To compose, or not to compose?

The Ideographic Description Character (IDC) sequence dual encoding problem—and a proposal for how to solve it.

Fredrick R. Brennan

孟福黎
Psiĥedelisto
フレッド・ブレンナン
copypaste@kittens.ph

15 August 2020

This document was typeset with SILE.

Acknowledgements

Ken Lunde (小林 劍 こばやし けん), Unicode Consortium

For his font development advice, and helpful advice regarding Unihan.

Deborah Anderson, University of California @ Berkeley Script Encoding Initiative

For her tireless review of script proposals by n00bs like me.

Peter Constable, Unicode Consortium

For having an interesting discussion with me about the problem of IDC shaping in general.

Table of Contents

Chapter 1
Quick summary

In short, I want U+200C, the Zero Width Non Joiner, to be allowed in Ideographic Description Character sequences.

If this proposal is accepted, the following change to the Ideographic Description Character sequence grammar in §18.2 of *The Unicode Standard* will be made:

1.1 Grammar (p. 733)

```
IDS := Ideographic | Radical | CJK_Stroke | Private Use | U+FF1F
     | IDS_BinaryOperator  Optional_NonJoiner IDS IDS
     | IDS_TrinaryOperator Optional_NonJoiner IDS IDS IDS
Optional_NonJoiner := ZWNJ | ""
ZWNJ := U+200C
CJK_Stroke := U+31C0 | U+31C1 | ... | U+31E3
IDS_BinaryOperator := U+2FF0 | U+2FF1 | U+2FF4 | U+2FF5 | U+2FF6 | U+2FF7 | U+2FF8 |
U+2FF9 | U+2FFA | U+2FFB
IDS_TrinaryOperator:= U+2FF2 | U+2FF3
```

1.2 Rendering (p. 735)

Append to the end:

> “An implementation must not render an Ideographic Description Sequence that contains any U+200C, Zero Width Non Joiner. Until version 14.0, U+200C was not part of a valid Ideographic Description Sequence; now that it is, document authors are recommended to use it if the sequence is not supposed to be composed, to prevent dual encoding.”

1.3 Equivalence (p. 734)

Append to the end:

> “No amibiguity is introduced: an IDC with Zero Width Joiners is absolutely equal to one without for comparison purposes. Zero Width Joiners should never be used in `USourceData.txt`, and for most database equality purposes should be stripped out; they only matter in terms of text shaping.”

Chapter 2
Rationale

2.1 IDC sequences and dual encoding

We can see from an October 22, 1998 proposal, SC2 N3186, from Japan's liaison to SC2, "The IDS describes the ideograph in the abstract form. It is not interpreted as a composed character and does not have any rendering implication." However, this has changed. Chapter 18, Section 2, page 735 of The Unicode Standard v13.0 reads in part: "An implementation may render a valid Ideographic Description Sequence either by rendering the individual characters separately *or by parsing the Ideographic Description Sequence and drawing the ideograph so described.*" [emphasis mine] There was a lot of precedent, for this, though. Indeed, John Jenkins was already doing it in 1999.[1] The Unicode Standard accepted this as valid as early as v3.0 of the standard, also released in 1999.[2] Perhaps the best known use of IDC shaping is of the character *biáng*, now at U+030EDE, 𰻞—the esteemed Dr. Ken Lunde produced a font which contains only this glyph for the benefit of users way back in 2014.[3]

This, of course, leads to dual encoding: that is to say, The Unicode Standard allows the same sequence of glyphs to appear two radically different ways in plaintext, and there's no way to differentiate between them. This has so far not been much of a problem; so much so that no one has complained in twenty years. IDC sequences are almost always in practice displayed uncomposed, that is to say, the IDC sequence for the name of my pet Pomeranian, Hitomi,[4] ⿰目童⿰羊大, will not be composed to 瞳美. In the rare cases where IDC sequences are currently composed, it is always because a document author has specifically chosen to include a font which will compose only a certain few sequences the author knows about, and which they can turn on and off.

