Evolutionary Assistance in Alliteration and Allelic Drivel*

Raquel Hervás[1], Jason Robinson[2], and Pablo Gervás[1]

[1] Dep. Ingeniería del Software e Inteligencia Artificial,
Universidad Complutense de Madrid, Spain
raquelhb@fdi.ucm.es, pgervas@sip.ucm.es
[2] University of Georgia, USA
srjrobin@uga.edu

Abstract. This paper presents an approximation towards an evolutionary generator of alliterative text. A simple text is given along with the preferred phoneme for alliterations as input. Then the evolutionary algorithm (with the aid of a phonemic transcriber, Microsoft Word and Google) will try to produce an alternative sentence while trying to preserve the initial meaning and coherence. A bigram language model and the evaluation of the phonetic analysis are used to assess the fitness of the sentences.

1 Introduction

Alliteration is a literary figure difficult to explain in precise terms. For example, some human poets consider alliteration to be the repetition of consonant sounds at the beginning of words; others consider it to be the repetition of consonant sounds in stressed syllables; and others may consider it to be the repetition of any sounds anywhere in the utterance. In any case, alliteration involves repetition of sound within a text of certain phonetic ingredients. However, feasible as it may be to carry out the phonetic analysis of an utterance, there is an intrinsically aesthetic component to the perception of alliteration by humans which is difficult to reduce to rules. A large number of ingredients seem to play a role in the process - certainly a repeated occurrence of certain phonemes, but also avoidance of radical departures from an intended meaning, rhythm, whether the given phoneme starts a word, similarities between the sound of the repeated phonemes and the sounds associated in real life with the concepts being mentioned... Moreover, the magic of a good alliteration lies in finding a perfect balance between this set of ingredients. We believe that an evolutionary solution - with possible modifications to a text modelled as mutation operators, and the ingredients that must be judged represented as fitness functions - is a good method for exploring the potential of applications for automated alliteration of input sentences. This

* Partially supported by the Spanish Ministry of Education and Science project TIN2006-14433-C02-01.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 537–546, 2007.

paper presents an initial exploration of this problem, using synonym substitution as an elementary modification mechanism, *synset* preservation as means of avoiding drastic shifts in meaning, and a combination of an n-gram language model and evaluation of phonetic analysis to evaluate the fitness of individuals. The n-gram language model is based on Google searches, calculating the probability of occurrence of a bigram in terms of its frequency of appearance as a Google search.

The current project isolates alliteration, in its broadest definition, in order to provide an inchoate step in a process that could eventually incorporate many other stylistic weapons in a writer's arsenal (be the writer human or machine). Once we have proven algorithms that generate semantically similar sentences with greater occurrences of alliteration, we can then apply this algorithm to other stylistic possibilities such as rhythm and rhyme and even more difficult semantic tricks.

2 A Strange Brew

This section outlines the combined use of typical research methodologies - Linguistics and Evolutionary Algorithms - with widely available commercial tools like Microsoft Word and Google.

2.1 Phonetics and Alliteration

The importance of alliteration in written Spanish style has often been considered inconsequential, deprecated or more a facet of germanic languages than it is of Spanish. Nevertheless, if one were to subscribe to theories such as Roman Jakobson's, which states that corresponding sounds suggest correspondence in meaning [1], alliteration would be a valid tool for all authors, regardless of language.

A valid phonemic (or phonetic) transcription is necessary for any algorithm that will be capable of measuring phonic qualities of an utterance in most languages. In Spanish it is true that a phonemic transcription is not as necessary for identifying alliteration as it would be for English due to the high correlation between phonemes and graphemes in Spanish; however, to evaluate rhyme and rhythm, the role of the phonemic transcription will become more critical.

An automatic phonemic transcription in Spanish is straightforward by following standard orthographic [2] and phonemic conventions [3,4]. The universal transcription algorithm used in this work was originally developed for a historical research project on Spanish syllabification, and revised and augmented in various projects since [5]. It will have some limitations namely dialectal differences and loan words. Nevertheless, the authors of this work have found that automatic transcription produced by this algorithm is sufficient for the majority of Spanish words, and therefore valid for this study.

Our computer generated transcription translates letters into phonemes and then groups these phonemes into syllables. Once this has been done, the theoretical stress can be calculated according to well defined Spanish grammatical

rules [6,3]. Note that the new transcription output will look very similar to the original Spanish spelling, because Spanish is so much more phonetic than other languages such as English or French.

