

Universal Multiple-Octet Coded Character Set
UCS

ISO/IEC JTC1/SC2/WG2 IRG N1308

Date: 2007-05-29

Source:	John Knightley
Title:	How to extend Annex S
Status:	For discussion
Distribution:	IRG Members and Ideographic Experts
Medium:	Electronic

Introduction:—

In deciding whether or not two characters should be unified the two most common challenges are (1) to reach agreement on whether or not the two characters have the same abstract shape and (2) to get sufficient information to decide whether or not the characters are cognate. In this document determining abstract shape is discussed.

Annex S states that if a pair of CJKV variants are cognate and have the same abstract shape then the pair should be unified, otherwise they should not be unified. The evidence rule also applies, only real characters are encoded, theoretical variants with no evidence are not encoded. This is one reason there exist both encoded traditional characters but the corresponding simplified variant is not encoded, and encoded simplified variants but the corresponding traditional variant is not encoded.

One purpose of Annex S is to give stability to the unification process, the unification rules for Extensions A, B, and C are the same, apart from the source separation rule removed in 1992 . Though by mistake some duplicate characters have been encoded, in general the unification rules have been applied well. However though the principles of Annex S should not be changed, the examples need to be extended. All the examples in Annex S come from the CJK Unified Ideographs range but it is however necessary to be able to correctly apply the principles of Annex S to new characters whether or not there are similar examples shown in Annex S.

Comparing Abstract Shapes

Annex S, S.1.3 describes how to decide whether or not two characters have the same abstract shape and can be summarized thus:-

“If it is possible to divide two characters into one or more components so that the corresponding components have the same abstract shape then the two characters have the same abstract shape, otherwise the two characters have a different abstract shape.”

We could take 搵 and 搵 as an example.

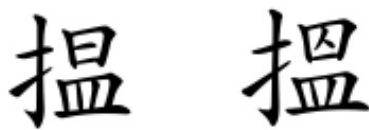


figure 1 : one component view



figure 2 : two component view



figure 3 : three component view

It would be wrong to say that if 日 does not have the same abstract shape as 囧, then 搵 and 搵 could not have the same abstract shape. It would also be wrong to say that because 搵 and 搵 have the same abstract shape, therefore 日 must have the same abstract shape as 囧. However, it would also be wrong because of difficulties like these to conclude that there is no correct way to make inferences about the abstract shape of characters not given in the lists in Annex S.

Combining Known Abstract Shapes

It is well known that if one combines components that are equivalent abstract shapes then the results have the same abstract shape. So, for example, since Annex S states clearly that 𠄎 and 𠄎 have the same abstract shape then it was easy for every one to agree that the two source glyphs of U+8158 (see figure 4) have the same abstract shape and should be unified.



figure 4: ISO_2003_charts_6810_824F_CJKUnifiedMain2.pdf page 135

What about combining a component with a pair of components known to have a different abstract shape? Whilst this can lead to new characters with the same abstract shape, this is the exception that proves the rule, the usual result is in fact a pair of new characters that have different abstract shapes. Annex S illustrates this in several ways, two of which are:-

1. S.1.4.3 gives U+7635 and U+7636 as an example of a character pair with different abstract shapes. At the end of Annex S, where examples are given of pairs not covered by the separation rule, this pair, and two new pairs 𠄎/𠄎 and 𠄎/𠄎 are shown as not having the same abstract shape because of S.1.4.3. These two new pairs are both examples of adding a component to a pair of characters with different abstract shapes, here U+7635 and U+7636, resulting in a pair of new characters that also are of different abstract shape. (see figure 5 below)

In accordance with the unification procedures described in clause S.1 of this annex the pairs (or triplets) of ideographs shown below are not unified. The reason for non-unification is indicated by the reference which appears to

the right of each pair (or triplet). For “non-cognate” see clause S.1.1

NOTE – The reason for non-unification in these examples is different from the source separation rule described in clause S.1.6.

