An alternate approach towards meaningful lyric generation in Tamil

Ananth Ramakrishnan A
AU-KBC Research Centre
MIT Campus of Anna University
Chennai, India
ananthrk@au-kbc.org

Sobha Lalitha Devi
AU-KBC Research Centre
MIT Campus of Anna University
Chennai, India
sobha@au-kbc.org

Abstract

This paper presents our on-going work to improve the lyric generation component of the Automatic Lyric Generation system for the Tamil Language. An earlier version of the system used an n-gram based model to generate lyrics that match the given melody. This paper identifies some of the deficiencies in the melody analysis and text generation components of the earlier system and explains the new approach used to tackle those drawbacks. The two central approaches discussed in this paper are: (1) An improved mapping scheme for matching melody with words and (2) Knowledge-based Text Generation algorithm based on an existing Ontology and Tamil Morphology Generator.

1 Introduction

In an attempt to define poetry (Manurung, 2004), provides three properties for a natural language artifact to be considered a poetic work, viz., Meaningfulness (M), Grammaticality (G) and Poeticness (P). A complete poetry generation system must generate texts that adhere to all the three properties. (Ananth et. al., 2009) explains an approach for automatically generating Tamil lyrics, given a melody, which attempts to generate meaningful lyrics that match the melody.

The existing approach (Ananth et. al., 2009) to automatically generate Tamil lyrics that match the given tune in *ABC* format (Gonzato, 2003) involves two steps. The first step is to analyze the input melody and output a series of possible syllable patterns in *KNM* representation scheme - a scheme for representing all words in the language, where, K stands for *Kuril* (**(C)V**, where V is a short vowel), N stands for *Nedil* (**(C)V**, where V is a long vowel) and M stands for *Mei* or *Ottru* (consonants) - that match the given melody, along with tentative word and sentence boundary. This melody analysis system was trained with sample film songs and their corresponding lyrics collected from the web. The tunes were converted to *ABC* Notation (Gonzato, 2003) and their lyrics were represented in *KNM* scheme. The trained model was then used to label the given input melody.

The subsequent step uses a Sentence Generator module to generate lines that match the given syllable pattern with words satisfying the following constraints: a) Words should match the syllable pattern and b) The sequence of words should have a meaning. This was achieved by using n-Gram models learnt from a Tamil text corpus.

Though the system manages to generate sentences that match the syllable pattern, it has the following limitations:

1) When no words are found matching a given syllable pattern, alternate patterns that are close to the given pattern, as suggested by the Edit Distance Algorithm, are considered. This algorithm treats the syllable patterns as strings for finding close patterns and hence, can provide choices that do not agree with the input melody.
2) The Sentence Generation is based on the n-Gram model learnt from a text corpus. This can result in sentences that do not have a coherent meaning. Also, since only bi-grams are considered, it can generate sentences that are ungrammatical due to Person-Number-Gender (PNG) agreement issues.

This paper is an attempt to propose alternate approaches in order to overcome the above limitations.

2 Limitations of existing approach

2.1 Finding close matches to syllable patterns

In the existing system, when no words are found matching the given syllable pattern (either due to a small corpus or rarity of the pattern), the closest patterns are considered as alternatives. The closest match to a given syllable pattern is generated based on the Edit Distance algorithm. For example, if the input sequence is given as "NKN" (*long vowel - short vowel - long vowel*) and if no words are found matching NKN, closest matches for NKN are generated. Thus, if an edit distance of 1 is considered, the alternate pattern choices are "KKN", "NKM", "NNN", "NMN", etc. However, not all of these syllable patterns can fit the original music notes.

