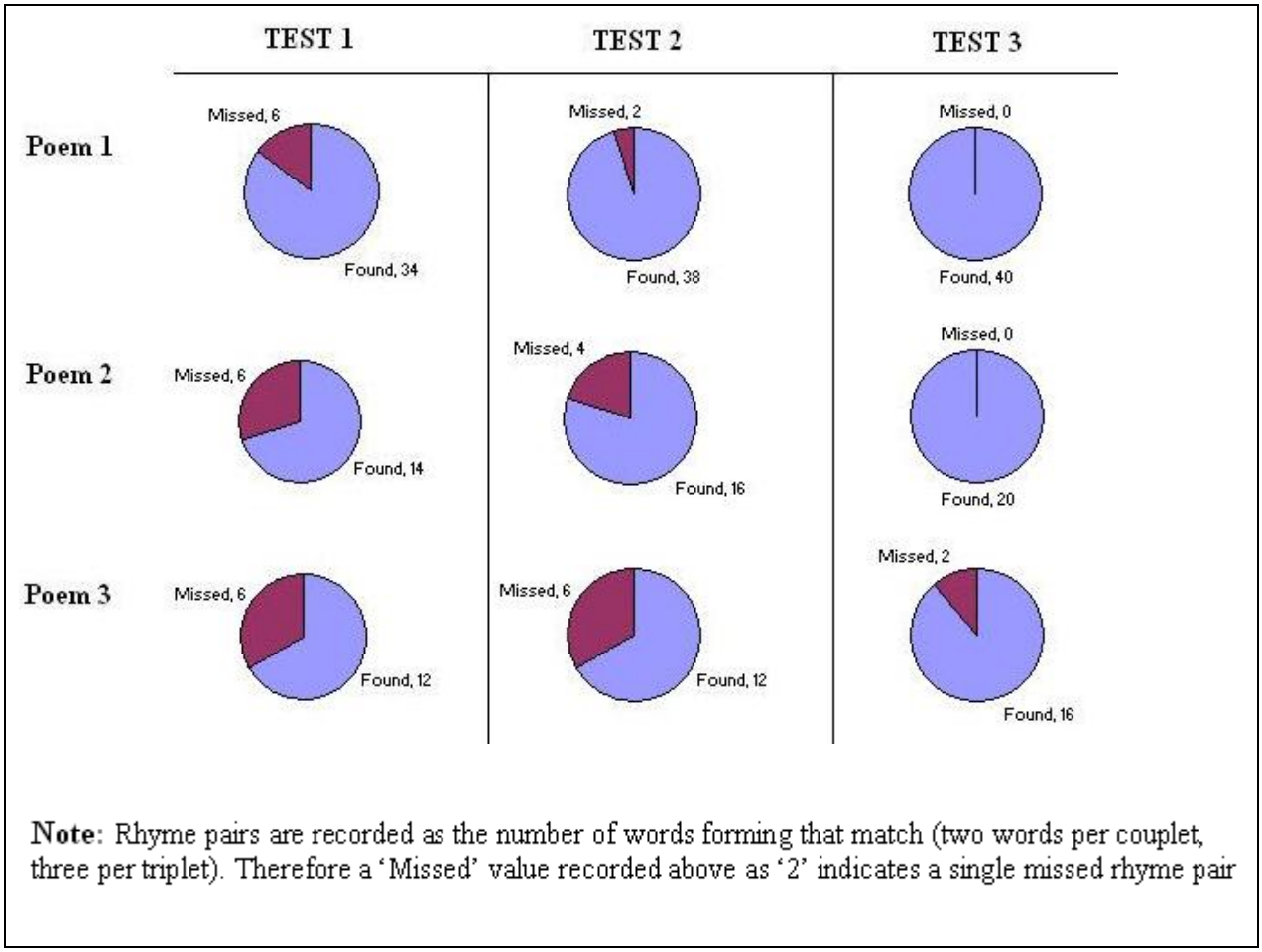


***Figure 5.1 - Comparative results for Tests 1 to 3 (Found and Missed rhymes)***

The increase in the accuracy of the rhyme identification by the addition of new rules is shown in Table 5.4 below.

| Poem number | Test | String representations used (c.f. Table 4.2) | Matched (x out of possible y) | Accuracy | Match Confidence |
|---|---|---|---|---|---|
| 1 | 1 | Word | 34/40 | 0.85 | 71% |
| | 2 | Word, RhymeX | 38/40 | 0.95 | 75% |
| | 3 | Word, RhymeX, Mapped Phoneme, Stressed Phoneme | 40/40 | 1.0 | 87% |

***Table 5.4 - Accuracy ratings for Tests 1 to 3 on Poem 1***

As the above table shows, the accuracy in identification of the rhyme increased with each successive test due to the addition of new rules for rhyme identification. As the different string representation rules were added the accuracy increased from 0.85 in Test 1 to 0.95 in Test 2 and then to a complete match in Test 3. In test 3 all string

representations are used in order to assess if the word matches, these are the basic word, RhymeX, mapped phoneme and stressed phoneme. In all the tests the only unidentified rhyme pairing was in the heavily accentuated poem 6 (‘Ye Flowery Banks’ by Robert Burns) this is the rhyme pairing ‘bough’ on line 6 and ‘true’ on line 8. This is not a true rhyme unless ‘bough’ is pronounced ‘boo’ or ‘true ‘ is pronounced ‘tr-aow’ and so is not deemed a failure of the application.

Although the results of the first test for poem 1 achieved a high accuracy rating of 0.85 the confidence for the matching was low at 71% whereas the confidence rating in test 3 was 85%. This is due to the higher confidence rating applied to the new string representations introduced through test 2 and test 3. Table 5.5 below shows the changes in rule matching and confidence rating for some of the selected rhyme pairs in poem 1 through tests 1, 2 and 3.

| **Correct rhyme pair (line numbers in brackets)** | **Test 1** | | **Test 2** | | **Test 3** | |
|---|---|---|---|---|---|---|
| | **Identifying Rule** | **Confidence rating** | **Identifying Rule** | **Confidence rating** | **Identifying Rule** | **Confidence rating** |
| form (26) storm (28) | 1d | 75 | 4d | 80 | 3h | 90 |
| left (29) bereft (31) | 1d | 75 | 4b | 100 | 4b | 100 |
| wake (33) break (35) | MISSED | | 4d | 66 | 3h | 90 |
| heart (34) apart (36) | 1d | 60 | 4d | 66 | 3h | 90 |

***Table 5.5 – Rule and confidence matching for Poem 1***

The rhyme pair ‘form’ (on line 26) and ‘storm’ (line 28) was initially matched using rule ‘1d’ with a confidence of 75%, in test 2 this confidence rating had been improved by a match using rule ‘4d’ with 80% and in test 3 the test using stressed phonemes rule ‘3h’ was producing a 90% confidence rating. During test 3 all matches were evaluated and reported and only the highest confidence match used to mark up the poem. Overall the benefits of using multiple rules to determine the match and applying a confidence rating to each match resulted in 15 of the originally matched pairs to be matched with new rules returning higher confidence ratings.

As tests one to three progressed the number of false positives increased. This can be seen in Figure 5.2. The number of false positive reported can be higher than the total number of rhymes. This is because the number of possible combinations of potential matches is always higher than the number of actual matches.

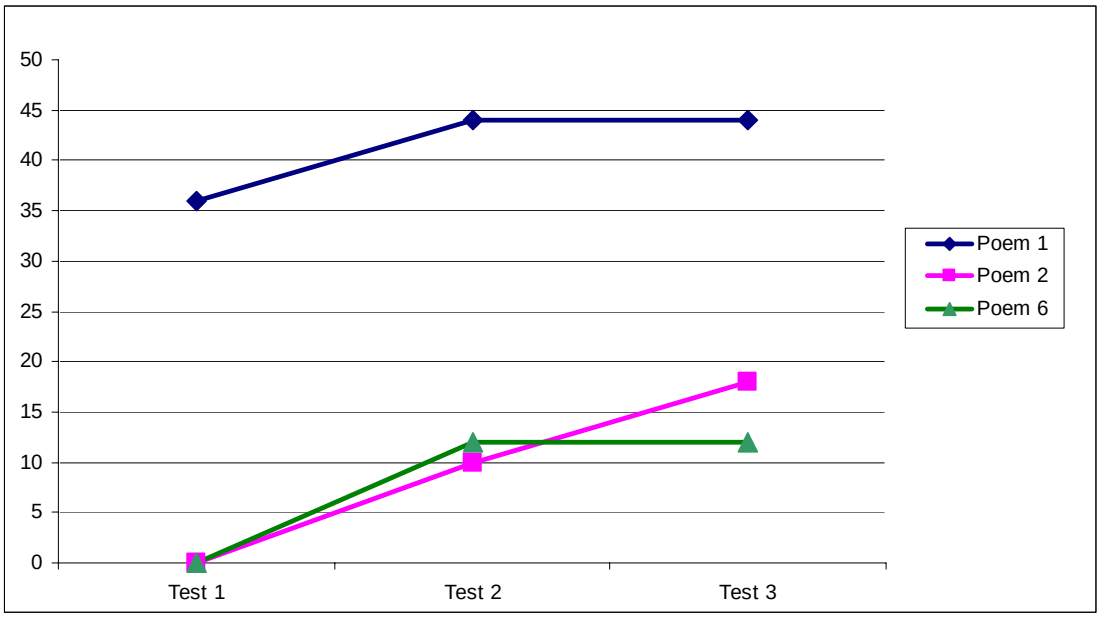


***Figure 5.2 - False positives for Tests 1 to 3***

False positives are always a cause for concern, however, on analysis these are not as bad as the initial raw figures show. All of the false positives in Test 3 for Poems 6 and 12 of the false positives for Poem 2 are based on a match for the word ‘me’. In the RhymeX representation the training ‘e’ is dropped leaving a solitary character ‘m’ and this is then matched as a substring or end character to numerous other words. The confidence rating of the false positives is also much lower as shown in Table 5.6 below:

| Poem Number | Average confidence Correctly matched rhymes | Average confidence False positives | Notes |
|---|---|---|---|
| 1 | 80.4 | 60.3 | |
| 2 | 92.0 | 68.2 | Confidence is high due to high confidence on the matches for the word ‘me’ |
| 3 | 93.3 | 76.0 | Confidence is high due to high confidence on the matches for the word ‘me’ |

***Table 5.6 - Average confidence ratings of correctly and incorrectly matched rhymes***

It is quiet simple to create filter rules to prevent matches on single characters or where the confidence is low. This would reduce or in some cases eliminate the false positives.

Test 4 introduced rules matching based on edit distance methods. The impact of these new rules can be assessed by comparison with the rules used to correctly match rhyme pairs using test 3. The rule matching for each matched pair is represented graphically in Figure 5.3 and Figure 5.4. Comparisons of successful matches used previously in other tables used a value of 2 for a successful rhyme pair.

