Journal of Trends in Computer Science and Smart Technology (ISSN: 2582-4104)
www.irojournals.com/tcsst/

Thirukkural Alagiduthal and Kural Venba Validation using Python

Thivaharan S[1], Gayathri C[2], Vijay Raj Kumar M[3]

[1]Assistant Professor, Department of Computer Science and Engineering, PSG Institute of Technology and Applied Research, India.

[2,3]UG Scholar, Department of Computer Science and Engineering, PSG Institute of Technology and Applied Research, India

E-mail: [1]thivahar@psgitech.ac.in, [2]21z111@psgitech.ac.in, [3]21z154@psgitech.ac.in

Abstract

Thirukkural, one of the prominent Tamil Literatures, written 2000 years ago, has great importance till date as it emphasizes everyday virtues of the individuals. As per Tamil Literature, it belongs to the class of 'Kural Venba' which is a subdivision of 'Venba' and is regarded as one of the poetic styles of Tamil Language. This research primarily aims on the validation of entered Thirukkural by checking the syntactical correctness of 'Kural Venba'. It uses 'Transliteration' methodology to convert the user given thirukkural into an intermediate form. Then, it processes the intermediate form to segregate the words (seer and thalai) into different asai as per the Yaapilakkanam known as Alagiduthal, compares it with the pre-defined norms of 'Kural venba' and produces the result. At any point of time, if the user input doesn't qualify a criterion, the program outputs a warning accordingly and gets terminated.

Keywords: Transliteration, Alagiduthal, Seer, Thalai, Kural Venba, Yaapilakkanam

1. Introduction

Language is a tool for expressing our thoughts and feelings to each other which ultimately helps people to develop into a civilized community. In earlier days, people used sign language for communication, then moved on to spoken language and finally evolved with written languages. As of now, there are approximately more than 7000 languages which are being used in vogue all around the world. Out of these languages, the oldest languages like Chinese, Greek, Hebrew, Sanskrit, and Tamil have attained classical status.

DOI: https://doi.org/10.36548/jtcsst.2023.1.004
Received: 07.03.2023, received in revised form: 07.04.2023, accepted: 22.04.2023, published: 02.05.2023

Tamil language is a Dravidian language being spoken predominantly in Tamil Nadu and Sri Lanka. It is known for its tradition and history. It has a rich collection of classical epics, grammar, and literatures, out of which Thirukkural is of great importance. Thirukkural was written 2000 years ago, by Thiruvalluvar. Thirukkural is so popular as it emphasizes the everyday virtues of the individual. Anything and everything starting from politics, education, love, ethics, and management are described in Thirukkural [17][18][19]. Venba is a form of classical Tamil poetry which consists of 2 – 12 lines. Kural Venba is a sub-category of Venba. Thirukkural is based on Kural Venba. It contains three major volumes called Aratthupaal (virtue), Porutpaal (wealth) & Inbathupaal (love). It has 1330 couplets called 'kural' which has 2 lines, each of which contains 4 seers and 3 seers respectively. Yappilakkanam describes the rules and methods of constructing a poem. The primary elements of Yappilakkanam are as below:

1. *Ezhuthhu* → The individual letters (vowels and consonants) are called Ezhuthu
    - Letters having very short sound are called 'Ottru'
    - Letters having short sound are called 'kuril'
    - Letters having long sound are called 'Nedil'

2. *Asai* → These are kuril, Nedil, Ottru form Nerasai or Niraiasai based on the duration of the pronunciation.

3. *Seer* → The asai are combined to form the Seer.

4. *Thalai* → It explains the position of same or opposite asai in the combination of the last asai in the preceding seer and the first asai in the succeeding seer.

