**MWG4-2**

# A Character Encoding Model for Preserving the Mongolian Alphabetical System

Surgaltu, Unenmongke, Erhim, Orlog

9 February 2020

## 1 Introduction

### 1.1 Scope

Mongolian scripts include traditional Mongolian script (for short as Mongolian script, exclude the other scripts such as Cyrillic Mongolian script and Pagspa character), Manchu script, Todo script, Sibe script and Ali Gali. This text discusses about Mongolian script except Ali Gali of it, but this model can be extended to all Mongolian scripts.

### 1.2 Aims

1.2.1 The model mainly aims to promote current Mongolian Unicode which exists repeated encoding problem (one grapheme with multiple codes is called as repeated encoding) to manually controlled level.

Repeated encoding is caused by several aspects. First, nominal character set was created from the traditional Mongolian alphabet which indexes grapheme by its phoneme, therefore a huge number of variants with a grapheme occur when mapping nominal characters to variants. For Example:

**Table 1-1:** Examples

| Translit. | Output | Character sequence |
|---|---|---|
| bodo | ᠪᠣᠳᠣ | < ᠪ 182A, ᠣ 1823, ᠳ 1833, ᠣ 1823 > |

| | | |
|---|---|---|
| budu | ᠪᠤᠳᠤ | < ᠪ 182A, ᠤ 1824, ᠳ 1833, ᠤ 1824 > |
| exe | ᠡᠬᠡ | < ᠡ 1821, ᠬ 182C, ᠡ 1821 > |
| ege | ᠡᠭᠡ | < ᠡ 1821, ᠭ 182D, ᠡ 1821 > |
| tata | ᠲᠠᠲᠠ | < ᠲ 1832, ᠠ 1820, ᠲ 1832, ᠠ 1820 > |
| tede | ᠲᠡᠳᠡ | < ᠲ 1832, ᠡ 1821, ᠳ 1833, ᠡ 1821 > |

Simultaneously, some combinations of graphemes are identical with the grapheme of another character or overlap another grapheme:

**Table 1-2:** Examples

| Translit. | Output | Character sequence |
|---|---|---|
| tegri | ᠲᠡᠭᠷᠢ | < ᠲ 1832, ᠡ 1821, ᠭ 182D, ᠷ 1837, ᠢ 1822 > |
| tngri | ᠲᠨᠭᠷᠢ | < ᠲ 1832, ᠨ 1828, ᠭ 182D, ᠷ 1837, ᠢ 1822 > |
| tŋri | ᠲᠩᠷᠢ | < ᠲ 1832, ᠩ 1829, ᠷ 1837, ᠢ 1822 > |
| siragdan | ᠰᠢᠷᠠᠭᠳᠠᠨ | < ᠰ 1830, ᠢ 1822, ᠷ 1837, ᠠ 1820, ᠭ 182D, ᠳ 1833, ᠠ 1820, ᠨ 1828 > |
| sirxatan | ᠰᠢᠷᠬᠠᠲᠠᠨ | < ᠰ 1830, ᠢ 1822, ᠷ 1837, ᠬ 182C, ᠠ 1820, ᠲ 1832, ᠠ 1820, ᠨ 1828 > |

In addition, following reasons deteriorate repeated encoding problem, they are: Encoding Mongolian script with Manchu, Todo, Sibe scripts that are from different writing systems and have mutually exclusive alphabet; Including all Mongolian graphemes in modern Mongolian character set as variants regardless of time period; allowing different academic notes exist in the same standard simultaneously, etc.

**Table 1-3:** Examples

| Output | Character sequence |
|---|---|
| ᠮᠣᡊᠭᠣᠯ | < ᠮ 182E, ᠣ 1823, ᡊ 184A, ᠭ 182D, ᠣ 1823, ᠯ 182F > |
| ᠠᡏᠠᡵᠠᠯᡨᠠ | < ᠠ 1820, ᡏ 184F, ᠠ 1820, ᡵ 1875, ᠠ 1820, ᠯ 182F, ᡨ 1868, ᠠ 1820 > |

According to the Encoding theory and Information theory, the situation mentioned above is not allowed to exist, for it acts as a disturbance so that reduces the reliability and the efficiency of processing Mongolian text. Practice of the past 20 years proved it, no matter how typing, inputting with OCR or inputting with speech recognition, people have to turn to a million-level corpus to proofread the input, even then they fail to eliminate mistakes completely. Even today people hardly actualize automated office and information sharing in Mongolian script. It is difficult to estimate the security problem of the alternative encodings and the spoofing when generalize the code.

### 1.2.2 Eliminating grammar rules from source script as much as possible.

Text Encoding is a process of semimanual encoding. Although mapping the key stroke to the character is implemented by computer, mapping the text recognizing to key stroke needs manual operation. Similarly, computer presents graphemes from the code, while people recognize text from graphemes sequence. Therefore, in encoding, at least on the user side, need to consider source script as random sequence. Otherwise, ambiguity on grammar cognition will cause error code. Current Mongolian encoding standards include a mass of orthography, even involve grammar to select a variant of the same grapheme, it requires users mastered knowledge of grammar, and a considerable vocabulary, otherwise the correctness of the string is questionable. For example:

**Table 1-4:** Examples

| Output | Character sequence | Rule |
|---|---|---|
| ᠭᠠᠷ | < ᠭ 182D, ᠠ 1820, ᠷ 1837 > | in the masculine |
| ᠭᠡᠷ | < ᠭ 182D, ᠡ 1821, ᠷ 1837 > | in the feminine |
| ᠮᠠᠨ | < ᠮ 182E, ᠠ 1820, ᠨ 1828 > | at the end of closed syllable |
| ᠮᠠᠨᠠ | < ᠮ 182E, ᠠ 1820, ᠨ 1828, ᠠ 1820 > | in the open syllable |
| ᠳᠠᠰᠭᠠᠯ | < ᠳ 1833, ᠠ 1820, ᠰ 1830, ᠭ 182D, ᠠ 1820, ᠯ 182F > | after Mongolian SA or DA |
| ᠳᠠᠭᠠᠯ | < ᠳ 1833, ᠠ 1820, ᠭ 182D, ᠠ 1820, ᠯ 182F > | after the other letters |

### 1.2.3 Avoiding the problem of grapheme and encoding divergence

To represent another grapheme through nominal character and Free Variant Selector combination is equivalent to using Variable-Length Code from the perspective of encoding theory. In current Mongolian encoding standards, corresponding methods of the grapheme and the code are unstable and incompatible. For instance, in Unicode12 version, the variant of MONGOLIAN LETTER NA is defined as below:

~ 1828 ᠨ first form (initial)

| | | |
|---|---|---|
| ~ 1828 | ‍ᠨ‍ | first form (medial) |
| ~ 1828 | ‍ᠨ | first form (final) |
| ~ 1828 180B | ᠨ᠋‍ | second form (initial) |
| ~ 1828 180B | ‍ᠨ᠋‍ | second form (medial) |

……

Essentially, this approach distributes the dotless NA two encoding ways, U+1828 and <U+1828 U+180B>, and as well as several ways for dotted NA including U+1828 and <U+1828 U+180B>. Such an unstable, intertwined encoding approach will make users be confused to choose the encoding way, by the way they also can't search and count the essential element of the text which is written by such a code.