胃胃 5191 80C4	non cognate	寶寶 5BF3 5BF6	S.1.4.3	胸胸 6710 80CA	non cognate	稻稻 7A32 7A3B	S.1.4.3
冲冲 51B2 6C96	S.1.4.3	廳廳 5EF0 5EF3	S.1.4.1	眺眺 6713 8101	non cognate	翱翱 7FF1 7FF6	S.1.4.3
决决 51B3 6C7A	S.1.4.3	懷懷 61D0 61F7	S.1.4.1	腩腩 6718 8127	non cognate	考考考 8007 8008 8009	S.1.4.3
况况 51B5 6CC1	S.1.4.3	戮戮 6560 656A	S.1.4.3	瞳瞳 6723 81A7	non cognate	聽聽聽 8074 807C 807D	S.1.4.1
塚塚 579B 579C	S.1.4.3	盼盼 670C 80A6	non cognate	朶朶 6735 6736	S.1.4.3	荊荊 8346 834A	S.1.4.2
孽孽 5B7C 5B7D	S.1.4.2	臄臄 670F 80D0	non cognate	灑灑 7054 7067	S.1.4.3	躲躲 8EB1 8EB2	S.1.4.3

figure 5 : the last section of Annex S

2. The last 3 examples in S.1.4.3 have dotted boxes to show that if another component is placed in that location the result is a pair of characters that have different abstract shapes.

S.1.4.3 Different structure of a corresponding component

The examples below illustrate rule c). The structure of one (or more) corresponding components within the two ideographs in each pair is different.

扌·擴	策·箒	灬·然	圣·聖
兪·僉	区·區	夾·夾	单·單
萑·藿	彘·彘	贊·贊	襄·襄
隹·隹	問·問	朶·朶	雋·雋
恒·恆	奂·奂	人·人	采·采
友·友			

figure 6 : 3 pairs, abstract shape different when combined

Decomposing Known Abstract Shapes

As with combining known abstract shapes, when decomposing known abstract shapes there is a case in which the results are 100% certain and also a case where the results are more relative to context.

The absolute case is that given a pair of characters with different abstract shape then if they can be decomposed in the same way at least one pair of corresponding components must have differing abstract shapes. For example Annex S tells us 間 and 閒 have differing abstract shapes. 間 can be decomposed into 門 and 日, and 閒 can be decomposed into 門 and 月. Since the 門 of 間 and the 門 of 閒 have the same actual shape then they also have the same abstract shape, therefore it follows logically that 日 and 月 do not have the same abstract shape.

The case that needs to be decided by context is that of decomposing character pairs that have the same abstract shape. If one of the basic components that defines the character is decomposed then the resulting corresponding pairs may well not all be pairs of components with the same abstract shape. If however no basic components are decomposed then the result is corresponding pairs that have the same abstract shape. Most CJKV characters consist of a phonetic component that indicates the sound and an ideographic component that indicates the meaning, when these two components are non-overlapping, then decomposing into a phonetic component and an ideographic component means that the resulting pairs are pairs that have the same abstract shape. An example would be if someone said since 侶 and 侶 have the same abstract shape as each other, then their phonetic components 侶 and 侶 also have the same abstract shape as each other. The last section of Annex S (see figure 5) that shows results deduced from the earlier sections, but this does not include any such examples.

To find more convincing examples of this we must look further afield. Annex S indicates U+6C5A 汚 and U+6C61 汚 have the same abstract shape, and Mr Taichai Kawabata suggests 𠂇 and 𠂇 have the same abstract shape in housetsu.html (item 62-3) and IRGN1154 (page 6) and in the former among other supporting examples are 7 compatibility ideographs.

2F840	𠂇	CJK COMPATIBILITY IDEOGRAPH-2F840	≡ 54A2 𠂇
2F86C	𠂇	CJK COMPATIBILITY IDEOGRAPH-2F86C	≡ 219C8 𠂇
2F8B4	𠂇	CJK COMPATIBILITY IDEOGRAPH-2F8B4	≡ 625D 𠂇
2F93D	𠂇	CJK COMPATIBILITY IDEOGRAPH-2F93D	≡ 25044 𠂇
2F941	𠂇	CJK COMPATIBILITY IDEOGRAPH-2F941	≡ 250F3 𠂇
2F9B5	𠂇	CJK COMPATIBILITY IDEOGRAPH-2F9B5	≡ 8667 𠂇
2F9CC	𠂇	CJK COMPATIBILITY IDEOGRAPH-2F9CC	≡ 27966 𠂇

figure 7 : from <<http://www.unicode.org/charts/PDF/U2F800.pdf>>

It should be noted that U+4E90 𠂇 and U+4E8F 𠂇 are non-cognate, and therefore could not be included in S.3 Source code separation examples. The fact that they are non-cognate does not mean however that they have different abstract shapes. It is suggested that U+4E90 𠂇 and U+4E8F 𠂇 should be included in an official list of pairs that have the same abstract shape.

There are other such pairs, however it should be noted that they need to be considered case by case.