2. Literature Survey

The research work aims in validating the entered Thirukkural and generating Alagiduthal Vaaipadu. The basic parameters used for the validation includes – Adi, Asai, Seer and Thalai. As per the Venbailakanam, it should contain 4 and 3 seers in the 2 lines respectively. Method of assigning separate values for space and carriage return, and generating a sequence to check the grammatical correctness of adi in the given input, has been adopted by some researchers [1]. The Thirukkural received as input in 'Tamil' is converted into intermediate form. Transliteration, the way of translating based on pronunciation, has been used widely for converting user input into an intermediate form [2].

The letters of Tamil language can be categorised into three types namely 'uyir', 'mei' and 'uyirmei'. For tokenisation [6][10] of the 'vowels' and 'consonants' in Tamil, different

strategies such as by constructing and optimising Context Free Grammar have been used [4]. Once the letters are split, they are easily categorised into ‘ner’ or ‘nirai’ asai.

The final stage of validation involves checking of correctness of thalai, based on the type of seer and forthcoming asai in next seer. In addition to the validation of a given Kural venba, the program also generates the ‘alagiduthalvaaipadu’ which adds a bit more value to the analysis. Various other research in Thirukkural such as analysing all the couplets and segregating the discourse markers and finding the precise meaning [13] of the Thirukkural with the help of the different categories of the discourse markers have been done [2]. Also the above similar validation method is been applied for other pa types in Tamil such as ‘kalipa’ [4]. Similar kind of algorithm has been applied in English literature for differentiating between prose and poetry based on shape, metric and rhymer using ‘Bayes’ Rule’ and ‘Multi-Layer Perceptron’ [12]. Also it has been implemented in various other languages [7][8][9].

3. Proposed work

The proposed ideology contains a set of criteria which the user-given input string should pass in order to qualify as Kural-Venba. These criteria are formed based on Yaapilakanam and Alagidudhal which defines how Kural-Venba should be. The paper divides the program into three main processes namely Transliteration, Alagiduthal and Kural-Venba validation.

3.1 Transliteration

The first and foremost process to be done before checking the input against the rules is to transliterate [14][15][20] the input from Tamil to English. For this, indic_transliteration package is used to convert text from one indic script to another i.e., from Tamil to English. It supports languages like Hindi, Sanskrit, Kannada, Devanagari, Tamil, Malayalam, Telugu, Gurumukhi, Gujarati, Bengali, and Oriya. Transliterate() function of the package is used for transliteration. To use this function, ‘sanscript’ is imported from the package indic_translitertion as described in Figure 1. The 'sanscript' module in the 'indic_transliterate' package is a Python library which provides various routines for transliterating text. The function ‘transliterate’ is imported from ‘sanscript’ module which takes three arguments: string that is to be transliterated, the source script and the target script.

```
from indic_transliteration import sanscript
from indic_transliteration.sanscript import transliterate
```

Figure 1. Code snippet to import indic-transliteration package

Tamil language consists of totally 247 letters out of which 12 are vowels (uyirezhuththu), 18 are consonants (meiezhuththu), 216 are the combination of both (uyir-meiezhuththu) and 1 ayuthaezhuththu [11]. Table 1 displays the complete list of transliterated Tamil letters. Here the column headers represent uyirezhuththukal, the row headers represent meiezhuththukal and the contents of the tables represent uyirmeiezhuththukal.

