ZWJ/ZWNJ

L2/05-307

behavior under Indic scripts with special reference to
chillu, conjuncts, etc in Malayalam

Rajeev J Sebastian
Rachana Akshara Vedi

1 Definitions

ZWJ and ZWNJ are format control characters with additional specific behavior in various scripts.

2 Uses of ZWJ and ZWNJ

In Malayalam script, the ZWJ and ZWNJ have the following uses:

ZWJ is used in two contexts:

- to indicate the half-consonant in other Indic scripts, while in Malayalam it is used to indicate chillu:
  ന്‍ = < ന chandrakkala ZWJ ‍ >
- to force C_2-conjoining forms in all Indic scripts:
  ‌പ്‍ = < പ ZWJ chandrakkala പ്‍ >

ZWNJ

- ZWNJ is used only in one context, and that is to indicate the overt-chandrakkala (explicit halant).
  ന്‌ = < ന chandrakkala ZWNJ ‌ >

3 What is the semantic load of ZWJ/ZWNJ ?

The present discussion and dispute regarding chillu (and even other presentation variants) makes one basic assumption: that ZWJ/ZWNJ cannot be used to provide a distinction between 2 visually different words such that in encoded form they differ only by a joiner, citing the reason that joiners do not encode 'semantic' differences.

From the above, we notice 2 basic statements about Unicode and the Indic encoding model:

1. Visual differences always translate to so-called 'semantic' differences

2. ZWJ/ZWNJ cannot be used to encode 'semantic' differences

To illustrate both these statements, some examples and contexts were provided in Malayalam.

The statement that whenever there is a visual difference between 2 words it is a semantic difference, cannot be substantiated very easily in the Indic scripts. The Unicode encoding is based on this principle, and it also is a fundamental principle of Indic scripts.

To see this, one need go no further than Devanagari. In Devanagari, half-forms exist both for consonants and for conjuncts. Although conjunct formation can be restricted using the ZWJ/ZWNJ, this is purely a contextual process, i.e., a user, at his discretion chooses which rendering he wishes according to the choices available in the rendering system (i.e., shaping engine and font).

This is the same model adopted in the Malayalam script. Conjuncts and restricted renderings of conjuncts, i.e., C_1-conjoing forms, C_2-conjoining forms and explicit chandrakkala forms, are basically equivalent and the appropriate rendering is chosen by the writer.

3A Format Control characters should not be script-overloaded

Script overloading is a part of the specification for ZWJ/ZWNJ, i.e., according to TUS Section 15.2,

> *"The ZWJ and ZWNJ also have specific interpretations in certain scripts as specified in this standard. For example, in Indic scripts the ZWJ provides an invisible neighbor to which a dead consonant may join to induce a half-consonant form (see Section 9.1, Devanagari)."*

Also, this paragraph implies that the functions of ZWJ/ZWNJ are bound to specific interpretations. Thus, ignoring of ZWJ/ZWNJ for various purposes need not be applied to these scripts as the case may be.

However, when ZWJ/ZWNJ appear in a text stream, independent from the specific cases or scripts, those occurrences are to be ignored.

3B Format Control characters do not have semantic load

Semantic load, or "meaning", have no basis in Unicode. For e.g., there is no need for the Unicode encoding to distinguish between the various meanings of "read". In a similar way, neither meaning of words (or non-words) like പിന്നിലാവും /pi̱nilāvum/, nor pronunciation of graphemes such as ർ /r/ have a basis in an encoding.

On the other hand, the Unicode does analyze the functional properties of characters and scripts, in order to achieve encoding. ZWJ/ZWNJ are 2 such characters which arise out of the functional properties of scripts. They correspond to mental operations of a user to choose the appropriate rendering of a sequence C_1 + chandrakkala + C_2. In this respect, the ZWJ/ZWNJ are the saving grace of Indic and Malayalam computing.

Within Indic scripts, the functions of ZWJ and ZWNJ have such well-defined properties. ZWJ/ZWNJ also have additional specific interpretations (or information content) in other scripts.

3C The sort order weight is something alien to encoding

Some persons have put forward certain arguments that the sort order has nothing to do with encoding a character. The sort order, or collation order, is indeed something alien to encoding from the perspective of numeric value of particular codepoints assigned by UTC.

