ECAI 2012
Luc De Raedt et al. (Eds.)

doi:10.3233/978-1-61499-098-7-115

Markov Constraints for Generating Lyrics with Style

Gabriele Barbieri[1,2] and **François Pachet**[1] and **Pierre Roy**[1] and **Mirko Degli Esposti**[2]

Abstract. We address the issue of generating texts in the style of an existing author, that also satisfy structural constraints imposed by the genre of the text. We focus on song lyrics, for which structural constraints are well-defined: rhyme and meter. Although Markov processes are known to be suitable for representing style, they are difficult to control in order to satisfy non-local properties, such as structural constraints, that require long distance modeling. We show that the framework of Constrained Markov Processes allows us to precisely generate texts that are consistent with a corpus, while being controllable in terms of rhymes and meter, a result that no other technique, to our knowledge, could achieve to date. Controlled Markov processes consist in reformulating Markov processes in the context of constraint satisfaction. We describe how to represent stylistic and structural properties in terms of constraints in this framework and we provide an evaluation of our method by comparing it to both pure Markov and pure constraint-based approaches. We show how this approach can be used for the semi-automatic generation of lyrics in the style of a popular author that has the same structure as an existing song.

1 INTRODUCTION

The style of a text is an important factor that determines its quality, its legibility and its identity. The idea of generating new texts in the style of an existing author has been popular since the invention of Markov processes, that have shown to capture, at least in a first approximation, elements of the style of a corpus [8, 9]. Markov processes represent faithfully local properties of sequences, at varying orders, which makes them well-suited for such a task.

However, a text is not just a Markovian sequence, and, notwithstanding the issue of meaning, it has been shown that texts also exhibit statistical long-range correlations [1]. For instance, poems or song lyrics often have rhymes or metric constraints, that are not always defined as local properties. They induce long-range dependencies that violate the hypothesis of short-term memory of Markov processes and demand long distance modeling. As a consequence, most approaches in automatic generation of stylistically imitative texts based on Markov models fail to capture higher-level properties of texts, which limit their use for practical applications, such as machine translation [21] or automatic summarization [2].

The recently proposed framework of constrained Markov processes (CMP), introduced in [14] and [15], addresses precisely the issue of imposing constraints to a Markov process, to control it without the need of other tools, such as, e.g., a second, high-order Markov process. CMP is a technique to generate structured Markov sequences by reformulating Markov Processes as constraint satisfaction problems. We show in this paper that lyrics generation can be formulated and solved in the CMP framework by expressing style as a set of *Markov constraints*, and properties of grammaticality, rhymes, meter and even to some extent, semantics, as constraints in that framework. We evaluate our approach by asking humans to rate texts generated by our technique against text generated using other techniques.

In Section 2, we review previous approaches on poetry generation. In Section 3, we describe constrained Markov processes. In Section 4, we show how structural constraints can be formulated as unary constraints in CMP. Section 5 describes the evaluation. In Section 6, we give the details of a semi-automatic generation of lyrics in the style of Bob Dylan that satisfy the structural constraints of an existing song (Yesterday, by The Beatles).

2 RELATED WORKS

Many Natural Language Generation systems have been proposed to generate *technical texts* in a given style. For example, Skillsum [18] generates feedback reports about people's literacy and numeracy skills. In Skillsum, the style is governed by rules based on a manual analysis of the corpus to imitate. Iconoclast [16] generates patient information leaflets. The style is parameterized by low-level parameters, such as paragraph length or the use of specific technical terms. However these system do not handle the structural constraints specific to poetry (rhymes, meter).

On the other hand, poetry generation systems generate well-formed texts in terms of meter and rhyme but they usually do not address the stylistic dimension. McGonagall [17] uses genetic algorithms to generate texts that are syntactically correct, following imposed meter patterns and which broadly convey a given meaning. The WASP system [5] and its evolution ASPERA [6] produce Spanish poetry from several input data, such as the choice of vocabulary and a poem type. In the field of lyrics generation, Tra-la-lyrics [13] automatically generates percussive lyrics to accompany a given piece of music. Titular [20] automatically generates novel song titles to inspire songwriter in lyrics production. None of these systems address explicitly the problem of style.