Table 1. Transliterated Tamil letters

H	a	A	i	I	u	U	è	e	ai	ò	o	au
gh	gha	ghA	ghi	ghI	ghu	ghU	ghè	ghe	ghai	ghò	gho	ghau
G	Ga	GA	Gi	GI	Gu	GU	Gè	Ge	Gai	Gò	Go	Gau
jh	jha	jhA	jhi	jhI	jhu	jhU	jhè	jhe	jhai	jhò	jho	jhau
J	Ja	JA	Ji	JI	Ju	JU	Jè	Je	Jai	Jò	Jo	Jau
Dh	Dha	DhA	Dhi	DhI	Dhu	DhU	Dhè	Dhe	Dhai	Dhò	Dho	Dhau
N	Na	NA	Ni	NI	Nu	NU	Nè	Ne	Nai	Nò	No	Nau
dh	dha	dhA	dhi	dhI	dhu	dhU	dhè	dhe	dhai	dhò	dho	dhau
n	na	nA	ni	nI	nu	nU	nè	ne	nai	nò	no	nau
bh	bha	bhA	bhi	bhI	bhu	bhU	bhè	bhe	bhai	bhò	bho	bhau
M	ma	mA	mi	mI	mu	mU	mè	me	mai	mò	mo	mau
y	ya	yA	yi	yI	yu	yU	yè	ye	yai	yò	yo	yau
r	ra	rA	ri	rI	ru	rU	rè	re	rai	rò	ro	rau
l	la	lA	li	lI	lu	lU	lè	le	lai	lò	lo	lau
v	va	vA	vi	vI	vu	vU	vè	ve	vai	vò	vo	vau
zh	zha	zhA	zhi	zhI	zhu	zhU	zhè	zhe	zhai	zhò	zho	zhau
L	La	LA	Li	LI	Lu	LU	Lè	Le	Lai	Lò	Lo	Lau
r2	r2a	r2A	r2i	r2I	r2u	r2U	r2è	r2e	r2ai	r2ò	r2o	r2au
n2	n2a	n2A	n2i	n2I	n2u	n2U	n2è	n2e	n2ai	n2ò	n2o	n2au

3.2 Alagiduthal

Alagiduthal is to classify the seer based on its asai [5] as Nerasai or Niraiasai as described in table 1. In the *asai*, the *ottru* letters does not have any value but used to identify the limits of asai in a word [16]. Excluding Ottru, asai would have a maximum of two letters.

Table 2. Formation of Asai

First letter	**Second Letter**	**Third letter**	**Asai formed**
Kuril	-	-	Ner
Nedil	-	-	Ner
Kuril	Ottru	-	Ner
Nedil	Ottru	-	Ner
Kuril	Kuril	-	Nirai
Kuril	Nedil	-	Nirai
Kuril	Kuril	Ottru	Nirai
Kuril	Nedil	Ottru	Nirai

This concept is implemented through a function nerai_ner() shown in figure 2 which is further explained in section 4.3

```
def nirai_ner(string,actual,cnt):
  ner_nir = []
  str1 = ""
  rules={"0":"நேர்","1":"நேர்","00":"நிரை","01":"நிரை"}
  p=string.split('2')
  for x in p:
    if x!="":
      if len(x)>2:
        n=2
        chunks = [x[i:i+n] for i in range(0, len(x), n)]
        for y in chunks:
          if y in rules:
            str1+=rules[y]
            str1+="/"
          else:
            h=[y[i:i+1] for i in range(0, len(y))]
            for z in h:
              str1+=rules[y]
              str1+="/"
      else:
        if x in rules:
          str1+=rules[x]
          str1+="/"
        else:
          h=[x[i:i+1] for i in range(0, len(x))]
          for z in h:
            str1+=rules[z]
            str1+="/"
    else:
      continue
  print("\n",str1)
  return(thalai_vaaipadu(str1,actual,cnt))
```

Figure 2. Code snippet showing the implementation of Alagiduthal process

3.3 Validation of Kural Venba

Kural Venba [3] holds the following properties:

- It consists of two lines. The first line consists of four seers, whereas the second line consists of three seers.
- The last seer in the second line will have any of the following vaaippadu: Naal, Malar, kaasu, Pirappu (as mentioned in Table 3).

Table 3. Criterion for last seer in Kural venba

Asai	**Vaaippaadu**
Ner	Naal
Nirai	Malar
Nerbu	Kaasu
Niraibu	Pirappu

- It must either contain eerasai seer- *Seer* having two *asai* or *moovasai* seer- *Seer* having three *asai*.
- It must either contain IyarseerVendalai or VensirVendalai (as described in Table 4). Other forms of thalai are not allowed.

