A Very Brief Introduction to Phonetic Languages and Poetry

Oleksandr Zhabenko

August 25, 2020

Abstract

In the performance paper a question of creating the somewhat suitable for poetry and music text is explored. An idea of ”phonetic” languages is introduced, for which the phonetic properties are of greater importance than the grammar rules with possibly preserving the semantics of the text. Several programs in Haskell are used to test the idea. An algorithm and respective programs in Haskell are introduced to evaluate the suitability of the text in such a phonetic language for poetry and music. The results are used to create and analyze some Ukrainian texts and music using the programs and additionally SoX. Shown that The idea can be extrapolated and extended for the other languages especially those ones that do not request a strict words order.

1 Introduction

There are different languages. They have different structure and rules. But there is a possibility to create and use (on the one of the existing vastly used and well spread languages basis, in this work, the Ukrainian one) the ”phonetic” language more suitable for poetry and music. Even there can be different variants of the phonetic language. This work proposes to create at least two new ”phonetic” languages on the Ukrainian basis and shows the way generalize the idea for the different languages.

Imagine, you can understand the information in the text no matter of the words order and only with the most needed grammar preserved (for example, the rule not to separate the preposition and the next word is preserved). Understand just like you can read the text (after some instruction and training might be) the text with the words where only the first and the last letters are preserved on their places and the rest are interchangeably mixed. So imagine, you can understand (and express your thoughts, feeling, motives and so on) the message of the text with no strict word order preserved.

In such a case, you can rearrange words (preserving the most important rules in this case to reduce or even completely eliminate ambiguity) so that they can obtain more interesting phonetic sounding. You can try to create poetic (at least somewhat rhythmic and expressive) text or music. This can be an inspiring and developing exercise itself. But how can you quickly find out which combinations are more or less suitable? Besides, can the complexity of the algorithms be reduced?

These are some of the interesting questions. The work does not at the moment answers them, but is experimental, still may be valuable.

Ukrainian is the language with no strict words order needed (though there do exist some preferences in it) and have rather pleasant sounding. So it can be a good example and instance. Besides for the author it is a native language.

Even if you would not like to create and use ”phonetic” languages where phonetics is of more importance than the grammar, then you can evaluate the phonetic potential of the words used in the text in producing specially sounding texts. This can also be helpful in poetry writing and other probably related fields [1].

2 Some historic notes

In the work [2] we can see a lot of information about the history of the Ukrainian literature from the 11th to the end of the 19th century. Brief introduction is in the work [3].

3 Basic concepts

As a criteria for evaluation of suitability of the text the following heuristics is used.

If we count the number of unique (not repeating) sounds in the text interval we can find out the magnitude of its phonetical diversity. The more is the number (or may be an average number), the more diverse phonetically is the text interval, so the more different sounds are needed to be pronounced by the speaker. There are limited number of possible sounds (phonemes) so this number cannon be greater than that limit. For Ukrainian it is definitely greater than 30. So if you get the text intervals (by some means) with rather long uniqueness intervals – this term can be considered the keywords one – the text is becoming more diverse and so suitable for pronunciation and therefore probably intonation changes. This can give rise to creating more dramatic music changes or text. It can be even hard in some variants to pronounce it.

The opposite variant is considered the following one. If there are some small uniqueness periods interval that means that some sound (phoneme) is often repeating in the interval and this repetition can additionally create some rhythm so the text can become more poetical and at least rhythmic.

4 Haskell functional programming in usage

Haskell is very suitable because of its peculiarities.

4.1 Algorithm of creating the more suitable interesting (first of all the Ukrainian one, but it can be generalized to other languages) text

1. Prepare all possible permutations of the (first of all Ukrainian, but it can be other language as well) words or their concatenations (without whitespaces) of small quantity (no more than 10 and better no more than 7).
2. For every combination compute one or several norms that are used then in order of their significance and importance. The influence on the result is the most for the most significant norm (it is applied further as the first one for filtering).
3. Using the collection of strings from the first step and the collection of norms from the second step calculate for every element all the norms in the respective order and store the results with the elements associated. Use uniqueness functions.
4. Find the maximum element (or minimum if define the minimum the same as the maximum here) in the collection with respect to the norms calculated previously. Print it or make distinguishable and callable by some means.
5. If needed more variants than just one, repeat the previous steps starting from the second one to get more maximum (minimum) elements. If there are enough of them (or just one needed) then end the computations and return them or print as needed.