To our knowledge, the only systems that generates literary texts with style are the poetry generator proposed in [7] and RKCP (Ray Kurzweil's Cybernetic Poet) [9]. In [7], the style is modeled as a "blending principles choice". Although the authors claim that an existing style can be approximated by carefully tuning these blending principles, they do not explain how to do it in practice. RKCP is the closest system that looks at solving the same problem that we have, and uses Markov processes trained on a corpus of given authors to generate poems in the same style. The generation is controlled by parameters such as the type of stanza and rhymes. However, RKCP uses a *generate-and-test* approach to validate the verses generated by the Markov processes. This approach is both incomplete and unbounded

[1] Sony CSL Paris, France, {barbieri,pachet,roy}@csl.sony.fr
[2] Università di Bologna, Italy, {gbarbieri,desposti}@unibo.it

in terms of execution time.

In this paper, we present a technique that, given a corpus, ensures real time generation of verses that satisfy structural constraints. We first describe the technique and, in Section 4, we show how it can be used.

3 FINITE-LENGTH MARKOV PROCESSES WITH CONSTRAINTS

A Markov process is a random process with a short-term memory: it generates states with a probability that depends only on a fixed, finite number of past states. The number of past states used to define the distribution of future states defines the *order* of the Markov process. A Markov process M of order n can be estimated from a corpus using *Maximum Likelihood Estimation* (MLE), by computing the relative frequencies (RF) of each *n-gram*, i.e., continuous sequence of n words that appear in the corpus. More sophisticated techniques, such as smoothing techniques [8], can be used, to deal with the *zero-frequency* problem caused by the sparsity of the RF estimate. We decided to use MLE, because it gives an acceptable estimate of the corpus used in this paper. However our approach is independent of the way the process M is estimated, and is therefore compatible with any of these techniques.

A Markov process can be used to generate new word sequences with a *random walk* procedure consisting of drawing random states according to the word transition probabilities. Each word w_i is generated with probability $P_M(w_i|w_{i-n},\ldots,w_{i-1})$ depending only on the $n-1$ words previously generated. For instance, the order-1 Markov model of the following corpus:

- *Clay loves Mary*
- *Mary loves Clay*
- *Clay loves Mary today*
- *Mary loves Paul today*

is represented in Figure 1. A random walk could produce sequences such as “loves Mary loves Clay loves”, or “Paul today”.

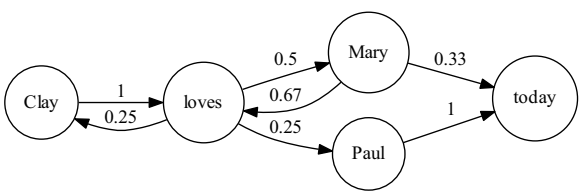

Figure 1. An order-1 Markov process learned from a corpus composed of five words.

Markov processes do not provide any control on the structure of the generated sequences. For instance, a constraint that imposes the *last word* of a 4-word sequence to be “today” and a constraint that imposes to the *first word* to rhyme with “today” create a long distance dependency between the first and the last word of the sequence. Indeed the only four-word sequences that satisfy these constraints are “Clay loves Mary today” and “Clay loves Paul today”. Therefore, the first word of the sequence *must be* “Clay”, excluding “Mary”, “Paul”, “loves” and “today” as possible first states. This implicit dependency cannot be represented in the initial Markov model. Obviously, the model generates texts that do not necessarily rhyme. Generate-and-test can be used to filter out incorrect sequences, but without any guarantee that correct sequences will be generated.

The framework of Constrained Markov Processes allows precisely to solve this issue, i.e., to generate Markov sequences that satisfy explicit control constraints, such as the rhyme constraint in the previous example.

Following [14], the Markov process is reformulated as a constraint satisfaction problem P. The sequence to generate is represented as the sequence of finite-domain constrained variables of P. The transition probabilities are represented as *Markov constraints* holding on these variables. Control constraints are represented as arbitrary constraints. The same authors showed, in [15], that if the control constraints are unary (i.e., they hold on a single variable) , the initial Markov process M can be transformed in a constrained Markov process $\tilde{M}$ with the following properties:

1. $\tilde{M}$ generates exactly the verses that satisfy the control constraints and,
2. the admissible verses are generated with the same probabilities in M and $\tilde{M}$ up to a constant factor.