Dealing with components not mentioned in Annex S

When comparing cognates, or variants, to see if they have the same abstract shape sooner or later it is necessary to deal with new basic components or new component variants not mentioned in Annex S, and that can not be decided by combining or decomposing known components. There are three main possibilities:-

1. The difference in actual shape is so big that the two components obviously have different abstract shape, as described in S.1.4
2. The abstract shape is obviously the same based upon S.1.5 as in figure 8 below



figure 8 : same abstract shape, from page 23 of

<<http://www.unicode.org/ivd/pri/pri98/pri98-partialcharts.pdf>>

3. They need to be decided upon by considering in general principles of what abstract shape equivalence means. Here it should be noted that abstract shape equivalence is different for different systems, and that these should reflect the unification policy of the IRG. An example pair is first worked through below to show how this might be done.

A Worked Example

Annex S only gives examples from the CJK Unified Ideographs range. What then are the expected features of two characters not covered by the source separation rule that have the same abstract shape :-

- (1) All CJK Unified Ideographs range pairs would be non-cognate
- (2) There should be many characters encoded using the standard form but only a few encoded using the variant form.
- (3) Most pairs differing by only these two characters as components should be cognate or at least variants.

U+4EB0 京 and U+4EAC 京 are an example of a pair of variants which seem to have the same abstract shape but were not included in Annex S because they are not covered by the source separation rule, and it is suggested that they should be included in an official list of pairs that have the same abstract shape.

(1) The pair 京 and 京 have the same reading and meaning in modern contexts, but are considered non-cognate because 京 is an ancient variant of 原. This is the only pair containing 京 and 京 where both are in the CJK Unified Ideographs range.

(2) Whilst 京 is used as a component in 124 encoded characters, 京 is only used as a component of 13 characters and all of these have corresponding encoded characters that have 京 as a component.

(3) 7 of the 13 pairs are well documented cognates, 2 further pairs are known variants

U+4EB0 京	U+4EAC 京	variants	
U+3B0C 景	U+666F 景	non-cognate	
U+416B 𣎵	U+7A24 𣎵		𣎵 is K3-2D59 𣎵 variant of 𣎵
U+2087D 勅	U+52CD 勅	cognate	勅與勅同。
U+22C4A 𣎵	U+63A0 𣎵	variants	
U+23136 𣎵	U+23135 𣎵	cognate	𣎵亦書作𣎵。 詳𣎵字註。
U+23137 𣎵	U+23134 𣎵	cognate	𣎵或作𣎵。 詳𣎵字註。
U+24692 𣎵	U+3E41 𣎵	cognate	𣎵同𣎵。
U+25218 𣎵	U+4041 𣎵	cognate	𣎵同𣎵。
U+260A0 𣎵	U+7DA1 𣎵	non-cognate	
U+2791A 鯨	U+27900 鯨	cognate	鯨同鯨。
U+28361 鯨	U+8F2C 鯨		
U+2A2D4 麋	U+9E96 麋	cognate	麋同麋。

Note #1

Though 𣎵, 𣎵, and 𣎵 are all variants of 𣎵, according to S. 1. 4. 2, 'Different relative positions of components', 𣎵 has the same abstract shape as 𣎵, but their abstract shape is different to that of 𣎵 and 𣎵.

Note #2

Though U+7A24 𣎵 is a variant of U+63A0 𣎵 their left hand components 禾 #115 and 扌 #064 are two different radicals and so 𣎵 has a different abstract shape than 𣎵.

(See Appendix A for Morohashi index and other references)

Widening the net

Over many years the IRG has made decisions about whether or not hundreds of thousands of source glyphs should be unified, apart from Annex S these decisions are recorded in the source glyph charts, compatibility ideographs and various IRG documents. By the Annex S definition if two characters are cognate, not covered by the separation rule and correctly encoded separately then they have a different abstract shape. This reminds us that at the end of the day abstract shape equivalence of a pair of characters is determined by the conventions of the IRG. This is a good thing.

By agreed IRG convention over the years variants differing only by 魚 versus 鱼 are encoded separately, in Annex S language this means that they have different abstract shapes. This is the result of the IRG discussing the relationship between traditional and simplified characters and agreeing they should be encoded separately. Some people might wonder what implication this has for the abstract shape of other characters, what about 魚 vs 𩺰 variants, or even 馬 vs 𩺰? The answer is that cases like these must sooner or later be discussed by the IRG. Nothing in Annex S says that just because 魚 and 鱼 have different abstract shape therefore 馬 and 𩺰 must have different abstract shapes. It is suggested that these are basic components and that to decompose them further is incorrect. Each pair must be considered on its own merits. It is suggested that in IRG contexts whilst 魚 vs 鱼, and 馬 vs 𩺰 are pairs with differing abstract shapes, however 魚 vs 𩺰 and 馬 vs 𩺰 are pairs with equivalent abstract shapes.