Table 4. Types of Vendalai accepted in Kural Venba

Type of Vendalai	**Preceding**		**Succeeding**	
	Seer	**Last asai**	**First asai**	**Seer**
IyarseerVendalai (VilamunNer)	Iyarseer (Eerasai)	Nirai	Ner	Iyarseer
IyarseerVendalai (MaamunNirai)	Iyarseer (Eerasai)	Ner	Nirai	Iyarseer
VenseerVendalai (kaaimunNer)	Venseer (Moovasai)	Ner	Ner	Venseer (Moovasai)

4. Stages

The algorithmic flow of the program along with different stages has been described in figure 3.

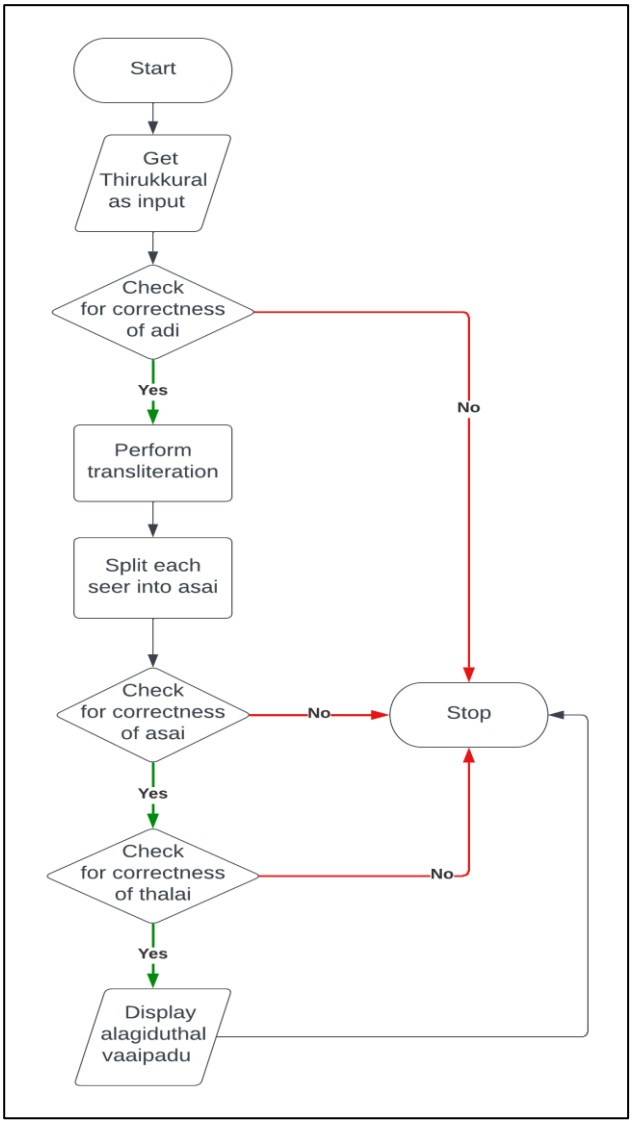

Figure 3. Flowchart describing the flow of the program

4.1 Verification of valid input

Verification of valid input is based on the following set of rules:

- Thirukkural consists of two sentences (adi).
- adi1 should have 4 words (seer) and adi2 should have 3 words (seer).

அகர முதல எழுத்தெல்லாம் ஆதி
பகவன் முதற்றே உலகு

Figure 4. Valid input

So, the input for Thirukkural is taken in its natural form i.e., two inputs (adi1 and adi2) from the user in Tamil. Length of adi1 and adi2 is checked by splitting it with space as delimiter using the split() function. If the condition is not met, the process is terminated otherwise adi1 and adi2 is passed together in the form of list of 7 words (4 words from adi1 and three word from adi 2) named seer to the function alagiduthal(). This function acts as a wrapper function which calls other routines defined in the program.