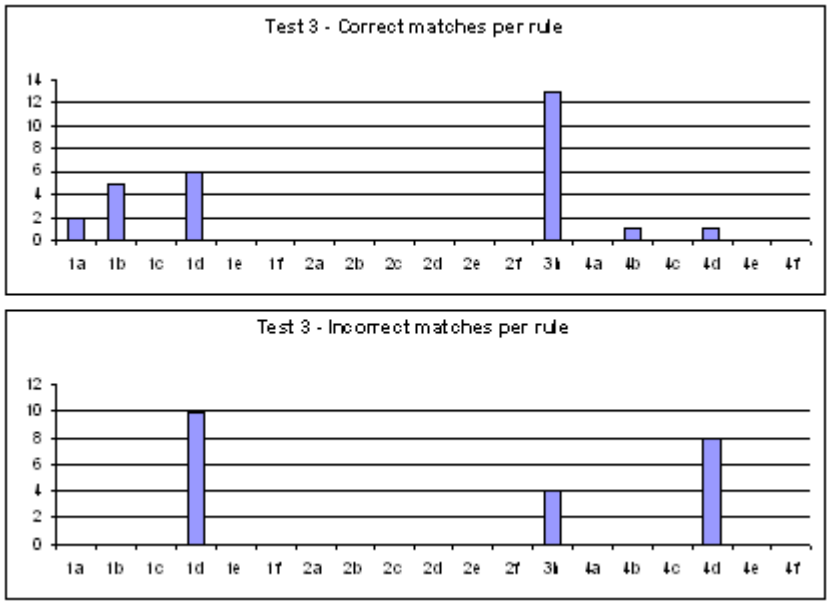


***Figure 5.3 - Test 3 - Poem 1 rules used to match rhymes***

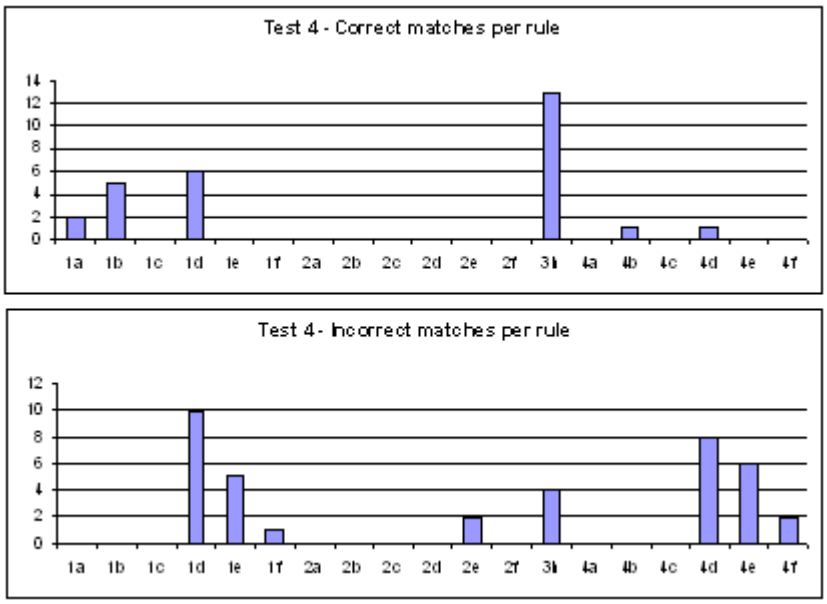


***Figure 5.4 - Test 4 - Poem 1 rules used to match rhymes***

As can be seen from the results shown in Figure 5.3 and Figure 5.4 above the inclusion of the edit distance matching rules (1e, 1f, 2e, 2f, 4e and 4f) provided no benefit in successfully matching the rhymes. No rule was correctly matched by these rules. There are no differences in the correct matching rules between Test 3 and Test 4. They did however greatly increase the number of false positives. As may be seen the rules 1d, 3h and 4d show the same number of incorrect matches between Test 3

and Test 4. However, in Test 4, the edit distance matching rules 1e, 1f, 2e, 4e and 4f have all produced new false positives.

Interestingly Figure 5.3 and Figure 5.4 show that the stressed phoneme tests (3h) produce the most correct matches than the mapped phoneme tests (2a to 2f) that use the phonetic dictionary value. It appears form this that once the stress of the phonemes are considered the accuracy is enhanced compared to using merely the phonetic spelling. The RhymeX representation (test 4a to 4f) produced few correct matches and many false positives. It was however the only representation capable of matching the rhyme pair ‘oblivion (line 30)/ gone (line 32)’. The RhymeX representation is based on phonix/editex string comparisons. These are focused on selecting possible matches allowing for typographic and transcription errors. These methods were only slightly adjusted to produce the RhymeX variant, most notably vowels were left in as these are crucial to rhyme. Using this rule in test 2 did produce better results than the base word alone.

False positives increased as the number of rule criteria were increased as shown in Figure 5.2. These reached a worst-case scenario in test 4 where for poem 1 forty rhyme pairs were shown as false positives. Table 5.7 shows a breakdown of these.

| Identifying Rule | Number of false positives | Average Confidence rating | Reason + Example of missed/ incorrect rhyme pair (line numbers in brackets) |
|---|---|---|---|
| 1d | 10 | 43.5 | Similar ending - two or more of the last letters match.<br>**Example:** bliss (2) nothingness (18); afternoon (5) oblivion (30) |
| 1e | 5 | 17 | Matched using edit distance - cost of change is small compared to length of string.<br>**Example:** forget (14) form (26); bare (21) bereft (31) |
| 1f | 1 | 70 | Matched using an edit distance based pattern match algorithm.<br>**Example:** mark (11) arm (25)<br>The similarity of 'ar' compared to the length of the string produces a high probability match. |
| 2e | 2 | 17 | Matched using edit distance using mapped phoneme representation - cost of change is small compared to length of string.<br>**Example:** bliss (2) oblivion (30); oblivion (30) blush (40) |
| 3h | 4 | 90 | Similar stressed phoneme<br>**Example:** warm (3) form (26); warm (3) storm (28)<br>Normally these would be a rhyme pair (and do in fact rhyme) however 'warm' is used to rhyme with 'arm' in this poem and so would never be manually marked in the rhyme pattern as also rhyming with 'form' and 'storm'. |
| 4d | 8 | 66.6 | Similar ending in RhymeX representation - two or more of the last letters match (RhymeX follows Phonix method of replacing substrings with a more phonetic representation).<br>**Example:** arm (1) form (26); boon (7) gone (32) |
| 4e | 8 | 15.5 | Matched using edit distance using RhymeX representation - cost of change is small compared to length of string.<br>**Example:** press (20) break (35); bare (21) apart (36) |
| 4f | 2 | 70 | Matched using an edit distance based pattern match algorithm.<br>**Example:** came (13) arm (25); name (15) arm (25)<br>The similarity of 'am' and 'arm' compared to the length of the string produces a high probability match. |

***Table 5.7 - False positives by rule for Test 4 Poem 1***

Test 5, the last of the comparison tests, used two examples of off rhyme and blank verse to determine how the application handled the verse. The application reported numerous rhyme pair matches due to false positives as shown in Table 5.8 for similar reasons to those in Table 5.7.

| | Actual rhyme pairs in poem | Found | Missed | False positives |
|---|---|---|---|---|
| **Poem 5** | 0 | 0 | 0 | 58 |

***Table 5.8 - Test 5 - Blank verse control test***

In this instance it was the effect of a line ending with the word ‘me’ matching to so many other words in the poem as a substring of another end word. In particular the RhymeX representation of the word ‘me’ is the single character ‘m’ as discussed earlier. This test demonstrates the need for more refined exclusion and filter rules to prevent these incorrect matches.

### 5.2.2 Classification tests

The algorithm should be able to classify the rhyme pairs found in addition to identifying them. The previous tests concentrated on the identification of rhyme pairs. Tests 6 to 10 defined classification tests to determine the rule based algorithms ability to identify various rhyme types. The rules implemented by them are outlined in Table 5.9

| | User match criteria rules | Software rules implemented by user selection | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Match criteria** | **a** | **b** | **c** | **d** | **e** | **f** | **g** | **h** | **i** | **j** |
| Test 6 | Internal Rhyme using all word representations | Y | Y | Y | Y | Y | Y | - | Y | - | - |
| Test 7 | Masculine and Feminine Rhyme using all word representations | Y | Y | Y | Y | Y | Y | - | Y | Y | - |
| Test 8 | Eye Rhyme using all word representations | Y | Y | Y | Y | Y | Y | - | Y | - | - |
| Test 9 | Alliteration using all word representations | Y | Y | Y | Y | Y | Y | Y | Y | - | - |
| Test 10 | Assonance and Consonance using all word representations | - | - | - | - | - | - | - | - | - | Y |

***Table 5.9 - Classification tests – rules used***

In these classification tests the false positives are ignored in all tests except for test 10. The reasons for the false positives are related to the identification of the rhyme pair

rather than the classification and these have already been discussed in the previous section.

Test 6 was conducted to determine internal rhyme. Two poems were used during this test and the results are summarised in Table 5.10 below. Each rhyme pair is given a value of 2 hence in the table ‘Poem 3’ is given an ‘Actual’ score of 4 this denotes two rhyme pairs.

| | **Actual rhyme pairs in poem** | **Found** | **Missed** | **Accuracy** |
|---|---|---|---|---|
| **Poem 3** | 4 | 4 | 0 | 1.0 |
| **Poem 7** | 28 | 16 | 12 | 0.6 |

***Table 5.10 - Test 6 - Internal rhyme summary results***

For Poem 7 twelve rhymes were missed (six rhyme pairs). These all occurred on different lines (i.e. the word from one line matched an internal word on another). The application currently only searches for internal rhyme within a single line so these missed rhymes are not deemed to be a test failure. If these ‘missed’ rhymes are adjusted to allow for the fact that they are specifically not searched for due to the limitations of the current implementation of the rule then the accuracy would be 1.0 for both tests.

Masculine and feminine rhyme was tested using test 7. These rhyme types are only tested for when the word pair has already been matched as a rhyme and are evaluated only on the basis of stressed phonemes. Table 5.11 shows the rhymes identified and missed.

| | | Actual rhyme pairs in poem | Found | Missed | Accuracy |
|---|---|---|---|---|---|
| **Poem 3** | **Masculine** | 12 | 12 | 0 | 1.0 |
| | **Feminine** | 2 | 0 | 2 | 0.0 |
| **Poem 7** | **Masculine** | 12 | 10 | 2 | 0.83 |
| | **Feminine** | 2 | 2 | 0 | 1.0 |

***Table 5.11 - Test 7 - Masculine and feminine summary results***

The rhyme pairs missed for poem 3 were ‘wring’ (line 2) and ‘comforting’ (line 3) and for poem 7 ‘lore’ (line 2) and ‘evermore’ (line 12). Both of these were due to the stressed vowel in each word having a different stress in the other. The rule matches the phoneme including the stress indicator. As the stress indictor had a different digit no match occurred (i.e. the phoneme AO in ‘lore’ represented as ‘L AO1 R’ is defined as ‘AO1’ identifying primary stress whereas in ‘evermore’ represented as 'EH1 V ER0 M AO2 R' it is defined as ‘AO2’ indicating secondary stress. ‘AO1’ and ‘AO2’ are not matched by the rule). The rule did identify the rhyme classification for all other instances.