In conclusion for the benefit of all concerned there is need at this time together to extend Annex S by producing both a longer list of basic components that have the same abstract shape and a longer list of pairs of basic components that have different abstract shapes. This conclusion is not a new one, IRGN956_Unification.pdf

starts with

"The IRG follows the unification rules given in ISO/IEC 10646, Annex S. The examples of the rules in Annex S are, however, typical examples, and not meant to be exhaustive. This means that there have been many cases where it wasn't clear whether two forms should be unified or not.

The Unification Ad Hoc group was requested to provide more detailed guidelines to the IRG on the issue of how to deal with doubtful cases.

The Ad Hoc group recommends that the IRG maintain a standing document listing the specific components which should be unified, and those which should not be unified. These lists should initially be populated with the examples from Annex S, and they should be updated as the IRG editors encounter more examples. As with Annex S, it should be understood that the examples in the standing document are not exhaustive."

Acknowledgments

The ideas contained in this document are those of the author, it should be noted that the acknowledgment, here or elsewhere, of the work or contribution of others should in no way be considered as saying that those individuals support the ideas contained in this document.

Thanks to James Kass for proof reading the many drafts of this document.

Thanks to Ken Lunde for researching the Morohashi in the preparation of "A Worked Example" .

Special thanks to Taichi Kawabata who for many years has in the spirit of open-source development made his IDS material freely available and without which this document would never have been written

Appendix One — References for “A Worked Example”

1.1 Morohashi References

Unicode	Morohashi (1994)	Vol–page	Morohashi index
U+3B0C	5–920		14072
U+3E41	7–656		20053
U+4041	8–228		23414
U+4EAC	1–545		299
U+4EB0	1–553		307
U+52CD	2–389		2370
U+63A0	5–282		12273
U+666F	5–893		13983
U+7DA1	8–1092		27545
U+8F2C	10–1040		38402
U+9E96	12–917		47661
U+2087D	2–399		2396
U+23134	5–714		13726
U+23135	5–714		13727
U+23136	5–714		13728
U+23137	5–714		13729
U+24692	7–658		20083
U+25218	8–243		23540
U+260A0	8–1138		27722
U+27900	Index–1067		49716
U+2791A	10–374		35134
U+2A2D4	12–920		47678

2.2 Unihan.txt for non Morohashi

U+416B	kIRGKangXi	0856.321
U+416B	kIRG_KSource	3–2D59
U+7A24	kIRGDaeJaweon	1281.101
U+7A24	kIRGHanyuDaZidian	42615.060

U+7A24 kIRGKangXi 0855.311
 U+7A24 kIRG_GSource 1-7D50
 U+7A24 kIRG_KPSource KP1-6233
 U+7A24 kIRG_KSource 2-4E64
 U+7A24 kKPS1 6233
 U+7A24 kKangXi 0855.311
 U+7A24 kMandarin LÜE4 LÜE3
 U+7A24 kMorohashi 99999
 U+7A24 kPseudoGB1 9348

 U+22C4A kIRGHanyuDaZidian 80025.140
 U+22C4A kIRGKangXi 0445.341
 U+22C4A kIRG_GSource HZ

 U+28361 kIRGHanyuDaZidian 53547.020
 U+28361 kIRGKangXi 1247.031
 U+28361 kIRG_GSource HZ

3.3 Hanyu DaZidian variant data

(Posted by Dr R. Cook on the Unihan mailing list)

[U+63a0][U+22c4a][U+5260][U+3a3c][U+7a24]

4.4 www.zdic.net data

U+4EB0 京 "【正字通】俗原字。◎按京字《字彙》不載，韻書無考。《正韻》十一先收原遼，亦闕京。《正字通》強增以爲京卽原字，不知京、京古通假不必別生枝節。詳前京字註。"

<<http://www.zdic.net/zd/zi/ZdicE4ZdicBAZdicB0.htm>>

U-0002087D 勅 "【字彙補】與勅同。"

<<http://www.zdic.net/zd/zi3/ZdicF0ZdicA0ZdicA1ZdicBD.htm>>

U-00023136 𣎵 "【篇海】𣎵亦書作𣎵。詳𣎵字註。"

<<http://www.zdic.net/zd/zi3/ZdicF0ZdicA3Zdic84ZdicB6.htm>>

U-00023137 鯨 "【篇海】鯨或作鯨。詳鯨字註。"