1.2.4 Final aim

Randomly ask people who are not able to write Mongolian script to type a Mongolian random text by a Mongolian keyboard to go through the test.

### 1.3 The other problems should be considered

1.3.1 Consider both Mongolian orthography and traditional perception of people.

1.3.2 The sorting problem of Mongolian letter of the traditional perception.

1.3.3 Reducing the requirement of systemic compatibility and the algorithm complexity of the searching and sorting.

## 2 Basic Character Set

For solving repeated encoding in Mongolian code, conserving script coherence of the traditional perception and simplifying the searching algorithm, firstly, separate phonetic and graphemic information in the alphabet, and define basic character set with graphemic alphabet, exclude grapheme overlap among the same position of different variants. Secondly, separate characters and graphemic variants to avoid over refinement which increases possibility of repeated encoding and decreases flexibility of font development. Meanwhile, need to get rid of the grammar information, simplify the mapping relation of character to variant.

### 2.1 Basic characters and their three forms of variants

This model classifies Mongolian ANG (U+1829) in nominal character set as anti-character (in one hand, the origin of the word ᠲᠠᠩᠭᠢ is controversial, in the other hand, searching algorithm relating ANG is too complex, for example, search ᠤᠨᠠᠭᠠᠪ through the stem ᠤᠨᠠ ). Besides, limit the variant in the same position (mainly, Initial form, Medial form, Final form) to one variant, and decrease the possibility of the grapheme overlap which is at the same position, under the control of Letter Conservation Algorithm.

Mongolian basic characters and their three forms of variants as below:

**Table 2-1:** Mongolian Basic Letters and Three Variants

| No. | Character | Code Point | Name | Init | Medi | Fina | Isol |
|---|---|---|---|---|---|---|---|
| 1 | ᠠ | U+1820 | MONGOLIAN LETTER A | ᠠ‍ | ‍ᠠ‍ | ‍ᠠ | ᠠ |
| 2 | ᠡ | U+1821 | MONGOLIAN LETTER E | ᠡ‍ | | | ᠡ |
| 3 | ᠢ | U+1822 | MONGOLIAN LETTER I | ᠢ‍ | ‍ᠢ‍ | ‍ᠢ | ᠢ |
| 4 | ᠣ | U+1823 | MONGOLIAN LETTER O | | | ‍ᠣ | ᠣ |
| 5 | ᠤ | U+1824 | MONGOLIAN LETTER U | ᠤ‍ | ‍ᠤ‍ | ‍ᠤ | ᠤ |
| 6 | ᠥ | U+1825 | MONGOLIAN LETTER OE | | | ‍ᠥ | ᠥ |
| 7 | ᠦ | U+1826 | MONGOLIAN LETTER UE | ᠦ‍ | ‍ᠦ‍ | | |
| 8 | ᠧ | U+1827 | MONGOLIAN LETTER EE | ᠧ‍ | ‍ᠧ‍ | ‍ᠧ | ᠧ |
| 9 | ᠨ | U+1828 | MONGOLIAN LETTER NA | ᠨ‍ | ‍ᠨ‍ | ‍ᠨ | |
| 10 | ᠪ | U+182A | MONGOLIAN LETTER BA | ᠪ‍ | ‍ᠪ‍ | ‍ᠪ | |
| 11 | ᠫ | U+182B | MONGOLIAN LETTER PA | ᠫ‍ | ‍ᠫ‍ | ‍ᠫ | |
| 12 | ᠬ | U+182C | MONGOLIAN LETTER QA | ᠬ‍ | ‍ᠬ‍ | ‍ᠬ | |
| 13 | ᠭ | U+182D | MONGOLIAN LETTER GA | ᠭ‍ | ‍ᠭ‍ | ‍ᠭ | |
| 14 | ᠮ | U+182E | MONGOLIAN LETTER MA | ᠮ‍ | ‍ᠮ‍ | ‍ᠮ | |
| 15 | ᠯ | U+182F | MONGOLIAN LETTER LA | ᠯ‍ | ‍ᠯ‍ | ‍ᠯ | |
| 16 | ᠰ | U+1830 | MONGOLIAN LETTER SA | ᠰ‍ | ‍ᠰ‍ | ‍ᠰ | |

| | | | | | | |
|---|---|---|---|---|---|---|
| 17 | ᠱ | U+1831 | MONGOLIAN LETTER SHA | ᠱ | ᠱ | ᠱ |
| 18 | ᠲ | U+1832 | MONGOLIAN LETTER TA | ᠲ | ᠲ | ᠲ |
| 19 | ᠳ | U+1833 | MONGOLIAN LETTER DA | ᠳ | ᠳ | ᠳ |
| 20 | ᠴ | U+1834 | MONGOLIAN LETTER CHA | ᠴ | ᠴ | ᠴ |
| 21 | ᠵ | U+1835 | MONGOLIAN LETTER JA | ᠵ | ᠵ | ᠵ |
| 22 | ᠶ | U+1836 | MONGOLIAN LETTER YA | ᠶ | ᠶ | |
| 23 | ᠷ | U+1837 | MONGOLIAN LETTER RA | ᠷ | ᠷ | ᠷ |
| 24 | ᠸ | U+1838 | MONGOLIAN LETTER WA | ᠸ | ᠸ | ᠸ |
| 25 | ᠹ | U+1839 | MONGOLIAN LETTER FA | ᠹ | ᠹ | ᠹ |
| 26 | ᠺ | U+183A | MONGOLIAN LETTER KA | ᠺ | ᠺ | ᠺ |
| 27 | ᠻ | U+183B | MONGOLIAN LETTER KHA | ᠻ | ᠻ | ᠻ |
| 28 | ᠼ | U+183C | MONGOLIAN LETTER TSA | ᠼ | ᠼ | ᠼ |
| 29 | ᠽ | U+183D | MONGOLIAN LETTER ZA | ᠽ | ᠽ | ᠽ |
| 30 | ᠾ | U+183E | MONGOLIAN LETTER HAA | ᠾ | ᠾ | ᠾ |
| 31 | ᠿ | U+183F | MONGOLIAN LETTER ZRA | ᠿ | ᠿ | ᠿ |
| 32 | ᡀ | U+1840 | MONGOLIAN LETTER LHA | ᡀ | ᡀ | ᡀ |
| 33 | ᡁ | U+1841 | MONGOLIAN LETTER ZHI | ᡁ | | |
| 34 | ᡂ | U+1842 | MONGOLIAN LETTER CHI | ᡂ | | |

The encoding model only encodes modern Mongolian script system without incompatible historic variant. The historic variant problem was solved through developing corresponding font, such as Uyghur Mongolian script of the 13th century or woodblock font of the 17th century. Thus, we can transform the modern Mongolian script and historic Mongolian script by font rather than modifying the text.