The needed functionality (extended for the Ukrainian and basic for other languages) is provided by the packages:

1. uniqueness-periods [4];
2. uniqueness-periods-general [5];
3. dobutokO-poetry-general [6];
4. dobutokO-poetry [7];
5. dobutokO-poetry-general-languages [8];
6. mmsyn6ukr [9];
7. mmsyn7s [10].

4.2 uniqueness-periods

The functionality in the package was first introduced in the mmsyn7s package [10] but there it is sorting used for the needs of mmsyn7 series of packages. This takes additional resources and is not needed because here we are not interested in the order of the unique phonemes for the text interval, but only in the distences between them.

4.3 uniqueness-periods-general

This package generalizes the functionality of the uniqueness-periods for other languages than Ukrainian.

For all the used conversion functions of the type g :: String -> Vector String it is important that they are stable for the repeated application (their result after the first application cannot be changed by the rules in the function into new variants). Otherwise, the recursive scheme of the functions in the module will lead to wrong results. So the conversion function should work the following way (xs denotes a word in the language) in GHCi:

```
let v = g xs
let ys = concat . toList $ v
let v2 = g ys
v == v2
True
```

Or in the other words, for the single word, g . concat . toList . g = g; [11]

It can be seen, that the condition is not very restrictive and, therefore, many languages can be represented in such a way. Ukrainian is just one of them.

Besides, it is not necessarily that the computation scheme must be recursive here, but it is just its straightforward realization.

4.4 dobutokO-poetry-general

The functionality in the module is used by the both packages dobutokO-poetry and dobutokO-poetry-general-languages, is common for them and that is why it could be also named dobutokO-poetry-common, but it is not.

4.5 dobutokO-poetry

Helps to order the 7 or less Ukrainian words (or their concatenations) to obtain somewhat suitable for poetry or music text. Can be also used as a research instrument with generalized

functions [7].

This is the main functionality module used in the work. It has also its generalization for other languages than Ukrainian. The generalization packages and the

4.6 dobutokO-poetry-general-languages

Helps to order the 7 or less words (or their concatenations) to obtain somewhat suitable for poetry or music text. Can be also used as a research instrument with generalized functions [8].

Generalizes the former package not only for Ukrainian but also for other languages, for which the appropriate variant of the function g from the uniqueness-periods-general package is provided.

4.7 mmsyn6ukr

The package is used to sound the Ukrainian text using special sound representations. They are not exact and not completely phonetically proper, but for the work give an impression of the Ukrainian sounding text and can be used to research rhythmics [9].

4.8 mmsyn7s

Can be used additionally for the Ukrainian, especially if you would like also deal with syllables [10].

5 Performance and Discussion

The first part of the program is the Taras Shevchenko's poem "Садок вишневий коло хати" [12]. The program was used for the prepared text: "садок вишневий колохати хрущі надвишнями гудуть плугатарі" for the first 20 output variants for the function

```
uniqNPoeticalUGNL_ K 1 20 (Data.Vector.fromList [norm4,norm513,norm51,norm5])
```

So the result is:

садок хрущі гудуть вишневий колохати надвишнями
хрущі гудуть садок вишневий колохати надвишнями
садок вишневий колохати надвишнями хрущі гудуть
колохати вишневий садок хрущі гудуть надвишнями
вишневий садок хрущі гудуть колохати надвишнями
надвишнями колохати садок вишневий хрущі гудуть
садок вишневий хрущі гудуть колохати надвишнями

садок хрущі гудуть надвишнями вишневий колохати
вишневий садок хрущі гудуть надвишнями колохати
вишневий колохати хрущі садок гудуть надвишнями
вишневий колохати садок хрущі гудуть надвишнями
вишневий хрущі гудуть садок надвишнями колохати
садок надвишнями хрущі гудуть вишневий колохати
хрущі гудуть надвишнями садок вишневий колохати
надвишнями садок хрущі гудуть вишневий колохати
хрущі гудуть вишневий садок надвишнями колохати
колохати надвишнями вишневий садок хрущі гудуть
колохати хрущі гудуть вишневий садок надвишнями
колохати надвишнями садок вишневий хрущі гудуть
колохати вишневий хрущі гудуть садок надвишнями

Just the third line is very close to the original and is a proper Ukrainian text.
So the sounding 20 variants are the first part of the performance.

