

# Decomposition for ISO/IEC 10646 Ideographic Characters

LU Qin, CHAN Shiu Tong, LI Yin, LI Ngai Ling

Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Hong Kong

{csluqin, cstchan, csyinli, [csnlli](mailto:csnlli@comp.polyu.edu.hk)}@comp.polyu.edu.hk

## Abstract


Ideograph characters are often formed by some smaller functional units, which we call character components. These character components can be ideograph radicals, ideographs proper, or some pure components which must be used with others to form characters. Decomposition of ideographs can be used in many applications. It is particularly important in the study of Chinese character formation, phonetics and semantics. However, the way a character is decomposed depends on the definition of components as well as the decomposition rules. The 12 Ideographic Description Characters (IDCs) introduced in ISO 10646 are designed to describe characters using components. The Hong Kong SAR Government recently published two sets of glyph standards for ISO10646 characters. The standards, being the first of its kind, make use of character decomposition to specify a character glyph using its components. In this paper, we will first introduce the IDCs and how they can be used with components to describe two dimensional ideograph characters in a linear fashion. Next we will briefly discuss the basic references and character decomposition rules. We will then describe the data structure and algorithms to decompose Chinese characters into components and, vice versa. We have also implemented our database and algorithms as an internet application, called the *Chinese Character Search System*, available at website <http://www.iso10646hk.net/>. With this tool, people can easily search characters and components in ISO 10646.

## Introduction

ISO/IEC 10646 (ISO 10646) in its current version, contains more than 27,000 Han characters, or ideograph characters as it is called, in its basic multilingual plane and another 40,000 in the second plane[1-2]. The complete

set of ideograph repertoire includes Han characters in all national/regional standards as well as all characters from the Kang Xi Dictionary( 康熙字典 ) and other major references. In almost all the current encoding systems including ISO 10646 and Unicode, each Han character is treated as a separate unique symbol and given a separate code point. This single character encoding method has some serious drawbacks. Consider most of the alphabet-based languages, such as English, even though new words are created quite frequently, the alphabet itself is quite stable. Thus the newly adopted words do not have any impact on coding standards. When new Han characters are created, they must be assigned a new code point, thus all codesets supporting Han characters must leave space for extension. As there is no formal rule to limit the formation of new Han characters, the standardization process for code point assignment can be potentially endless. On the other hand, new Han characters are almost always be created using some existing character components which can be existing radicals, characters proper, or pure components which are not used alone as characters. If we can use coded components to describe a new character, we can potentially eliminate the standardization process.

Han characters can be considered as a two dimensional encoding of components. The same set of components when used in different relative positions can form different characters. For example the two components 大 and 小 can form two different characters: 尖 忝 depending on the relative positions of the two components. However, the current internal code point assignments in no way can reveal the relationship of the these characters with respect to their component characters. Because of the limitation of the encoding system, people have to put a lot of efforts to develop different input methods. Searching for characters with similar shapes are also quite difficult. The 12

Ideographic Description Characters (IDCs) were introduced in ISO 10646 in the code range of 2FF0 - 2FFB to describe the relative positions of components as shown in Table 1. Each IDC symbol shows a typical ideograph character composition structure. For example,  (U+2FF0) indicates that a character is formed by two components, one on the left-hand side and one on the right-hand side. All IDCs except U+2FF2 and U+2FF3 have cardinality of two because the decomposition requires two components only. Details of these symbols can be found in Annex F of ISO 10646 2nd Edition [1] and in John Jenkins' report [3].














Smb1	Code point	Name in ISO 10646	Cardinality	Label
	2FF0	IDC LEFT TO RIGHT	IDC2	A
	2FF1	IDC ABOVE TO BELOW	IDC2	B
	2FF2	IDC LEFT TO MIDDLE AND RIGHT	IDC3	K
	2FF3	IDC ABOVE TO MIDDLE AND BELOW	IDC3	L
	2FF4	IDC FULL SURROUND	IDC2	I
	2FF5	IDC SURROUND FROM ABOVE	IDC2	F
	2FF6	IDC SURROUND FROM BELOW	IDC2	G
	2FF7	IDC SURROUND FROM LEFT	IDC2	H
	2FF8	IDC SURROUND FROM UPPER LEFT	IDC2	D
	2FF9	IDC SURROUND FROM UPPER RIGHT	IDC2	C
	2FFA	IDC SURROUND FROM LOWER LEFT	IDC2	E
	2FFB	IDC OVERLAID	IDC2	J

Table 1. The 12 Ideograph Description Characters

The IDCs can be used to describe not only unencoded characters, but also coded characters to reveal their internal structures and relationships among components. Thus applications for using these structural symbols can be quite useful. In fact the most common applications are in electronic dictionaries and on-line education [4].