The identification of eye rhyme (words that look the same but are pronounced differently i.e. ‘food’ and ‘good’) is examined by Test 8. Eye rhyme can only be identified by the phonetic pronunciation of the word and so the word must exist in the phonetic dictionary. The algorithm identifies eye rhymes by checking if there is an apparent rhyme match on one of the four representations of the string as outlined in Table 4.2. It then checks the phonetic representation to see if the stressed vowels rhyme, if not then it is assumed to be an eye rhyme. Table 5.12 below summarises the results of the test.

| | Actual rhyme pairs in poem | Found | Missed | Accuracy |
|---|---|---|---|---|
| **Poem 8** | 6 | 4 | 2 | 0.66 |
| **Poem 9** | 4 | 0 | 4 | 0.0 |

***Table 5.12 - Test 8 – eye rhyme summary results***

It should be noted that the chart above follows the convention of scoring a rhyme pair as 2, not 1, and so the 'Actual' column reports respectively six and four rhyme words (three and two rhyme pairs) With an accuracy of 0.66 for poem 8 and 0.0 for poem 9 initial results appear to be very poor, however further examination reveals a much better performance by the algorithm. The eye rhyme pairs missed by the algorithm are provided in Table 5.13 below.

| | LINE | WORD | LINE | WORD | RULE | CONF | TYPE |
|---|---|---|---|---|---|---|---|
| Poem 8 | 9 | simmer | 11 | summer | 1d | 66 | FullRhyme |
| | 9 | simmer(siMe) | 11 | summer(s3Me) | 2d | 50 | FullRhyme |
| | 9 | simmer(8I55EAH) | 11 | summer(8U55EAH) | 4d | 71 | FullRhyme |
| | 9 | simmer(SIH1MER0) | 11 | summer(SAH1MER0) | 5 | 67 | EyeRhyme |
| Poem 9 | 1 | wind(WAY1ND) | 2 | unkind(AH0NKAY1ND) | 3h | 90 | FullRhyme |
| | 14 | warp(WA4H1) | 15 | sharp(8HA4H1) | 4d | 80 | FullRhyme |
| | 14 | warp(WAO1RP) | 15 | sharp(SHAA1RP) | 5 | 76 | EyeRhyme |

***Table 5.13 - Test 8 - Rhyme matches identified by algorithm***

Checking the rhyme table it can be seen that in poem 8 the missed eye rhyme 'simmer' and 'summer' was matched with a 67% confidence rate, however a full rhyme match with a confidence rate of 71% was also identified using the 'RhymeX' representation. This resulted in the full rhyme rather than the eye rhyme being reported. The issue here is caused by the confidence reporting and is not a failure to identify eye rhyme. Poem 9 has two eye rhymes 'wind/unkind' and 'warp/sharp'. The 'wind/unkind' pair is missed because the phonetic dictionary has multiple pronunciations for 'wind'; the verb 'wind' (to wind a watch) and the noun 'wind' (the west wind). The verb 'to wind' does rhyme with 'unkind' and so is not determined to

be an eye rhyme. To allow for this the phonetic dictionary would have to differentiate the word classes for each pronunciation and the poem would have to be tagged with word classes. As neither case is true it is impossible to allow for this. The ‘warp/sharp’ was reported as an eye rhyme pair but, again, also as a full rhyme with a higher confidence rating. The algorithm therefore was very accurate in identifying the eye rhymes.

Alliteration was next tested for. The rule contained two tenets for identifying alliteration; the words must start with the same letters and the words must be consecutive. Obviously this basic implementation of the rule would need amending for other examples of alliteration. A summary of the results is provided in Table 5.14 below

| | **Actual rhyme pairs in poem** | **Found** | **Missed** | **Accuracy** |
|---|---|---|---|---|
| **Poem 3** | 20 | 16 | 4 | 0.8 |
| **Poem 7** | 16 | 10 | 6 | 0.6 |

***Table 5.14 - Test 9 – alliteration summary results***

The missed samples from poem 3 were the two alliteration pairs ‘herds-long huddle’ on line five and ‘mind has mountains’ on line nine. These are both missed due to the words ‘long’ and ‘has’ occurring between the alliterated words (hyphenated words are processed as two separate words). The alliteration in poem 7; 'weak and weary' (line 1), 'surcease of sorrow' (line 10), 'rare and radiant' (line 11) was missed due to the intervening words ‘and’ and ‘of’. Adjusting for these the accuracy on both tests was 1.0 (100%). The rule should be amended to ignore these conjunctions.

The final test was used to determine assonance and consonance. The rules for the identification of this form of slant rhyme are perhaps the loosest of all rules. There are no rules of identifying assonance or consonance that can easily be modelled in a

software algorithm. As such the algorithm normally applies the filter that this slant rhyme should only be checked for where no rhyme match has been identified or where the confidence of any identified rhyme match is very small (less than 40%). For the purposes of this test, this filter was removed and so all possible end rhyme pairs were tested for assonance and consonance. This has led to a number of false positives as true rhymes are also included. Table 5.15 below gives a summary of the results.

| | | Actual rhyme pairs in poem | Found | Missed | Accuracy | False positives |
|---|---|---|---|---|---|---|
| **Poem 4** | **Assonance** | 0 | 0 | 0 | n/a | 236 |
| | **Consonance** | 6 | 6 | 0 | 1.0 | 28 |
| **Poem 8** | **Assonance** | 0 | 0 | 0 | n/a | 90 |
| | **Consonance** | 12 | 12 | 0 | 1.0 | 10 |

***Table 5.15 - Test 10 - Slant rhyme summary results***

As mentioned above all filters were turned off for this test so a high number of false positives was expected. For each match the algorithm provides a confidence rating this gives a clearer picture of the matching. Comparing the results using the average confidence matching as shown in Table 5.16 below:

| | | Total matched pairs | Average confidence rating |
|---|---|---|---|
| **Poem 4** | False positives | 132 | 14.6212 |
| | Correct | 3 | 28.6667 |
| | | | |
| **Poem 8** | False positives | 50 | 12.32 |
| | Correct | 6 | 35.3333 |

***Table 5.16 - Test 10 - Average confidence ratings***

This indicates that correct matches achieve a far higher confidence rating and so a simple filter rule would eliminate much of the false positives. The algorithm did correctly identify the true matches correctly and with relatively high confidence ratings for this type of test.

## 5.3 Validation

The tests carried out and the analysis of the results show that a rule based algorithm is capable of both identifying and classifying the rhyme types. The analysis of the comparison tests summarised in Figure 5.1 and Figure 5.2 has shown that rules can be modelled to identify rhyme structure and new rules may be added to it to improve the ability to identify that structure. The classification tests have shown that the rules based approach facilitates the identification of the actual rhyme type.

The identification of rhyme type is not perfected and some tests produce a number of false positives. It must be considered that the purpose of this research was not to produce a software prototype to perfectly match and classify rhyme but merely to determine if it was possible. For this reason the application included very few exclusion rules and this resulted in most of the false positives. It also modelled some edit distance cost based rules for evaluation. The tests outlined in this chapter have shown these to be of little benefit and to produce numerous false positives.

This research has answered the research question posed “Can the application of a rule based algorithm assist in the automatic determination of rhyme structure”. It has been proven to be possible to use such an algorithm to identify rhyme structure. However it should be considered that the relatively few rules implemented during this research also produced a number of false positives. Also it may not be possible to match some heavily accentuated or ‘wrenched’ rhymes.

## 5.4 Summary

The research has proven capable of answering the posed research question. Tests for both identifying rhyme patterns and for the classification of the rhyme type show the

promise of the use of rule-based algorithms. The research also provided many pointers and groundwork for future development. The use of a phonetic dictionary was shown to improve the extent of matching and the accuracy and confidence of the match. In particular the use of stressed phonemes to identify rhyme was very successful. The RhymeX variation of string comparison algorithms also proved to be useful in assisting the identification of rhyme, however it may need further changes to focus on rhyme as its current implementation is perhaps still to closely tied with the phonix/ editex algorithm designed to provide matches for typographic and transcription errors.

Classification of the rhyme types was also shown to be possible and analysis of the results showed that the algorithm was often more successful than initial summary data indicates. Changes to the confidence ratings applied and the addition of some filter and exclusion rules would be required to help fine tune these classification rules.

# Chapter 6 Conclusions

## 6.1 Project review

This research was conducted to assess the use of a rule based approach to the identification of rhyme. The use of a rule based approach facilitates the use of phonetic dictionaries. This assists in the elimination of manually updating rhyme pairs identified in Section 1.2 as a drawback to the existing approach. Both Plamondon (2006) and Adams and Birnbaum (1996) had suggested the use of phonetic dictionaries or on line lexicons to assist in the identification of rhyme.

The use of a rule based approach has advantages over existing methods of rhyme matching that use a database lookup table to find rhyme word pairs (Plamondon, op. cit.). Taking an example from test 6 the poem 'No worst, There is none' contains the following sextet shown in Figure 6.1 below:

O the mind, mind has mountains; cliffs of fall
    Frightful, sheer, no-man-fathomed. Hold them cheap
May who ne'er hung there. Nor does long our small
    Durance deal with that steep or deep. Here! creep,
Wretch, under a comfort serves in a whirlwind: all
    Life death does end and each day dies with sleep.

***Figure 6.1 – 'No worst, there is none' stanza 2***

This sextet holds the rhyme triplet 'cheap/ creep /sleep' and the internal rhyme 'steep/ deep/ creep' on line four. In the database approach the database would have to be manually updated to match every combination of these five words as rhymes. If a new rhyme word was found, for instance the word 'beep', then it would not be identified as a rhyme match and would have to be added manually to the database (Plamondon, op. cit.). It is quite likely this would only be added to the word it rhymed with in that instance and so would, at a later point, have to be added to the other four words too.

In the rule based approach all these rhymes were found automatically based on the string representation of them and their phonetic representation (indeed the results of the tests show that all these words were matched by either of the RhymeX and base string representation so even if the word did not exist in the phonetic dictionary the rhyme match would have been identified). Therefore the rules already modelled for this research are capable of matching all those words and new words such as ‘beep’ without any manual intervention. Plamondon’s (op. cit.) stated that the ultimate goal of his work on ‘AnalysePoems’ was a complete metrical and phonetic analysis of the verse. The inclusion of the phonetic dictionary with rhyme matching rules can facilitate that goal.

Russell (1971) demonstrated the problems of rhyme identification caused by geographic accentuation and archaic pronunciation. With the rule based approach different phonetic dictionaries may be used. This facilitates using different phonetic pronunciation to test the verse of William Shakespeare or a contemporary writer such as Seamus Heaney. This compares favourably to the database approach that would have to store all variants of the words pronunciation, both current dialect accentuations and archaic usage. Deciding which variant held in the database should be used for rhyme comparison could be problematic.

A second drawback to existing approaches identified in Section 1.2 was the identification of the end rhyme only. The research affirmed the question that a rule based algorithm approach can assist in the determination of all rhyme structures. The success in matching the rhyme structure as shown in Table 5.4, although not perfect, showed that rules might be modelled and applied in order to automatically determine the rhyme pattern and the rhyme type. The successful identification of the rhyme was

also accompanied by a volume of false positives as shown in Figure 5.2. It has not been possible to compare the level of false positives from this approach with others. Plamondon's (op. cit.) work matches only known word rhyme pairs and only produces false positives when the word rhymes with an alternative pronunciation of the word other than the one used. An example of this is provided by Plamondon (op. cit.) for the words 'subject' and 'reject'. These words only rhyme if they are both used as verbs but not if one is used as a noun. Adams and Birnbaum (op. cit) do not give any figures for false positives. It was stated however in Section 3.1 that the purpose of this research was not to produce a perfect match.