However, the existence of codepoints, i.e., whether a codepoint should be assigned to provide contrast, has to resort to collation, in the same way as rendering, line breaking and other applications.

For e.g., the functional properties of characters typically begin with a specification of its position in the sort order in Malayalam, just as in English. In short, the encoding, or description of fundamental units of a script for general purpose computing, impact applications greatly.

Hence in Malayalam (and for the Malayalam script), the functional properties or the "value" (or, information content, weight, information unit-ness, etc) of the codepoint U+0D03 begin with the statement that it is the first character of the Malayalam "aksharamala" or alphabet set (but not necessarily numerically the first in the Unicode Malayalam block); it is a vowel; and so on.

Thus the sort order is definitive, just like the aksharamala, in determining the unit-ness of characters, i.e., their existence, their distinction from other characters, and so on.

Malayalam grammarians themselves have voiced that the chillus are not in the aksharamala; this is because chillu is not a basic character, rather it is a presentation variant. Thus, the opinion that sort order has no influence on encoding issues is absolutely wrong, and was generated without knowing the real facts regarding Malayalam and Unicode stated above.

In the case of chillus, the aksharamala is not enough to establish the information content for chillus, since chillus are not in the aksharamala. For this we have recourse only to the sort order which gives definitive values to the chillu. Under the Unicode scheme, this is the critical difference between the 'basic character' and 'glyph' perspectives: the Malayalam sorting scheme places the chillu as a variant (glyph perspective) of the corresponding consonant (basic character perspective). In particular, the sorting scheme places the chillu as a special rendering of the vowelless form of the consonant (see the relevant section of L2 05-210)

It is also important to note that the determination of sort order cannot be made in a callous way as suggested by some persons. The sort order is important for various reasons; the economic implications of choosing a wrong sort order is immense, not to mention the loss of ergonomy for the user and efficiency of the sort algorithm; also, the sort order cannot be meddled with on some persons' whim or fancy.

4 ZWJ/ZWNJ at the application level

Some persons are of the opinion that ZWJ/ZWNJ are purely format control characters and so can only be used for rendering and they cannot be used within spell checkers, etc. According to them, the consequence of this is that the chillus need to be encoded as unique codepoints.

This may be due to the provision of ZWJ/ZWNJ in the General Category class Cf (Other, Format).

Even though the ZWJ/ZWNJ are mentioned as format control characters in the relevant section of TUS 15.2 and in UCD, they are in fact capable of gaining more functionality depending on the script in which they are embedded. Also, applications are free to give language/script dependent or independent interpretations to ZWJ/ZWNJ within limits of good reason.

So, for e.g., in collation, the ZWJ/ZWNJ can be given a weight in a sequence which also contains some non-ignorable characters. In line breaking and text boundaries, ZWJ/ZWNJ are not considered as Control characters.

Just as for CGJ, the Canonical Combining Class value of ZWJ/ZWNJ is 0, and thus from TUS Section 15.2,

> *"The presence of the ccc=0 combining grapheme joiner blocks the reordering of hiriq before patah by canonical reordering. That allows the two sequences to be reliably distinguished, whether for display or for other processing."*

Although the use of CGJ and ZWJ/ZWNJ is not comparable (the use of CGJ illustrated is in combining character reordering algorithm), it is illustrative of the use of character properties to give 'semantic' value to otherwise ignorable characters- just like the ZWJ/ZWNJ, the CGJ is also by default a completely ignorable character.

4A Collation and collation properties

In the UCA standard, in Section 3.1.5 Collation Grapheme the following is mentioned:

> *"A collation ordering determines a collation grapheme cluster (also known as a collation grapheme or collation character), which is a sequence of characters that is treated as a primary unit by the ordering. For example, ch is a collation grapheme for a traditional Spanish ordering. These are generally contractions, but may include additional ignorable characters. [...]"*

Also from Section 3.1.6 Combining Grapheme Joiner,

> *"Sequences of characters which include the combining grapheme joiner or other completely ignorable characters may also be given tailored weights. [...]"*

From the data files, it seems that ZWJ and ZWNJ being "completely ignorable" characters, they can participate in contractions, and such contractions can be given tailored weights via the rules mentioned above.