As an example, consider the Tamil word "*thA-ma-rai*" (*lotus*) that fits the pattern NKN. Suppose no words that match the pattern NKN was present in the corpus and other close patterns were opted for, we get:

Pat.	**Word**	**Meaning**	**Match**
KKN	*tha-va-Lai*	*Frog*	No match
NKM	*thA-ba-m*	*Longing*	No match
NNN	*kO-sA-lai*	*Cow Hut*	Close Match
NMN	*pA-p-pA*	*Child*	No match

Table 1. Alternative patterns for "NKN"

None of the above words can be used in the place of "*thA-ma-rai*", a good fit for a NKN pattern, as they don't *phonetically* match (except for a close-but-not-exact "*kO-sA-lai"*) and hence cannot be used as part of the lyric without affecting the intended melody.

2.2 Ungrammatical or meaningless generation

The Sentence Generation algorithm was based on the n-Gram model built from a text corpus. Given that n-Gram based generation schemes have in-built bias towards shorter strings, it can end-up generating meaningless and ungrammatical sentences. As observed in (Ananth et.al., 2009), we can get sentences such as:

அவன் நடந்து சென்றாள்
(avan-He-3sm nadandhu-walk sendrAlY-3sf)*
(He reached by walking)

Here, the subject *avan (He),* which is a 3rd person, singular, masculine noun, does not agree with the verb *sendrAlY* , which is 3rd person, singular, feminine. Thus, the noun and the verb do not agree on the gender. The correct sentence should be:

அவன் நடந்து சென்றான்
(*avan-3sm nadandhu sendrAn*-3sm)

This is happening because the bi-gram score for நடந்து சென்றாள் could be greater than நடந்து சென்றான்.

Similar disagreements can happen for other aspects such as person or number. Though performing a joint probability across words would help in reducing such errors, it would slow down the generation process.
In addition to the above ungrammatical generation problem, the system can also generate meaningless sentences. Though, some of them can be considered as a *poetic license*, most of them were just non-sensical. For example, consider the following sentence generated by the n-Gram sentence generation system:

* அது இது என்
(adhu-that idhu-this en-my)
(*that this my)*

The above sentence does not convey any coherent meaning.

2.3 Ability to control theme/choice of words

Given the nature of the Sentence generation algorithm, it is not possible for the program to hand-pick specific words and phrases. That is, the whole generation process is guided by the probability values and hence it is not possible to bias the algorithm to produce utterances belonging to a particular theme.

In the subsequent sections, we explain the alternative approaches to tackle the above limitations.

3 Closest Syllable Patterns

The existing approach uses the *KNM Notation* for representing all words in the language. This phonetic representation is at the most basic level, i.e., alphabets, and hence can be used to represent all words in the language. The *KNM notation* is generated by the melody analyzer and is used throughout the system for generating sentences. Though this representation scheme is at the most basic level, it does not help in cases where we are looking for alternate or close matches. Thus, we need to come up with a representation scheme at a higher level of abstraction that will help us in providing valid choices without compromising the requirements of the melody. To this end, we hereby propose to use elements from classic poetry metric rules in Tamil Grammar (Bala et. al., 2003) as defined in the oldest Tamil Grammar work, *Tholkappiyam* (Tholkappiyar, 5th Century B.C.).

3.1 Meter in classical Tamil Poetry

Meter is the basic rhythmic structure of a verse and the basic term that refers to Tamil meter is *pA*. Each line in the poem is called an *adi,* which, in turn, is made up of a certain number of metrical feet known as the *ceer (*words/tokens*)*. Each *ceer* is composed of a certain metrical units called *asai* (syllables) which are made up of letters (vowels and consonants) that have certain intrinsic length/duration, known as *mAthirai*. The above entities are known as the core structural components of a Tamil poem (Rajam, 1992)

The basic metrical unit *asai* is mostly based on vowel length. There are two basic types of *asai*: *nEr asai* (straightness) and *niRai asai* (in a row; array). The *nEr asai* has the pattern **(C)V(C)(C)** and *niRai asai*, **(C)VCV(C)(C)**. These longest-matching basic *asai* patterns are expanded to represent non-monosyllabic words, but for our needs, we use these two basic *asai* patterns for the new representation scheme.