The analysis of these false positives generated by this rule based approach (Figure 5.3) show them to be generated primary by three rules. Rule types '1d' and '4d' match for similar endings on the base word type and the RhymeX representation. These were generated by matches on single (such as the pronoun 'I') or double (such as the pronoun 'we') characters. A basic rule filter requiring a minimum of characters for the match would eliminate these false positives. The third rule responsible for a large number of false positive was rule '3h'. This matches on similar stressed phonemes and produced numerous results based on the combination of matches between the words 'warm/ form/ storm'. Normally these would match as correct rhyme pairs but the author has used 'warm' to rhyme with 'arm' earlier and therefore 'warm' cannot, in this instance, rhyme with 'form/ storm'. In order to allow the application to identify this, a series of rhyme pattern templates based on rhyme patterns such as those outlined in Table 1.2 could be defined so the application could match to a known pattern. This approach was originally rejected for the research as it would possibly lead to a bias in the output towards known patterns. It could however be applied in other applications.

Although the use of the rule based algorithm has been shown to be a valid approach, the application did not produce a perfect match. Several factors have been decisive in this. The first factor is the lack of exclusion rules to assist in filtering out false positives. A deliberate decision was made early on not to model exclusion rules as the aim of the project was to assess if it was possible to identify the rhyme pairs. Exclusion rules are generally used for exception handling and time is required to identify those exceptions, unfortunately on this project it was not possible to afford the time to do so.

A second factor was the lack of word classes. In test 8 the eye rhyme pair 'wind/unkind' was not detected because the verb 'wind' rhymes with 'unkind' even though the noun 'wind' does not. Adams and Birnbaum's (op. cit) work also suffered from this difficulty in matching identical words with alternative pronunciations, as did Plamondon (op. cit.) with the effects of noun/ verb pronunciation. This issue could only be overcome if the poem contained mark up identifying the word class and the phonetic dictionary identified what word class the separate entries related too. As my raw input was plain text representation of the poem and the CMU (1998) phonetic dictionary does not contain word class identification, this was not possible.

The use of a phonetic string matching algorithm was also examined. In this case it was a new variant of editex (described by Zobel and Dart (1996)) that I called 'RhymeX'. While this representation proved it could assist and improve the rhyme identification I think it would need more work to truly describe this as a phonetic matching algorithm for rhyme. It is still too closely related to the editex version that has been developed to match for typographic and transcription errors. The use of an editex type string matching algorithm does show a lot of promise. In the event of there

being no entry in a phonetic dictionary the probability of correctly identifying a rhyme pair is increased by the use of such an algorithm as shown in Figure 5.1 by the increased detection rate between Test 1 and Test 2.

Although the application was, in general, very good at identifying rhyme patterns (as shown in Figure 5.1) there are certain aspects it not capable of identifying. Blank rhyme is not specifically identified. The application tries to find a rhyme pattern in every poem and will return a pattern even if only one rhyme pair is found. Conceptual rhymes are also not identified. Conceptual rhymes are analogous; an example of this would be the Sea 'rhyming' with Time or Death with Winter (Fry 2005, p124). The application currently matches the rhyme purely on the words spelling or pronunciation and never on its meaning or an idea related to the word.

### 6.2 Future research

As already mentioned the application currently produces numerous false positives due to the lack of exclusion rules. The first and most obvious improvement to the application would be to model some of these rules. The confidence calculations also need some refinement as these have caused some issues with the reporting back of the correct rhyme type as shown in Table 5.13 for eye rhyme.

The use of word classes would also be an area of interest to assist in the identification of the correct pronunciation for a word. This has impacted on my work here and has also been noted, as mentioned previously, in the work of both Plamondon (op. cit.) and Adams and Birnbaum (op. cit). This work would have to be coupled with the work of word class identification in a phonetic dictionary. Poetry also makes heavy use or a process called aphaeresis (Fry, 2005 p135). This means the dropping of the

first letter or letters and inserting an apostrophe (i.e. ‘neath for beneath). This prevents words being identified in the dictionary and so some method of handling this would be required. The coupling of this application and one to identify the scansion would also assist the rhyme identification because the scansion would identify the stressed syllable in the word. Poetic stress is not always applied in the same manner as phonetic stress.

## References


ACM. (1997) '*ACM Code of Ethics and Professional Conduct*' Association for Computing Machinery, Inc. (last updated 5/12/03)
(Available from http://www.acm.org/constitution/code.html accessed 29/04/2007)

Adams, L.D. and Birnbaum, D.J. (1996) 'Perspectives on Computer Programming for the Humanities' *Text Technology* 7/1, Spring, 1997, 1-17
(Available from http://clover.slavic.pitt.edu/~djb/rhyme/tt.html accessed 20/02/2007)

American Poems, http://www.americanpoems.com (accessed 23/07/2007)

Archer, D. Ernst-Gerlach, A. Kempken, S. Pilz, T. Rayson, P. (2006) 'The Identification of spelling Variants in English and German Historical Texts: Manual or Automatic?', Procedures of Digital Humanities (ALLC-ACH Conference) 2006

Barnard, G.A. (1946) "Sequential Tests in Industrial Statistics", *Supplement to the Journal of the Royal Statistical Society*, Vol. 8, No. 1. (1946), pp. 1-26.

Bauschatz, P. (2003) 'Rhyme and the structure of English consonants' *English Language and Linguistics* 7.1, pp 29-56, Cambridge University Press 2003.

Blake, J. (2005) 'Understanding Poetry Online: an Internet Application for Teaching', Procedures of ALLC-ACH Conference 2005

Burnard, L. and Sperberg-McQueen, C. M (2002), 'TEI Lite: An Introduction to Text Encoding for Interchange', *Text Encoding Initiative*.
(Available from http://www.tei-c.org/Lite/teiu5_en.pdf accessed 24/06/2007)

Chisholm, D. and Robey, D. (1995) 'Encoding Text Verse', *Computers and the Humanities* 29 pp 99-111, 1995

Cole, R. Mariani, J. Uszkoreit, H. Batista Varile, G. Zaenen, A. Zampoli, A. Zue, V. (editors) (1997) *Survey of the State of the Art in Human Language Technology (Web Edition)*, Cambridge University Press and Giardini.

CMU (Carnegie Mellon University) (1998) Phonetic Dictionary
(Available from http://www.speech.cs.cmu.edu/cgi-bin/cmudict accessed 24/06/2007)

Dublin, D. Birnbaum, D.J. (2005) 'A Declarative Framework for Modeling Pronunciation and Rhyme' Presented at ALLC/ACH, Victoria, British Columbia, June 2005
(Available from http://mustard.tapor.uvic.ca/cocoon/ach_abstracts/xq/pdf.xq?id=198 accessed 24/06/2007)

Elite Skills, http://www.eliteskills.com (accessed 23/07/2007)

Every poet, http://www.everypoet.com (accessed 17/07/2007)

Fabb, N. (1999) 'Verse constituency and the locality of alliteration', *Lingua* 108 (19999) pp 223-245

Fry, S. (2005) '*The Ode Less Travelled*' Hutchinson, London

Giordano, R. (1995) 'The TEI Header and the Documentation of Electronic Texts' in N. Ide. and J. Veronis, eds. The Text Encoding Initiative: Background and Contexts, special triple issue of *Computers and the Humanities*, 29:1, 1995, pp. 75-84

Hayward M. (1991) 'A Connectionist Model of Poetic Meter', *Poetics*, Volume 20, pp. 303-317

Hayward M. (1996) 'Application of a Connectionist model of poetic meter to Problems in Generative Metrics', *Research in Humanities Computing 4*. (pp. 185-192). Oxford: Clarendon P.

Hascall, D. (1969) 'Some contributions to the Halle-Keyser Theory of Prosody', *College English*, Volume 30, N0. 5 (Feb., 1969).pp. 357-365

Hockey, S. (2000) '*Electronic Texts in the Humanities*' Oxford University Press, Oxford.

Houdek, S. (1997) '*Electronic tagging of verse : a review of the literature*' University of Minnesota
(Available from http://mh.cla.umn.edu/houdek2.html accessed 24/06/2007)

Levenshtein, V. I. (1966) 'Binary codes capable of correcting deletions, insertions and reversals', *Soviet Physics* Doklady 10(8) pp. 707-710, Feb 1966.

Morgan, L. Z. (1991) 'Computational Analysis of Franco-Italian: St Mark's French Manuscript 13', *Literary and Linguistic Computing*, Volume 6, No. 1, 1991.

Mitton, R. (1986) 'A Partial Dictionary of English in Computer-Usable Form' *Literary and Linguistic Computing* 1986 1(4):214-215

Open University (2000) '*T834 Quality: Delivering Excellence'* Block 8 Inspection, Third Edition, 2000. Open University (ISBN 07492 98944)

Open University (2002) '*M880 Software Engineering*' Professional Issues version 2.1. Open University (ISBN 07492 55218)

Plamondon M. (2005) 'Computer-Assisted Phonetic Analysis of English Poetry: A Preliminary Case Study of Browning and Tennyson', TEXT Technology Volume 14 Number 2 153-174. (Received in correspondence from Author but also available at http://texttechnology.mcmaster.ca/pdf/vol14_2/plamondon14-2.pdf accessed 24/06/2007)

Plamondon, M. (2006) 'Virtual Verse Analysis: Analysing Patterns in Poetry" *Literary and Linguistic Computing 2006* vol. 21(Supplement 1): pp. 127-141

Joël Plisson, Nada Lavrac, Dunja Mladenic (n/d) 'A Rule Based Approach To Word Lemmatization'
(Available from http://eprints.pascal-network.org/archive/00000715/01/Pillson-Lematization.pdf accessed 24/06/2007)

Pressman, R. (2000) '*Software Engineering a practitioners approach*' fifth edition (European) Adapted for Open University by Darrel Ince. McGraw Hill, Maidenhead, Berkshire.

Representative Poetry Online, http://rpo.library.utoronto.ca/ (accessed 23/07/2007)

Ristad, E. S. and Yianilos, P. N. (1998) 'Learning String-Edit Distance', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 20, No. 5, May 1998.

Robey, D. (1990) 'Analysing Italian Renaissance Poetry: the Oxford Text Searching System', *Literary and Linguistic Computing*, Volume 5, No 4, 1990.

Robey, D. (1993) 'Scanning Dante's the Divine comedy: A Computer-based Approach', *Literary and Linguistic Computing*, Volume 8, No 2, 1993.

Robey, D. (1999) 'Counting syllables in the 'Divine Comedy': A Computer Analysis', *The Modern Language Review*, Volume 94, No 1 (Jan 1999), pp. 61-86)

Russell, G. (1971) "Dialectal and Phonetic Features of Edward Taylor's Rhymes: A Brief Study Based Upon a Computer Concordance of his Poems"
*American Literature*, Vol. 43, No. 2 (May, 1971), pp. 165-180

SAMPA (2005), University College London http://www.phon.ucl.ac.uk/home/sampa/

Sharp, J. Peters, J. Howard K (2002) '*The Management of a Student Research Project'*, 3rd Edition, Gower, England

Smolinsky, S. and Sokoloff, C. (2006) 'Introducing the Pattern-Finder', Procedures of Digital Humanities (ALLC-ACH Conference) 2006

Sproat, R. Taylor, P, Tanenblatt, M., Isard, A (1997) 'A Mark-up Language for Text-To-Speech Synthesis' *Eurospeech97*, Rhodes, Greece.