5.0.1 The second part of performance

The poems of the famous Ukrainian poet Oleksandr Oles' were used for the second part of the performance.

They are (in the order of the performance rewriting):

1. "Раз високо над горами..." [13];
2. Над колискою [14];
3. "Рідна мова в рідній школі!" [15];
4. "Дух наш пречистий, дух наш народний..." [16];
5. "Ні, забуття не дасть мені й сама природа..." [17];
6. "Яка краса: відродження країни!.." [18];
7. В роковини Шевченка [19];
8. "В золотій смушевій шапці..." [20];
9. "Італійська ніч підкралась..." [21].

The performance includes the result of the work for the following program in Haskell.

```
module Main where

import System.Environment (getArgs)
import qualified Data.Vector as V
```

```
import DobutokO.Poetry.Data
import DobutokO.Poetry.Norms
import DobutokO.Poetry.General
import DobutokO.Poetry.Ukrainian.PrepareText (prepareText)

main :: IO ()
main = do
  args <- getArgs
  let file = concat . take 1 $ args
  contents <- readFile file
  let flines = prepareText contents
  -- Now we have a list of lines to be processed
  let lasts = map (\ts -> if null . words $ ts then [] else last . words $ ts) flines
  -- And a list of the last words there that can be rhymed in poetic text
  uniqNPoeticalUGNL_ (PA [] (concat . take 1 $ lasts)) 1 1 (normF) . concat . take 1 .
          map (unwords . (\xss -> if null xss then [] else init xss) . words) $ flines
  -- Now the first converted line is ready
  circle2 (concat . take 1 $ lasts) (drop 1 flines)

normF :: V.Vector ([Int] -> Int)
normF = V.fromList [normSplitWeightedG (*) [5,-1] (V.fromList [norm513, norm4])]

circle2 :: String -> [String] -> IO ()
circle2 xs xss
 | null xss = return ()
 | otherwise = let (zss,tss) = splitAt 1 xss in do
     let rs = words . concat $ zss
     uniqNPoeticalUGNL_ (PA xs (if null rs then []
       else last rs)) 1 1 (normF) . unwords $
           (if null rs then [] else init rs)
     circle2 (if null rs then [] else last rs) tss
```

Afterwards, after the running this program its output was saved to the textual file. Then on the file there was run the following program.

```
module Main where

import Data.Char (isAlpha)
import System.Environment (getArgs)
```

```
main :: IO ()
main = do
  args <- getArgs
  let file = concat . take 1 $ args
  xs <- readFile file
  mapM_ putStrLn . noDoubleWords $ xs

noDoubleWords :: String -> [String]
noDoubleWords ys = let xss = lines ys in
  (concat . take 1 $ xss):(map (unwords . drop 1 . words) . drop 1 $ xss)
```

The resulting text was piped to the text file and then it was pronounced with mmsyn6ukr program. This generates the second part of the performance.

If you would like some experimenting, you can use SoX [22] to obtain... e. g. whistle. To do so, type the next command on one line.

```
sox haidamakySound.flac rhythm-generated-temp.flac vol 0.8 rate -h 44100
  vol 1.2 channels 2 echo 0.88 0.8 20 0.2 tempo 1.3 reverb -w 1 10 30
     vol 1.8 sinc 50-5500 biquad 1 0.3 0 1 0.1 0.4 sinc 150-400 vol 5
       echo 0.2 0.8 10 0.8 reverb -w 10 2 100 30 tempo 1.62 gain -n -6
         reverb -w 10 2 100 30 tempo 1.62 gain -n -6
```

Then again:

```
sox rhythm-generated-temp.flac rhythm01.flac sinc 800-11025
sox rhythm01.flac rhythm02.flac gain -n -7 sinc 1000-2000 gain -n -6
sox rhythm02.flac test.flac compand .1,.2 -inf,-62.1,-inf,-62,-62 0 -90 .1
sox test.flac test930-940.flac sinc 930-940 gain -n -6
```

The 62 in the last command is a calibrated value so that there is discreteness in the sound obtained.