## 2.2 The other characters

For reducing the complexity of variants and simplifying the searching algorithm, the original Mongolian Vowel Separator, U+180E, is set as Anti-character. The other control characters, punctuations and numbers are not changed.

### 2.3 About punctuation

Writing direction, line breaking direction and page changing direction are different in handwriting or presenting on screen. From the intuition of a script writing in horizontal direction, people should rotate all characters 90 degrees clockwise and then flip over whole page 180 degrees. Whereas Chinese character in vertical writing system, people should rotate all characters 90 degrees anti-clockwise, and then turn whole page in 90 degrees clockwise. That is to say, Mongolian vertical writing system and Chinese vertical writing system are mutually exclusive.

Currently, Mongolian Unicode standard loan plenty of general punctuations and CJK vertical punctuations. It increases the complexity of Chinese and Mongolian vertical writing format in the official applications like Microsoft Office. Mongolian punctuations can't match corresponding font correctly or present in horizontal direction in a majority of editors. Besides, programing web page or under the condition of multilingual text, will cause direction disaccord of punctuation. Therefore, we should distribute code point for all punctuations of Mongolian scripts individually.

## 3 Presentation Control

Some letters in the writing system lose the function of individually indicating phoneme because of isomorphous grapheme, and it is considered as breaking rule of the alphabetical system. By contrast with the other phonetic writing, traditional Mongolian faces severer problems of the breaking rule and it exposes in NLP.

Redefinition of the basic character set and carding variant solve a big amount of problems relating to traditional perception of Mongolian alphabetical system, but repeated encoding still exists. This model aims to figure out that Unicode repeated encoding problem through the Presentation Control, simultaneously, get rid of over-split due to traditional orthographic perception.

For Mongolian, for example, spelling ᠨᠡᠶᠡᠡ as ᠨ ᠡ ᠶ ᠡ ᠡ, has little infulence on meaning, thus this model neglects problem of repeated encoding of it.

### 3.1 Separate different writing systems

Mixing the Mongolian, Todo, Manchu and Sibe letters should be forbidden in a word through limiting the deformation between the boundary of different scripts to avoid the repeated encoding problem.

**Transitional proposal:** Require font vendors to realize with OpenType layout engine.

**Target proposal:** Separate Mongolian, Todo, Manchu and Sibe as different writing systems, and let Mongolian layout engine add language tag between words boundaries.

Separation of different writing systems as Table 3-1:

**Table 3-1:** Examples

| Current standard | Character sequence | New model |
|---|---|---|
| ᠠᡊᠭᠠᠯ | < ᠠ 1820, ᡊ 184A, ᠭ 182D, ᠠ 1820, ᠯ 182F > | ᠠᡊᠭᠠᠯ |
| ᡋᠠᠷᠢᠮᡨᠠ | < ᡋ 184B, ᠠ 1820, ᠷ 1837, ᠢ 1822, ᠮ 182E, ᡨ 1868, ᠠ 1820 > | ᡋᠠᠷᠢᠮᡨᠠ |
| ᡨᠠᡵᠢ | < ᡨ 1868, ᠠ 1820, ᡵ 1875, ᠢ 1822 > | ᡨᠠᡵᠢ |

The character sequence in the table is based on the current standard, just contrast presenting form of the identical or corresponding grapheme in current standard and new model here, the same below.

## 3.2 Mandatory Separating Approach

To trade off traditional perception of letter and texts digitization, adopt mandatory separating approach to solve the repeated encoding problem, which is caused by breaking Mongolian spelling rules, namely, cancel the connection between the confused grapheme and context, shown as below:

**Table 3-2:** Examples

| Current standard | Character sequence | New model |
|---|---|---|
| ᠮᠠᠨᠳᠤᠬᠤ | < ᠮ 182E, ᠠ 1820, ᠨ 1828, ᠳ 1833, ᠤ 1824, ᠬ 182C, ᠤ 1824 > | ᠮᠠᠨᠳᠤᠬᠤ |
| ᠮᠠᠠᠳᠤᠬᠤ | < ᠮ 182E, ᠠ 1820, ᠠ 1820, ᠳ 1833, ᠤ 1824, ᠬ 182C, ᠤ 1824 > | ᠮᠠᠠᠳᠤᠬᠤ |
| ᠮᠬᠳᠤᠬᠤ | < ᠮ 182E, ᠬ 182C, ᠳ 1833, 1824, ᠬ 182C, ᠤ 1824 > | ᠮᠬᠳᠤᠬᠤ |
| ᠳᠤᠨᠳᠠ | < ᠳ 1833, ᠤ 1824, ᠨ 1828, ᠳ 1833, ᠠ 1820> | ᠳᠤᠨᠳᠠ |
| ᠳᠤᠠᠳᠠ | < ᠳ 1833, ᠤ 1824, ᠠ 1820, ᠳ 1833, ᠠ 1820 > | ᠳᠤᠠᠳᠠ |
| ᠳᠳᠳᠠ | < ᠳ 1833, ᠳ 1833, ᠳ 1833, ᠠ 1820 > | ᠳᠳᠳᠠ |

Red graphemes indicate inputs breaking spelling rules.

**Transmitting proposal:** require all font vendors to implement the separating approach with OpenType layout.

**Target proposal:** Add contextual connection property, which indicates whether a character should be connected with another in some conditions, and let layout engine insert "Dashed Nirugu" between the boundary of characters according to contextual connection property when mandatory separating character is needed. The model distributes code point for "Dashed Nirugu", meanwhile, font vendors should guarantee the insertion of it having visual separating area, and also it is different from any space and has no presentation feature. Concrete algorithm is in section 5.1.