Text Encoding Initiative Consortium. (2007) "TEI Guidelines" P5 Release.
(Available from http://www.tei-c.org/P5/ accessed 23/07/2007)

Warwick, C. Buchanan, G. Gow, J. Blandford, A. Rimmer, J. (2006) 'Code. Comments and consistency, a Case Study of the Problems of Reuse of Encoded Texts', *Procedures of Digital Humanities* (ALLC-ACH Conference) 2006

Zobel, J., and Dart, P. (1996) "Phonetic String Matching: Lessons from Information Retrieval." *Proceedings of the 19th International Conference on Research and Development in Information Retrieval (SIGIR'96)*. Ed. H.-P. Frei, D. Harman, P. Schäuble and R. Wilkinson. New York: ACM, 1996. 166-172.
(Available from http://goanna.cs.rmit.edu.au/~jz/fulltext/sigir96.pdf accessed 02/06/2007)

## Index

## Appendix A – CMU Phoneme translation table

The following is the mapping table implemented to convert double character CMU phonemes to single character equivalents to facilitate easier comparison of the strings. For comparison it also lists the equivalent RhymeX character code

```
##
## CMU Phoneme translation table
##
## Matching phonemes to individual characters + Editex variant (rhymex) numeral
## to facilitate rhyme identification using Analyse Poems
##
## Both Upper and lower case characters are used and both have different sounds
##
## Structure = Phoneme        Map Char            Rhymex Code
##
##
AA      1       O
AE      2       A
AH      3       U
AO      4       O
AW      5       O
AY      6       Y
B       B       1
CH      C       2
D       D       3
DH      d       3
EH      E       E
ER      e       U
EY      7       A
F       F       7
G       G       6
HH      H       6
IH      i       I
IY      I       Y
JH      J       6
K       K       2
L       L       4
M       M       5
N       N       5
NG      n       5
OW      O       O
OY      o       Y
P       P       7
R       R       4
S       s       8
SH      S       8
T       T       3
TH      t       3
UH      u       U
UW      U       U
V       V       7
W       W       W
Y       Y       Y
Z       Z       8
ZH      z       8
```

## Appendix B – Rhyme identified for poems used in data input

The following is a list of the poetic texts used as sample input data during this research project and the rhyme structures identified as a preliminary step in the research. The identified rhyme patterns were used as the benchmark for analysis.

Alliteration indicated in blue italic font, internal rhyme in red, bold and underlined font.

**A Shropshire Lad XXVI Along the field as we came by ( AE Housman)**

Along the field as we came by
A year ago, my love and I,
The aspen over stile and stone
Was talking to itself alone.
"Oh who are these that kiss and pass?
A country lover and his lass;
Two lovers looking to be wed;
And time shall put them both to bed,
But she shall lie with earth above,
And he beside another love."

And sure enough beneath the tree
There walks another love with me,
And overhead the aspen heaves
Its rainy-sounding silver leaves;
And I spell nothing in their stir,
But now perhaps they speak to her,
And plain for her to understand
They talk about a time at hand
When I shall sleep with clover clad,
And she beside another lad.

**It was not death, for I stood up (510) (Emily Dickinson)**

| | | |
|---|---|---|
| **A** | It was not death, for I stood up, | - |
| **A** | And all the dead lie down. | - |
| **B** | It was not night, for all the bells | - |
| **B** | Put out their tongues for noon. | - |
| **C** | | |
| **C** | It was not frost, for on my flesh | - |
| **D** | I felt siroccos crawl, | - |
| **D** | Nor fire, for just my marble feet | - |
| **E** | Could keep a chancel cool. | - |
| **E** | | |
| | And yet it tasted like them all, | - |
| **F** | The figures I have seen | - |
| **F** | Set orderly for burial | - |
| **G** | Reminded me of mine, | - |
| **G** | | |
| **H** | As if my life were shaven | - |
| **H** | And fitted to a frame | - |
| **I** | And could not breathe without a key, | - |
| **I** | And 'twas like midnight, some, | - |
| **J** | | |
| **J** | When everything that ticked has stopped | - |
| | And space stares all around, | **A** |
| | Or grisly frosts, first autumn morns, | - |
| | Repeal the beating ground; | **A** |
| | | |
| | But most like chaos, stopless, cool, | - |
| | Without a chance, or spar, | - |
| | Or even a report of land | - |
| | To justify despair. | - |

**Sympathetic Portrait of a Child (William Carlos Williams)**

The murderer's little daughter -
who is barely ten years old -
jerks her shoulders -
right and left -
so as to catch a glimpse of me -
without turning round. -
Her skinny little arms -
wrap themselves -
this way then that -
reversely about her body! -
Nervously -
she crushes her straw hat -
about her eyes -
and tilts her head -
to deepen the shadow— -
smiling excitedly! -

As best as she can -
she hides herself -
in the full sunlight -
her cordy legs writhing -
beneath the little flowered dress -
that leaves them bare -
from mid-thigh to ankle— -

Why has she chosen me -
for the knife -
that darts along her smile? -

**No Worst, There is None (Gerard Manley Hopkins)**

No worst, there is none. *Pitched past pitch* of grief, **A**
More pangs will, schooled at forepangs, *wilder wring*. **B**
Comforter, where, where is your comforting? **B**
*Mary, mother* of us, where is your relief? **A**
My cries *heave, herds-long; huddle* in a main, a chief- **A**
*Woe, wórld*-sorrow; on an áge-old ánvil wínce and sing **B**
—
Then lull, then leave off. Fury had shrieked "No ling- **B**
Ering! Let me be *fell: force* I must be brief." **A**

O the *mind, mind has mountains*; cliffs of fall **C**
Frightful, sheer, no-man-fathomed. Hold them cheap **D**
May who ne'er hung there. Nor does long our small **C**
Durance deal with that **steep** or **deep**. Here! **creep**, **D**
Wretch, under a comfort serves in a whirlwind: all **C**
Life death does end and each day dies with sleep. **D**

## Ye Flowery Banks (Bonie Doon) ( Robert Burns)

| | |
|---|---|
| Ye flowery banks o' bonie **Doon**, | **A** |
| How can ye **blume** sae fair? | **B** |
| How can ye chant, ye little birds, | **C** |
| And I sae fu' o' care? | **B** |
| | |
| Thou'll break my heart, thou bonie bird, | **C** |
| That sings upon the bough; | **D** |
| Thou minds me o' the happy days, | **-** |
| When my fause love was true. | **D** |
| | |
| Thou'll break my heart, thou bonie bird, | **C** |
| That sings beside thy mate; | **E** |
| For *sae I sat, and sae I sang*, | **-** |
| And wist na o' my fate. | **E** |
| | |
| Aft hae I rov'd by bonie Doon | **A** |
| To see the wood-**bine twine**, | **F** |
| And ilka bird sang o' its luve, | **-** |
| And sae did I o' mine. | **F** |
| | |
| Wi' lightsome heart I pu'd a rose | **G** |
| Frae aff its thorny tree; | **H** |
| And my fause luver staw my rose | **G** |
| But left the thorn wi' me. | **H** |

## Out of Pompeii (William Wilfred Campbell)

| | |
|---|---|
| She lay, face downward, on her beaded arm, | **A** |
| In this her new, sweet dream of human bliss, | **B** |
| Her heart within her *fearful, fluttering*, warm, | **A** |
| Her lips yet pained with love's first timorous kiss. | **B** |
| She did not note the darkening afternoon, | **C** |
| She did not mark the lowering of the sky | **D** |
| O'er that great city. Earth had given its boon | **C** |
| Unto her lips, love touched her and passed by. | **D** |
| | |
| In one dread moment all the sky grew dark, | **E** |
| The hideous rain, the panic, the red rout, | **F** |
| Where *love lost love*, and all the world might mark | **E** |
| The city overwhelmed, blotted out | **F** |
| Without one cry, so quick oblivion came, | **G** |
| And life passed to the black where all forget; | **H** |
| But she,'we know not of her house or name,' | **G** |
| In love's sweet musings doth lie dreaming yet. | **H** |
| | |
| The dread hell passed, the ruined world grew still, | **I** |
| And the great city passed to nothingness: | **J** |
| The ages went and mankind worked its will. | **I** |
| Then men stood still amid the centuries' press, | **J** |
| And in the ash-hid ruins opened bare, | **K** |
| As she lay down in her shamed loveliness, | **J** |
| Sculptured and frozen, late they found her there, | **K** |
| Image of love 'mid all that hideousness. | **J** |
| | |
| Her head, face downward, on her bended arm, | **A** |
| Her single robe that showed her shapely form, | **L** |
| Her wondrous fate love keeps divinely warm | **A** |
| Over the centuries, past the slaying storm, | **L** |
| The heart can read in writings time hath left, | **M** |
| That linger still through death's oblivion; | **N** |
| And in this waste of life and light bereft, | **M** |
| She brings again a beauty that had gone. | **N** |
| | |
| And if there be a day when all shall wake, | **O** |
| As dreams the hoping, doubting human heart, | **P** |
| The dim forgetfulness of death will break | **O** |
| For her as one who sleeps with lips apart; | **P** |
| And did God call her suddenly, I know | **Q** |
| She'd wake as morning wakened by the thrush, | **R** |
| Feel that red kiss across the centuries glow, | **Q** |
| And make all heaven rosier by her blush. | **R** |

## The Raven (Edgar Allan Poe)

| | | |
|---|---|---|
| Once upon a midnight dreary, while I pondered, *weak and weary*, | - | |
| Over many a quaint and curious volume of forgotten lore— | **A** | |
| While I *nodded, nearly* ***napping***, suddenly there came a **tapping**, | - | |
| As of some one gently **rapping, rapping** at my chamber door. | **A** | |
| "'Tis some visiter," I muttered, "**tapping** at my chamber door— | **A** | |
| Only this and nothing more." | **A** | |
| | | |
| Ah, distinctly I **remember** it was in the bleak **December**; | - | |
| And each separate dying **ember** wrought its ghost upon the floor. | **A** | |
| Eagerly I wished the **morrow**;—vainly I had sought to **borrow** | - | |
| From my books *surcease of* ***sorrow***—**sorrow** for the lost Lenore— | **A** | |
| For the *rare and radiant* maiden whom the angels name Lenore— | **A** | |
| Nameless here for evermore. | **A** | |
| | | |
| And the *silken, sad*, **uncertain** rustling of each purple **curtain** | - | |
| **Thrilled** me—**filled** me with fantastic terrors never felt before; | **A** | |
| So that now, to still the beating of my heart, I stood **repeating** | - | |
| "'Tis some visiter ***entreating*** *entrance* at my chamber door— | **A** | |
| Some late visiter ***entreating*** *entrance* at my chamber door;— | **A** | |
| This it is and nothing more." | **A** | |

## Miners (Wilfred Owen)

| | | |
|---|---|---|
| There was a whispering in my hearth, | **A** | Slant rhyme |
| A sigh of the coal, | **B** | " |
| Grown wistful of a former earth | **A** | " |
| It might recall. | **B** | " |
| | | |
| I listened for a tale of leaves | **C** | " |
| And smothered ferns; | **D** | " |
| Frond-forests; and the low, sly lives | **C** | " |
| Before the fawns. | **D** | " |
| | | |
| My fire might show steam-phantoms simmer | **E** | " |
| From Time's old cauldron, | **F** | " |
| Before the birds made nests in summer, | **E** | " |
| Or men had children. | **F** | " |

