ISO/IEC JTC 1/SC22/WG20 N898R

L2/02-008R

Universal multiple-octet coded character set
International organization for standardization
Organisation internationale de normalisation

Title: String ordering weighting roadmap

Source: Kent Karlsson
Date: 2002-02-04
Status: Expert contribution
Document type: Working group document
Action: For consideration by the UTC and JTC 1/SC 22/WG 20

1 Introduction

The Unicode Collation Algorithm has a default weighting table, and similarly the ISO/IEC 14651 standard (International string ordering and comparison) has a Common Template Table (CTT). The weights used/implied by these weighting tables, normally after tailoring for particular collation traditions, are used to create collation keys for strings. The weights (unless drastic tailoring is done) are ordered in a particular way to enable shorter collation keys.

The current scheme is only partial, however, and does this particular ordering only for level 3 weights versus level 2 weights, and level 2 weights versus level 1 weights. In the latest tables there is in addition, within the level 1 weights, a division between “ordinary” (or “lead”) weights and Hàn trail weights, since Hàn characters now have double weights.

However, there is also a need to further divide the level 1 weights due to that some scripts use grouping of characters for collation purposes. E.g. for Hangul, the grouping is in terms of consonant clusters, and vowel clusters (note: *not* in terms of Hangul syllables), and for several Brahmic derived scripts, the collation grouping is, it appears, in terms of orthographic syllables (it may be better analysed as grouped by consonants clusters and (singleton) vowel clusters). One way of achieving this is to insert, after each group, a light weight, either as part of the collation key construction, or have the prehandling insert ZWNBSP (say) characters after each group, and weight ZWNBSP at level 1 before all letters and digits (part of this will be maintained below).

Many of the ZWNBSPs insertions can be avoided, however, (except one per Hangul syllable, just after the lead consonants) if the weighting is done in a particular way, explained below.

Note that ZWNBSP should not occur in the original input, and if it does, the prehandling must replace it with WORD JOINER. So the only ZWNBSPs that are to be weighted are those that mark the end of a collation cluster, and in addition could not be avoided by utilising the particular weighting layout for level 1 weights explained below.

2 Weighting scheme

Nominally, a collation key consists of a number of subkeys, one for each level in the collation table, normally 3 or 4. (APIs can be used to look only at fewer levels.) Each subkey is made up of a number of weights, and a weight is a positive integer. Nominally the subkeys are independent: the level 2 subkeys are compared only if the level 1 subkeys are equal, etc. In practice, however, one concatenates the subkeys into a single key. However, the comparison must still work *as if* the subkeys were independent. This is nominally done by inserting a very small weight, usually 0 or 1, that is never used as a proper weight, between the subkeys:

$k = s_1, 0, s_2, 0, ..., 0, s_n$

where s_i is the subkey at level i, and commas mark concatenation of sequences of weights.

2.1 Key terminator weight

0 (or nothing). The 0 is only needed between keys if several keys are concatenated, to form a multi-string collation key. This is not part of UCA not 14651, but may be used for proper word-by-word (or other grouping) ordering.

2.2 Subkey terminator weight

1 (or 0). These can in most cases be avoided easily, see below. However, in order to be able to reuse weight values at the fourth level, these are used after the third level weights when using the fourth level.

2.3 Third level weights

In order to avoid having to have a subkey terminator weight after the second level weights in a collation key, all the third level weights are lighter (come before) the second level weights. Third level weights are usually used for case and similar differences.

2.4 Second level weights

In order to avoid having to have a subkey terminator weight after the first level weights in a collation key, all the second level weights are lighter (come before) the first level weights. Second level weights are usually used for accent and similar differences.

2.5 First level weights

2.5.1 Delimiter, punctuation, and symbol weights

It is sometimes useful to consider a string divided into substrings in various ways when ordering; e.g. sentence-by-sentence or word-by-word. Instead of actually dividing up the string into several strings, that are given keys that are then concatenated (with a key terminator weight in-between each), an easier way is to just certain “part terminating” characters as having light primary weights. Indeed, the latter even allows for a hierarchy. It is quite possible to have a, say, syllable clustered ordering, and on top of that have a word-by-word ordering; and indeed on top of that have a sentence-by-sentence ordering.

Some scripts always use a clustered ordering. Hangul uses a clustered ordering where the clusters are sequences of consonants and sequences of vowels (*note that Hangul does **not** use orthographic syllable clustering in its ordering rules, despite popular claims to that effect*). The Brahmic derived scripts use a clustering in ordering that is based on the orthographic syllable clusters.

For the most part, we can weight the characters so that the appropriate ordering clustering falls out as a result. This is not fully the case for Hangul, however. Most of it can be achieved by assigning the collation weights in a good way. However, one **must** still insert a lightweight terminator weight (that for ZWNBSP below) after each Choseong cluster. None is needed after Jungseong or Jongseong clusters **if** the weights are assigned appropriately (see below).