4.2 Transliteration and kuril-nedil identification

The parameter passed to the alagiduthal() function is transliterated from Tamil to English, letter by letter. This is done using the Transliterate() function of indic_transliteration library.

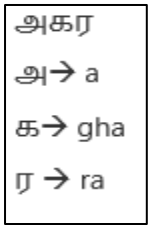

Figure 5. Example of Transliterated word

The transliterated letter is passed to kuril_nedil_finder(). It works based on string comparison. The last character or last two characters of the transliterated letter is compared with the already defined lists namely Kuril and Nedil [16] which contains valid list of characters kuril letters and nedil letters respectively.

```
kuril=['a','i','u','è','ò']

nedil=['A','I','U','e','ai','o','au','H']
```

Figure 6: Kuril and nedil list used

Based on the comparison, it returns integers as described in Table 5.

Table 5. Working of kuril_nedil_finder()

Category	**Integer returned**
Kuril	0
Nedil	1
Others	2

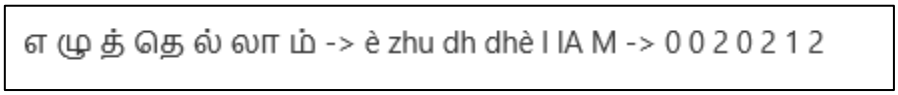

Figure 7. Intermediate word

Such intermediate output as shown in Figure 7, is stored in a variable and passed on to nirai_ner() function.

4.3 Identification of niraiasai and nerasai

The nirai_ner() function identifies the argument passed either as niraiasai or nerasai, based on string comparison. A dictionary namely rules, is defined as follows for the niraiasai and nerasai identification:

```
rules={"0":"நேர்","1":"நேர்","00":"நிரை","01":"நிரை"}
```

The argument passed is primarily split into smaller string with ‘2’ as delimiter using split function. Such words are further split into smaller string until it matches any of the keys of the dictionary. Once the word matches any of the key, corresponding key’s value is printed along with a ‘\’ appended at the end of the value. Also it is stored in a variable and passed to the function thalai_vaaipadu().

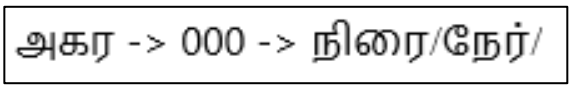

Figure 8. Sample output of nirai_ner()

4.4 Segregation of thalai

This is the stage where the input is tested against various conditions. For execution of this, two global lists namely start and vaaipadu have been maintained. Each seer may contain one or more asai. The start list is used to store the first asai of each seer. The vaaipadu list is used to store integer value which corresponds a specific combination of asai as described in Table 6.

Table 6. Description of vaaipadu list

Description	Integer
vilam seer	0
ma seer	1
kaai seer	2
Ner	3
Nirai	4
Nerbu	5
Niraibu	6

The string passed on to the thalai_vaaipadu() is split into smaller string using split() function with '/' as delimiter and stored in a list which is processed using a for loop. This function majorly focusses on two cases, parameter passed to the function is the last word of the thirukkural or parameter passed to the function is any word other than the last word. This is identified with the help of cnt variable which is passed as parameter to the function.

4.4.1 The parameter passed is the last word of the Thirukkural (given as input)

In this case, the length of the list is checked in which the split strings are stored. If the length is more than 3, error is displayed and -1 is returned. If the length of the list is 3 (along with the back slash), it means that it has two asai (eerasai). Kural Venba requires that in case of eerasai, and the second asai must be 'ner'. If this condition is not met, error is displayed and -1 is returned. Otherwise the working proceeds further and checks whether the last character belongs to kutrilugaraelzhuththukal (in Tamil the Kutrailugaraeluthukal includes " கு, சு, ஞு, து, பு, று" which on transliteration gives "ghu,jhu,Dhu,dhu,bhu,r2u" respectively). If not, error is displayed and -1 is returned. Otherwise, if the eerasai is 'NerNer', 'Nerbu' and 5 are appended to the list start and vaaipadu respectively, or if the eerasai is 'NiraiNer', 'Niraibu' and 6 are appended to the list start and vaaipadu respectively.