$\tilde{M}$ is obtained in two steps. The first step makes the constraint satisfaction problem P arc-consistent [10]: for each variable, the values that violate at least one constraint are removed, until a fixed-point is reached. An intermediary Markov process is built from M and P by zeroing state transitions that are filtered out by the arc-consistency procedure, i.e., the transitions that correspond to the removed values. This step guarantees that only correct sequences are generated (Property 1. above). The first step affects the transition probabilities, therefore, a second step is applied that adjusts the local transition probabilities to get the initial global probability distribution of M (Property 2.). $\tilde{M}$ is the resulting process. For more details about the construction of $\tilde{M}$ and the proof that the obtained model satisfies 1. and 2., see [15].

Coming back to the previous example, the Markov process M_{ex} is transformed in the process $\tilde{M}_{ex}$ in Figure 2.

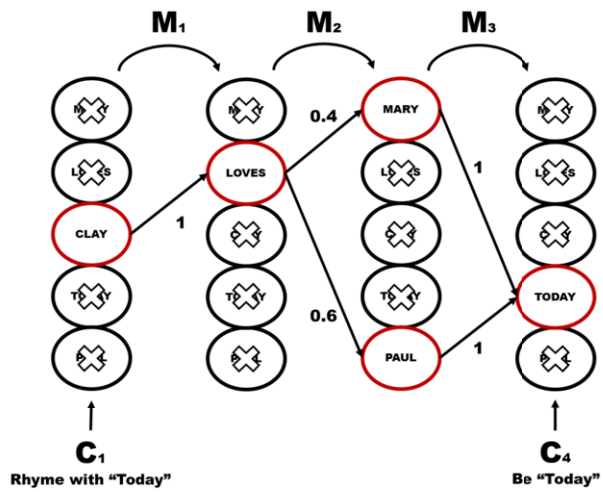

Figure 2. A constrained Markov process $\tilde{M}_{ex}$ that generates verses composed of 4 words and rhymes with the word *yesterday*. $\tilde{M}_{ex}$ and M_{ex} have the same probability distribution. M_1, M_2 and M_3 represent the Markov constraints, C_1 represents the control constraint “rhyme with today”, C_4 represents the control constraint “be today”. The arrow labels indicate the transition probabilities.

4 UNARY CONSTRAINTS FOR LYRICS GENERATION

In this section, we challenge the problem of representing most of the structural aspects of poetic texts [17]: rhyme, meter, syntactic correctness, as well as semantic relatedness, as unary constraints in the framework of CMP.

4.1 Meter and Rhyme

We implement rhyme and meter respectively by *rhyme constraints* and *rhythmic templates*. These constraints are hard constraints: they ensure that the generated verses *fully* respect an imposed meter and an imposed rhyme structure.

4.1.1 Rhyme Constraints

Rhymes are naturally represented as unary constraints on the ending words of verses. Given a target word s, a *rhyme constraint* is satisfied by all the words in the corpus that rhyme with s, according to its phonetic spelling. In this paper, we chose the CMU pronunciation dictionary [19] to extract the phonetic spelling of words. Vowels are tagged with their lexical stress (0 for non stressed vowels, 1 for stressed vowels). For example the spelling of the word "today" is [T, AH0, D, EY1]. In our system two words are said to rhyme if the suffixes of their spellings from the last stressed vowels are the same. For example "today" ([T, AH0, D, EY1]) rhymes with "pray" ([P, R, EY1]).

4.1.2 Rhythmic Templates

Meter is the position of the stressed syllables in a verse. Using the the CMU dictionary, each word is labeled with a rhythmic tag, defined by the sequence of the lexical stresses. For instance, the rhythmic tag of "today" [T, AH0, D, EY1] is 01. The meter of a verse is composed by the rhythmic tags of its words, e.g., the meter of "Innocence of a story I could leave today" is [101, 1, 1, 10, 1, 1, 1, 01]. Meter can be imposed on a verse by a sequence of unary constraints, called a *rhythmic template*. Each unary constraint imposes a rhythmic tag on a word.

A corpus (for instance the lyrics of Bob Dylan) provides a collection of rhythmic templates induced by the meter of each verse.

