

# Indian Institute of Technology, Delhi



## B.Tech Project report

**Conversion of text in standard format to braille**

Supervisor - Prof. M. Balakrishnan

**Submitted by**

Prem Ranjan (2013CS10247)

Anoop Kumar (2013CS10211)

# 1. Introduction

There is so much data present everywhere which is in the digital format. The challenge is to convert it into braille so that it can be accessed by the visually-impaired people. Accessibility to digital content is a big challenge for the visually-impaired people.

epub is the standard for all electronic publications. So, the input files can be assumed to be in epub format. There are some standard rules in almost every language to write in braille script. Once the epub file is converted into braille, it can be accessed either digitally or by embossing it on a paper. To convert a file into braille, the formatting information needs to be preserved to the extent possible.

There are various formats available for a file in braille. The most popular are 'brl' and 'pef'. 'pef' stands for Portable Embosser Format. These files can be embossed on paper using some softwares. They are also readable by the Refreshable Braille Display, which displays some characters at a time.

## 2. Types of braille

There are mainly three types of braille:-

- (a) **Grade 1 (Uncontracted)** - There is braille representation for every single character. So, the number of characters in the text file is almost equal to the number of characters in the braille file. This kind of braille is mainly used by the beginners who are new to braille.
- (b) **Grade 2 (Contracted)** - Some of the group of letters which occur very frequently are represented by just one or two characters in braille. So, the number of characters in total decreases. It is also more convenient to the users. Once the users become little proficient in reading braille, he immediately switches to Grade-2 braille. The contraction may be in the form of prefix, suffix, infix or the whole word.
- (c) **Grade 3 (Densely contracted)** - This kind of braille is highly contracted. It is used by those users who are highly experienced in using braille.

## 3. Work flow

To convert an epub file into braille, there are mainly two steps which needs to be followed – conversion of epub to txt, and conversion of txt to braille.

### 3.1. epub to txt

While converting epub to txt, the formatting needs to be preserved to the extent possible. First, the subscript and subscript information needs to be captured, so appropriate characters were added at the beginning and the end of the scripts in the epub file.

The steps to be followed for that are:-

- (a) Extract the contents of the zip file to get the individual .html or .htm files.
- (b) Use the **BeautifulSoup** library in python to find the elements under the sup, sub tags inside scripts.
- (c) Add the appropriate characters required for representing the subscripts and subscripts.

So, the simple mathematical expressions which are defined using the scripts can be captured using this. But even the slightly complicated ones likes the ones defined using the 'math' tag in html can't be expressed.

There are some formatting information which are difficult to preserve, e.g. - bullet points, italics, color of the element, the numbering of the points, etc.

In this process, all the images, their captions etc. are removed as it can't be represented in the brl or the pef format. Any character which is not convertible into braille is replaced by '\*'.

After this is done, the **ebook-convert** library by **calibre** is used for the conversion.

### 3.2. txt to pef

The main challenge is to convert the individual word into its corresponding braille representation. Almost every language has its mappings for braille script. These mappings also depend on whether the braille is grade 1, grade 2 or grade 3. Grade 1 braille is relatively simple. There is a one-to-one mapping between the characters. In Grade 2 braille, there are various contractions for some group of letters which occur frequently, e.g. – **en** - ⠠ , **tion** - ⠠⠠

The contraction can be in the form of prefix, suffix or infix of a word or for the whole word. The algorithm for conversion of a complete word is the following.