## 3.3 Equivalent Sequences and Normalization

In some cases, although some graphemes follow Mongolian spelling rules, repeated encoding still occurs. In these cases, treat them as Compatible-equivalent sequence. They look similar or identical but contain different meaning in the context. For instance, in the Table 3-3:

**Table 3-3：** Examples

| Equivalent sequences | Position | Display |
|---|---|---|
| < ᠬ 182C, ᠠ 1820 > ≈ < ᠠ 1820, ᠬ 182C > | Medial | ᠬᠠ |
| < ᠳ 1833, ᠋ 180B > ≈ < ᠤ 1824, ᠨ 1828, ᠋ 180B > | Medial/Final | ᠳ᠋ / ᠳ᠋ |

Normalizing grapheme is a proper method to solve these problems. In other words, under the condition of preserving main features of the grapheme, add a conspicuous boundary between adjacent characters through lengthening the Nirugu and require Font vendors to satisfy items of the Norm as much as possible. Besides, further normalization of the Mongolian orthography is welcomed to make some improvement. Grapheme normalizing result as below:

**Table 3-4：** Examples

| Current standard | Character sequence | New model |
|---|---|---|
| ᠨᠤᠷᠬᠠᠴᠢ | < ᠨ 1828, ᠤ 1824, ᠷ 1837, ᠬ 182C, ᠠ 1820, ᠴ 1834, ᠢ 1822 > | ᠨᠤᠷᠬᠠᠴᠢ |
| ᠨᠤᠷᠠᠭᠴᠢ | < ᠨ 1828, ᠤ 1824, ᠷ 1837, ᠠ 1820, ᠭ 182D, ᠴ 1834, ᠢ 1822 > | ᠨᠤᠷᠠᠭᠴᠢ |
| ᠠᠷᠠᠳᠴᠢ | < ᠠ 1820, ᠷ 1837, ᠠ 1820, ᠳ 1833, ᠴ 1834, ᠢ 1822 > | ᠠᠷᠠᠳᠴᠢ |

## 3.4 Variable-Length Code Management

This model categorizes Mongolian Free Variants as Soft Variant, Hard Variant and Phonetic Variant according to difference of basic letters and their graphemes of the variants or grammatical meaning. Add FVS1, FVS2 and FVS3 behind the relating basic letter to present them. From the view of encoding, Soft Variant, Hard Variant and Phonetic Variant are differentiated whether they ignore control character in searching and sorting algorithm. Concretely, as Table 3-5:

**Table 3-5:** Spelling and encoding implications of free variants

| Variant Type | Spelling Meaning | | Encoding Meaning | | Control character | Code Point | Alias |
|---|---|---|---|---|---|---|---|
| | Grapheme difference | Grammatical difference | Searching | Sorting | | | |
| Soft | Smaller | Small | Ignored | Ignored | FVS1 | U+180B | SVS |
| Hard | Big | Small | Not Ignored | Ignored | FVS2 | U+180C | HVS |
| Phonetic | None | Big | Ignored | Not Ignored | FVS3 | U+180D | PVS |

In order to specify and limit the scope, give an alias to the Mongolian Free Variation Selector 1, 2, 3 as Mongolian Soft Variation Selector (SVS), Mongolian Hard Variation Selector (HVS) and Mongolian Phonetic Variation Selector (PVS). This phonetic variant is not the variant in phonetics.

In Mongolian encoding standard, using invisible control characters deteriorates repeated encoding problem, encoding consistency and information security problem. In this model, Mongolian script just includes SVS and HVS in the table 3-6 and PVS in the table 4-4. Mongolian layout engine and font just present corresponding grapheme when the variable-length code (basic character with control character) is valid, otherwise present the control characters' default grapheme. Any control character does not allow to exist as invisible form, namely no control character was neglected in presenting. Editor should treat replace operation as searching algorithm, any sorting and searching irrelevant operation, such as copying, pasting is not supposed to ignore any control characters.

Mongolian Soft Variation Selector (SVS) / Mongolian Hard Variation Selector (HVS) as table 3-6:

**Table 3-6:** Modern Mongolian Soft/Hard Variations

| No. | Name | Unicode | Init | Medi | Fina | Isol |
|---|---|---|---|---|---|---|

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | MONGOLIAN LETTER SEPARATING A/E | <1820 180B> | | | ᠠ᠋ | |
| 2 | MONGOLIAN LETTER OLD I | <1822 180B> | ᠢ᠋ | ᠢ᠋ | | ᠢ᠋ |
| 3 | MONGOLIAN LETTER OLD U | <1824 180B> | ᠤ᠋ | | | ᠤ᠋ |
| 4 | MONGOLIAN LETTER DOTLESS AN | <1828 180B> | | ᠨ᠋ | ᠨ᠋ | |
| 5 | MONGOLIAN LETTER QE | <182C 180C> | ᠬ᠌ | ᠬ᠌ | ᠬ᠌ | |
| 6 | MONGOLIAN LETTER OLD QE | <182D 180C> | ᠭ᠌ | ᠭ᠌ | | |
| 7 | MONGOLIAN LETTER AD | <1833 180B> | | ᠳ᠋ | ᠳ᠋ | |
| 8 | MONGOLIAN LETTER OLD IA | <1836 180B> | ᠶ᠋ | ᠶ᠋ | ᠶ᠋ | |

MONGOLIAN LETTER OLD I, OLD U and OLD IA is used for spelling the letter reserving Uyghur Mongolian spelling rules in Medial Age, always appear in separated suffix ᠶᠢᠨ ᠤᠨ ᠤ ᠶᠢ ᠢ ᠶᠢᠶᠢ ᠶᠢᠶᠢᠨ, combination with separated A/E ᠢ ᠠ and some special word like ᠨᠠᠶᠢᠨ. MONGOLIAN LETTER OLD I possesses vowel letter's feature, and OLD IA possesses consonant letter's feature. As no letter in modern Mongolian script correspond to MONGOLIAN LETTER OLD QE, this model reserves it.

About Mongolian Phonetic Variant please refer to section 4.5.

**Transitional proposal:** Require font vendors to realize with OpenType layout engine.

**Target proposal:** Include presenting form with nominal character + control character in variable-length code scope and managed by Unicode Standard. Variable-length code combination undefined in Unicode Standard, has no access to corresponding feature label.

Variable-length code Management result as below:

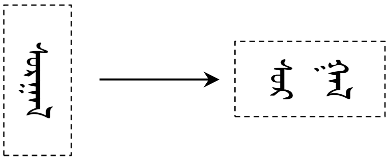
**Table 3-7:** Examples

| Current standard | Character sequence | New model |
|---|---|---|
| ᠰᠠᠷᠠ | < ᠰ 1830, ᠠ 1820, ᠷ 1837, ᠠ 1820 > | ᠰᠠᠷᠠ |
| ᠰᠠᠷᠠ | < ᠰ 1830, FVS2 180C, ᠠ 1820, ᠷ 1837, ᠠ 1820 > | ᠰ HVS ᠠᠷᠠ |
| ᠰᠠᠷᠠ | < ᠰ 1830, ᠠ 1820, ᠷ 1837, FVS1 180B, FVS3 180D, ᠠ 1820 > | ᠰᠠᠷ SVS PVS ᠠ |

### 3.5 About Word Breaking problem in the end of the line

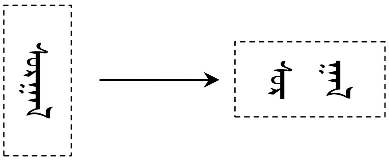
Currently, as Mongolian layout engine breaks word in the end of the line, and doesn't support the word breaking the line automatically in editor, or size of the textbox is smaller than the word size, it causes breaking the line in the word and this word will be taken apart into two isolated word, hence the engine presents different word with a same sequence. For example, textboxes in the Figure 3-1 have identical content before one of them is changed. However, the Mongolian words presented distinctly different meaning after the change.