4.2 Syntax and Semantics

A poetic text is of course not simply a concatenation of words that satisfy some formal requirements, but must be syntactically well-formed and convey some meaningful message. We show that unary constraints are, again, well-adapted to ensure, at least partially, syntactical correctness. Although we do not contribute here to the general issue of representing meaning, we show that unary constraints can impose, to some extent, a semantic bias to the generated verse. Following [5] and [12], we use *part-of-speech templates* to ensure syntactical correctness and *semantic constraints* to approach the problem of meaning. These constraints do not ensure that the generated verses are always syntactically correct and clearly communicate a message. So these constraints define the two questions we will ask in Section 5.

4.2.1 Part-of-Speech Templates

Syntactic correctness is enforced by the combination of Markov probabilities and a *template-based* approach [4]. A *part-of-speech template* is a sequence of *part-of-speech* (POS), such as Verb, Noun, etc. In this experiment, the corpus is tagged using the Stanford Log-Linear part-of-speech tagger [22], tagging each word with one or more POSs (for instance the word "run" is tagged with the POSs Noun and Verb). A part-of-speech template is represented as a sequence of unary constraints. For example the template [PRP, VBD, IN, PRP, RB] (Personal pronoun, Verb past tense, Preposition, Adverb) is satisfied by the verse "she knocked upon it anyway".

Similarly to rhythmic templates, a corpus induces a collection of POS templates. However, according to [20], we only retain the templates that appear in at least two verses in the corpus, to reject incorrect templates that may occur due to errors in the POS tagging procedure.

4.2.2 Semantic Constraints

The dimension of meaning is, of course, the most difficult computational task to achieve in general. We show that our framework is well-adapted to enforce a semantic relatedness between a target concept and the generated verse. This is done by imposing *semantic constraints* on the word on which the semantic bias has to be enforced. For any word w in the corpus, the semantic constraint is satisfied by the n words in the corpus most related to w. We use the *Wikipedia Link-Based Measure* [11], to compute semantic relatedness. We decide for this measure because it is an effective compromise between ease of computation and accuracy. Note here that a more flexible technique could be to use cardinality constraints, holding on all the words of the verse. This solution does not fit with the CMP approach, which considers only unary constraints, but is envisaged as future work.

5 EVALUATION

This section is an evaluation of our technique, to investigate to what extent the constraints proposed in Section 4.2 ensure that the generated texts are syntactically correct and semantically related to the concepts imposed by the user.

Firstly, we compare our technique (hereafter *CM*) against a pure Markov approach (*PM*) that generates texts using only the initial Markov model without control constraints, to evaluate how much the constraints increase the syntactic correctness and semantic relatedness.

Secondly, we compare our technique against a pure constraint solving approach (*PC*) that generates texts using only the control constraints, because this approach is similar to the approaches in [5], [6], [12] and [20]. We cannot compare directly these techniques because they differ in many ways from *CM* (for instance the choice of the language, the dictionary and the semantic relatedness measure used) and these parameters affect the quality of the generated texts, as pointed out in [5]. Therefore we use the same constraints in *PC* and *CM* to ensure a fair evaluation.

Automatic evaluations of both syntactic correctness and semantic relatedness are still open problems (see respectively [3] and [23]) and to our knowledge there is no reliable method to automatically evaluate these properties. An automatic evaluation of semantic relatedness is presented in [17], but this evaluation requires an optimal solution

to the problem (for example a human-generated text). We cannot exploit the proposed evaluation because we are interested in generating novel material instead of rewriting existing lyrics.

We propose an empirical evaluation, by asking humans to rate syntactic correctness and semantic relatedness of texts generated by *CM*, *PM* and *PC*.

5.1 Generation of the Test Poems

For each technique, we generate 16 poems, to make a total of 48 poems. Each of the 16 poems is entitled with a one-word title, that defines the concept to which the poem should be related, and four verses. We manually select the 16 titles (listed in Table 1). The rhyme structure is ABAB.

Table 1. The 16 titles of the poems. Each title is used for three poems, generated using the three different techniques (CM, PM and PC). A title imposes a topic to the corresponding poems.

television	religion	politics	sky
jealousy	envy	joy	laugh
love	moon	sun	music
paradise	hell	peace	war

We train an order-2 Markov process M_{Dylan} on a corpus composed of 12408 verses by Bob Dylan, hereafter referred to as the *Dylan corpus*. We chose Bob Dylan because he is a fine and prolific songwriter, with a personal and consistent style. The Markov model made up from the complete set of his lyrics (see Table 2) is not too sparse while of a reasonable size.