Note that the clusters here are *not* directly related to the "grapheme clusters" elsewhere!

Here we list the collation clustering characters in order of increasing first level weight (when not IGNOREd).

2.5.1.1 Top level termination characters

These punctuation characters should be IGNOREd by default, but have smallest first level weights in case they are not to be IGNOREd.

The characters here are FS/IS4, as well as "end of documents" visible punctuation: 17DA (khmer sign koomuut), 0E5B (thai character khomut).

2.5.1.2 Paragraph termination characters

These punctuation characters should be IGNOREd by default, but have first level weights in case they are not to be IGNOREd.

The characters here are GS/IS3, as well as "end of paragraph" invisible and visible punctuation: PS, CR, LF, NEL, <CR, LF>.

2.5.1.3 Sentence termination characters

These punctuation characters should be IGNOREd by default, but have first level weights in case they are not to be IGNOREd. The characters here are RS/IS2, as well as "end of (partial) sentence" invisible and visible punctuation: LS, HT, VT, FF, <CR, VT>, <CR, FF>, FULL STOP, QUESTION MARK, EXCLAMATION MARK, COMMA, SEMICOLON (and their variants), possibly also EM DASH. (Ethiopic?)

2.5.1.4 Word termination characters

These punctuation characters should be IGNOREd by default, but have first level weights in case they are not to be IGNOREd. The characters here are US/IS1, as well as "end of (partial) word": any Zs, BREAK PERMITTED HERE. (Ethiopic?) (currency signs?)

2.5.1.5 Subword termination characters

These punctuation characters should be IGNOREd by default, but have first level weights in case they are not to be IGNOREd. The characters here are: SHY, MTSHY, WORD JOINER.

2.5.1.6 Syllable or subsyllable termination character

The character here, ZWNBSP, should **not** be ignored by default. This is a character that does not occur in input (if it does such occurrences are replaced by WORD JOINER). Thus all occurrences in weighted input is the result of prehandling inserting ZWNBSP after (sub)syllables in (sub)syllable clustering ordered scripts (Brahmic derived scripts and Hangul). Given the special weighting introduced here, the only (sub)syllable position where they still need to be inserted is after each maximal sequence of Hangul lead consonant jamos (handling Hangul compatibility letters is optional, and not part of the required prehandling, just the optional prehandling).

The weight for ZWNBSP has sometimes been referred to as the <LOW> weight. However, there are even lower weights, as listed above.

2.5.2 Symbols

Next come symbols, IGNOREd on levels 1-3 by default, but they have first level weights (in UCA) in case they are set (via an API) not to be ignored. That is except MINUS SIGN, PLUS SIGN, and INFINITY, which are given weights for numerals at the first level; see below.

2.5.3 Integral numeral weights

The weighting scheme per se only handles digits, but can also handle negative and positive infinity. For accurate ordering of natural, integral, or more general numerals, special prehandling is needed. See section C.3 of ISO/IEC 14651 (this is not covered in UTS 10).

Again we present them in order of increasing weighting. However, these are not IGNOREd by default, but have their first level weights always used.

2.5.3.1 Negative infinity

First among the numerals (most negative) is negative infinity. For this, have a contraction for <MINUS SIGN, INFINITY>.

2.5.3.2 Negative values

MINUS SIGN should not be IGNOREd by default. However, proper handling of numerals for negative values requires one of a number of particular prehandlings to be done (see section C.3 of ISO/IEC 14651). The latter is not part of UTS 10 (nor 14651), but handling of integral numerals is recommended.

2.5.3.3 Positive values

At this weighting group the digits (and the PLUS SIGN, if not IGNOREd) are weighted. Again, proper handling of numerals requires one of a number of particular prehandlings to be done (see section C.3 of ISO/IEC 14651).

2.5.3.4 Positive infinity

Last among the numerals is positive infinity. Transfinite numerals (ordinal or cardinal) should not be handled by default.

2.5.4 Script weights

The script weights (for letters/ideographs of scripts, not numbers or punctuation, which already has been dealt with) should be split into several groups. This is partially done already: the Hàn trail weights are in a separate group after all the "normal" letters. There is also an incomplete proposal to handle Hangul a bit better (L2/02-010) as well as a more complete proposal, that however only handles the "split" in commentary after the table (L2/01-469). The split for Hangul is given in this paper (without complete table though). Hangul is collation clustered by consonant clusters and vowel clusters. Also the Brahmic derived scripts can be seen as collation clustered according to consonants cluster and (single) vowel cluster).