Now considering the case where the length of the list is 2 (along with back slash - which means it has one asai): if the asai is 'Ner ' , 'Ner' and 3 are appended to the list start and vaaipadu respectively, or if the asai is 'Nirai ', 'Nirai' and 4 are appended to the list start and vaaipadu respectively.

4.4.2 The parameter passed is not the last word of the Thirukkural (given as input)

Here, the code first checks for the length of the list in which the split strings are stored and analyse and append values in the list start and vaaipadu based on Table 6 and Table 7.

Table 7. Different combinations of asai and its name

Combination of asai	**Name**
NerNer	Thema
NiraiNer	Pulima
NerNirai	Koovilam

NiraiNirai	Karuvilam
NerNerNer	Themangai
NerNerNirai	Themangani
NerNiraiNer	Koovilankai
NerNiraiNirai	Koovilankani
NiraiNerNer	Pulimangai
NiraiNerNirai	Pulimangani
NiraiNiraiNer	Karuvilangai
NiraiNiraiNirai	Karuvilankani

4.5 Final validation

It is first verified whether any part of the verification process has returned -1 which shows that the input did not satisfy a property of Kural Venba. If not so, the process is continued for the last validation, where it is tested for the criterion of Iyarseervendalai and Venseervendalai. Kural Venba does not accept other forms of thalai. This is carried out using string comparison. If the input passes this test too, then it is Kural Venba.

```
for i in range(0,6):
  if((vaaipadu[i]==1 and start[i+1]=="நிரை") or (vaaipadu[i]==1 and start[i+1]=="நிரைபு")):
    pass
  elif((vaaipadu[i]==0 and start[i+1]=="நேர்") or (vaaipadu[i]==0 and start[i+1]=="நேர்பு")):
    pass
  elif((vaaipadu[i] == 2 and start[i+1]=="நேர்") or (vaaipadu[i]==2 and start[i+1]=="நேர்பு")):
    pass
  else:
    check=1
    print("Thalai did not belong to iyarseer or venseer vendalai in",i+1,"th seer")
    break
```

Figure 9. Code snippet to test the criterion of Iyarseervendalai and Venseervendalai

5. Analysis

This proposed work checks for various grammatic exceptions of thirukkural and once it satisfies all aspects of tamil grammar, it displays the entire alagiduthal vaaipadu and thereby verifies the entered thirukkural. Else, it acknowledges the user by intimating the type of violation occurred. Few stages of verification process are listed below.

5.1 Case-1

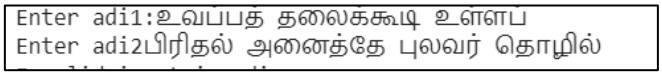

Figure 10. Thirukkural with error in adi

The 1st adi of the above input has '3 seer' and the last adi has '4 seer', which is contradictory to the Venba-ilakanam. So, it displays as follows and terminates the process.

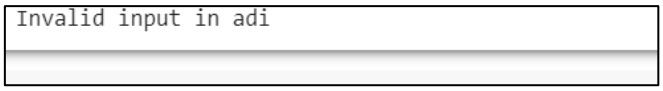

Figure 11. Output for fig. 10

5.2 Case-2

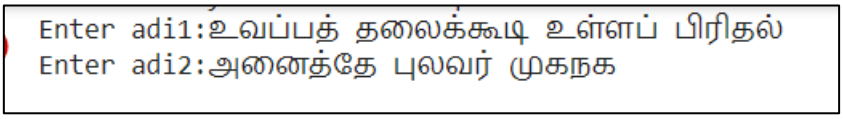

Figure 12. Thirukkural with error in last seer

The last seer of last adi for the above given input has 'Nirai-Nirai' asai ,which can't be categorised into 'Ner (or) Nirai (or) Nerbu (or) Niraibu', which is mandatory for an eetru seer of a thirukkural. Hence, it concludes it as invalid input and outputs as follows.