**For each word do:**

**If word in Contractions:**

**res = Contraction[word];**

**else:**

**lpp = longest\_possible\_prefix(word);** *//result = lpp + remaining\_string\_in\_braille*

**res = contraction[ lpp];** *//initialize*

**s = word.subString( length(lpp), end );** *//initialize*

**while(not isEmpty(s) ) do:**

**if s in SuffixContraction:**

**res = res + SuffixContraction[s];**

**break;**

**else: //Check for infix**

**lpi = longest\_possible\_infix(s);**

**if isEmpty(lpi):**

**res = res + braille(s[0]);**

**s = s.subString( 1, end );**

**else:**

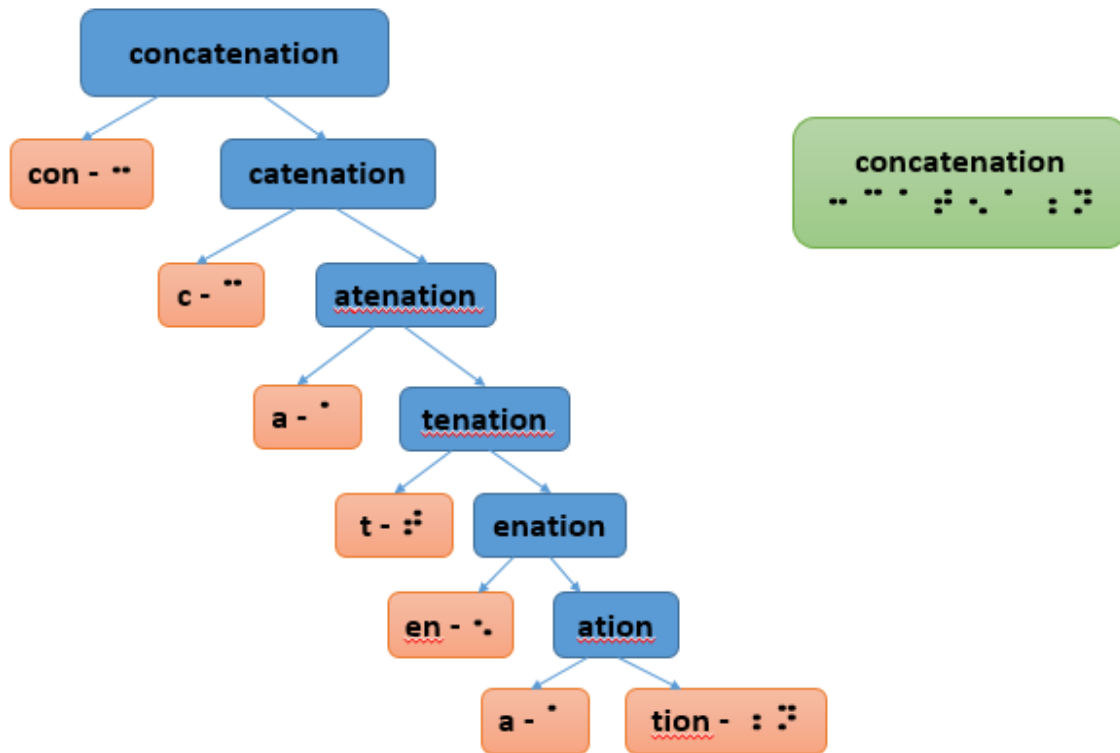
**res = res + InfixContraction[lpi];**

**s = s.subString( length(lpi), end );**

**end while**

**end for**

An example of the conversion for the word ‘concatenation’ is



The mapping in Braille script is different in different languages. Even for the same language, the rules can be different. For English, the standard is defined by **Unified English Braille (UEB)**.

The liblouis library in python can be used to convert a string into its braille representation. It contains the translation tables for most of the languages.

## PEF format

Once the string of text is converted, it needs to be written in proper format specified for the pef file. In pef format, the maximum number of rows per page and the maximum number of characters per line can be specified. We used the pef extension for calibre to convert the text file to pef. It was an open source software, so the source code was available, but some bugs were also present. We fixed the bugs and did appropriate changes in the code so that our purpose is served.

### 3.3. User interface

A desktop application was built with talkback accessibility so that it could be used by a visually impaired person. The **pyqt library** in python was used to create the user interface. It contains the features required for accessibility. The user can navigate to the various options and choose the file by using the buttons on the keyboard. The user can also add his own translation tables, but it has to be in the format specified by liblouis. Some softwares are available to view the pef file generated. One is present on the [official site](#) of the pef format. The file can be viewed and can also be connected to the embosser.

## 4. Conclusion

### 4.1. Some of the limitations of the app are :-

- The characters which contain some special characters, can't be converted. Those characters are replaced by the braille translation of the character '\*'.
- Only simple text can be converted to braille. The files which contain complex mathematical equations, chemical equations etc. also can't be translated correctly.
- As the epub and html files are converted into pef, some formatting in the file may be lost, e.g. – bullet points, bold or italics etc. Also, the spaces between the lines and paragraphs may increase, but it will never decrease.
- The pef does not support images, as it can't be represented by braille dots. It needs some tactile graphics, which is currently not supported in pef format.

### 4.2. However, there is a lot of scope for future development. Some of them are :-

- Support for complex mathematical expressions and other technical materials
- Support to display the data in tabular format with rows and columns
- Add translation tables for regional languages, other than English and Hindi
- Tactile displays in pef files

The braille converter app is useful in many ways for the visually impaired people. It can also be used to represent braille music.

### 4.3. References

- <http://www.brailletranslator.org/>
- <http://pef-format.org/>
- <http://www.iceb.org/ueb.html>
- <http://liblouis.org/>
- <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>
- <https://manual.calibre-ebook.com/>
- <https://docs.python.org/2/library/zipfile.html>