Table 2. Statistics of the Dylan corpus.

Number of verses	12 408
Total number of words	96 089
Number of different words	7 600

For each of the 16 poems, we define a set of control constraints as follows. The Dylan corpus provides a collection of rhythmic templates and a collection of POS templates as explained in Section 4. For each verse, a rhythm template and a POS template of the same length are randomly selected. In each verse, one position is randomly chosen among the open tags (i.e., adjectives, nouns, adverbs, and verbs) of the POS template. We impose the corresponding word to be semantically related to the title word. We specify rhyme constraints to ensure that the first (respectively second) and third (resp. fourth) verses rhyme with the word w_A (resp. w_B). w_A and w_B are randomly chosen from the set of words that rhyme with at least 10 other words of the corpus.

The *CM* approach generates the 16 poems with a constrained Markov process combining M_{Dylan} with the control constraints. The *PC* approach generates the poems by drawing each verse randomly in the space of the verses that satisfy the control constraints. The *PM* approach generates the poems with a naive random walk on M_{Dylan}. The length of each verse is randomly chosen between 5 and 8 words. For each approach, we generate automatically poems such as the ones shown in Table 3.

5.2 Empirical Evaluation

Following [5], we asked a team of 12 volunteers to rate syntactic correctness and semantic relatedness of each poem. The evaluators were instructed to rate syntactic correctness according to the following scale: 1) the poem is strongly grammatically incorrect, 2) the poem presents some grammatical errors, 3) the poem is almost or completely grammatically correct. The evaluators were instructed to rate semantic relatedness according to the following scale: 1) the poem is semantically unrelated to the title, 2) the poem is weakly semantically related to the title, 3) the poem is strongly semantically related to the title.

The results of these evaluations are shown in Figure 3.

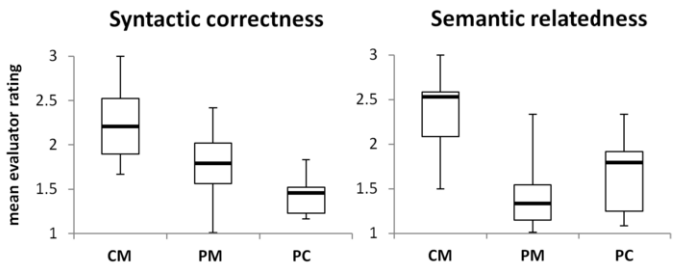

Figure 3. Box plots illustrating the distribution of "syntactic correctness" and "semantic relatedness" ratings assigned by evaluators to *CM*, *PM* and *PC* outputs. Boxes represent the interquartile ranges, with the medians indicated by thick black lines. Whiskers on either side span the minimum and maximum values of each distribution.

5.3 Discussion

Results clearly show that the *CM* technique performs better regarding syntactic correctness and semantic relatedness. The results are statistically significant for both experiments (Mann-Whitney test with $p < 0.05$, multiple tests are corrected using the Bonferroni method). Surprisingly, *PC* is the lowest-preforming technique with respect to syntactic correctness, whereas the results performed by *PM* and *CM* are more similar. This may be due to the fact that the POS tags we use are not detailed enough. This is clear when observing the second verse of the poem in the examples shown in Table 3. In this verse there is a clear mistake: "you wakes". This mistake is because the tag PRP (personal pronoun) does not contain any information about the person. Of course, a better POS tagging will improve the grammaticality rating of *PC*, however *CM* will also benefit from such as an improvement. It is interesting to observe that mistakes like "you wakes" are prevented by the use of a Markov model. Therefore the interaction between the information carried by the Markov model and the POS templates explains the high grammatical score of *CM*.

As expected, *PM* generates semantically unrelated poems, because no information about semantics is provided to the generator. The fact that the average semantic relatedness is not exactly 1 is due to the presence, by chance, of words related to the desired concept, such as, in the *PM* example in Table 3, the word "heard", related to concept of "music". The fact that the meaningfulness rating of *CM* is better than the rating of *PM* is again probably due to the information carried by the Markov model.

6 AN EXAMPLE: LYRICS IN THE STYLE OF BOB DYLAN

This section describes an application of our technique to the semi-automatic generation of lyrics in the style of Bob Dylan with an imposed structure. One example is shown in Table 4. Many examples in the style of more than 60 popular authors can be found at `http://www.csl.sony.fr/MarkovCt/lyrics/`.