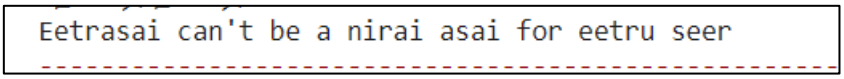

Figure 13. Output for fig. 12

5.3 Case-3

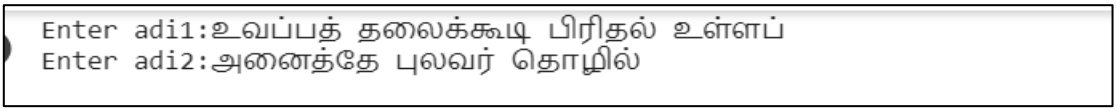

Figure 14. Thirukkural with error in thalai

In the above input, the third and fourth seer are interchanged, and so the thalai did not belong to iyarseer or venseervendalai (i.e., 'Maa munnirai' , 'Vilammunner' , 'Kaaimunner'). So, it concludes as an invalid input and outputs as follows.

```
Thalai did not belong to iyarseer or venseer vendalai in 2 th seer
```

Figure 15. Output for fig. 14

5.4 Case-4

```
Enter adi1:உவப்ப பார்ப்பவருக்கு உள்ளப் பிரிதல்
Enter adi2:அனைத்தே புலவர் முகநக
```

Figure 16. Thirukkural with error in seer

The 2nd seer in the aforementioned input is neither ‘Eerasai’ nor ‘Moovasai’ seer, which violates the tamil – venbailakanam. Therefore, it doesn’t proceed further in listing the alagiduthalvaaipadu and generates the following error message to the user.

```
Seer is neither eerasai or moovasai seer
```

Figure 17. Output for fig .16

5.5 Case-5

```
Enter adi1:உவப்பத் தலைக்கூடி உள்ளப் பிரிதல்
Enter adi2:அனைத்தே புலவர் தொழில்
```

Figure 18. Thirukkural with no error

For the above input, it satisfies all the ‘adi’ , ‘seer’ and ‘thalai’ (iyarseer and venseervendalai) completely as per the tamil – venbailakanam. As a result, it completely accepts thirukkural and displays as follows

```
நிரை/நேர்/
புளிமா

நிரை/நேர்/நேர்/
புளிமாங்காய்

நேர்/நேர்/
தேமா

நிரை/நேர்/
புளிமா

நிரை/நேர்/
புளிமா

நிரை/நேர்/
புளிமா

நிரை/
நிரை
Thirukural is valid
```

Figure 19. Output for fig. 18

6. Conclusion and Future Works

The prime objective of this research is to validate the entered Thirukkural based on Tamil ilakanam by initially transforming it into English with the help of transliteration. The algorithm starts with the analysis of ‘adi’, followed by splitting of each word as a sequence of ‘nirai’ or ‘ner’ asai based on their respective criteria, prior to which it converts the words into an intermediate string based upon ‘kuril’ and ‘nedil’ in Tamil language. Finally, the code checks for the correctness of ‘thalai’ for the given Thirukkural and then generates ‘Alagiduthalvaaipadu’, which is an additional accomplishment of this work. The type of algorithmic approach for the analysis of Thirukkural (which is a Kural Venba) can also be extended to other poetic styles in Tamil literature such as ‘Asiriyappa’ , ‘Vanjipa’ etc., and also for the subdivisions of the poetic styles such as ‘Nerisaivenba’, ‘Innisai venba’ and so on.