3.2 *asai*-based Representation Scheme

In the new representation scheme, the constituents of the KNM representation scheme are converted to *nEr* or *niRai asai* before being sent to the Sentence Generator module. The Sentence Generator module, in turn, makes use of this new representation scheme for picking words as well as for finding alternatives. In this new representation scheme, a *nEr asai* is represented as *Ne* and a *niRai asai* is represented as *Ni*.

The following table illustrates the mapping required for converting between the two representation schemes:

KNM Representation	*asai* representation
K	*Ne*
KM(0….2)	*Ne*
N	*Ne*
NM(0…2)	*Ne*
KK	*Ni*
KKM(0…2)	*Ni*
KN	*Ni*
KNM(0…2)	*Ni*

Table 2. KNM to *asai* representation

For example, an output line such as, for example, "KK KK KKK" in the old representation scheme will be converted as "*Ni Ni NiNe*" in the new representation based on *asai*. This means that the line should contain three *ceer*(words/tokens) and the first word should be a *nirai asai*, second word should be a *nirai asai* and the third word contains two syllables with a *nirai asai* followed by *nEr asai*.

This new representation scheme helps in coming up with alternatives without affecting the metrical needs of the melody as the alternatives have the same *mAthirai* (length/duration). Thus, if we are given a pattern such as "***NiNe***", we have several valid choices such as "KKK" (originally given), "KKMK", "KKMKM", "KKN", "KKMN" and "KKMNM". We can use words that match any of the above patterns without compromising the duration imposed by the original music note. This way of choosing alternatives is much better than using the Edit Distance algorithm as it is based on the original meter requirements as against matching string patterns.

To use the previous example of "*thA-ma-rai*" (*lotus*) (NKN) in this new representation scheme, we get, "***NeNi***" and all the following words will match:

Word	KNM scheme
nE-ra-lai (straight wave)	NKN
Sa-nj-nja-la-m (doubt)	KMKKM
Ma-ng-ka-la-m (auspicious)	KMKKM
a-m-bi-kai (goddess)	KMKN
vE-ng-ka-ta-m (Venkatam – a name)	NMKKM

Table 3. NKN alternatives using *asai* representation

The above (valid) choices such as KMKKM, NMKKM, etc. are not possible with just using the Edit Distance algorithm. Thus, the architecture of the system now consists of a new component for this conversion (Figure 1)

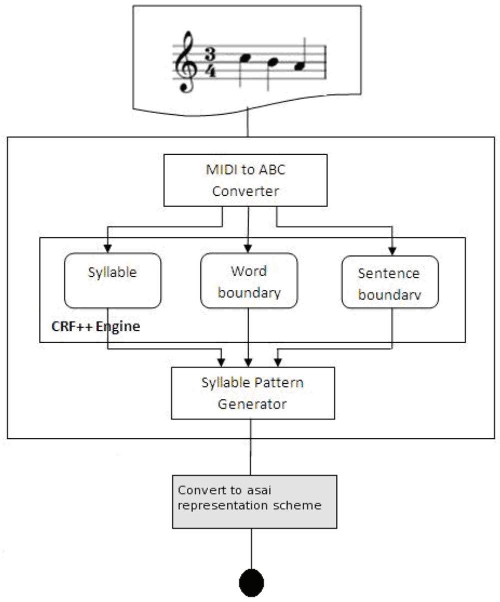

Figure 1. System Approach with new *ASAI* converter

4 Knowledge-based Sentence Generation

The goal of the Sentence Generation module is to generate sentences matching the input pattern given in the new *asai* representation scheme. The existing system generated sentences based on the n-Gram language model created from a text corpus of poems and film songs. However, as explained earlier, this can result in ungrammatical or meaningless sentences being generated. In order to overcome this limitation, the Sentence Generation module is completely overhauled using a knowledge-based approach. A Tamil Morphology generator component, built in-house, is used to generate grammatically correct sentences from this knowledge base.