Table 3. Examples of poems used in the evaluation. The first poem is generated using constrained Markov processes (*CM*). The second poem is generated using a pure constraint solving approach (*PC*). The third poem is generated using a pure Markov process approach (*PM*). Each poem is intended to be related to the concept "Music". Note that, unlike *CM* and *PC*, the *PM* poem do not satisfy the structural constraints, such as the rhyme

MUSIC (*CM*)	**MUSIC** (*PC*)	**MUSIC** (*PM*)
There is a note in his eyes	Its pine this notes from all tries	Swimmy from the cold eyes of Judas on
He backs the beat of the key	You wakes no band like all three	Handful of rain tempting you to be heard
Down the song in my eyes	So half beat worth whose ties	Therefore i remain at my window wishing
You back the beat of the sea	That leaves those beats of this flea	Nighttime is the one said

Table 4. The lyrics of the song Today, generated using constrained Markov processes. The style of the lyrics is clearly Dylanesque (see e.g. the 8th verse "Wind is blowing in the light in your alleyway", made up from words of the hits "Blowing in the wind" and "Subterranean homesick blues", that Dylan fans would easily recognize) . Each verse follows the rhythm and the rhyme structure defined by the Beatles' song "Yesterday". The words on which a semantic constraint is imposed are in italic. The corresponding constraints are listed on the right column of the table.

Today (lyrics generated using the constrained Markov approach)	Semantic constraints
Innocence of a story I could leave *today*	*today* imposed
When I go down in my hands and pray	
She knocked upon it anyway	
Paradise in the dark side of love it is a sin	*paradise* imposed by the semantic constraint related to "pray"
And I am getting weary looking in	
Their promises of paradise	*paradise* is imposed
Now I want to know you would be spared this day	
Wind is blowing in the light in your alleyway	
Innocence in the wind it whispers to the day	*innocence* is imposed
Out the *door* but I could leave today	*door* imposed by the semantic constraint related to "knocked"
She knocked upon it anyway	equal to the 3rd verse, as in the original song.

We impose the same rhyme and meter structure as that of the song "Yesterday" by the Beatles. In other words, we want to map Bob Dylan's songwriting style onto the structure of "Yesterday", very much like one can map a texture onto an existing shape.

The generation process is semi-automatic. Although a fully automatic generation process is possible (as shown in section 5), the interaction with a human user improves the global coherence of the lyrics. The verses are generated one by one, prompting the user at each step to select one verse out of five different candidates.

The initial Markov process is M_{Dylan}, as described in Section 5.1. The constraints imposed on the song to enforce rhyme are as follows. Initially, no constraint is set. The rhyme structure of Yesterday is AAABBCAAAAA, as shown in table 5. This implies that after the first verse v_1 is selected by the user, all the verses tagged A (verses v_2, v_3, and v_7 to v_{11}) will be generated with a unary constraint that forces them to rhyme with v_1. Similarly, the verse v_4 is generated with no rhyme constraint, but the verse v_5 will be constrained to rhyme with v_4.

The rhythm constraints are enforced by rhythmic templates defined as explained in Section 4.1.2, and according to the rhythmic structure of "Yesterday" (see table 5 for the complete structure).

For each verse, a POS template is drawn randomly from the POS templates induced by the corpus that have the length than the corresponding verse in Yesterday. For instance, the POS templates for the first verse are those templates with eight words.

The song is entitled intentionally "Today", a word that has the same meter as "away", the last word of the first Yesterday's verse. Accordingly, we impose the word "today" as the last word of the first verse. As a consequence, verses v_2, v_3, and v_7 to v_{11} are constrained to rhyme with "today". Note that, after imposing "today" at the end of the first verse, the candidate POS templates are those that: have eight words; appear at least twice in the corpus; are compatible with "today", i.e., end with NN. An example is [NN, IN, DT, NN, PRP, MD, VB, NN], the one actually chosen.

These constraints control a constrained Markov process that generates the candidates for the first verse. Figure 4 shows the verses proposed. We select "Innocence of a story I could leave today".