Considering the text in section 15.2 of the Unicode Standard,

> *"ZWJ and ZWNJ are format control characters. Like other such characters, they should be ignored by processes that analyze text content. For example, a spelling-checker or find/replace operation should filter them out."*

As far as searching and sorting operations are concerned, the default behavior of ZWJ and ZWNJ being filtered out for those operations, occurs only after any contractions are applied. Thus, for spell-checkers and search engines the default behavior is adequate for Malayalam computing.

Greedy regular expressions also match the ZWJ/ZWNJ characters based on the level of collation algorithm. This matches fully with the expectations users have when using find/replace operations.

The default collation weights provided by Unicode in the DUCET, must necessarily be language independent. This is to prevent conflicts and to provide unique weights for a sequence which may not be correct (e.g., "ch" in Spanish and English have different weights and thus both languages cannot be supported by the DUCET). This is also why language dependent sorting is provided in tailorings of the data tables.

In the case of Malayalam, giving a sequence containing ZWJ/ZWNJ a weight in a contraction with other non-ignorable, does not create conflicts with other language and so it is possible not only to implement it in tailorings, but also, if necessary, in the default collation tables of Unicode.

4B Text boundaries

From the point of view of break iteration, the ZWJ and ZWNJ are in the class Combining Mark (CM), which conforms to the default behavior expected for Indic script usage.

In fact, according to UAX#29 Table 1, for detecting grapheme clusters, ZWJ/ZWNJ are specifically not Control characters. Also, in UAX#29 Table 2, for detection of word boundaries, ZWJ/ZWNJ are specifically not Format characters. In UAX#29 Table 3, for detection of default Sentence boundaries, ZWJ/ZWNJ are specifically not Format characters. So, as far as default behaviour is concerned, ZWJ/ZWNJ meets expectations.

However, tailorings can go beyond this, and the new standard language for description of break iteration

tailorings is a welcome addition to Unicode in this regard, though not strictly necessary for Malayalam.

4C Additional behavior of ZWJ/ZWNJ in various scripts

Considering the text in section 15.2 of the Unicode Standard,

> *"The ZWJ and ZWNJ also have specific interpretations in certain scripts as specified in this standard. For example, in Indic scripts the ZWJ provides an invisible neighbor to which a dead consonant may join to induce a half-consonant form (see Section 9.1, Devanagari)."*

Also, mentioned in the section is that there is additional behavior of ZWJ/ZWNJ in various Indic scripts (and, it seems from some mailing list messages and other websites, for Arabic as well). So, ZWJ/ZWNJ may have additional value (or information content) when it appears in a text stream bounded by Indic codepoints.

4D Codepoints for consonant+explicit chandrakkala?

Some persons have put forward the proposal for encoding chillus stating that the ZWJ is 'not semantic' and will be filtered by higher-level applications with a semantic operation. This is not true as mentioned above, since

1. various applications are allowed to make their own interpretations regarding codepoints
2. the standardized applications of Unicode (rendering, collation, text boundaries, etc) give a 'semantic' value to ZWJ/ZWNJ inasmuch as they are not considered to be purely format control characters.

In addition, if, as these persons claim, chillus must be encoded to discontinue the use of ZWJ, then so must the ZWNJ. Consider, for e.g., the case of ദൃക്സാക്ഷി. In this word, by convention, the ക്സ must occur with explicit chandrakkala and should not be combined, even though either rendering is fundamentally the same. In the existing model, this is forced by using ZWNJ.

As a corollary to the chillu encoding proposal, if chillus have to be encoded to replace the function of ZWJ, i.e., if chillu-na must be encoded to replace the function of <ന chandrakkala ZWJ>, then explicit-chandrakkala-consonant must be encoded to replace the function of ZWNJ, for e.g., explicit-chandrakkala-ka must also be encoded to replace the function of <ക chandrakkala ZWNJ>.

When considering that the consonant+explicit chandrakkala rendering has to be encoded, we will require the allocation of a huge number of codepoints, one for each consonant.

In general, it is true that the value (or information content) of codepoints may be different for various applications, i.e., the application determines the information content of characters and text streams. For

e.g., whether the specific stream “read” is in the past tense or present tense is not required for a collation algorithm but it is required for a grammar checker. Similarly, whether the Malayalam codepoint U+0D31 റ is the sound /r̲a/ or /t̲a/ is not required for a shaping engine, but is required for a text-to-speech system.