## As You Like It (Act 2, scene VII) (William Shakespeare)

| | | |
|---|---|---|
| Blow, blow, thou winter wind. | **A** | }Eye rhyme |
| Thou art not so unkind | **A** | } |
| As man's ingratitude; | **B** | |
| Thy tooth is not so keen, | **C** | |
| Because thou art not seen, | **C** | |
| Although thy breath be rude. | **B** | |
| Heigh-ho! sing, heigh-ho! unto the green holly: | **D** | |
| Most friendship is feigning, most loving mere folly: | **D** | |
| Then, heigh-ho, the holly! | **D** | |
| This life is most jolly. | **D** | |
| Freeze, freeze, thou bitter sky, | **E** | |
| That dost not bite so nigh | **E** | |
| As benefits forgot: | **F** | |
| Though thou the waters warp, | **G** | }Eye rhyme |
| Thy sting is not so sharp | **G** | } |
| As friend remember'd not. | **F** | |

## Appendix C – Phonix replacement table

The following is the table implemented for the RhymeX conversion of a string. This table is the table developed for the Phonix string matching algorithm described by Zobel and Dart (1996). Each string being converted to a Phonix/ RhymeX representation is first checked for the listed substrings and the necessary character substitution made. The application first checks the start characters and then the end characters before finally checking the middle characters.

```
#list
#vowels list (vowels=... followed by comma separated list)
vowels=a,e,i,o,u
#consonant list (consonants=... followed by comma separated list)
consonants=b,c,d,f,g,h,j,k,l,m,n,p,q,r,s,t,v,w,x,y,z
#begin
#start
aa      ar
augh    arf
btl     tl
ca      ka
ce      se
chrV    krV
ci      si
ck      k
clV     klV
co      ko
crV     krV
ct      kt
cu      ku
cy      si
cz      z
dg      g
exci    ecs
gh      g
ghn     n
ght     t
gn      n
hroug   rew
jr      dr
kn      n
lle     le
lough   low
lz      lsh
mnV     nV
mps     ms
mpt     mt
mpts    mps
nc      nk
nx      nks
```

| | |
|---|---|
| ough | of |
| pf | f |
| ph | f |
| pnV | nV |
| ps | s |
| pt | t |
| q | kw |
| rz | rsh |
| sch | sh |
| tch | ch |
| tjV | chV |
| tsV | tV |
| tsjV | chV |
| wr | r |
| x | ecs |
| yjV | yV |
| yth | ith |
| zs | s |
| zz | ts |
| #middle | |
| Cz | Cts |
| VhrC | VahC |
| VjV | VyV |
| VqV | VkwV |
| VrC | VahC |
| Vstl | Vsl |
| Vwsk | Vvskie |
| Vwz | Vz |
| aa | ar |
| augh | arf |
| btl | tl |
| ca | ka |
| ce | se |
| ci | si |
| ck | k |
| co | ko |
| ct | kt |
| cu | ku |
| cy | si |
| cz | ch |
| dg | g |
| exci | ecs |
| ghn | n |
| ght | t |
| gnC | nC |
| hroug | rew |
| jr | dr |
| ljV | ldV |
| lle | le |
| lough | low |
| lz | lsh |
| mps | ms |
| mpt | mt |
| mpts | mps |
| nc | nk |
| nx | nks |
| ough | of |
| ph | f |
| rz | rsh |

| | |
|---|---|
| sch | sh |
| tch | ch |
| x | ecs |
| yth | ith |
| zV | sV |
| zs | s |
| zz | ts |
| #end | |
| Cle | Cile |
| Cles | Ciles |
| Vgh | Ve |
| Vhr | Vah |
| VhrC | VahC |
| Vmb | Vm |
| Vr | Vah |
| VrC | VahC |
| Vss | Vas |
| Vstl | Vsl |
| Vwsk | Vvskie |
| aa | ar |
| augh | arf |
| btl | tl |
| ca | ka |
| ce | se |
| ci | si |
| ck | k |
| co | ko |
| ct | kt |
| cu | ku |
| cy | si |
| dg | g |
| dl | dil |
| dt | t |
| e | % |
| eaux | oh |
| ee | ea |
| es | s |
| exci | ecs |
| gc | k |
| ghn | n |
| ght | t |
| gn | n |
| gnC | nC |
| gne | n |
| hroug | rew |
| jc | k |
| jr | dr |
| lle | le |
| lough | low |
| lz | lsh |
| mps | ms |
| mpt | mt |
| mpts | mps |
| nc | nk |
| ned | nd |
| nx | nks |
| ough | of |
| ph | f |
| re | ar |

```
rz      rsh
sch     sh
tch     ch
tl      til
tnt     ent
x       ecs
yth     ith
zs      s
zz      ts
#commentary
V       any vowel
C       any consonant
%       null
Order is important -- more general strings (ie substrings of later patterns)
should come first.
V, C can't be embedded.
```

## Appendix D – Sample report output for poem

The following is an example of the output report for a poem. This shows the rules enabled, full list of matches found (including alliteration, internal and slant rhyme matches not shown in the final pattern), a list of the highest confidence matches between two words, the pattern per stanza and the confidence of the match and finally the poem text with the rhyme patterns. The following is an example of the report for the poem ‘A Shropshire Lad XXVI Along the field as we came by’ by A. E. Housman.

Note: The output report is formatted by padding strings with spaces. It is best viewed therefore in a text editor that uses a proportional font such as ‘Courier’. Viewing on Windows using Notepad.exe is not recommended. Please use Wordpad.exe or a third party viewer such as Textpad, Edit Plus or Ultra Edit.

```
============================================================
=                                ANALYSE POEMS
=
=                                 REPORT FILE
============================================================


Processing poem: analyse_poem_A Shropshire Lad XXVI Along the field as we came by_42947.xml

User selected rhyme identification rules:
Rule 1 - End + Internal Rhyme - Try match on words is ENABLED
Rule 2 - End + Internal Rhyme - Try match on mapped phonemes is ENABLED
Rule 3 - End + Internal Rhyme - Try match on stressed phonemes is ENABLED
Rule 4 - End + Internal Rhyme - Try RhymeX match is ENABLED
Rule 5 - Check for Eye Rhyme for end rhyme only is ENABLED
Rule 6 - Check Masculine and feminine rhyme for end rhyme only is ENABLED
Rule 7 - Check for Internal rhyme is ENABLED
Rule 8 - Check for alliteration is ENABLED
Rule 9 - Check for assonance/ consonance for end rhyme only is ENABLED

Processing rules (software modelled rule used to process rhyme identification rules)
Rule a - First check to see if the words are identical
Rule b - Check if one words is identical to the ending of the other
Rule c - Check if the words are substrings of each other
Rule d - Check for identical endings
Rule e - Check for similarity using Edit Distance
Rule f - Check for similarity using a Patern Match
Rule g - ALLITERATION - Check there is a min of 3 chars and see how many first chars match
Rule h - Check the words on stressed phoneme matches
Rule i - Check the words on stressed phoneme matches to see if a masculine or feminine rhyme can be
determined
Rule j - Check the words for possible assonance/ consonance


APPLIED PROCESSING RULES
```

```
 Rule 1 applies processing rules (A, B, C, D, E, F, )
 Rule 2 applies processing rules (A, B, C, D, E, F,)
 Rule 3 applies processing rules (H, )
 Rule 4 applies processing rules (A, B, C, D, E, F,)
 Rule 5 applies processing rules (A, B, C, D, )
 Rule 6 applies processing rules (I, )
 Rule 7 applies processing rules (A, B, C, D, G, H, )
 Rule 8 applies processing rules (G, )
 Rule 9 applies processing rules (J, )
```