4.1 Knowledge Base

The knowledge base consists of: (a) set of verbs along with their selectional restriction rules (b) hand-coded sub-categorization Ontology with nouns and (c) list of adjectives and adverbs learned from a text corpus.

4.1.1 Verbs and Selectional Restrictions

Selectional restriction is defined as the right of the verb to select its arguments. Verb is the nucleus of a sentence and has the nature of choosing its arguments. Any particular verb can take its arguments only according to its selectional restriction constraints. When these constraints are violated, the meaning of the sentence is affected. This violation of selectional restriction rules may lead to semantically wrong sentences or figurative usages. Correctness of a sentence not only depends on the syntactic correctness, but also with the semantic interpretation of the sentence.

4.1.2 Syntactic Classification

Verbs can be broadly classified into three divisions, viz., monadic, dyadic and triadic verbs. Monadic verbs can have only one argument - the subject. Dyadic verbs can have two arguments - subject and object. Triadic verbs can take three arguments - subject, direct and indirect objects. But there is no strict rule that the triadic verbs should have all three arguments or the dyadic verbs should have the two arguments filled. There can be overlaps between these groups of verbs. Triadic verb can drop the indirect object and have a Prepositional Phrase (PP) attached with the sentence. Dyadic verb can drop the object and still give a valid sentence. The verbs are grouped according to the sub-categorization information of the subject and object nouns. The sub-categorization features are explained in the following section. At present, we are using only Monadic and Dyadic verbs for our sentence generation purposes.

4.1.3 Sub-Categorization

Sub-categorization features explain the nature of the noun. The subject and object nouns are analyzed using these features. These features may include the type of noun, its characteristics, state etc. Sub-categorization information includes the features such as [±animate], [±concrete], [±edible] etc.

Some of the features and the meanings are listed below:

[+animate]	All animals, human beings
[+human]	All human beings
[+female]	Animals/human beings of feminine gender
[+solid]	Things that are in solid state
[+vehicle]	All vehicles
[+concrete]	Things that physically exist
[-concrete]	Things that do not physically exist
[+edible]	Things that can be eaten
[-edible]	Things that cannot be eaten
[+movable]	Things that are movable
[-movable]	Things that are not movable

Table 4. Sub-categorization Features

4.1.4 Ontology of Nouns

The sub-categorization features are used in the formulation of general Ontology of Nouns. It is made with respect to the usage of language. The Ontology that is developed has the following salient features:

- It is a language-based Ontology originally developed for English and has been customized for Tamil
- Nodes in the Ontology are the actual sub-categorization features of Nouns
- It is made according to the use of nouns in the Tamil language
- Each node will have a list of nouns as entries for that node

The complete Ontology can be found in (Arulmozhi, et. al., 2006)

4.1.5 Contents of Knowledge Base

At present, the knowledge-base consists of 116 unique verbs, 373 selectional restriction rules and 771 Nouns in the Ontology.

The verbs list includes both cognitive as well as non-cognitive verbs. Examples of verbs include *pAr* (to see), *kelY* (to listen), *vA* (to come), *thEtu* (to search), *piti* (to catch), *po* (to go), *kal* (to learn), etc.

The selectional restriction rules are stored as follows:

Verb=>subject_category;subject_case=>object_category;object_case.

When a verb does not take any object, the keyword *[no_obj]* is used to denote the same. In addition to the subject and object categories, the rule also contains the appropriate case markers to be used for the subject and object nouns. This additional information is stored for use by the Morph Generation component.

Some examples of selectional restriction rules are given below:

pAr=>[+living,+animate,+vertebrate,+mammal,+human];NOM=>[no_obj]

pAr=>[+living,+animate,+vertebrate,+mammal,+human];NOM=>[+living,+animate,+vertebrate,+mammal,+human];ACC

piti=>[+living,+animate,+vertebrate,+mammal,+human];NOM=>[living,+concrete,+movable,+artifact,+solid,+instrument,-vehicle,+implements];NOM

piti=>[+living,+animate,+vertebrate,+mammal,+human];NOM=>[no_obj]

Here, *ACC, NOM, DAT*, etc. denote the case markers to be used for the subject and object nouns.