Then again:

```
sox -T "|sox -r44.1k -n -p synth tra 935 channels 2" test930-940.flac
  Not-Very-Interesting-Routine.flac vol 5 sinc 1130-1140 vol 100
    compand .1,.1 -25.1,-25,-inf,0,-inf 25 -90 .1 vol 5 reverb -w 10 1 30
      vol 3 echo 0.8 0.88 60 0.3 trim 0 `soxi -D test930-940.flac`
```

The last one is also one line command in bash shell on Linux. Sometimes the background sounds in it can be very interesting.

Besides, you can use more postprocessing. Enter the command further:

```
sox Not-Very-Interesting-Routine.flac Not-Very-Interesting-Routine-m.flac
  sinc 1860-1880 vol 850 reverb -w 10 20 30 vol 5 echo 0.8 0.88 60 0.4
```

The resulting file contains changing whistle.

6 Conclusions

So we can see an interesting field for investigation. The idea of the ”phonetic” languages is interesting and needs further development. The used algorithm is sensible and rather interesting.

7 Acknowledgements

The author thanks Donya Quick for her clarifications for many questions connected with the FARM performance event. Besides the author thanks Julia’s Synyavska family, Lina’s Hlavchuk family and Lesia’s Horova family. They truly encouraged him to conduct such a research.

8 Additional information in Ukrainian

It can be found here. It deals with syllables.

9 Technical requirements

The programs in the text have moderate system requirements to work. Nevertheless, they use memory (especially dobutokO-poetry and dobutokO2) for their work and mmsyn6ukr also uses disk space to function properly. The programs were tested to work well on Raspberry PI B3 with the following technical characteristics. The systems with higher resources should work also. Usage for the less memory (down to 256MB of RAM) is expected as well, but is not recommended. If you need more RAM then in Linux since 4.19 series (at least, but even earlier versions may also have it) there is a kernel module ZRAM that allows approximately to double the possible memory amount for the programs by compression of the RAM. For more information, please, refer to its documentation [23].

Technical specifications for Raspberry PI B3

SoC:	Broadcom BCM2837
CPU:	4× ARM Cortex-A53, 1.2GHz (note: the real working frequency is about 700MHz if the CPU is not overclocked);
GPU:	Broadcom VideoCore IV;
RAM:	1GB LPDDR2 (900 MHz);
Storage:	microSD;

[24]

References

[1] Oleksandr Zhabenko. *dobutokO-poetry README documentation*, 2020. https://hackage.haskell.org/package/dobutokO-poetry-0.16.2.0.

[2] D. Cyzevs'kyj, D. Tschižewskij, D.I. Čyževs'kyj, D. Ferguson, G.S.N. Luckyj, Ukrainian Academy of Arts, Sciences in the United States, D. Gorsline, and U. Petyk. *A History of Ukrainian Literature: From the 11th to the End of the 19th Century*. Annals of the Ukrainian Academy of Arts and Sciences in the United States. The Ukrainian Academy of Arts and Sciences and Ukrainian Academic Press, 1997. http://uvan.org/wp-content/uploads/2014/06/Annals-of-UVAN-1997-History-of-Ukr-Lit_1-of-2.pdf.

[3] Ukrainian literature. May 2011. https://www.britannica.com/art/Ukrainian-literature.

[4] Oleksandr Zhabenko. *uniqueness-periods description*, 2020. https://hackage.haskell.org/package/uniqueness-periods-0.1.0.0.

[5] Oleksandr Zhabenko. *uniqueness-periods-general description*, 2020. https://hackage.haskell.org/package/uniqueness-periods-general-0.2.0.0.

[6] Oleksandr Zhabenko. *dobutokO-poetry-general description*, 2020. https://hackage.haskell.org/package/dobutokO-poetry-general-0.1.0.0.

[7] Oleksandr Zhabenko. *dobutokO-poetry description*, 2020. https://hackage.haskell.org/package/dobutokO-poetry-0.16.2.0.