| Line | Word | Line | Word | Rule | Conf. | Type |
|---|---|---|---|---|---|---|
| 1 | by(B6) | 2 | I(6) | 2b | 100 | Full Rhyme |
| 2 | I(I) | 15 | stir(83IAH) | 4c | 20 | Full Rhyme |
| 2 | I(6) | 2 | my(M6) | 2b | 100 | Full Rhyme |
| 3 | stone | 4 | alone | 1d | 60 | Full Rhyme |
| 3 | stone(S T OW1 N) | 4 | alone(AH0 L OW1 N) | 3h | 90 | Full Rhyme |
| 3 | stone(83O5) | 4 | alone(A4O5) | 3h | 50 | Full Rhyme |
| 3 | stone(S T OW1 N) | 4 | alone(AH0 L OW1 N) | 6i | 91 | Masculine |
| 3 | stone | 7 | wed | 9j | 14 | Assonance |
| 3 | stone | 8 | bed | 9j | 14 | Assonance |
| 3 | stone | 9 | above | 9j | 28 | Assonance |
| 3 | stone | 10 | love | 9j | 28 | Assonance |
| 3 | stone | 11 | tree | 9j | 14 | Assonance |
| 3 | stone | 12 | me | 1f | 70 | Full Rhyme |
| 3 | stone(83O5) | 12 | me(5) | 4b | 100 | Full Rhyme |
| 3 | stone | 13 | heaves | 9j | 11 | Assonance |
| 3 | stone | 14 | leaves | 9j | 11 | Assonance |
| 3 | stone | 15 | stir | 9j | 28 | Consonance |
| 3 | stone | 16 | her | 9j | 14 | Assonance |
| 3 | stone(sTON) | 17 | understand(3NDesT2ND) | 2e | 17 | Full Rhyme |
| 3 | stone | 17 | understand | 9j | 21 | Consonance |
| 3 | stone | 3 | The | 1f | 70 | Full Rhyme |
| 3 | stone | 3 | stile | 5g | 75 | Alliteration |
| 4 | alone | 5 | pass | 9j | 14 | Assonance |
| 4 | alone | 6 | lass | 9j | 14 | Assonance |
| 4 | alone | 7 | wed | 9j | 14 | Assonance |
| 4 | alone | 8 | bed | 9j | 14 | Assonance |
| 4 | alone | 9 | above | 9j | 42 | Assonance |
| 4 | alone | 10 | love | 1e | 17 | Full Rhyme |
| 4 | alone | 10 | love | 9j | 28 | Assonance |
| 4 | alone | 11 | tree | 9j | 14 | Assonance |
| 4 | alone | 12 | me | 1f | 70 | Full Rhyme |
| 4 | alone(A4O5) | 12 | me(5) | 4b | 100 | Full Rhyme |
| 4 | alone | 13 | heaves | 9j | 23 | Assonance |
| 4 | alone | 14 | leaves | 9j | 23 | Assonance |
| 4 | alone | 16 | her | 9j | 14 | Assonance |
| 4 | alone | 17 | understand | 9j | 7 | Assonance |
| 4 | alone | 18 | hand | 9j | 14 | Assonance |
| 4 | alone | 19 | clad | 9j | 14 | Assonance |
| 4 | alone | 20 | lad | 9j | 14 | Assonance |
| 5 | pass | 6 | lass | 1d | 75 | Full Rhyme |
| 5 | pass(P AE1 S) | 6 | lass(L AE1 S) | 3h | 90 | Full Rhyme |
| 5 | pass(1A48) | 6 | lass(4A48) | 3h | 75 | Full Rhyme |
| 5 | pass(P AE1 S) | 6 | lass(L AE1 S) | 6i | 91 | Masculine |
| 5 | pass | 9 | above | 9j | 14 | Assonance |
| 5 | pass | 13 | heaves | 9j | 11 | Assonance |
| 5 | pass | 14 | leaves | 9j | 11 | Assonance |
| 5 | pass | 17 | understand | 9j | 7 | Assonance |
| 5 | pass | 18 | hand | 9j | 17 | Assonance |
| 5 | pass | 19 | clad | 9j | 17 | Assonance |
| 5 | pass | 20 | lad | 9j | 17 | Assonance |
| 5 | pass(1A48) | 5 | are(AA4) | 4f | 70 | Full Rhyme |
| 5 | these | 5 | that | 5g | 75 | Alliteration |
| 5 | pass | 5 | kiss | 1d | 50 | Full Rhyme |
| 6 | lass | 9 | above | 9j | 14 | Assonance |
| 6 | lass | 13 | heaves | 9j | 11 | Assonance |
| 6 | lass | 14 | leaves | 1e | 17 | Full Rhyme |
| 6 | lass | 14 | leaves | 9j | 23 | Consonance |
| 6 | lass | 17 | understand | 9j | 7 | Assonance |
| 6 | lass | 18 | hand | 9j | 17 | Assonance |
| 6 | lass | 19 | clad | 9j | 17 | Assonance |
| 6 | lass | 20 | lad | 9j | 17 | Assonance |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 6 | lass(4A48) | 6 | A(A) | 4c | 25 | Full Rhyme |
| 6 | lass | 6 | lover | 5g | 50 | Alliteration |
| 7 | wed | 8 | bed | 1d | 66 | Full Rhyme |
| 7 | wed(W EH1 D) | 8 | bed(B EH1 D) | 3h | 90 | Full Rhyme |
| 7 | wed(WE3) | 8 | bed(1E3) | 3h | 66 | Full Rhyme |
| 7 | wed(W EH1 D) | 8 | bed(B EH1 D) | 6i | 91 | Masculine |
| 7 | wed | 9 | above | 9j | 14 | Assonance |
| 7 | wed | 10 | love | 9j | 17 | Assonance |
| 7 | wed | 11 | tree | 1f | 70 | Full Rhyme |
| 7 | wed | 12 | me | 9j | 23 | Assonance |
| 7 | wed | 13 | heaves | 9j | 11 | Assonance |
| 7 | wed | 14 | leaves | 9j | 11 | Assonance |
| 7 | wed | 16 | her | 9j | 23 | Assonance |
| 7 | wed | 17 | understand | 9j | 7 | Assonance |
| 7 | lovers | 7 | looking | 5g | 75 | Alliteration |
| 8 | bed | 9 | above | 9j | 14 | Assonance |
| 8 | bed | 10 | love | 9j | 17 | Assonance |
| 8 | bed | 11 | tree | 1f | 70 | Full Rhyme |
| 8 | bed | 12 | me | 9j | 23 | Assonance |
| 8 | bed | 13 | heaves | 9j | 11 | Assonance |
| 8 | bed | 14 | leaves | 9j | 11 | Assonance |
| 8 | bed | 16 | her | 9j | 23 | Assonance |
| 8 | bed | 17 | understand | 9j | 7 | Assonance |
| 8 | bed | 8 | both | 5g | 50 | Alliteration |
| 9 | above | 10 | love | 1d | 75 | Full Rhyme |
| 9 | above(AH0 B AH1 V) | 10 | love(L AH1 V) | 3h | 90 | Full Rhyme |
| 9 | above(A1O7) | 10 | love(4O7) | 3h | 66 | Full Rhyme |
| 9 | above(AH0 B AH1 V) | 10 | love(L AH1 V) | 6i | 91 | Masculine |
| 9 | above | 11 | tree | 9j | 14 | Assonance |
| 9 | above | 12 | me | 1f | 70 | Full Rhyme |
| 9 | above | 13 | heaves | 9j | 23 | Assonance |
| 9 | above | 14 | leaves | 9j | 23 | Assonance |
| 9 | above | 16 | her | 9j | 14 | Assonance |
| 9 | above | 17 | understand | 9j | 7 | Assonance |
| 9 | above | 18 | hand | 9j | 14 | Assonance |
| 9 | above | 19 | clad | 9j | 14 | Assonance |
| 9 | above | 20 | lad | 9j | 14 | Assonance |
| 9 | above | 9 | she | 1f | 70 | Full Rhyme |
| 9 | she | 9 | shall | 5g | 75 | Alliteration |
| 9 | above | 9 | lie | 1f | 70 | Full Rhyme |
| 10 | love | 11 | tree | 9j | 17 | Assonance |
| 10 | love | 12 | me | 1f | 70 | Full Rhyme |
| 10 | love | 13 | heaves | 9j | 11 | Assonance |
| 10 | love | 14 | leaves | 1e | 17 | Full Rhyme |
| 10 | love | 14 | leaves | 9j | 23 | Consonance |
| 10 | love | 16 | her | 9j | 17 | Assonance |
| 10 | love | 17 | understand | 9j | 7 | Assonance |
| 10 | love | 10 | he | 1f | 70 | Full Rhyme |
| 11 | tree | 12 | me | 1f | 70 | Full Rhyme |
| 11 | tree(T R IY1) | 12 | me(M IY1) | 3h | 90 | Full Rhyme |
| 11 | tree(T R IY1) | 12 | me(M IY1) | 6i | 91 | Masculine |
| 11 | tree(T R IY1) | 13 | heaves(HH IY1 V Z) | 3h | 90 | Full Rhyme |
| 11 | tree(T R IY1) | 14 | leaves(L IY1 V Z) | 3h | 90 | Full Rhyme |
| 11 | tree | 15 | stir | 9j | 35 | Consonance |
| 11 | tree | 16 | her | 9j | 17 | Assonance |
| 11 | tree | 17 | understand | 9j | 7 | Assonance |
| 11 | tree(T R IY1) | 11 | beneath(B AH0 N IY1 TH) | 3h | 90 | Full Rhyme |
| 11 | tree | 11 | the | 1f | 70 | Full Rhyme |
| 11 | tree | 11 | the | 5g | 50 | Alliteration |
| 11 | the | 11 | tree | 5g | 50 | Alliteration |
| 12 | me(M IY1) | 13 | heaves(HH IY1 V Z) | 3h | 90 | Full Rhyme |
| 12 | me(M IY1) | 14 | leaves(L IY1 V Z) | 3h | 90 | Full Rhyme |
| 12 | me | 16 | her | 9j | 23 | Assonance |
| 12 | me(5) | 17 | understand(U53EAH83A53) | 4c | 9 | Full Rhyme |
| 12 | me | 17 | understand | 9j | 7 | Assonance |
| 12 | me(5) | 18 | hand(HA53) | 4c | 25 | Full Rhyme |
| 12 | me | 12 | There | 1f | 70 | Full Rhyme |
| 12 | me(5) | 12 | another(A5O3HEAH) | 4c | 12 | Full Rhyme |
| 12 | me | 12 | love | 1f | 70 | Full Rhyme |
| 13 | heaves | 14 | leaves | 1d | 83 | Full Rhyme |
| 13 | heaves(HH IY1 V Z) | 14 | leaves(L IY1 V Z) | 3h | 90 | Full Rhyme |
| 13 | heaves(HEA78) | 14 | leaves(4EA78) | 3h | 80 | Full Rhyme |
| 13 | heaves(HH IY1 V Z) | 14 | leaves(L IY1 V Z) | 6i | 91 | Masculine |
| 13 | heaves | 16 | her | 9j | 11 | Assonance |
| 13 | heaves | 17 | understand | 9j | 14 | Assonance |
| 13 | heaves | 18 | hand | 9j | 11 | Assonance |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 13 | heaves | 19 | clad | 9j | 11 | Assonance |
| 13 | heaves | 20 | lad | 9j | 11 | Assonance |
| 14 | leaves | 16 | her | 9j | 11 | Assonance |
| 14 | leaves | 17 | understand | 9j | 14 | Assonance |
| 14 | leaves | 18 | hand | 9j | 11 | Assonance |
| 14 | leaves | 19 | clad | 9j | 11 | Assonance |
| 14 | leaves | 20 | lad | 9j | 11 | Assonance |
| 14 | sounding | 14 | silver | 5g | 50 | Alliteration |
| 15 | stir(sTe) | 16 | her(He) | 2f | 70 | Full Rhyme |
| 15 | stir(S T ER1) | 16 | her(HH ER1) | 3h | 90 | Full Rhyme |
| 15 | stir(83IAH) | 16 | her(HEAH) | 3h | 50 | Full Rhyme |
| 15 | stir | 17 | understand | 9j | 14 | Consonance |
| 15 | stir(83IAH) | 15 | I(I) | 4c | 20 | Full Rhyme |
| 15 | stir | 15 | spell | 5g | 50 | Alliteration |
| 15 | stir | 15 | their | 1d | 50 | Full Rhyme |
| 15 | stir(83IAH) | 15 | their(3HEIAH) | 4d | 60 | Full Rhyme |
| 16 | her | 17 | understand | 9j | 7 | Assonance |
| 17 | understand | 18 | hand | 1d | 75 | Full Rhyme |
| 17 | understand(AH2 N D ER0 S T AE1 N D) | 18 | hand(HH AE1 N D) | 3h | 90 | Full Rhyme |
| 17 | understand(U53EAH83A53) | 18 | hand(HA53) | 3h | 75 | Full Rhyme |
| 17 | understand(AH2 N D ER0 S T AE1 N D) | 18 | hand(HH AE1 N D) | 6i | 91 | Masculine |
| 17 | understand | 19 | clad | 9j | 7 | Assonance |
| 17 | understand | 20 | lad | 9j | 7 | Assonance |
| 17 | understand | 17 | And | 1d | 66 | Full Rhyme |
| 17 | understand(3NDesT2ND) | 17 | And(2ND) | 2b | 100 | Full Rhyme |
| 17 | understand(U53EAH83A53) | 17 | And(A53) | 2b | 100 | Full Rhyme |
| 18 | hand | 19 | clad | 9j | 17 | Assonance |
| 18 | hand | 20 | lad | 9j | 17 | Assonance |
| 18 | They | 18 | talk | 5g | 50 | Alliteration |
| 18 | hand | 18 | a | 1c | 25 | Full Rhyme |
| 18 | hand(HA53) | 18 | a(A) | 1c | 25 | Full Rhyme |
| 19 | clad | 20 | lad | 1b | 100 | Full Rhyme |
| 19 | clad(24A3) | 20 | lad(4A3) | 1b | 100 | Full Rhyme |
| 19 | clad(K L AE1 D) | 20 | lad(L AE1 D) | 6i | 101 | Masculine |
| 19 | shall | 19 | sleep | 5g | 50 | Alliteration |
| 19 | clad | 19 | clover | 5g | 75 | Alliteration |
| 19 | clover | 19 | clad | 5g | 75 | Alliteration |