The 771 Nouns are stored across several files according to their position in the Ontology. An Ontology map is used to determine the list of nouns present in a particular node position.

4.1.6 Adjectives and Adverbs

In addition to the verbs and nouns mentioned above, the knowledge-base also contains a list of adjective-noun and adverb-verb bi-grams learnt from a text corpus. This information is used to augment the Sentence Generator with words from these POS categories.

4.2 Tamil Morphological Generator

Tamil is a densely agglutinative language and displays a unique structural formation of words by the addition of suffixes representing various senses or grammatical categories, to the roots or stems. The senses such as person, number, gender and case are linked to a noun root in an orderly fashion. The verbal categories such as transitive, causative, tense and person, number and gender are added to a verbal root or stem. Thus, with the given knowledge-base and a Tamil Morphological generator component one can generate grammatically correct sentences.

We use the Tamil Morphological Generator component (Menaka et. al., 2010) to generate inflections of subject/object nouns with appropriate number & case and the verbs with person, number and gender suffixes.

4.3 Sentence Generation

Given a line in *asai* representation scheme, the sentence generation module is responsible for generating a grammatically correct and meaningful sentence matching the given *asai* scheme. It achieves the same by using the knowledge-base along with the Tamil Morphology Generator component (Figure 2). In addition to the *asai* representation, the module also accepts the tense in which the sentence must be written. The rest of the parameters such as person, gender and case are automatically deduced by the module.

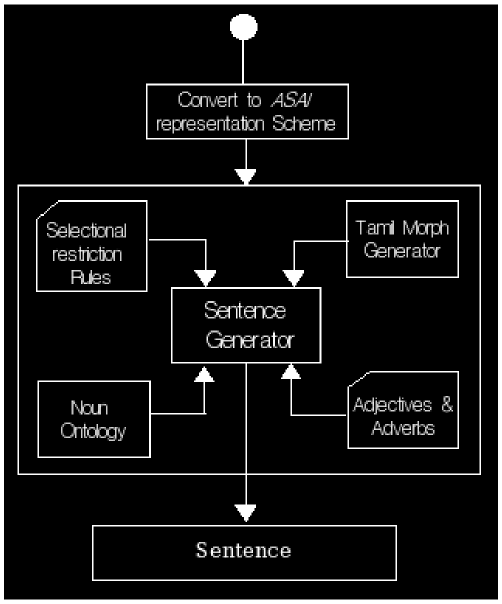

Figure 2. Sentence Generator module

The algorithm for generating a matching sentence is as follows:

```
1. Pick a selectional restriction rule, R in random

2. For each noun, SUB_N in subject_category of rule, R:

  2.1 Guess the gender for SUB_N based on subject_category

  2.2 For each noun, OBJ_N in object_category:

    2.2.1 Use Morphology Generator component to get morphed nouns & verbs based on tense, person, gender and case.

    2.2.2 Generate sentences of the form [SUB_N] [OBJ_N] [VERB]

    2.2.3 Add adjectives or adverbs, if needed

    2.2.4 Repeat words, if needed

    2.2.4 Add to list of sentences generated

3. Check the list of sentences against the asai pattern. If matches, return sentence. Otherwise, go to step 1.
```

Table 5. Sentence Generation Algorithm

Details about steps such as matching against *asai* pattern, gender identification, word repetition and adding adjectives/adverbs are explained below.

4.3.1 Matching against *asai* pattern

The list of sentences generated from the module are compared against the given *asai* pattern. The matching could either be an exact match or a re-ordered match. That is, since Tamil is a relatively free word-order language, the generated sentence can also be re-ordered, if required, to match the given *asai* pattern. However, when adjectives or adverbs are added to the sentence, they need to maintain their position in front of the noun or verb respectively and hence they are not re-ordered. For now, we do not weight the sentences and hence return the first matching sentence.