A way of handling this clustering is to insert a ZWNBSP between each collation cluster. That leads to an excessive number of ZWNBSPs to insert. However, by assigning the weights in a good way, the ZWNBSPs can be reduced to need to be inserted only after Hangul lead consonant jamo clusters. The others aren't needed, provided that the weighting is done as described in this paper.

2.5.4.1 Independent letters grouped by script

First among the letter groups is the "ordinary" (independent) letter group. This is where letters are ordered at present in the UCA default table and the CTT. This group thus include many alphabetic scripts, like Latin, Greek, Cyrillic, but also most syllabic scripts, like Yi, Canadian Syllabics.

This group includes Hàn lead weights (Hàn characters are now doubly weighted), weights for Hangul lead consonants, and for independent letters in Brahmic derived scripts.

This group does not include Hangul vowels, Hangul trail consonants, Hàn trail weights, nor dependent vowels or dependent consonants (often introduced via a VIRAMA in the preceding combining sequence, but sometimes combining letters of their own). These are weighted after this group.

Note that for Hangul compatibility letters, they are weighted here regardless of their compatibility decomposition (which is erratic). Prehandling may *optionally* insert a CGJ between the Hangul compatibility vowel letters and the compatibility Hangul consonant letters that follow each cluster of such vowels. That will result in proper clustering for the Hangul compatibility letter sequences. However, compatibility Hangul letters are not supported by default in the UCA or 14651; they construction is too complex (see L2/01-469).

Note that prehandling must canonically decompose precomposed Hangul Syllables, and must insert a ZWNBSP after each Choseong cluster, and should ideally do so also after a sequence of (truly) leading compatibility Hangul consonants.

2.5.4.2 Dependent vowel letters grouped by script, except Hangul vowels

Nominally, to get the proper low level clustering, subsyllable (Hangul) or syllable (Brahmic) collation clusters at the first level, one should insert a (sub)syllable terminator character (ZWNBSP) between each low level collation cluster. However, that leads to an excessive amount of ZWNBSPs inserted, as well as corresponding weights inserted into the resulting collation key. One theoretical possibility is to have contractions for all possible collation clusters. However, that would be unwieldy due to the number of possible collation clusters. To achieve this clustering without lots of

ZWNBSP, there is no need to have a contraction for each possible cluster. Instead we nominally use ZWNBSP as low-level cluster separator, but most of the ZWNBSP can be avoided if characters that lead collation clusters are given lower weights that those that can only come inside a collation cluster. Ordinary letters are order before the following ones for just that reason. The ordinary (independent) letters lead collation clusters.

This is the first group of letters-post-ordinary-letters: dependent Brahmic-derived vowels (except for Thai and Lao, these are combining characters). These dependent letters occur only towards the end of a collation cluster for these scripts, always attached to a (sequence of) consonant(s). The dependent vowels for these scripts seem to be consistently collation ordered before dependent consonants (joined into the collation cluster via a VIRAMA character, or being subjoined consonants).

KHMER SIGN VIRIAM, TIBETAN HALANT, THAI PHINTHU are silent dependent vowels, not viramas. As vowels they should be order among the dependent vowels of the respective scripts.

2.5.4.3 Letter links (mostly used to make independent consonant dependent)

Letter links are the VIRAMAs (by various names) that make a letter combining sequence conjoining with the following letter combining sequence. This is used to adjoin further independent consonants (or, in some cases, independent vowels) to an orthographic syllable. The scripts that use this mechanism appear to consistently order adjoined consonants after dependent vowels. One can see this as that the collation clustering is really subsyllable: a consonants cluster is a collation cluster, and the dependent vowel is a collation cluster of its own. However, with the weighting given here, no ZWNBSP need be inserted between the consonant sequence and the vowel.

Note that also COMBINING GRAPHEME JOINER should be weighted here, in the default weighting table. That way a CGJ-joined letter pair (e.g. <c, CGJ, h>) automatically by default come after, in the collation order, the first letter of the pair, also when further letters follow in the strings that are collated: e.g.

c, ch, cz, <c,CGJ,h>, d, ...

2.5.4.4 Dependent consonant letters grouped by script

Some dependent consonants have characters of their own, rather than using some kind of VIRAMA. For instance Tibetan subjoined consonants. As already mentioned, these appear to come after, in the collation order, the dependent vowels (judging from the CTT).

Here we put also Hangul trailing consonant jamos (though they could alternatively have been put in any of the two preceding groups).

2.5.4.5 Hangul vowel letters

Hangul is collated by clusters of consonants and clusters of vowels. Note that it is *not* collated by syllable clusters (i.e. the collation clustering is *not* grapheme clustering). Historically this clustering has been achieved by listing, and making not just contractions for these, but indeed making such clusters into characters, selecting just those thought to be used. *However, those lists have been incomplete, and, more importantly, Hangul is an alphabetic script and sequences of single letters (not cluster characters) are not only sufficient but indeed more general.*

Inserting ZWNBSPs after each collation cluster for Hangul, then results in a structure like (ignoring ill-formed Hangul syllable jamo sequences):

...<ZWNBSP>LLL<ZWNBSP>VV<ZWNBSP>TT<ZWNBSP>L...