By default the phonemic transcription used in this project includes many other features such as syllabification, accentuation and various statistics functions. The statistics functions return counts of syllables, words, unique words, individual phonemes and total phonemes. Many of these counts will be utilized by the evolutionary algorithm.

2.2 Synonyms in Microsoft Office

Who loves to hate Microsoft? A powerful and primary programmatic element of our project is a COM[1] interface with Microsoft Word. Microsoft has invested considerable time and money into the linguistic capabilities of their products. No matter what stance the readers may have towards this company and its products, we wanted to illustrate the tremendous benefit that one can realize by reusing what is already available as opposed to reinventing the wheel. Furthermore, the ability to lookup synonyms and antonyms for almost any given word in almost any given language seemed beneficial enough to justify the use of this common commercial product.

Our function that accesses this benefit is less than 100 lines of C++ code and is flexible enough for us to change the language for which synonyms are retrieved with one single parameter. An example in Spanish shows how Microsoft's API returns a series of synonyms for each word queried. These series, though less structured and consistent, are somewhat analogous to Wordnet's *synset* [7]. In this case a search for "pago" will return (as formatted by our interface):

- [wdNoun, desembolso (desembolso, reembolso, cancelación, liquidación, dispendio, entrega)]
- [wdVerb, pagar(pagar)]
- [...]

These sets of synonyms are organized by their grammatical function, which can be filtered. The importance of this is obvious: if you are replacing a noun in a sentence with a synonym, you do not want to replace it with a verb.

2.3 Statistical Language Modelling n-Gram Models

Statistical Language Modelling (SLM) [8] is the attempt to capture regularities of natural language for the purpose of improving the performance of various natural language applications. By and large, statistical language modelling amounts to estimating the probability distribution of various linguistic units, such as words, sentences, and whole documents.

Ironically, the most successful SLM techniques use very little knowledge of what language really is. The most popular language models (n-grams, that are

[1] Component Object Model.

consecutive sequences of n words in a text or sentence) take no advantage of the fact that what is being modelled is language - it may as well be a sequence of arbitrary symbols, with no deep structure, intention or thought behind them.

2.4 Evolutionary Algorithms

We propose the use of evolutionary algorithms (EAs) to deal with the generation of alliterations in Spanish. In the case under consideration, the main advantage we can find in evolutionary algorithms is that they do not need specific rules to build a solution, but rather a form of quantitative evaluation.

Evolutionary techniques have been shown in the past to be particularly well suited for the generation of verse. The work of Manurung [9] and Levy [10] proposed different computational models of the composition of verse based on evolutionary approaches. In both cases the main difficulty lay in the choice of a fitness function to guide the process. Although Levy only addressed a simple model concerned with syllabic information, his overall description of the architecture in terms of a population of poem drafts that evolve, with priority given to those drafts that are evaluated more highly, is an important insight. The work of Manurung addresses the complete task, and it presents a set of evaluators that grade the candidates solutions according to particular heuristics. An important conclusion to draw from these efforts is the suitability of evolutionary techniques for natural language generation tasks in which the form plays a significant role, to the extent of even interfering with the intended content.

3 An Evolutionary System for Alliterations

The work presented here is intended to be an evolutionary generator of alliterations from a given sentence and the phoneme that must be repeated in the intended alliteration. The operation of the system is based on the phonetics of the initial sentence and the use of synonyms for the words in the phrase. The algorithm produces a solution phrase with a similar meaning than the initial one, but formed by words containing the specified phoneme.

In this evolutionary algorithm, each word of a sentence is treated as a gene. The initial population is generated randomly, using for each word from the initial text a synonym obtained from Microsoft Word. The system works over this population for the number of generations determined by the user. In each generation two typical genetic operators are used: crossover and mutation. Finally, at the end of each generation each sentence is evaluated and a selection of the population is passed to the next one, in such way that the phrases with a higher fitness value have more possibilities of being chosen.

3.1 Data Representation and Genes

The population with which the system operates is made up of various instances of the sentence that has to be alliterated. The words of this phrase are considered to

be the genes. Each word of the sentence has an associated tag indicating its part of speech. So, the synonyms searched for are only the ones that correspond to the same part of speech as the corresponding word, thus avoiding inconsistencies.

When looking for the synonyms of a given word, Microsoft Word returns a list of synonym sets or *synsets*. To avoid significant departures from meaning, a unique *synset* is considered for each word of the initial sentence. The mutations of each word will be carried out with this stored *synset*, including the initial word if it was not in the first *synset*.