However, there are good reasons to believe that this will soon change. In one regard it's already changed: the wildly popular Source Han fonts, also by Dr. Lunde, compose the following four IDC sequences by default:[5]

㵐 ⇐ ⿰氵 恩
𰻞 ⇐ ⿺辶⿱穴⿰月⿰⿰⿰幺长⿰言马⿰幺长 刂心
𰻞 ⇐ ⿺辶⿱穴⿰月⿰⿰⿰幺長⿰言馬⿰幺長 刂心
龘 ⇐ ⿱雲⿰雲龍雲⿰龍龍

Given that Source Han is an extremely high quality open source font, it's already shown up in many open source operating systems. Its documentation is essentially a to-do list for any company producing a similar font, and Dr. Lunde is recognized widely as an expert on CJK fonts, so its influence is not going away any time soon. Indeed, the Noto CJK project also shapes all four sequences in its Sans font and the IDC sequence for the 𰻞's in its serif: I am using it in this very paper.

1. John H. Jenkins (1999). "New Ideographs in Unicode 3.0 and Beyond". San Jose, CA: 15th International Unicode Conference. pp. 12–21.
2. The Unicode Standard, version 3.0. p. 270–271.
3. Lunde, Ken (2014). "IDS + OpenType: Pseudo-encoding Unencoded Glyphs". Adobe CJK Type Blog.
4. Hitomi likes ideographs.
5. Lunde, Ken (2019). ReadMe for Source Han Sans Version 2.001. pp. 13–14.

One can say, "well simply disable `ccmp`". And indeed that's what I've done in this paper to typeset the above examples. But Unicode is for plaintext, and there's no way to disable `ccmp` in a tweet, certainly not one that's already sent.

There is a second reason to expect change in the area of IDC sequences: The Unicode Standard allows implementors to shape arbitrary IDC sequences, and I am very interested in doing so. I recently wrote an entire paper about it,[6] and plan to do whatever I can to further the state of the art on this matter. As I demonstrate in that paper, it is already simple to shape arbitrary IDC sequences up to a certain recursion depth allowing for some ugliness of glyphs. In the paper I mainly talk about ways we can make the glyphs nicer and increase the recursion depth, but it is indeed already possible to shape the majority of IDC sequences with already existing standards and technology without breaking any of the standards. Stroke fonts in OpenType (with font-specifiable caps) and ways to specify regular expressions defining runs that can (or cannot, as the case may be) would be nice, but are not imperative for shaping IDC sequences.

Why would one want to shape arbitrary IDC sequences? I see two main use cases: first, to use popular characters which are currently somewhere along the winding path between proposal to encoding, such as 鱻, which took five years to be encoded. Second, to support neologisms, which no one knows whether or not they will become popular enough to be encoded or not. In the past, it was common for new Chinese (and Japanese) characters to be invented, but now, old characters are simply given new meanings or new compounds are made; to create a neologism and have it stick is quite impossible in the digital age, as the current system places too many barriers in the way of users. I went into more detail on Quora in 2018, which is when I began considering this issue.[7]

I don't mean to say that we should stop encoding new blocks of CJK characters. Not at all. And, indeed, no matter how much I work to tweak IDC shaping, there will always be edge cases that are not well handled, and a font with a glyph specifically drawn for a given character will look better than a computer's attempt to shape it. Shaped IDC's will always be below encoded characters.

In any event, it is possible right now to shape IDC's, and with the wide dissemination as of this writing of HarfBuzz, I'd say inevitable. Someone in China is probably working on it right now; there may indeed already be a font out there which can shape IDC sequences for all the Kanxi radicals to a recursion depth of 2, and my inability to speak Chinese well (I speak Japanese only marginally better) means I cannot find it. ☺

And this is only considering OpenType; if we throw LaTeX into the mix, there is a long history of Chinese character description languages and LaTeX;[8] Certainly, LaTeX can shape arbitrary IDC sequences. Graphite fonts may be able to do so as well.

6. Brennan, Fredrick (2020). "Shaping Ideographic Description Sequences". Text Shaping Working Group (TSWG).

7. Brennan, Fredrick (2018). Answer to "What are the disadvantages of typography?". Quora. In 2019 I solved the problem I dubbed *Uniformity* to my satisfaction with the release of TT2020 (I also consider the excellent work many have done on adopting the OpenType `rand` feature part of the answer), now I am to solve the one I dubbed *Stagnation*.

8. Wong, Wai, et al. (2003). "Typesetting Rare Chinese Characters in LaTeX". *TUGboat*, Volume 24 (2003), No. 3