In the case of Malayalam, making non-standard interpretations is not required, since the values of ZWJ/ZWNJ are already perfect for Malayalam computing from all perspectives.

5 The undisputed status of ZWJ/ZWNJ in Devanagari

One notable feature of Malayalam is the case of chillu and its relationship with the explicit halant form of consonants in Devanagari. If we look closely, they are both the same, even though they are derived from Dravidian and Sanskrit environments respectively. In Sanskrit, word-final vowellessness is represented by the halant. Similarly, in Malayalam for five consonants, it is marked as chillu.

It is quite interesting to note that, the tail-like halant points downwards in Sanskrit, whereas in Malayalam, it goes upwards! In almost all contexts, they behave the same way.

It is important to note that there is no comparison between the half forms of Devanagari and the chillu forms of Malayalam. However, in general, both may be considered as specialized presentation variants of the underlying consonants, i.e., C_1-conjoining forms. From this perspective, forcing the half-form in Devanagari and the chillu in Malayalam with ZWJ is entirely appropriate.

In Devanagari, the sequence <C_1 chandrakkala ZWNJ> is rendered as explicit halant on C_1 and has been fixed for several years beyond dispute. The same sequence renders as overt-chandrakkala in Malayalam. Also, in Devanagari, the ZWJ is used to produce half-forms of consonants, i.e., variant presentation of vowelless consonants or C_1-conjoining forms. In Malayalam, the sequence <C_1 chandrakkala ZWJ> is rendered as chillu.

The ZWJ/ZWNJ is effectively utilized for Devanagari computing. In that case, how is the same so highly disputed in Malayalam ?

Also, it should be noted that, the said equivalent characters of chillu in Devanagari, i.e., halant and half forms, do not have specific codepoints in Unicode.

Thus, if the chillu codepoints are accepted for Malayalam, then it will be against the general Indic encoding model.

6 Features and problems in input and rendering

Input and rendering algorithms generally work the same way as Indic scripts such as Devanagari. The exceptions are the specific features of Malayalam such as gopi, ന്റ and ൺഠ, and the issue of font expressivity.

6A Repham or Gopi of Malayalam script

Questions regarding the repham of Malayalam script has been raised by some persons, and they connected those issues with the application of ZWJ/ZWNJ. There are a few important issues to consider when using the repham:

1. The gopi mark is formed by the sequence ര + chandrakkala + C_2 and it is considered a conjunct in Malayalam.
2. In recent times, the repham or gopi mark is considered a deprecated symbol in Malayalam, both in Original and Typewriter scripts. This occurred as part of the natural evolution of Malayalam.
3. The user does not expect that a sequence ര + chandrakkala + C_2 would render the gopi mark on C_2. This is quite unlike the similar case of Devanagari, where the reph-form is rendered for such sequences.

Considering the points above in light of the Unicode standard, the gopi is not the default rendering for a sequence ര + chandrakkala + C_2 and so should not be rendered by default. Instead, the standard fallback measures should be applied to render this as ർ + C_2. This is done adequately by providing the mechanism in the font: when the font does not contain the gopi mark, then standard fallback measures should be used.

6B Issue of font expressivity

Unicode text is rendered using a combination of a shaping engine and a set of fonts. Text is interpreted by the shaping engine to capture intent of the author of the text.

By expressivity, we refer to the ability of the rendering engine (combination of shaping engine and font) to express this intent. This ability improves with a larger number of conjuncts, etc.

This is the core issue that faces Malayalam: when a piece of text produced with a font with a lower expressivity is viewed with one with higher expressivity, or vice-versa, there is a loss of information, i.e., the

intent of the author would be lost during the interchange of text.

Consider for example a font without the ന്മ conjunct. A user targeting the ൻമ rendering will automatically see it with the sequence <ന chandrakkala മ> (due to fallback rendering).

When this text is subsequently passed to a computer with any font with ന്മ conjunct, this renders as ന്മ, although the intent of the author was ൻമ.

Although ന്മ and ൻമ are linguistically equal as far as value is concerned, clearly, the rendering engine was not able to reproduce the intent of the author.

Although, this looks like a case for encoding chillu forms, the problem does not end here.

Consider for instance a font without the സ്ന conjunct. A user targeting the rendering സ്ന uses the sequence /സ chandrakkala ന/ which achieves what he desires.