**Figure 3-1**



**Solution:** Cancel word break in the end of the line, as Figure 3-2 below:

**Figure 3-2**



Hence the measure above relates to the encoding efficiency, should specify in the standard.

# 4 Text Representation

### 4.1 Contextual Variant

Contextual Variant is a variant that adjusts the junction between letters according to the contextual letter, under the premise of keeping main feature of the variant.

Contextual Variant is categorized as Common Contextual Variant and Special Contextual Variant. Common Contextual Variant formed in middle age or earlier, while special Contextual Variant is only relevant with font style. For simplifying the searching and sorting algorithm, the model implements the all Contextual Variants with OpenType layout, and avoids ligature as much as possible.

Common Contextual Variants list as Table 4-1:

**Table 4-1:** Examples

| Character sequence | Position | Intermediate | Output |
|---|---|---|---|
| < ᠪ 182A, ᠠ 1820 > | Initial / Medial | ᠪᠠ | ᠪᠠ |
| < ᠪ 182A, ᠠ 1820 > | Final / Isolate | ᠪᠠ | ᠪᠠ |
| < ᠪ 182A, ᠢ 1822 > | Initial / Medial | ᠪᠢ | ᠪᠢ |
| < ᠪ 182A, ᠢ 1822 > | Final / Isolate | ᠪᠢ | ᠪᠢ |
| < ᠪ 182A, ᠤ 1824 > | Initial / Medial | ᠪᠤ | ᠪᠤ |
| < ᠪ 182A, ᠤ 1824 > | Final / Isolate | ᠪᠤ | ᠪᠤ |
| < ᠪ 182A, ᠦ 1826 > | Initial / Medial | ᠪᠦ | ᠪᠦ |
| < ᠪ 182A, ᠥ 1825 > | Final / Isolate | ᠪᠥ | ᠪᠥ |
| < ᠪ 182A, ᠧ 1827 > | Initial / Medial | ᠪᠧ | ᠪᠧ |
| < ᠪ 182A, ᠧ 1827 > | Final / Isolate | ᠪᠧ | ᠪᠧ |

Mongolian letter ᠪ presents differently in front of different vowels, the vowel behind ᠪ will reshape itself as well. Simultaneously, for simplifying the input method algorithm and editing difficulty, set final form and isolated form of the ᠪᠤ as < ᠪ 182A, ᠤ 1824 > combination. Besides ᠫ ᠹ ᠬ ᠭ ᡦ ᠺ ᠻ follow the same rule.

Special Contextual Variants list as Table 4-2:

**Table 4-2:** Examples

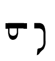
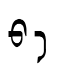
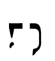
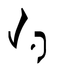
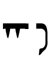
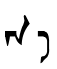
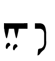
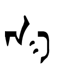
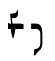
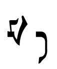
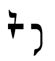
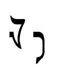
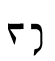
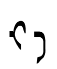
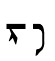
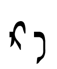
| Character sequence | Font Style | Intermediate | Output |
|---|---|---|---|
| < ᠬ 182C, ᠌ 180C, ᠠ 1820, ᠷ 1837, ᠯ 182F, ᠠ 1820 > | Light |  | ᠬ᠌ᠠᠷᠯᠠ |
| < ᠬ 182C, ᠌ 180C, ᠠ 1820, ᠪ 182A, ᠯ 182F, ᠠ 1820 > | Light | ᠬ᠌ᠠᠪᠯᠠ | ᠬ᠌ᠠᠪᠯᠠ |
| < ᠰ 1830, ᠠ 1820, ᠳ 1833, ᠤ 1824, ᠨ 1828, ᠋ 180B > | Hawang |  | ᠰᠠᠳᠤᠨ᠋ |
| < ᠰ 1830, ᠤ 1824, ᠳ 1833, ᠤ 1824, ᠨ 1828, ᠋ 180B > | Hawang | ᠰᠤᠳᠤᠨ᠋ | ᠰᠤᠳᠤᠨ᠋ |

## 4.2 Separated A/E

Mongolian separated A/E derive from isolated variant Aleph, so although the preceding character locates in the medial of the word, it is given as final form. The model doesn't distribute code point for separated A/E, they will be applied by OpenType layout.

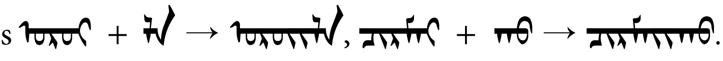
Variant of the Separated A/E as table 4-3:

**Table 4-3:** Examples

| Character sequence | Position | Intermediate | Output |
|---|---|---|---|
| < ᠤ 1824, ᠠ 1820, ᠋ 180B > | Final |  |  |
| < ᠨ 1828, ᠠ 1820, ᠋ 180B > | Final |  |  |
| < ᠬ 182C, ᠠ 1820, ᠋ 180B > | Final |  |  |
| < ᠭ 182D, ᠠ 1820, ᠋ 180B > | Final |  |  |
| < ᠮ 182E, ᠠ 1820, ᠋ 180B > | Final |  |  |
| < ᠯ 182F, ᠠ 1820, ᠋ 180B > | Final |  |  |
| < ᠶ 1836, ᠋ 180B, ᠠ 1820, ᠋ 180B > | Final |  |  |
| < ᠷ 1837, ᠠ 1820, ᠋ 180B > | Final |  |  |

## 4.3 Double teeth I

In current Mongolian encoding system, double teeth I (ᠢ) is encoded and decoded in several ways, it causes severe encoding divergence. This model regards double teeth I as a contextual variant, and just turn into double teeth form behind letters E, O, U and EE:

- In modern Mongolian, double teeth I occurs behind A, E, O, U and EE, while behind I, OE, UE, I appears in single tooth form.
- When any word stem end with I combines with any suffix, follows the rules above to turn to double teeth I, such as .
- Treating double teeth I as contextual variant of Mongolian letter I facilitates the searching algorithm and sorting algorithm.

This model requires including all possibilities to alphabetical preserving system, avoids the homograph combination through mandatory separate approach.