[8] Oleksandr Zhabenko. *dobutokO-poetry-general-languages description*, 2020. https://hackage.haskell.org/package/dobutokO-poetry-general-languages-0.2.0.0.

[9] Oleksandr Zhabenko. *mmsyn6ukr Readme*, 2019–2020. https://hackage.haskell.org/package/mmsyn6ukr.

[10] Oleksandr Zhabenko. *mmsyn7s package documentation*, 2020. https://hackage.haskell.org/package/mmsyn7s-0.8.0.0.

[11] Oleksandr Zhabenko. *dobutokO-poetry-general-languages String.UniquenessPeriodsG module description*, 2020. https://hackage.haskell.org/package/uniqueness-periods-general-0.2.0.0/docs/String-UniquenessPeriodsG.html.

[12] Taras Shevchenko. Sadok vyshnevyi kolo khaty. Тарас Шевченко. Садок вишневий коло хати. https://www.ukrlib.com.ua/books/printit.php?tid=63.

[13] Oleksandr Oles'. Raz vysoko nad horamy... Олександр Олесь. "Раз високо над горами..." Джерело: друковане видання. http://poetyka.uazone.net/oles/oles05.html.

[14] Oleksandr Oles'. Nad kolyskoyu. Олександр Олесь. Над колискою Джерело: друковане видання. http://poetyka.uazone.net/oles/oles10.html.

[15] Oleksandr Oles'. Ridna mova v ridnii shkoli. Олександр Олесь. "Рідна мова в рідній школі!". Джерело: Бібліотека школяра. Поезія. "Наукова думка", К., 1998. http://poetyka.uazone.net/oles/oles36.html.

[16] Oleksandr Oles'. Dukh nash prechystyi, dukh nash narodnyi. Олександр Олесь. "Дух наш пречистий, дух наш народний..." Джерело: друковане видання. http://poetyka.uazone.net/oles/oles42.html, 1918.

[17] Oleksandr Oles'. Ni, zabuttya ne dast' meni j sama pryroda. Олександр Олесь. "Ні, забуття не дасть мені й сама природа..." Джерело: Бібліотека школяра. Поезія. "Наукова думка", К., 1998. http://poetyka.uazone.net/oles/oles50.html, 1905.

[18] Oleksandr Oles'. Yaka krasa: vidrodzhennya krajiny! Олександр Олесь. "Яка краса: відродження країни!.." Джерело: Бібліотека школяра. Поезія. "Наукова думка", К., 1998. http://poetyka.uazone.net/oles/oles54.html, 1908.

[19] Oleksandr Oles'. V rokovyny shevchenka. Олександр Олесь. В роковини Шевченка. Джерело: Бібліотека школяра. Поезія. "Наукова думка", К., 1998. http://poetyka.uazone.net/oles/oles63.html, 1928.

[20] Oleksandr Oles'. V zolotij smushevij shaptsi. Олександр Олесь. "В золотій смушевій шапці..." Джерело: Бібліотека школяра. Поезія. "Наукова думка", К., 1998. http://poetyka.uazone.net/oles/oles64.html, 1930.

[21] Oleksandr Oles'. Italijska nich pidkralas'. Олександр Олесь. "Італійська ніч підкралась..." Джерело: http://pysar.net/Бібліотека Кошового Писаря. http://poetyka.uazone.net/oles/oles75.html, 1913.

[22] Chris Bagwell, Lance Norskog SoX Contributors, and Sundry Contributors. *SoX*. http://sox.sourceforge.net/.

[23] Nitin Gupta. *zram: Compressed RAM based block devices*. https://www.kernel.org/doc/Documentation/blockdev/zram.txt.

[24] Russell Barne. Raspberry pi 3: Specs, benchmarks & testing. https://magpi.raspberrypi.org/articles/raspberry-pi-3-specs-benchmarks.

[25] Solomija N. Buk, Ján Macutek, and Andrij A. Rovenchak. Some properties of the ukrainian writing system. *CoRR*, abs/0802.4198, 2008. http://arxiv.org/abs/0802.4198.

[26] *Glasgow Haskell Compiler User's Guide. 9.1. Language options. extension-BangPatterns*, 2020. https://downloads.haskell.org/~ghc/latest/docs/html/users_guide/glasgow_exts.html#extension-BangPatterns.