4.3.2 Gender Identification

As noted in the algorithm, the gender needs to be automatically guessed. In Tamil, the gender of the subject is denoted by the appropriate suffix in the verb. If a personal pro-noun such as *nAnY* (I) or *nI* (you) is used as subject, then any of masculine or feminine gender can be used without affecting the grammatical correctness of the verb. In this case, the program uses the default value of masculine gender. If the subject is not a personal pronoun, the gender for the verb is guessed based on the subject_category of the subject noun. If the subject_category explicitly mentions [+human, +living, +female,…], then feminine gender is returned. If the subject_category explicitly mentions [+human, +living, -female,…], then masculine gender is returned. Otherwise, if [+human, +living,…] is present, but there is no explicit mention of +female or –female, it defaults to honorific suffix. In all other cases, neuter gender is returned.

4.3.3 Adding adjectives and adverbs

The Sentence Generator module using the selectional restriction rules can only create sentences of the form "*[subject] [object] [verb]*". However, typical lyrics will not always contain just three word sentences and thus, the ability to put more words in a sentence generated by our system is required. In such cases, a look-up list of adjectives and adverbs is used for filling the additional words required by the syllable pattern. This look-up list is generated from a POS-tagged text corpus from which the list of adjective-noun, adverb-verb bigrams are added to the look-up list. Whenever a sentence needs more than three words, this look-up list is consulted to generate sentences that add the relevant adjectives to subject or object nouns and relevant adverbs before the verb. Each possible combination of such sentences is generated and added to the list of sentences.

4.3.4 Word repetition

An additional approach to handle lines with more than three words is to repeat certain words already present in the "*[subject] [object] [verb]*" output. If an adjective or adverb is already added to the sentence, then preference for repetition is given to the adjective/adverb subject to the constraints of the input *asai* scheme. Otherwise, the verb is chosen for repetition. Finally, the subject and object nouns are considered.

5 Experiments

The goal of the experiment was to validate whether the sentences generated using the Knowledge-based approach are more grammatical and meaningful than the n-Gram approach. In order to test this hypothesis, a set of 10 syllable patterns was given to the old n-Gram system and 30 sentences were generated from them. The new knowledge-based approach was also given the syllable patterns and the resulting 32 sentences were collected. In order to avoid any bias, these 62 sentences were interleaved in a single document and this document was given to five human evaluators for scoring each sentence. The scoring methodology is as follows:

Score	Meaning
1	Incorrect
2	Grammatically perfect, but no meaning at all
3	Grammatically correct but only partially meaningful
4	Both Grammar and Meaning are only partially correct
5	Perfect

Table 6. Scoring methodology

Based on the scores given by the human evaluators, the sentences generated using the n-Gram approach scored an average of **2.06**, whereas the sentences generated using the knowledge-based approach scored an average of **4.13**. This clearly demonstrates that the new approach results in consistently more grammatical and meaningful sentences.

A break-down of statistics based on the scores given by each evaluator is given below (Table 7):

	E-1	*E-2*	*E-3*	*E-4*	*E-5*
*Avg. Score (KB)**	4.5	4.38	4.06	4.09	3.63
Avg. Score (n-G) *	2.37	1	3.3	2.13	1.5
# Sentences scoring 5 (KB)	25	25	23	20	14
# Sentences scoring 5 (n-G)	6	0	14	1	0
# Sentences scoring 1 (KB)	2	0	7	4	7
# Sentences scoring 1 (n-G)	16	30	11	19	25

Table 7. Detailed Statistics

*KB = Knowledge-based approach and n-G = n-Gram based approach.