However, viewing this text with a font of higher expressivity, it renders the സ്ന. Here too, the intent of the author was not captured by the rendering engine.

This problem exists immaterial of encoding issues: irrespective of the encoding scheme for the theoretical sequence <C_1 chandrakkala C_2> and its variants, the font that renders this sequence must support all variants. Fallback rendering is not at fault here.

Neither is this problem something to do with Original vs Typewriter script, as commented by some persons: even within all several Typewriter script schemes and software packages, this problem will happen.

It is important to note that in the Original script of Malayalam, as in the case of Devanagari, a definite and consistent scheme to formulate the conjuncts exists. Due to this, such serious problems do not occur.

This problem in all its complexities appears in the Typewriter script, where the concept and scheme of conjuncts have been demolished through the illogical reform. It is due to this that Malayalam was alienated from the rest of the Indic scripts.

The question is, how to allow a user to input the unambiguous sequence, and how to render them unambiguously. The answer to both involves the entire rendering apparatus: even if shaping engines can unambiguously interpret such sequences, the expressivity of the font is still a limiting factor.

One possible solution for this problem, is to remove the fallback precedence hierarchy for rendering Malayalam. However, this too has issues; Conjuncts and chillus will not be represented properly or easily, and there will be a disconnect from the Indic encoding model. It also makes great difficulties for users which may lead to conflict.

This has severe consequences which are not warranted, considering the historical development of the Malayalam script, and the needs of users.

This problem does not exist in other Indic scripts. This is because of the existence of a standard set of glyphs.

The only possible way to solve this problem is to include a standard set of glyphs in every font. As noted above, this makes Malayalam conform to the other Indic scripts. This standard set of glyphs must necessarily include the largest set of allowed combinations. This is true also for the so-called Reformed script. This is because only 2 "reforms" have occurred as a result of the natural evolution of the Malayalam script - that of deprecation of repham (or gopi) mark, and that of dropping the left part of the two part AU vowel.

6C Chillu is equivalent to C_1 + chandrakkala + ZWJ

Some persons are of the opinion that chillus must be encoded. This is done under the guise that wherever chillus are rendered on screen, the renderer will have in the backing store a chillu codepoint.

However, this is a fallacy: chillus can also occur via fallback rendering of the C_1 + chandrakkala + C_2 sequence (e.g., ൽപ).

The only way to make the chillu encoding successful would be to remove fallback rendering, because by this, chillus would have a guaranteed encoding as C_1 + chandrakkala + ZWJ.

Removing fallback encoding would invalidate the chillu encoding proposal by removing the concerns of this proposal.

In other words, chillu encoding is equivalent to the sequence C_1 + chandrakkala + ZWJ, without fallback rendering, and it is precisely with fallback rendering that C_1 + chandrakkala + ZWJ finds its use.

Therefore, the chillu encoding proposal is not necessary to meet its concerns, since the guarantee of a chillu rendering can be provided using the ZWJ.

In the current scheme,

1. C_1 + chandrakkala automatically renders the chillu.
2. In C_1 + chandrakkala + C_2, if a conjunct is not available, (and if C_2-conjoining form is not available), C1 is transformed into the C1-conjoining form.
3. Also, C_1 + chandrakkala + ZWJ forces the C_1-conjoining form.

With the chillu encoding proposal, the sequence C_1 + chandrakkala + ZWJ is replaced with the chillu

codepoint, i.e.,

1. C_1 + chandrakkala automatically renders chillu.
2. In C_1 + chandrakkala + C_2, if the conjunct is not available (and if C_2-conjoining form is not available), C_1 is transformed into the explicit chandrakkala form.
3. The sequence C_1 + chandrakkala + ZWJ + C_2 is deprecated, and instead it should be converted (either during normalization, or in the input method) into chillu-C_1 + C_2.
4. The sequence chillu-C_1 + C_2 would be rendered as chillu-C_1 + C_2

Considering the removal of fallback rendering,

1. If (1) is not allowed to form chillu, and instead it shows the explicit chandrakkala, vowellessness will be shown which is not particularly a problem (except from the point of view of ergonomy, and expectation of the user) with the Original script, but it is a problem with the Typewriter script, since the user may interpret a pseudo-samvruthokaram.
2. If (2) is not allowed to form chillu, and instead it shows the explicit chandrakkala form, immense problems occur: നമ will be shown as നമ് if the മ conjunct is not available. ന്റ will show ൻറ if ന്റ conjunct is not available.