An example of input sentence could be

```
"El sonido con que rueda la profunda tormenta"
```

It is an alternative version of the well-known alliteration from José Zorrilla *"El ruido con que rueda la grave tempestad"*. The associated part of speech tags of this sentence are

```
[other,noun,preposition,conjunction,verb,other,adjective,noun]
```

3.2 The Genetic Operators

Two genetic operators have been used: crossover and mutation.

For the crossover operator, two sentences are selected randomly and crossed by a random point of their structure, so that each of the children will have part of each of the parents' traits.

In the case of the mutation operator, some of the genes are chosen randomly to be mutated. If a word is selected to mutate, the system asks Microsoft Word for synonyms corresponding to the same part of speech as the word to mutate. One synonym is randomly selected to substitute the previous word. The *synsets* are used as described in Section 3.1.

A special case in the mutation are the articles "el", "la", and their corresponding plural forms. Even though they do not have synonyms, it is necessary to exchange them with their opposite gender form. This is because the *synsets* for nouns do not distinguish between the feminine and masculine gender. In order to obtain coherent results in the final sentences we have to mutate the articles from "el" to "la" and vice versa, using the same mutation probability applied to the rest of words.

3.3 The Fitness Function

The key to the evolutionary algorithm lies in the choice of the fitness function. A sentence in the population is considered better or worse depending on two factors: the number of appearances of the desired phoneme and the coherence of the sentence.

The phonemic fitness value of a phrase is calculated as the percentage of appearances of the phoneme out of the total number of phonemes in the phrase[2]. In

[2] Even when the common definition of alliteration requires the target phoneme to appear at the start of the word, there are some exceptions. We have chosen not to take into account the position of the phoneme in this approximation.

the extreme, the maximum value for a sentence would be 100% if every phoneme in the sentence were the same. Of course this would be impossible, but that measure is enough to study the resulting phrases provided by the evolutionary algorithm.

The coherence fitness value is calculated using bigrams. For each pair of consecutive words in the sentence we query Google to get the number of appearances of the pair in the Internet. This value is normalized using the number of appearance of each of the words separately compared to the number of times they appear side by side, as shown in Formula 1, and the coherence of all the bigrams is sumed up to obtain the coherence fitness of the whole sentence.

$$coherence(w1, w2) = app(w1 + w2)/(app(w1) + app(w2)) \quad (1)$$

The final fitness value for the whole sentence is the sum of the phonetic and the coherence fitnesses. With this measure, we are trying to obtain the most alliterations in a phrase with the minimum loss of coherence.

3.4 Example of Execution

For the operation of the algorithm three linguistic inputs and four genetic parameters are required.

For linguistic inputs we consider the initial sentence that is going to be alliterated, the tags indicating the part of speech of each word in the sentence, and the phoneme with which we desire to alliterate. The genetic parameters required are the number of individuals in the population, the number of generations to be executed, and the crossover and mutation operator probabilities.

Considering the example and the initialization exposed in Section 3.1, some sentences from the initial population could be

```
El sonido con que vira la recóndita precipitación
El sonido con que gira la oscura inclemencia
```

For the crossover operator pairs of sentences are selected randomly to be crossed by a random point of their structure. In this example, if the previous sentences are crossed by the fifth word, the result could be the following:

```
El sonido con que vira la oscura inclemencia
El sonido con que gira la recóndita precipitación
```

For the mutation operator, some words from the sentences are randomly chosen to be mutated. Mutation consists then in substituting the word for one of its synonyms from Microsoft Word, always corresponding to the same part of speech. In this example, if the word "sonido" from the first sentence is selected to mutate, we must choose its synonym from the stored *synset* provided by Microsoft Word:

```
[eco,resonancia,retumbo,cacofonía,asonancia,eufonía,fama,monotonía]
```

In this example, the chosen word has mutated to "retumbo", and it is the only word that mutates.

After crossover and mutation the fitness values for the two sentences of the example are given in Table 1. This table also includes the fitness values of the original alliterative text by José Zorrilla in order to compare our output with one "ideal" result. We are supposing that the searched phoneme is *'r'*. For the coherence fitness, the sentences are evaluated using bigrams. If we take as example the pair of words "El retumbo", Google returns a count of 35.100.000 for the word "El", 80.000 times for "retumbo" and 1.390 times for both words as a phrase. So, for this pair of words the coherence fitness is 3,95E-5.