A subset of syllable patterns given to the system and the sentences generated by the system are given below:

Input	**NM KKMKMKMK KMNM**
Intermediate Form	***Ne NiNeNeNe NeNe***
Sentences நாம் அரங்கத்துக்கு வந்தோம் (*nAm*-we *arangathukku*-stadium *vanthOm*-came) (*We came to the stadium*) நீ சிறைச்சாலைக்கு வந்தாய் (*nee*-You *siraichAlaikku*-prison *vanthAi*-came) (*You came to the prison*)	

Input	**NN KKNN NMNM**
Intermediate Form	***NeNe NiNeNe NeNe***
Sentences * ராஜா நடனத்தை கேட்டார் (* *rAjA*-King *nadanathai*-dance *kEttAr*-listen) (*The King listened to the dance*) நீங்கள் பிடித்தீர்கள் கையை (*neengal*-You *piditheergal*-caught *kaiyai*-hand) (*You caught the hand*)	

Here, the sentence "*rAjA*-King *nadanathai*-dance *kEttAr*-listened" (*The King listened to the dance*) is generated due to the fact that the noun *dance* is taken from the Ontology node "*content*" that also contains nouns for *music*, *drama*, etc. for which the verb *listen* matches perfectly. Thus, this semantically meaningless sentence is generated due to the present sub-categorization levels of the nouns Ontology. In addition to this, Ontology based generation can also create semantically meaningless sentences when a verb has more than one sense and the appropriate sense is not taken into consideration.

The next sentence "*neengal*-You *piditheergal*-caught *kaiyai*-hand" (*You caught the hand*) is an example of a sentence in which the verb and object noun were re-ordered to match the input pattern.

6 Limitations and Future Work

From the initial set of experiments, we see that the knowledge-based approach results in generating grammatically correct and mostly meaningful sentences. Also, unlike the Edit Distance algorithm, the new *asai* representation scheme consistently provides valid choices and alternatives for syllable patterns, thus resulting in better coverage.

We are also currently working on introducing cohesion across multiple lines of the verse by (a) grouping related verbs, (b) using semantically related verbs (such as Synonym, Antonym, Hyponym, etc.) from previous sentences and (c) picking rules that can result in using the same subject or object.

The main drawback of the current knowledge-based approach is the lack of poetic sentences and hence the poetic aspect of the verse needs improvement. Although we attempt to introduce structural poeticness by rhyme and repetition, the content aspect of the poem remains a bottleneck given our approach of using selectional restriction rules that does not lend well for figurative sentences.

References

Ananth Ramakrishnan, Sankar Kuppan, and Sobha Lalitha Devi. 2009. *Automatic Generation of Tamil Lyrics for Melodies*. Proceedings of the Workshop on Computational Approaches to Linguistic Creativity, CALC'09, Boulder, Colorado:40-46.

Arulmozhi P, Sobha. L. 2006. *Semantic Tagging for Language Processing*. 34th All India conference for Dravidian Linguistics (June 22-24, 2006), Trivandrum, India.

Bala Sundara Raman L, Ishwar S, and Sanjeeth Kumar Ravindranath. 2003. *Context Free Grammar for Natural Language Constructs – An implementation for Venpa Class of Tamil Poetry*. 6th International Tamil Internet Conference and Exhibition, Tamil Internet 2003 (August 22-24, 2003), Chennai, India.

Guido Gonzato. 2003. *The ABCPlus Project* http://abcplus.sourceforge.net.

Hisar Maruli Manurung. 2004. *An evolutionary approach to poetry generation*. Ph.D. Thesis, University of Edinburg.

Menaka S, Vijay Sundar Ram, and Sobha Lalitha Devi. 2010. *Morphological Generator for Tamil*. Proceedings of the Knowledge Sharing event on Morphological Analysers and Generators (March 22-23, 2010), LDC-IL, Mysore, India:82-96.

Rajam V.S. 1992. *A Reference Grammar of Classical Tamil Poetry (150 B.C.-pre-5th/6th century A.D.)*. Memoirs of the American Philosophical Society, Philadelphia: 113-240.

Tholkaappiyar. 5th Century B.C. *Tholkaapiyam - http://www.tamil.net/projectmadurai/pub/pm0100/tolkap.pdf*.