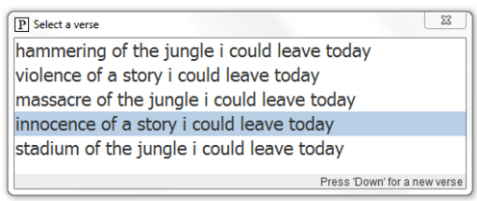

Figure 4. Five verses proposed by the constrained Markov process for the first verse of the song. Each of them satisfies the control constraints: rhythmic template [100, 1, 1, 10, 1, 1, 1, 01]; POS template [NN, IN, DT, NN, PRP, MD, VB, NN]; "today" imposed as the last word.

The complete generation process of the song is the following.

v_1: "today" is imposed as being the last word. The rhythm template is that of "Yesterday, all my troubles seem so far away", i.e., [101, 1, 1, 10, 1, 1, 1, 01].

v_2: Yesterday's rhythmic templates, namely [1, 1, 1, 1, 1, 1, 1, 1, 1]; rhyme with "today".

v_3: Rhythm [1, 1, 01, 1, 101]; rhyme with "today".

v_4: Rhythm [100, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]. Semantic constrain related to "pray" on the first word. Free rhyme.

v_5: Rhythm [1, 1, 1, 10, 10, 10, 1]. Rhyme with "sin".

v_6: Rhythm [1, 101, 1, 100]. "Paradise" imposed as being the last word.

Table 5. The lyrics of the song Yesterday by The Beatles. Each verse is associated to its automatically extracted rhythmic template. The last column of the table shows the rhyme structure.

Yesterday	Rhythmic templates	Rhyme structure
Yesterday all my troubles seemed so far away	101, 1, 1, 10, 1, 1, 1, 01	A
Now it looks as though they are to stay	1, 1, 1, 1, 1, 1, 1, 1, 1	A
Oh I believe in yesterday	1, 1, 01, 1, 101	A
Suddenly I'm not half to man I used to be	100, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1	B
There's a shadow hanging over me	1, 1, 1, 10, 10, 10, 1	B
Oh yesterday came suddenly	1, 101, 1, 100	C
Why she had to go I don't know she wouldn't say	1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1	A
I said something wrong now I long for yesterday	1, 1, 10, 1, 1, 1, 1, 1, 101	A
Yesterday love was such an easy game to play	101, 1, 1, 1, 1, 10, 1, 1, 1	A
Now I need a place to hide away	1, 1, 1, 1, 1, 1, 1, 01	A
Oh I believe in yesterday	1, 1, 01, 1, 101	A

v_7: Rhythm [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]. Rhyme with "today".

v_8: Rhythm [1, 1, 10, 1, 1, 1, 1, 1, 101]. Rhyme with "today".

v_9: Rhythm [101, 1, 1, 1, 1, 10, 1, 1, 1]. "Innocence" imposed as being the first word. Rhyme with "today".

v_{10}: Rhythm [1, 1, 1, 1, 1, 1, 1, 01]. Semantic constraint related to "knock" on the third word. Rhyme with "today".

v_{11}: As in the original song this verse is equal to the third verse, therefore it is not generated by a constrained Markov process, but is simply a copy.

7 CONCLUSION

Structural properties of texts demand long distance modeling which Markov processes at the word level do not cope with. We have demonstrated that constrained Markov processes can be used to generate texts that imitate a given style while satisfying structural properties. We formulate the properties of syntactic correctness, rhymes and meter as unary control constraints. We show that this same framework enables the verses to be semantically biased towards a given semantic constraint. By construction, each verse generated by a constrained Markov process fulfills the property of rhyme and meter. We empirically evaluated syntactic correctness and semantic relatedness by asking humans to evaluate texts generated by our approach against texts generated by two other approaches. This evaluation shows that constrained Markov processes generates better texts in terms of syntactic correctness and semantic relatedness. Finally we showed how this approach can be used to create the lyrics of a song that is both stylistically coherent while satisfying structural constraints.

We want to improve the approach to a deeper control on the texts we generate, by extending the palette of constraints to non-unary constraints. For instance, semantic biases could be improved by allowing cardinality constraints on a whole verse or stanza, instead of unary constraints on chosen words. Form could be constrained by using other constraints such as the number of characters (sum), or constraints holding on paragraphs.