## 4.4 Mandatory Hyphen and Nirugu

In Modern Mongolian, when mandatorily connect two or more words, people follow rules as below:

Rule 1: turn preceding word's final form into medial form, for example: [illegible] + [illegible] → [illegible] ;

Rule 2: if the next word start with a vowel, write the onset straightly and connect to the previous word , like: [illegible] + [illegible] → [illegible], [illegible] + [illegible] → [illegible] ;

Rule 3: if a word starts with a consonant, alter the initial form of the consonant to medial form, then connect to the previous word. The vowel of the first syllable in latter word is supposed to be retained. If both words have a same gender, change the latter word to be non-first syllabic form, such as: [illegible] + [illegible] → [illegible] , [illegible] + [illegible] → [illegible] .

Rule 2 does not match the modern Mongolian spelling rules, thereby the model connects preceding and subsequent words in the rule 2 through the Mandatory Hyphen, count the graphemic change behind the hyphen as contextual variant.

Mongolian Mandatory Hyphen is similar to Nirugu (MONGOLIAN NIRUGU, U+180A). Nirugu can be inserted between the initial form and the medial form, the medial form and the medial form, the medial form and the final form to lengthen the spine between them without changing the connection grapheme, while the hyphen can connect the medial form and the subsequent initial form. Thus, Mongolian Mandatory Hyphen is a variant of Nirugu, the model indicates Mongolian Mandatory Hyphen with <180A 180B>.

Nirugu mainly uses for showing that writings are exceptions to modern Mongolian as a grapheme connector. For instance, type a word by grapheme or syllable, such as [illegible] [illegible] [illegible] [illegible]. In general, Nirugu in word should be long enough (at least not short than half of the full-width character) for avoiding visual confusion. According to the modern Mongolian orthography, [illegible], in the word [illegible], can't occur behind consonant letter, Nirugu connects [illegible] and [illegible]. Form the aesthetic perspective, the length of the Nirugu should be shorten.

## 4.5 Phoneme Indicating Lebel

Through declaring basic character set and presentation control, different character or character combination show differently on visual so as to eliminate the repeated encoding problem in the former

Mongolian encoding standard. However, this encoding system departures from the traditional perception of Mongolian alphabetical system, namely, Mongolian text can't be sorted by Unicode itself. To trade off the contradiction between the traditional perception of Mongolian alphabetical system and the information carrying capacity of the grapheme, this model applies Phonetic Indicating Label (PIL) to discriminate heteronym in modern Mongolian. As Table 4-4:

**Table 4-4:** Modern Mongolian Phonetic Variations

| Grapheme | Position | Conditions | First phonetic variant | | Second phonetic variant | |
|---|---|---|---|---|---|---|
| | | | Phonetic | Code | Phonetic | Code |
| ‍ᠠ‍ | Medial | None | ɑ | 1820 | e | 1820 PIL |
| ‍ᠠ | Final | None | ɑ | 1820 | e | 1820 PIL |
| ‍ᠠ᠋ | Final | None | ɑ | 1820 180B | e | 1820 180B PIL |
| ᠡ‍ | Initial | None | e | 1821 | ɑ | 1821 PIL |
| ᠡ | Isolate | None | e | 1821 | ɑ | 1821 PIL |
| ᠢ‍ | Initial | None | ɪ | 1822 | i | 1822 PIL |
| ‍ᠢ‍ | Medial | None | ɪ | 1822 | i | 1822 PIL |
| ‍ᠢ | Final | None | ɪ | 1822 | i | 1822 PIL |
| ᠢ | Isolate | None | ɪ | 1822 | i | 1822 PIL |
| ‍ᠣ | Final | None | o | 1823 | u | 1823 PIL |
| ᠣ | Isolate | None | o | 1823 | u | 1823 PIL |
| ᠤ‍ | Initial | None | o | 1824 | u | 1824 PIL |
| ‍ᠤ‍ | Medial | First syllable & Non-semivowel | o | 1824 | u | 1824 PIL |
| ‍ᠤ‍ | Medial | First syllable & Masculine | o | 1824 | u | 1824 PIL |
| ‍ᠤ‍ | Medial | First syllable & Feminine | ö | 1824 | ü | 1824 PIL |
| ‍ᠤ‍ | Medial | Non-first syllable & Masculine | o | 1824 | u | 1824 PIL |
| ‍ᠤ‍ | Medial | Non-first syllable & Feminine | ö | 1824 | ü | 1824 PIL |

| | | | | | | |
|---|---|---|---|---|---|---|
| ᠤ | Final | First syllable & Non-semivowel | u | 1824 | ü | 1824 PIL |
| ᠤ | Final | First syllable & Masculine | o | 1824 | u | 1824 PIL |
| ᠤ | Final | First syllable & Feminine | ö | 1824 | ü | 1824 PIL |
| ᠤ | Final | Non-first syllable & Masculine | o | 1824 | u | 1824 PIL |
| ᠤ | Final | Non-first syllable & Feminine | ö | 1824 | ü | 1824 PIL |
| ᠤ | Final | Before separating A/E | o | 1824 | w | 1824 PIL |
| ᠤ | Isolate | None | u | 1824 | ü | 1824 PIL |
| ᠤ᠋ | Initial | None | u | 1824 180B | ü | 1824 180B PIL |
| ᠤ᠋ | Isolate | None | u | 1824 180B | ü | 1824 180B PIL |
| ᠥ | Final | None | ö | 1825 | ü | 1825 PIL |
| ᠥ | Isolate | None | ö | 1825 | ü | 1825 PIL |
| ᠦ | Initial | None | ö | 1826 | ü | 1826 PIL |
| ᠦ | Medial | None | ö | 1826 | ü | 1826 PIL |
| ᠬ | Medial | None | x | 182C | g | 182C PIL |
| ᠬ᠌ | Initial | None | x | 182C 180C | g | 182C 180C PIL |
| ᠬ᠌ | Medial | None | x | 182C 180C | g | 182C 180C PIL |
| ᠲ | Initial | None | t | 1832 | d | 1832 PIL |
| ᠳ | Medial | None | t | 1833 | d | 1833 PIL |
| ᠶ᠋ | | Before separating A/E | j | 1836 180B | y | 1836 180B PIL |
| ᠸ | Initial | None | w | 1838 | o | 1838 180B PIL |
| ᠸ | Medial | None | w | 1838 | o | 1838 180B PIL |

In the table, *ɪ* is masculine, *i* is feminine. Treat the syllable behind the Mandatory Hyphen or gender changed syllable as first syllable.

Trough Phonetic Indicating Lebel separates the phonetic information and graphemic information in the alphabet to avoid unstable phonetic information causing bias. PIL is a fault tolerance encoding, that means during the whole typing process, user (source) is permitted to input PIL that does not conform standard Mongolian (Barimjiya Abiya), recipient(sink) can utilize the phonetic information from the PIL in the text, while also ignore it because of its uncertainty. To repeal the PIL will turn the phonetic code into graphemic code.

Phonetic Variation Selector (PVS, U+180D) is selected as PIL in the Model.