Table 1. Fitness values for examples of sentences

Sentence	Appearances of phoneme	Total number of phonemes	Phoneme Fitness	Coherence Fitness	Total Fitness
El retumbo con que vira la oscura inclemencia	2	37	5.41%	0.89%	6.30%
El sonido con que gira la recóndita precipitación	3	41	7.32%	4.92%	12.24%
El ruido con que rueda la ronca tempestad	2	33	6.06%	4.37%	10.43%

4 Experiments and Results

To test the feasibility of the idea of alliterating words within a sentence using EAs, we have carried out experiments taking the initial sentence "El sonido con que rueda la profunda tormenta" and setting this as our goal. We have executed several experiments using different population sizes and number of generations. The crossover and mutation probabilities are constant in all the experiments. The searched phoneme is "'rr".

In Table 2 we can see the numerical results of the experiments, and in Figure 1 its graphical representation. This chart illustrates the correlations between the rise in population size and the rise in phonemic fitness with the decrease in coherence fitness regardless of the number of iterations per example sentence. For each resulting sentence the graph shows the number of individuals and iterations, indicated by the bar on the left, to be measured against the fitness measures on the bar to the right. The phonemic fitness, the line plotted with squares, rises as the population size rises; while at the same time the coherence fitness drops, denoted by the triangle plotted line.

With small populations (10 and 25 individuals) the coherence and phonemic fitness values are quite similar. Sometimes the coherence fitness value is greater; sometimes the phonemic. However, as the population size increases the phonemic fitness raises while the coherence drops. This means that the individuals with the most total fitness are the ones with higher phonemic fitness, and are not

Table 2. Table of numerical results

Population size	Number of generations								
	10			25			50		
	Phon. Fitness	Coh. Fitness	Total fitness	Phon. fitness	Coh. fitness	Total fitness	Phon. fitness	Coh. fitness	Total fitness
10	3,03%	5,17%	8,20%	5,56%	4,60%	10,16%	5,56%	5,19%	10,75%
25	8,11%	4,63%	12,74%	8,11%	4,63%	12,74%	8,11%	4,63%	12,74%
50	10,53%	0,40%	10,93%	10,81%	0,40%	11,21%	10,47%	0,35%	10,81%
100	10,37%	0,44%	10,80%	10,46%	0,39%	10,85%	10,53%	0,41%	10,93%
300	10,53%	0,41%	10,93%	10,81%	0,40%	11,21%	10,81%	0,40%	11,21%

necessarily good from the point of view of coherence. This problem can be seen easily in Figure 1, where both fitness values start more or less at the same level, but as the phonemic increases the coherence tends to decrease. In our first approximation both coherence and phonemic fitness values have equal weights in the total fitness, but the algorithm should be adjusted to find the proper weights for each one and improve the results.

From the point of view of the coherence fitness, the values obtained are quite low. The maximum value for a pair of words in the sentence would be 1 if the number of appearances of both words together is equal to the sum of the appearances of each of the words separately. This is almost impossible. In addition, when one of the words is a very common one - for example "el" -, the number of appearances is much higher than the number of appearances of the bigram, provoking low values for the coherence fitness.

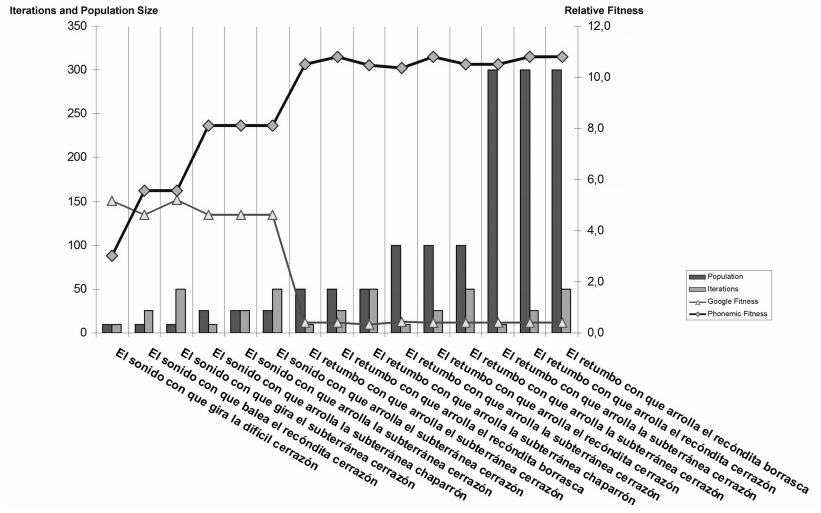

Fig. 1. Graphical representation of the results

In addition, we have also obtained strange results from the point of view of coherence. For example, in a generation of the system, we found that the sentence "El sonido con que gira el subterránea cerrazón" has more coherence fitness than "El sonido con que gira la subterránea cerrazón", which is the most correct one. This problem is due to the fact that we are normalizing the fitness of each pair of words and then summing. This means that pairs of words that have many appearances according to Google are considered to have the same weight as the combinations with fewer appearances. In this case, the fact that "gira el" is more common than "gira la" is enough to give more fitness to the first sentence even when "la subterránea" has many more appearances than "el subterránea". It is clear that we must study ways to prevent this type of incongruence.