REFERENCES

[1] E. Alvarez-Lacalle, B. Dorow B, J.P. Eckmann, and E. Moses, 'Hierarchical structures induce long-range dynamical correlations in written texts', *Proc. Nat. Acad. Sci. USA*, **103**, 7956–7961, (2006).

[2] R. Barzilay, 'Probabilistic approaches for modeling text structure and their application to text-to-text generation', in *Empirical methods in natural language generation*, pp. 1–12. Springer-Verlag, (2010).

[3] A. Budanitsky and G. Hirst, 'Evaluating wordnet-based measures of lexical semantic relatedness', *Comput. Linguist.*, **32**(1), 13–47, (2006).

[4] K. Deemter, M. Theune, and E. Krahmer, 'Real versus template-based natural language generation: A false opposition?', *Computational Linguistics*, **31**(1), 15–24, (March 2005).

[5] P. Gervás, 'Wasp: Evaluation of different strategies for the automatic generation of spanish verse', in *Time for AI and Society*, pp. 93–100, (2000).

[6] P. Gervás, 'An expert system for the composition of formal spanish poetry', *Journal of Knowledge-Based Systems*, **14**(3-4), 181–188, (2001).

[7] J. A. Goguen and D. Fox Harrell, 'Style as a choice of blending principles', in *Style and Meaning in Language, Art Music and Design*, pp. 49–56. AAAI Press, (2004).

[8] D. Jurafsky and J. H. Martin, *Speech and Language Processing*, Prentice Hall, 2009.

[9] R. Kurzweil. Cybernetic poet. http://www.kurzweilcyberart.com, 2001.

[10] A. K. Mackworth, 'Consistency in networks of relations', *Artificial Intelligence*, **8**, 99–118, (1977).

[11] D. Milne and I. H. Witten, 'An effective, low-cost measure of semantic relatedness obtained from wikipedia links', in *Proceedings of AAAI 2008*, (2008).

[12] Y. Netzer, D. Gabay, Y. Goldberg, and M. Elhadad, 'Gaiku: Generating haiku with word associations norms', in *Proc. of the Workshop on Computational Approaches to Linguistic Creativity (CALC)*, pp. 32–39. ACL Press, (2009).

[13] H. R. Gonçalo Oliveira, F. Amlcar Cardoso, and F. C. Pereira, 'Tra-la-lyrics: an approach to generate text based on rhythm', in *Proc. of the 4th. International Joint Workshop on Computational Creativity*, (2007).

[14] F. Pachet and P. Roy, 'Markov constraints: steerable generation of Markov sequences', *Constraints*, **16**(2), 148–172, (2011).

[15] F. Pachet, P. Roy, and G. Barbieri, 'Finite-length Markov processes with constraints', in *Proc. of IJCAI 2011*, pp. 635–642, Spain, (2011).

[16] R. Power, D. Scott, and N. Bouayad-Agha, 'Generating texts with style', in *Proc. of the 4th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing03)*, pp. 444–452, Mexico, (2003).

[17] F. Rahman and H. M. Manurung, 'Multiobjective optimization for meaningful metrical poetry', in *Proc. of ICCC-11*, pp. 4–9, Mexico, (2011).

[18] E. Reiter and S. Williams, 'Generating texts in different styles', in *The Structure of Style: Algorithmic Approaches to Manner and Meaning*, pp. 59–75. Springer-Verlag, (2010).

[19] A. Rudnicky. The Carnegie Mellon pronouncing dictionary, version 0.7a. http://www.speech.cs.cmu.edu/cgi-bin/cmudict, 2010.

[20] B. Settles, 'Computational creativity tools for songwriters', in *Proc. of the NAACL-HLT Workshop on Computational Approaches to Linguistic Creativity*, pp. 49–57. ACL Press, (2011).

[21] H. Somers, 'Review article: Example-based machine translation', *Machine Translation*, **14**, 113–157, (1999).

[22] K. Toutanova, D. Klein, C. Manning, and Y. Singer, 'Feature-rich part-of-speech tagging with a cyclic dependency network', in *Proc.of HLT-NAACL 2003*, pp. 252–259, (2003).

[23] J. Wagner, J. Foster, and J. van Genabith, 'Judging grammaticality: Experiments in sentence classification', *CALICO Journal. Special Issue on the 2008 Automatic Analysis of Learner Language*, (2009).