## 4.6 About Separated Suffix

This Model doesn’t treat differently on separate suffix. The feature that separate suffix follows the preceding word is realized by NARROW NO-BREAK SPACE, U+202F, to avoid line-break in the text, but NNBSP is banned to present any variant.

# 5 Realization

## 5.1 Mandatory Separation Approach

Mandatory Separation Approach is presented by a list of characters' contextual connectable property which contains "Current Character", "the Position of Current Character", "List Type", "Previous Character", "the Position of Previous Character", "Next Character", "the Position of Next Character" seven fields.

The Model requires the list of characters' contextual connectable property includes information as Table 5-1:

**Table 5-1:** Mongolian characters' contextual connectable property

| Current character | | List type | Previous character | | Next character | |
|---|---|---|---|---|---|---|
| Code points | Position | | Code points | Position | Code points | Position |
| 1820 | | deny | 1820 | | | |
| 1820 | | deny | 1821 | | | |
| 1820 | | deny | 1822 | | | |
| 1820 | | deny | 1822 180B | | | |
| 1820 | | deny | 1824 | | | |
| 1820 | | deny | 1826 | | | |
| 1820 | | deny | 1827 | | | |
| 1820 | | deny | 1828 180B | | | |
| 1820 | | deny | 1833 180B | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| 1820 | medial | permit | 1836 180B | medial | 1828 180B | final |
| 1820 | medial | permit | 1836 180B | medial | 1837 | final |
| 1821 | medial | deny | | | | |
| 1821 | final | deny | | | | |
| 1822 | | deny | 1820 180B | | | |
| 1822 | | deny | 1820 | | 1822 | |
| 1822 | | deny | 1821 | | 1822 | |
| 1822 | | deny | 1822 | | 1822 | |
| 1822 | | deny | 1824 | | 1822 | |
| 1822 | | deny | 1824 180B | | 1822 | |
| 1822 | | deny | 1826 | | 1822 | |
| 1822 | | deny | 1827 | | 1822 | |
| 1822 180B | initial | permit | | | 1836 180B | |
| 1822 180B | medial | permit | 1820 | | | |
| 1822 180B | medial | deny | | | 1822 | |
| 1822 180B | medial | deny | | | 1822 180B | |
| 1822 180B | medial | deny | | | 1836 180B | |
| 1823 | initial | deny | | | | |
| 1823 | medial | deny | | | | |
| 1824 180B | initial | permit | | | 1824 | final |
| 1824 180B | initial | permit | | | 1828 180B | final |
| 1824 180B | initial | permit | | | 1833 180B | final |
| 1825 | initial | deny | | | | |
| 1825 | medial | deny | | | | |
| 1826 | final | deny | | | | |
| 1827 | | deny | 1820 | | | |
| 1827 | | deny | 1821 | | | |
| 1827 | | deny | 1822 | | | |
| 1827 | | deny | 1824 | | | |
| 1827 | | deny | 1826 | | | |
| 1827 | | deny | 1827 | | | |
| 1828 180B | | permit | 1820 | | | |
| 1828 180B | | permit | 1821 | | | |
| 1828 180B | | permit | 1822 | | | |
| 1828 180B | | permit | 1824 | | | |
| 1828 180B | | permit | 1826 | | | |
| 1828 180B | | permit | 1827 | | | |
| 1828 180B | | deny | | | 1820 | |
| 1828 180B | | deny | | | 1822 | |
| 1828 180B | | deny | | | 1823 | |
| 1828 180B | | deny | | | 1824 | |
| 1828 180B | | deny | | | 1825 | |
| 1828 180B | | deny | | | 1826 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| 1828 180B | | deny | | | 1827 | |
| 1828 180B | | deny | | | 1828 180B | |
| 1828 180B | | deny | | | 1833 180B | |
| 182C | medial | permit | 1820 | | | |
| 182C | medial | permit | 1822 | | | |
| 182C | medial | permit | 1824 | | | |
| 182C | medial | permit | | | 1820 | |
| 182C | medial | permit | | | 1822 | |
| 182C | medial | permit | | | 1823 | |
| 182C | medial | permit | | | 1824 | |
| 1833 180B | | permit | 1820 | | | |
| 1833 180B | | permit | 1821 | | | |
| 1833 180B | | permit | 1822 | | | |
| 1833 180B | | permit | 1824 | | | |
| 1833 180B | | permit | 1826 | | | |
| 1833 180B | | deny | | | 1820 | |
| 1833 180B | | deny | | | 1822 | |
| 1833 180B | | deny | | | 1823 | |
| 1833 180B | | deny | | | 1824 | |
| 1833 180B | | deny | | | 1825 | |
| 1833 180B | | deny | | | 1826 | |
| 1833 180B | | deny | | | 1827 | |
| 1833 180B | | deny | | | 1828 180B | |
| 1833 180B | | deny | | | 1833 180B | |
| 1836 | final | deny | | | | |
| 1836 180B | initial | permit | | | 1822 | final |
| 1836 180B | initial | permit | | | 1822 | medial |
| 1836 180B | medial | permit | | | 1820 | medial |
| 1836 180B | medial | permit | | | 1820 180B | final |
| 1836 180B | medial | permit | | | 1822 | final |
| 1836 180B | medial | deny | 1822 | medial | | |
| 183E | | deny | 182F | | | |
| 1841 | initial | permit | | | 1822 | |
| 1841 | medial | deny | | | | |
| 1841 | final | deny | | | | |
| 1842 | initial | permit | | | 1822 | |
| 1842 | medial | deny | | | | |
| 1842 | final | deny | | | | |

In the table above:

- Character: Character or Ligature.
- Position: initial, medial, final three possible forms.

- List Type: Permit and Deny, for the same current character, List Type must be one of the two types. If the value is Permit, all situation will be denied beyond the list, and vice versa.
- Null means no limitation. If a character or its contextual connection description on a position is not included in the list, the character is no limitation on contextual connection.
- If a character has multiple Previous Character in the same position, or multiple Next Character, the logic relationship of them is OR.
- If a character has a Previous Character and a Next Character in the same position, and List Type of the Previous Character and the Next Character is identical, then their logic relationship is XOR, if isn't, then their logic relationship is AND.
- Inserting position of Dashed Nirugu: If current character's position is initial or medial, insert Dashed Nirugu behind the variant, while if current character's position is final, insert it in front of the variant.

## 5.2 Sorting Algorithm

Sorting Algorithm accords with code point's sequence, ignores SVS, HVS by default and except PVS.

- Sorting algorithm ignores the Nirugu.
- SVS and HVS don't refer to character or string comparing directly, but last character < current character + SVS < current character +HVS < next character.
- PVS takes part in character or string comparing, and its code value is bigger than any Basic Characters' code value.

For sorting algorithm, < ᠬ 182C, ᠍ 180D > is equivalent to < ᠭ 182D >, < ᠬ 182C, ᠌ 180C, ᠍ 180D > is equivalent to < ᠭ 182D, ᠌ 180C >.