In this paper, however, we introduce a new

application where the IDCs and components are used in the standardization of Han character glyphs. As we all know that ISO 10646 is a character standard, which allows different glyph styles for the same character and different regions can develop different glyph styles to suit their own needs. The ideographic repertoire in ISO 10646 has a so called Horizontal Extension, where each coded ideograph character is listed under the respective CJKV columns. The glyph of each character can be different under different columns because ISO 10646 is a character standard, not a glyph standard. We normally call these different glyphs as variants. For example, the character bone  can take three different forms(variants):

		
HK	Mainland	Taiwan

Even with the ISO 10646 horizontal extensions, people in Hong Kong still get confused as to which styles to use, as only some characters in the Hong Kong style deviate from both G column(mainland China) and T column(Taiwan). Consequently, the Hong Kong SAR Government has decided to develop the Hong Kong glyph standards for ISO 10646 which can serve as a reference guide for font vendors when developing products for Hong Kong. The standards, being the first of its kind, makes uses of character decomposition to specify a character glyph using its components.

The rest of the paper is organized as follows. Section 1 gives the rationale for the use of character components, the references and decomposition rules. Section 2 describes the data structure and algorithms to decompose Chinese characters into components and, vice versa. Section 3 discusses performance considerations and Section 4 is the conclusion.

## 1. Character Decomposition Rules

At the beginning of the glyph standardization, one important requirement was agreed by the working group, namely, *extensibility*. That is, the specifications should be easily extended by adding more characters into later versions of the ISO/IEC 10646, which we refer to as the *new characters*. The specifications should also not contain any internal inconsistency, or inconsistency in relation to the ISO/IEC 10646's

source standards. In order to satisfy both consistency requirements, we have concluded that listing every character in ISO/IEC 10646 is not desirable. Instead, we decided to produce the specifications by giving the correct glyphs of character components based on a common assumption that if a component or a character is written in a certain way, all other characters using it as a component should also write it in the same way. For example if the character “bone” 骨 (U+9AA8) is written in a certain way, all characters using “bone” as a component, such as “滑” (U+6ED1) and “髂” (U+9ABC), should have the bone “骨” component follow the same style. In this way, the specification can be extended very easily for all new characters using bone “骨” as a component. In other words, we can assume that component glyphs are standardized for general usage. By using components to describe a character, we can also avoid inconsistency. That is, by avoid listing all characters with bone, “骨” as a component, we do not need to be concerned about producing inconsistent glyphs in the specifications. This is important because the working group does not have any font vendor as a member, because of an implicit rule that was specified by the Government of the HKSAR to avoid any potential conflict of interest. The glyph style is mostly based on the book published by the Hong Kong Institute of Education in 2000[5]

In principle, for producing glyph specifications, we have to produce a concrete, minimal, and unique list of basic components. In order to achieve this, we need to have a set of rules to decompose the characters systematically. In our work, we have used the GF 3001-1997 [6] as our major component reference. The following is a brief description of the rules. (For a detailed description, please refer to the paper “The Hong Kong Glyph Specifications for ISO 10646’s Ideographic Characters”[7].)

- Use GF 3001-1997 specifications as the basis to construct a set of primary components. Components for simplified Chinese are removed. The shapes are modified to match the glyph style for Hong Kong.
- Characters are decomposed into components according to their structure and etymological origin.
- In some cases, an “ad-hoc” decomposition occurs if the

etymological origin and its glyph shape are not consistent, or the etymological origin is not clear, or to avoid defining additional components.

- Characters are not decomposed if it appears in GF 3001-1997 as a component.
- Detached parts can be further decomposed.
- Merely touched parts that do not have overlapping or crossing can be decomposed.
- In some cases, we do not decompose some components to prevent the components from getting too small.
- In some cases, a single component will be distinguished as two different components. This is the concept of *variant* or *related* component.

This set of rules, together with 644 basic components and the set of intermediate components defined, enables us to decompose Chinese characters that appear in the first version ISO 10646 with 20,902 characters, Ext. A in the second version of ISO 10646[1] and Hong Kong Supplementary Character Set [8-9]. The 644 basic components play a very important role because they form all the Chinese characters in our scope.

In order to describe the position relationship amongst components in a character, we have used the 12 Ideographic Description Characters (IDC) in ISO/IEC 10646 Part1:2000 in the range from 2FF0 to 2FFB, and defined an extra IDC “M” (which indicates that a particular component is a basic component and will not be further decomposed), as shown in Table 1. Every character can be decomposed into up to three components depending on the cardinality of the IDC used.