<<http://www.zdic.net/zd/zi3/ZdicF0ZdicA3Zdic84ZdicB7.htm>>

U-00024692 鯨 "【篇海】同鯨。"

<<http://www.zdic.net/zd/zi3/ZdicF0ZdicA4Zdic9AZdic92.htm>>

U-00025218 鯨 "【正字通】同鯨。"

<<http://www.zdic.net/zd/zi3/ZdicF0ZdicA5Zdic88Zdic98.htm>>

U-0002791A 鯨 "【篇海類編】同鯨。字書無鯨字，當是鯨字之譌。"

<<http://www.zdic.net/zd/zi3/ZdicF0ZdicA7ZdicA4Zdic9A.htm>>

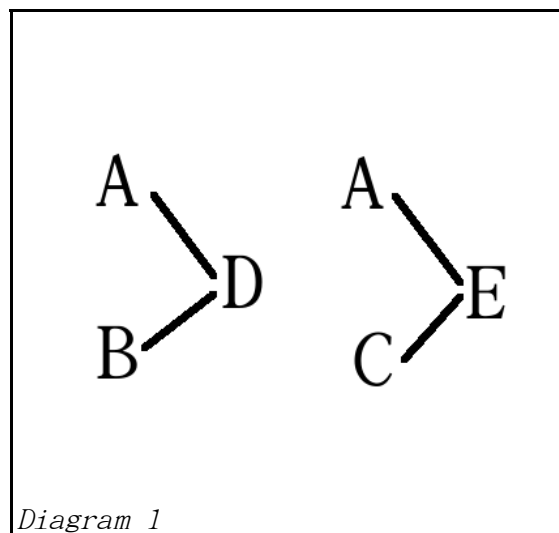
U-0002A2D4 麋 "【字彙補】同麋。"

<<http://www.zdic.net/zd/zi3/ZdicF0ZdicAAZdic8BZdic94.htm>>

Appendix Two — The main argument in a more abstract form

The component tree dilemma

If one uses the component tree model in S.l.2 out of context to decide whether or not two characters have the same abstract shape one runs into the following dilemma.



If in Diagram 1 B and C are corresponding components, that is they are in the same relative position, then if B and C have the same abstract shape then D has the same abstract shape as E. That is

(R1) if $B = C$ then $D = E$.

From this it also follows that if D and E have different abstract shapes then B does not have the same abstract shape as C. That is

(R2) if $D \neq E$ then $B \neq C$.

However if B does not have the same abstract shape as C then the relationship between D and E is left undefined

(R3) if $B \neq C$ then either $D = E$ or $D \neq E$

and also if D has the same abstract shape as E then the relationship between D and E is left undefined

(R4) if $D = E$ then either $B = C$ or $B \neq C$.

A possible solution might run as follows:-

Whilst the logic of “(R3) if $B \neq C$ then either $D = E$ or $D \neq E$ ” is correct however it should be noted that the implication of S.1.4.3 is that in general if B does not have the same abstract shape as C the result is that D and E do not have the same abstract shape, and that when D and E have the same abstract shape this is the exception that proves the rule.

To consider an example here would help. If for example A was \ddagger and there was no overlap between either A and B or between A and C then the conventions of CJKV characters dictate that if B and C have a different abstract shape then the resulting $\boxed{\ddagger} \ddagger B = D$ and $\boxed{\ddagger} \ddagger C = E$ also have different abstract shapes. In this case the argument would be made that because D and E obviously have at least to separate components, left hand side and right hand side, then the resulting D and E have the same abstract shape relationship as B and C – that is in this case if B and C are equivalent abstract shapes then so are D and E, and visa versa .

It is put forward that given B has a different abstract shape than C then the result D has the same abstract shape as E only occurs when D and E

themselves are 'basic components', as in if $D = \square \square \text{貝}$ and $E = \square \triangle \text{貝}$, and that clarifying the concept of a 'basic component' is one thing that needs to be done. This gives rise to

(R3a) if $(B \neq C)$ and $(D \text{ and } E \text{ are non-basic components})$ then $D \neq E$

By similar reasoning if D is not the same abstract shape as E , and D and E are compounds of two or more basic components then B is not the same abstract shape as C . That is

(R4a) if $(D \neq E)$ and $(D \text{ and } E \text{ are non-basic components})$ then $B \neq C$

R3a and R4a more accurately reflect the thinking of Annex S the R3 and R4.

In practice these rules can be summarized in a much more user friendly form:-

“If it is possible to divide two characters into one or more components so that the corresponding components have the same abstract shape then the two characters have the same abstract shape.”