For instance:

**Table 5-2:** Examples

| Grapheme | Position | Phonetic | Character sequence | Sort results |
|---|---|---|---|---|
| ᠠ | Medial | ɑ | < ᠠ 1820 > | 1 |
| ᠠ | Medial | e | < ᠠ 1820, ᠍ 180D > | 2 |
| ᠢ | Medial | i | < ᠢ 1822 > | 3 |
| ᠤ | Medial | o | < ᠤ 1824 > | 4 |
| ᠤ | Medial | u | < ᠤ 1824, ᠍ 180D > | 5 |

| | | | | |
|---|---|---|---|---|
| ᠦ | Medial | ö | < ᠦ 1826 > | 6 |
| ᠦ᠍ | Medial | ü | < ᠦ 1826, ᠍ 180D > | 7 |
| •••• | | | | |
| ᠬᠠ | Isolate | xa | < ᠬ 182C, ᠠ 1820 > | 1 |
| ᠬ᠌ᠠ᠍ | Isolate | xe | < ᠬ 182C, ᠌ 180C, ᠠ 1820, ᠍ 180D > | 2 |
| ᠬ᠌ᠢ | Isolate | xi | < ᠬ 182C, ᠌ 180C, ᠢ 1822 > | 3 |
| ᠬᠣ | Isolate | xo | < ᠬ 182C, ᠣ 1823 > | 4 |
| ᠬᠣ᠍ | Isolate | xu | < ᠬ 182C, ᠣ 1823, ᠍ 180D > | 5 |
| ᠬ᠌ᠥ | Isolate | xö | < ᠬ 182C, ᠌ 180C, ᠥ 1825 > | 6 |
| ᠬ᠌ᠥ᠍ | Isolate | xü | < ᠬ 182C, ᠌ 180C, ᠥ 1825, ᠍ 180D > | 7 |
| ᠭᠠ | Isolate | ga | < ᠭ 182D, ᠠ 1820 > | 8 |
| ᠬ᠌᠍ᠠ᠍ | Isolate | ge | < ᠬ 182C, ᠌ 180C, ᠍ 180D, ᠠ 1820, ᠍ 180D > | 9 |
| ᠬ᠌᠍ᠢ | Isolate | gi | < ᠬ 182C, ᠌ 180C, ᠍ 180D, ᠢ 1822 > | 10 |
| ᠭᠣ | Isolate | go | < ᠭ 182D, ᠣ 1823 > | 11 |
| ᠭᠣ᠍ | Isolate | gu | < ᠭ 182D, ᠣ 1823, ᠍ 180D > | 12 |
| ᠬ᠌᠍ᠥ | Isolate | gö | < ᠬ 182C, ᠌ 180C, ᠍ 180D, ᠥ 1825 > | 13 |
| ᠬ᠌᠍ᠥ᠍ | Isolate | gü | < ᠬ 182C, ᠌ 180C, ᠍ 180D, ᠥ 1825, ᠍ 180D > | 14 |
| •••• | | | | |
| ᠲᠠ | Initial | ta | < ᠲ 1832, ᠠ 1820 > | 1 |
| ᠲᠠ᠍ | Initial | te | < ᠲ 1832, ᠠ 1820, ᠍ 180D > | 2 |
| ᠲᠢ | Initial | ti | < ᠲ 1832, ᠢ 1822 > | 3 |
| ᠲᠤ | Initial | to | < ᠲ 1832, ᠤ 1824 > | 4 |
| ᠲᠤ᠍ | Initial | tu | < ᠲ 1832, ᠤ 1824, ᠍ 180D > | 5 |

| | | | | |
|---|---|---|---|---|
| ᠲᠦ | Initial | tö | < ᠲ 1832, ᠦ 1826 > | 6 |
| ᠲᠦ᠍ | Initial | tü | < ᠲ 1832, ᠦ 1826, ᠍ 180D > | 7 |
| ᠲ᠍ᠠ | Initial | da | < ᠲ 1832, ᠍ 180D, ᠠ 1820 > | 8 |
| ᠲ᠍ᠠ᠍ | Initial | de | < ᠲ 1832, ᠍ 180D, ᠠ 1820, ᠍ 180D > | 9 |
| ᠲ᠍ᠢ | Initial | di | < ᠲ 1832, ᠍ 180D, ᠢ 1822 > | 10 |
| ᠲ᠍ᠤ | Initial | do | < ᠲ 1832, ᠍ 180D, ᠤ 1824 > | 11 |
| ᠲ᠍ᠤ᠍ | Initial | du | < ᠲ 1832, ᠍ 180D, ᠤ 1824, ᠍ 180D > | 12 |
| ᠲ᠍ᠦ | Initial | dö | < ᠲ 1832, ᠍ 180D, ᠦ 1826 > | 13 |
| ᠲ᠍ᠦ᠍ | Initial | dü | < ᠲ 1832, ᠍ 180D, ᠦ 1826, ᠍ 180D > | 14 |
| •••• | | | | |

## 5.3 Searching Algorithm

Searching algorithm ignores SVS and PVS by default, while HVS is detected. Thus, can ignore the control character in string contrast. Under the condition that switch off whole word searching function, the model can locate word like ᠪᠠᠶᠢᠭᠤᠯᠤᠨ by ᠪᠠᠢ ᠭ, ᠲᠠᠭᠠᠨᠠᠯᠠᠭᠠᠪ by ᠲᠠᠭᠠᠨ ᠭ, ᠰᠤᠷᠤᠯᠴᠠᠪ by ᠰᠤᠨ. When we switch on the whole word searching function, we need to revoke the ignorance of the SVS.

Besides, in searching algorithm, < ᠺ 183A> is strictly equal to < ᠻ 183B >, < ᠯ 182F, ᠾ 183E > is strictly equal to < ᡀ 1840 >.

## 5.4 Input Method Algorithm

The model mainly considers keyboard mapping complexity and algorithm complexity of Mongolian Soft Variant, Hard Variant and Phonetic Variant. Input method can alter graphic input and phonetic input easily through Phonetic Indicating Label and traversing standard word corpus, so that supports spell-check and proofreading, pronunciation obfuscation and proofreading.

## References

[1] *Surgaltu, A Character Encoding Proposal of the Unicode for Preserving the Mongolian Alphabetical System, http://www.erillab.cn/, 2019.*

[2] *The Unicode® Standard Version 12.0 – Core Specification, http://www.unicode.org/.*

[3] *DONG Zhi-jiang, WU Jian, ZHONG Yi-xin, Research and Implementation on Complex Text Processing Based on the OpenType, Application Research of Computers, 2004.*

[4] *Quejingzhabu, Mongolian Information Processing Specialty, 2014.*