By weighting all 'leading' letters (Hangul lead jamos, as well as any other letters) before the Hangul vowels and the Hangul trail letters, we need fewer ZWNBSPs:

...LLL<ZWNBSP>VV<ZWNBSP>TTL...

Then by weighting the Hangul trail consonants before(!) the Hangul vowels, as recommended here, we can reduce the number of ZWNBSPs further:

...LLL<ZWNBSP>VVTTL...

However, there is still a need for a ZWNBSP after each sequence of Hangul lead consonants. To remove that, the Hangul vowels would need to be ordered before the Hangul lead consonants, and that is prevented by that they have to be (for ZWNBSP reduction, as noted here) ordered after the trail consonants, which are after the 'lead' letters of any script.

Furthermore, the cluster jamo characters should be weighted according to the single letters they "consist" of, and not be weighted separately.

2.5.5 Han extension weights

Finally, we have the Hàn extension weights. They could alternatively be among any of the four preceding groups, since they can occur only after Hàn lead weights.

2.6 Fourth level weights

A subkey separator weight is used between the third and fourth level subkeys when constructing the full key, so the fourth level weights can be assigned independently of the other level weights. However, the special weight <PLAIN> must be last among the fourth level weights. It is used in special ways by the key construction where it is assumed to be the largest fourth level weight.

3 Conclusions

Some changes are needed to the UCA required prehandling, as well as to the UCA/14651 default weighting tables to get good default behaviour for Hangul, Brahmic derived scripts, for the CGJ, and to be able to get hierarchical collation clustering.

In addition to NFD and Logical_Order_Exception rearrangement, the required part (in UTS 10) of the prehandling should also do the following:

1. Replace all occurrences of ZWNBSP in the collation input with WORD JOINER (BOM use of ZWNBSP is not expected for collation input).
2. Algorithmic decomposition of Hangul precomposed syllables (part of NFD formation).
3. Insert a ZWNBSP after each sequence of Hangul lead consonant jamos. This is needed to get the proper collation clustering for Hangul.
4. Hangul jamos should be weighted according to their letter content (see *L2/01-469*).

In summary, the weight assignment should be done as follows:

third level weights	ok as is
second level weights	ok as is
first level weights for cluster terminators in a hierarchy, most of these are normally ignored on levels 1-3; except that ZWNBSP must not be ignored (needed for Hangul)	fix needed for ZWNBSP (and preferably also for hierarchical clustering)
first level weights for digits and other numeral related characters	fix preferred for MINUS SIGN and INFINITY
first level weights for various alphabets	weights for some characters need to be made heavier as noted here (partial proposal put forward by Davis L2/02-010; see this paper for a more complete proposal)
dependent vowels in Brahmic derived scripts (including non-link viramas—silent dependent vowels)	needs to be fixed
letter link characters (many (not all) viramas and CGJ)	needs to be fixed
dependent consonants in Brahmic derived scripts as well as Hangul trail consonants	needs to be fixed
Hangul vowels	(partial proposal put forward by Davis; but the vowels are misweighted, and the insertion of ZWNBSPs are missing)
Hàn continuation weights (could be anywhere in this last group)	already fixed

Finally, the Hangul jamos that encode clusters of letters should be weighted according to their single letter content as detailed in L2/01-469.

4 References

ISO/IEC 10646-1:2000 Information Technology – Universal multiple-octet coded character set (UCS), Part 1, second edition.

Unicode 3.0 The Unicode standard, version 3.0.

UCD 3.2 Unicode character database, version 3.2.

ISO/IEC 14651:2001	International string ordering and comparison – Method for comparing character strings and description of the common template tailorable ordering.
UTS 10	Unicode technical standard 10, Unicode collation algorithm.
ISO/IEC JTC 1/SC22/WG20 N891R	Kent Karlsson, Hangul ordering rules. 2001-11-29.
ISO/IEC JTC 1/SC2/WG2 N2405R	Same as ISO/IEC JTC 1/SC22/WG20 N891R.
L2/01-469	Same as ISO/IEC JTC 1/SC22/WG20 N891R.
ISO/IEC JTC 1/SC22/WG20 N896	Kent Karlsson, Khmer ordering rules. 2001-12-19.
L2/01-476	Same as ISO/IEC JTC 1/SC22/WG20 N896.
	Segmental collation and the UCA. Mark Davis. "Very draft", 2001-11-16.
L2/02-010	UCA Hangul Problem.

--- end ---