Essentially, as far as rendering is concerned, there is no difference between C_1 + chandrakkala + ZWJ and the proposed chillu codepoint.

The only other concern is the 'semantic' issue with regard to ZWJ/ZWNJ. This is a baseless claim, and it is has already been dealt with.

6d C2-conjoining consonants

Some persons have shown a contrast involving C_1 + chandrakkala + C_2 sequences where both C_1-conjoining and C_2-conjoining form fallback is possible, which they claim is not possible to encode since ZWJ and ZWNJ are 'not semantic'.

First, we would like to state that ഴ്യ and ഴ്വ will not render as natural conjuncts in Malayalam. It needs further explanation, which is irrelevant in this context. What is relevant is the rendering mechanism for the sequence <ഴ chandrakkala ര>.

Apart from the ZWJ and ZWNJ 'semantics' issue, which has already been dealt with, there is no specific reason why the chillu should be encoded, and the same is also applicable to the request for specific codepoints for the C_2-conjoining consonant marks.

In the case of fallback rendering of a sequence C_1 + chandrakkala + C_2, where both C_1-conjoining and C_2-conjoining form is available in the current standard, C_2-conjoining form is preferred. This entirely within the expectations of Malayalam users:

1. C_1 + chandrakkala + C_2, rendered with a C_2-conjoining form is also considered as a conjunct.

2. Specialized cases such as ന്ദ്ര്യാ where instead of the conjunct ന്ദ്ര്യ, the യ postbase consonant mark should be shown, is easily handled with the ZWJ + chandrakkala + C_2 encoding.
3. In the Typewriter script, the sequence C_1 + chandrakkala + ര renders with the ര prebase consonant mark, i.e., the C_2-conjoining form. In fact, although the ര consonant mark is less than optimal in usage of Malayalam, at the encoding level and conceptually it conforms to the usage of existing consonant marks of Original Malayalam.

6e /nta/ vs /nra/

From the point of view of input: consider that there is a mechanism to produce ൻറ using the chillu character i.e., as the chillu-ന key followed by the റ key. Irrespective of the method to produce ന്റ, a general user will use the ൻറ method also for the ന്റ, since

1. they consist of the same components ന and റ
2. there is already a precedent of writing ന്റ just like ൻറ. This is reinforced by the ൻ rendering component within the ന്റ glyph.
3. given a word, with the chillu-ന side by side with the റ, the human requires context in order to understand whether it is ന്റ or ൻറ. It is not easy for input methods to implement this contextual process.

Thus, it will not be possible to disambiguate the ന്റ from the ൻറ.

6F Disambiguating pseudo-samvruthokaram from the chandrakkala

Some persons have raised the issue that a separate codepoint should be encoded in order to provide a distinct encoding for the psuedo-samvruthokaram.

However, this is not a practical solution. This is because

1. the user will use the chandrakkala to represent the pseudo-samvruthokaram in text, even if a key is provided to input the potential pseudo-samvruthokaram codepoint
2. it is extremely difficult to create basic input methods that are capable of recognizing when the user intends the pseudo-samvruthokaram rather than vowellessness, and vice-versa.

In any case, practically, the pseudo-samvruthokaram could usefully be distinguished from vowellessness, only in the limited case of certain noun formations (dative and nominative case of words ending with ന). It is not worthwhile to grant an interpretation of word-ending visible chandrakkala as a possible pseudo-samvruthokaram. A sort scheme that tries to achieve this will be very difficult to create, since it would have to use a dictionary to denote those cases. This has far reaching implications when considering Malayalam databases, etc.

7 Conclusion

In all respects, ZWJ/ZWNJ meets the requirements for Malayalam computing. The debate raised to encode chillus hinges on dropping the "semantic" use of ZWJ/ZWNJ. However, such a use of ZWJ/ZWNJ is not made in Malayalam, just as the use of ZWJ/ZWNJ in Devanagari does not raise any specific concerns of "semantics".

Ultimately, ZWJ/ZWNJ is a blessing for Malayalam and Indic computing, and does not create any problems for achieving Malayalam chillu.