5 Conclusions and Future Work

Our experiment with evolutionary algorithms and alliteration denotes obvious strengths and weaknesses.

An important difficulty in the task of generating alliterative sentences is the fact that it requires taking into account information at different levels of linguistic competence, such as phonetics, morphology, syntax and semantics. In accepted examples of good alliteration, the information at these different levels interacts in non-trivial ways to produce the desired literary effect. Although the evolutionary approach presented in this paper is still far from reproducing the competence of even a mediocre human writer, it has shown that a combination of mutation operators and fitness functions of varying complexity might be a manageable way of modeling these interactions computationally. This we consider the most important strength of the approach. Given the complexity of the interactions identified in examples from the literature, it seems improbable that acceptable results be obtained by means of simpler algorithmic or rule-based approaches. Of our experiment we are most proud of the broad, flexible base that we have laid and the promise that this base holds for interesting future work. The weaknesses, though not discouraging, have illustrated some areas that we need to improve. Many of these are in the process of being addressed.

The addition of the Google searches did improve the overall semantics slightly. Further refinement of the Google searches - for instance by extending it from bigrams to trigrams - should improve the quality of the sentences. The search "el difícil" and "difícil fama" may prove frequent, incorrectly validating sentences like "el difícil fama". This would be avoided using trigrams. On-line queries of another more precise Spanish corpus such as [11] might also yield better results.

The addition of a morphosyntactic disambiguator may help providing the part of speech tags as input of the system and checking output sentences for overall validity. We may implement our own, or simply query The Grupo de Estructuras de Datos' Morphosyntactic Disambiguator online in [12].

For the computational evaluation of sentence quality we considered two variables, alliteration versus frequency, with equal weight. This may be refined. Most people prefer a completely mundane sentence that makes sense to line of

alliterative drivel. Our future work will strive to find an acceptable balance between these extremes, and then apply this balance to the evolutionary algorithm.

In this work we have considered that an alliteration is based in the repetition of a phoneme, but it can also be provoked by the repetition of syllables. Also we can consider apart vowels and consonant, or even the repetition of two or more phonemes.

More mutation operators can also be added. For adjectives, to add another adjective with the same meaning is a good option. For instance, "la grave ronca tempestad" is a very common structure in Spanish. In the same line, another operator that duplicates structures can also provides interesting results. For instance, "rueda la profunda tormenta, grave tempestad", where the pair adjective-noun has been duplicated.

Once the aforementioned weaknesses have been attended to, we will then apply what we have learned to more patterns in language other than alliteration, such as rhythm and rhyme.

References

1. Jakobson, R.: Language in Literature. Cambridge: Belknap Press of Harvard UP (1996)
2. R.A.E.: Ortografía de la lengua española. Espasa Calpe, Madrid (1999)
3. Hammond, R.: The Sounds of Spanish: Analysis and Application (with Special Reference to American English). Somerville: Cascadilla Press (2001)
4. Navarro, T.: Manual de pronunciación española. Consejo Superior de Investigaciones Científicas, Instituto "Miguel de Cervantes", Madrid (1977)
5. Robinson, J.: Colors of poetry: Computational deconstruction. Master's thesis, Athens, Georga, USA, University of Georgia (2006)
6. Llorach, E.: Gramática de la lengua española. Espasa Calpe, Madrid (2003)
7. Miller, G.A.: Wordnet: a lexical database for English. Commun. ACM **38** (1995) 39–41
8. Rosenfeld, R.: Two decades of statistical language modeling: Where do we go from here? Proceedings of the IEEE **88** (2000)
9. Manurung, H.: An evolutionary algorithm approach to poetry generation. PhD thesis, School of Informatics, University of Edinburgh (2003)
10. Levy, R.P.: A computational model of poetic creativity with neural network as measure of adaptive fitness. In: Proccedings of the ICCBR-01 Workshop on Creative Systems. (2001)
11. Davies, M. (www.corpusdelespanol.org)
12. (http://www.gedlc.ulpgc.es/investigacion/desambigua/morfosintactico.htm)