HIGHEST CONFIDENCE MATCHES

| Line | Word | Line | Word | Rule | Conf. | Type |
|---|---|---|---|---|---|---|
| 1 | by | 2 | I | 2b | 100 | Full Rhyme |
| 2 | I | 15 | stir | 4c | 20 | Full Rhyme |
| 2 | I | 2 | my | 2b | 100 | Internal |
| 3 | stone | 4 | alone | 6i | 91 | Masculine |
| 3 | stone | 12 | me | 4b | 100 | Full Rhyme |
| 3 | stone | 17 | understand | 2e | 17 | Full Rhyme |
| 3 | stone | 3 | The | 1f | 70 | Internal |
| 3 | stone | 3 | stile | 5g | 75 | Alliteration |
| 4 | alone | 10 | love | 1e | 17 | Full Rhyme |
| 4 | alone | 12 | me | 4b | 100 | Full Rhyme |
| 5 | pass | 6 | lass | 6i | 91 | Masculine |
| 5 | pass | 5 | are | 4f | 70 | Internal |
| 5 | these | 5 | that | 5g | 75 | Alliteration |
| 5 | pass | 5 | kiss | 1d | 50 | Internal |
| 6 | lass | 14 | leaves | 1e | 17 | Full Rhyme |
| 6 | lass | 6 | A | 4c | 25 | Internal |
| 6 | lass | 6 | lover | 5g | 50 | Alliteration |
| 7 | wed | 8 | bed | 6i | 91 | Masculine |
| 7 | wed | 11 | tree | 1f | 70 | Full Rhyme |
| 7 | lovers | 7 | looking | 5g | 75 | Alliteration |
| 8 | bed | 11 | tree | 1f | 70 | Full Rhyme |
| 8 | bed | 8 | both | 5g | 50 | Alliteration |
| 9 | above | 10 | love | 6i | 91 | Masculine |
| 9 | above | 12 | me | 1f | 70 | Full Rhyme |
| 9 | above | 9 | she | 1f | 70 | Internal |
| 9 | she | 9 | shall | 5g | 75 | Alliteration |
| 9 | above | 9 | lie | 1f | 70 | Internal |
| 10 | love | 12 | me | 1f | 70 | Full Rhyme |
| 10 | love | 14 | leaves | 1e | 17 | Full Rhyme |
| 10 | love | 10 | he | 1f | 70 | Internal |
| 11 | tree | 12 | me | 6i | 91 | Masculine |
| 11 | tree | 13 | heaves | 3h | 90 | Full Rhyme |
| 11 | tree | 14 | leaves | 3h | 90 | Full Rhyme |
| 11 | tree | 11 | beneath | 3h | 90 | Internal |

```
|   11|                         tree|   11|                          the|  5g|    50|   Alliteration|
|   11|                         tree|   11|                          the|  1f|    70|       Internal|
|   11|                          the|   11|                         tree|  5g|    50|   Alliteration|
|   12|                           me|   13|                       heaves|  3h|    90|     Full Rhyme|
|   12|                           me|   14|                       leaves|  3h|    90|     Full Rhyme|
|   12|                           me|   17|                   understand|  4c|     9|     Full Rhyme|
|   12|                           me|   18|                         hand|  4c|    25|     Full Rhyme|
|   12|                           me|   12|                        There|  1f|    70|       Internal|
|   12|                           me|   12|                      another|  4c|    12|       Internal|
|   12|                           me|   12|                         love|  1f|    70|       Internal|
|   13|                       heaves|   14|                       leaves|  6i|    91|      Masculine|
|   14|                     sounding|   14|                       silver|  5g|    50|   Alliteration|
|   15|                         stir|   16|                          her|  3h|    90|     Full Rhyme|
|   15|                         stir|   15|                            I|  4c|    20|       Internal|
|   15|                         stir|   15|                        spell|  5g|    50|   Alliteration|
|   15|                         stir|   15|                        their|  4d|    60|       Internal|
|   17|                   understand|   18|                         hand|  6i|    91|      Masculine|
|   17|                   understand|   17|                          And|  2b|   100|       Internal|
|   18|                         They|   18|                         talk|  5g|    50|   Alliteration|
|   18|                         hand|   18|                            a|  1c|    25|       Internal|
|   19|                         clad|   20|                          lad|  6i|   101|      Masculine|
|   19|                        shall|   19|                        sleep|  5g|    50|   Alliteration|
|   19|                         clad|   19|                       clover|  5g|    75|   Alliteration|
|   19|                       clover|   19|                         clad|  5g|    75|   Alliteration|
-------------------------------------------------------------------------------------------------------

Stanza 1 rhyme pattern = 'AABBCCDDEE'
Stanza 2 rhyme pattern = 'DBDDAABBFF'

Confidence = 94% for 20 identified end rhymes


POEM RHYME PATTERN


Along the field as we came by A
A year ago, my love and I,    A
The aspen over stile and stone       B
Was talking to itself alone.  B
"Oh who are these that kiss and pass? C
A country lover and his lass; C
Two lovers looking to be wed; D
And time shall put them both to bed,  D
But she shall lie with earth above,   E
And he beside another love."  E

And sure enough beneath the tree     D
There walks another love with me,    B
And overhead the aspen heaves D
Its rainy-sounding silver leaves;    D
And I spell nothing in their stir,   A
But now perhaps they speak to her,   A
And plain for her to understand      B
They talk about a time at hand       B
When I shall sleep with clover clad,  F
And she beside another lad.   F


============= END OF REPORT ============================
```

## Appendix E – Sample XML output for poem

The following is an example of the XML output for the poem ‘A Shropshire Lad XXVI Along the field as we came by’ by A. E. Housman. This output complies with the TEI guidelines with additional labelling for alliteration and rhyme type. The TEI header block has been added

```
<?xml version="1.0" ?>
<!-- Formatted to TEI version 5 by AnalysePoems -->
<!-- copyright Frank Kavanagh -->
<TEI>
  <teiHeader>
    <fileDesc>
      <titleStmt>
        <title />
        <author />
        <respStmt>
          <resp>Auto generated from manual text input by</resp>
          <name>Analyse Poems (copyright Frank Kavanagh)</name>
        </respStmt>
      </titleStmt>
      <publicationStmt>Unknown - Auto generated from manual text input</publicationStmt>
      <sourceDesc>Manual input</sourceDesc>
    </fileDesc>
  </teiHeader>
  <text>
    <body>
      <lg type="" met="" rhyme="AABBCCDDEE">
        <l n="1" real="">
          Along the field as we came
          <rhyme label="A" type="Full Rhyme">by</rhyme>
        </l>
        <l n="2" real="">
          A year ago,
          <rhyme label="A" type="Internal">my</rhyme>
          love and
          <rhyme label="A" type="Full Rhyme">I</rhyme>
          ,
        </l>
        <l n="3" real="">
          <rhyme label="B" type="Internal">The</rhyme>
          aspen over
          <rhyme label="" type="" alliteration="st">stile</rhyme>
          and
          <rhyme label="B" type="Full Rhyme" alliteration="st">stone</rhyme>
        </l>
        <l n="4" real="">
          Was talking to itself
          <rhyme label="B" type="Full Rhyme">alone</rhyme>
          .
        </l>
        <l n="5" real="">
```

```
            "Oh who
            <rhyme label="C" type="Internal">are</rhyme>
            <rhyme label="" type="" alliteration="th">these</rhyme>
            <rhyme label="" type="" alliteration="th">that</rhyme>
            <rhyme label="C" type="Internal">kiss</rhyme>
            and
            <rhyme label="C" type="Masculine">pass</rhyme>
            ?
          </l>
        - <l n="6" real="">
            <rhyme label="C" type="Internal">A</rhyme>
            country
            <rhyme label="" type="" alliteration="l">lover</rhyme>
            and his
            <rhyme label="C" type="Masculine" alliteration="l">lass</rhyme>
            ;
          </l>
        - <l n="7" real="">
            Two
            <rhyme label="" type="" alliteration="lo">lovers</rhyme>
            <rhyme label="" type="" alliteration="lo">looking</rhyme>
            to be
            <rhyme label="D" type="Masculine">wed</rhyme>
            ;
          </l>
        - <l n="8" real="">
            And time shall put them
            <rhyme label="" type="" alliteration="b">both</rhyme>
            to
            <rhyme label="D" type="Masculine" alliteration="b">bed</rhyme>
            ,
          </l>
        - <l n="9" real="">
            But
            <rhyme label="E" type="Internal" alliteration="sh">she</rhyme>
            <rhyme label="" type="" alliteration="sh">shall</rhyme>
            <rhyme label="E" type="Internal">lie</rhyme>
            with earth
            <rhyme label="E" type="Masculine">above</rhyme>
            ,
          </l>
        - <l n="10" real="">
            And
            <rhyme label="E" type="Internal">he</rhyme>
            beside another
            <rhyme label="E" type="Masculine">love</rhyme>
            ."
          </l>
      </lg>
    - <lg type="" met="" rhyme="DBDDAABBFF">
        - <l n="11" real="">
            And sure enough
            <rhyme label="D" type="Internal">beneath</rhyme>
            <rhyme label="D" type="Internal" alliteration="t">the</rhyme>
            <rhyme label="D" type="Masculine" alliteration="t">tree</rhyme>
          </l>
        - <l n="12" real="">
            <rhyme label="B" type="Internal">There</rhyme>
            walks
```

<rhyme label="B" type="Internal">another</rhyme>
<rhyme label="B" type="Internal">love</rhyme>
with
<rhyme label="B" type="Full Rhyme">me</rhyme>
,
</l>
- <l n="13" real="">
And overhead the aspen
<rhyme label="D" type="Masculine">heaves</rhyme>
</l>
- <l n="14" real="">
Its rainy
<rhyme label="" type="" alliteration="s">sounding</rhyme>
<rhyme label="" type="" alliteration="s">silver</rhyme>
<rhyme label="D" type="Masculine">leaves</rhyme>
;
</l>
- <l n="15" real="">
And
<rhyme label="A" type="Internal">I</rhyme>
<rhyme label="" type="" alliteration="s">spell</rhyme>
nothing in
<rhyme label="A" type="Internal">their</rhyme>
<rhyme label="A" type="Full Rhyme" alliteration="s">stir</rhyme>
,
</l>
- <l n="16" real="">
But now perhaps they speak to
<rhyme label="A" type="Full Rhyme">her</rhyme>
,
</l>
- <l n="17" real="">
<rhyme label="B" type="Internal">And</rhyme>
plain for her to
<rhyme label="B" type="Full Rhyme">understand</rhyme>
</l>
- <l n="18" real="">
<rhyme label="" type="" alliteration="T">They</rhyme>
<rhyme label="" type="" alliteration="T">talk</rhyme>
about
<rhyme label="B" type="Internal">a</rhyme>
time at
<rhyme label="B" type="Masculine">hand</rhyme>
</l>
- <l n="19" real="">
When I
<rhyme label="" type="" alliteration="s">shall</rhyme>
<rhyme label="" type="" alliteration="s">sleep</rhyme>
with
<rhyme label="" type="" alliteration="cl">clover</rhyme>
<rhyme label="F" type="Masculine" alliteration="cl">clad</rhyme>
,
</l>
- <l n="20" real="">
And she beside another
<rhyme label="F" type="Masculine">lad</rhyme>
.
</l>
</lg>

```
        </body>
    </text>
</TEI>
```

## Appendix F – Application software

Please refer to the attached CD. The JAVA application software is contained in the folder ‘AnalysePoems’.

The software may be run by executing the Java class ‘RunApp.class’. This class expects one parameter; the path and filename of the poem to be analysed.

The software was developed using the Eclipse IDE and was tested using the Java Runtime Environment version 1.06.0_02

## Appendix G –Test outlines, results and analysis

Please refer to the attached CD. The tests run, their results and the analysis of those results are contained in the folder ‘Analysis’.

The spreadsheet ‘TestOutline.xls’ provides a summary and description of the tests run and the poems used for each test. Each test has its own subfolder within the ‘Results’ folder containing the output from the application (the XML version of the poem and the report). From these outputs the results analysis spreadsheets ‘Textx_ResultsAnalysis.xls’ were compiled. These spreadsheets contain the data used during the documentation of this project.