References

[1] Madhavan, K. V., S. Nagarajan, and Rajeswari Sridhar. "Rule based classification of tamil poems." *International Journal of Information and Education Technology* 2.2 (2012): 156.

[2] Anita, R., and C. N. Subalalitha. "Building discourse parser for Thirukkural." *Proceedings of the 16th International Conference on Natural Language Processing*. 2019.

[3] Balasundararaman, L., S. Ishwar, and Sanjeeth Kumar Ravindranath. "“Context Free Grammar for Natural Language Constructs an Implementation for Venpa Class of Tamil Poetry." *Proceedings of Tamil Internet, India* (2003).

[4] Sridhar, Rajeswari, et al. "Recognition of kalippa class of tamil poetry." *In 12th International Tamil Internet Conference*. 2013.

[5] Venkatsubhramaniyen, Subasree, Subha Rashmi, and Rajeswari Sridhar. "Classification of tamil poetry using context free grammar using tamil grammar rules." *CS & IT Conference Proceedings*. Vol. 3. No. 6. CS & IT Conference Proceedings, 2013.

[6] Ramalingam, Anita, and SubalalithaChinnaudayarNavaneethakrish. "A discourse-based information retrieval for Tamil literary texts." *Journal of Information and Communication Technology* 20.3 (2021): 353-389.

[7] Rakshit, Geetanjali, et al. "Automated analysis of bangla poetry for classification and poet identification." *Proceedings of the 12th international conference on natural language processing*. 2015.

[8] Almuhareb, Abdulrahman, et al. "Recognition of Modern Arabic Poems." *J. Softw.* 10.4 (2015): 454-464.

[9] Kaur, Jasleen, and Jatinderkumar R. Saini. "Automatic classification of Punjabi poetries using poetic features." *International Journal of Computational Intelligence Studies* 7.2 (2018): 124-137.

[10] Subalalitha, ChinnaudayarNavaneethakrishnan, and ParthasarathiRanjani. "A unique indexing technique for discourse structures." *Journal of Intelligent Systems* 23.3 (2014): 231-243.

[11] Dhanalakshmi, V., and S. Rajendran. "Natural language processing tools for tamil grammar learning and teaching." *International journal of Computer Applications* (2010): 0975-8887.

[12] Tizhoosh, Hamid R., FarhangSahba, and Rozita Dara. "Poetic features for poem recognition: A comparative study." *Journal of Pattern Recognition Research* 3.1 (2008): 24-39

[13] Markandan, Rubavathanan. "Metafunctions in the Thirukkural: A Systemic Functional Linguistics Analysis." *International Journal of Linguistics, Literature and Translation* 4.6 (2021): 01-06.

[14] Mammadzada, Sabina. "A review of existing transliteration approaches and methods." *International Journal of Multilingualism* (2021): 1-15.

[15] Kaur, Kamaljeet, and Parminder Singh. "Review of machine transliteration techniques." *International Journal of Computer Applications* 107.20 (2014).

[16] Pugazhendhi, D. "Comparison between the Grammar of Greek Sapphic and Tamil Seppal Songs." *Athens Journal of Philology* 7.3 (2020): 147-170.

[17] Madhusudanan, S., and R. Nalini. "Life Skills in Classical Tamil Literature, Thirukural." *ZENITH International Journal of Multidisciplinary Research* 5.8 (2015): 90-95.

[18] Madhusudanan, S., and R. Nalini. "Indigenizing social casework principles in the light of Thirukural." *International Journal of Advances in Social Sciences* 3.3 (2015): 107-110.

[19] Samraj, P. Lazarus. "IMPORTANCE OF REVISITING THIRUKKURAL FOR GOOD GOVERNANCE." *The Indian Journal of Political Science* 76.3 (2015): 267-270.

[20] Oh, J., K. Choi, and Hitoshi Isahara. "A comparison of different machine transliteration models." *Journal of Artificial Intelligence Research* 27 (2006): 119-151.