Each *Character* is decomposed according to the following definition:

$$\text{Character} = \text{IDC2 } CC(1) \text{ } CC(2) \\ | \text{IDC3 } CC(1) \text{ } CC(2) \text{ } CC(3) \\ | M$$

where

$$\text{IDC2} \in (2\text{FF0} - 2\text{FFB})$$

$CC(i)$  is a set of character components and  $i$  indicates its position in the sequence

*M* is a special symbol indicating  
*Character* will not be further  
 decomposed

By our definition, a *CC* can be formed by three subsets: (1) coded radicals, (2) coded components and ideographs proper, and (3) intermediate components that are not coded in ISO 10646. The intermediate components are maintained by our system only. The decomposition result is stored in the database. Conceptually, every entry in the database can be treated as a Chinese component, having a data structure described above.

## 2. Decomposition/Formation Algorithms

As mentioned above, the decomposition database only gives information on how a character is decomposed in a minimal way. However, some characters have nested components. For instance, the character “準” can be decomposed into two components: “淮” and “十”, but “淮” being a character can be further decomposed into two components. In order to handle nesting and finding components to the most elementary form (no further decomposition), we have defined the decomposition and formation algorithms. There are mainly two algorithms, one for the decomposition of a character into a set of components (the algorithm is called *Char-to-Compnt*), another one for the formation of a set of characters from a component (the algorithm is called *Compnt-to-Charr*).

```

Let x be the seed (x = starting character for
search);
Stop = false
WHILE NOT stop DO
  IF Struct_Symbol(CD[x]) = “M”
    Stop = True
  ELSE
    LCmp = { cc[x] ∈ CC }
  ENDFOR
ENDWHILE
  
```

Figure 1. Pseudo-code of “Char-to-Compnt”

Both algorithms are very similar. They recursively retrieve all characters/components appearing in the decomposition database by using the characters/components themselves as a seed, but their directions of retrieval are opposite to each other. In the “*Char-to-Compnt*”, the decomposition goes from its current level down, one level at a time, until no more decomposition

can be done. Figure 1 the pseudo code of the algorithm for one level only and they can be done recursively to find all components of a character. Table 2 shows the entries related to the character “盟”. Notice that the number of components for “盟” is not two, but 4 because one of the components “明” can be further decomposed into two more components.

Character	IDC	Comp1	Comp2	Comp3
盟	B	明	皿	
明	A	日	月	
皿	M			
日	M			
月	M			

Table 2. Component Entries of character “盟”

On the other hand, the “*Compnt-to-Char*” algorithm searches from its current level up until no more character can be found using the current component. Figure 2 shows the pseudo code of the upward search algorithm where *x* is considered the seed to start the search and the variable contains all characters formed using the current component *x*.

```

Let x be the seed (x = starting component for
search);
Stop = false
Char_List = { x }
WHILE NOT stop DO
  IF No Change to Char_List
    Stop = True
  ELSE
    FOR each x in Char_List
      Char_List = Char_List ∪ { Char[x] }
    ENDFOR
  ENDFOR
ENDWHILE
  
```

Figure 2. Pseudo-code of “Compnt-to-Char”

Character	IDC	Comp1	Comp2	Comp3
口	M			
吾	B	五	口	
語	A	言	吾	
齧	A	齒	吾	
唔	A	口	吾	
...				

Table 3. Example character entries of component “口”

Table 3 shows some of the search results involving the component “口”. Note that the result not only find the character “吾”, but also the characters using “吾” as components as well.

Further more, due to the fact that there are two IDCs with cardinality of three, the decomposition is not unique. Based Han characters formation rules, some characters should be decomposed into two components first before considering further decomposition. For instance, “鋤” should be decomposed into “金” and “則” whereas “街” should be decomposed into “行” and “圭”. However, for upward search we certainly want the character “鋤” to be found if the search component is “鋤”. Therefore, in addition to using the most reason decomposition at the first level, we also maintain different decompositions for applications where character formation rule are less important. In other words, we also provided composition and decompositions independent of certain particular character formal rules. Again taking the character “鋤” as an examples, its components should not only be “金” and “則”, but also “鋤”, “則”, “鋤” as well as “貝” and “ ”. In fact, in our system, “鋤” is decomposed into “金”, “貝” and “ ” as shown in Table 4. The “Char-to-Compnt” algorithm will take the relative positions of the components into consideration based on the IDC defined in each entry to find other three possible components “鋤”, “則” and “鋤”. This can be done because the combination of “金” and “貝” will form “鋤”; similarly “貝” and “ ” will form “則”; and “金” and “ ” will form “鋤”. Note that in the first two cases of the OR clause, “鋤” and “則” will be identified. In the third case of the OR clause, the character “鋤” will be identified. You may argue the validity of the third case of the OR clause, but for the character “街”, finding the component “行” would be very important.

Character	IDC	Comp1	Comp2	Comp3
鋤	K	金	貝	
鋤	A	金	貝	
則	A	貝		
鋤	A	金		

Table 4 An example of handling a character with three components

The basic principle of the algorithm, as shown in Figure 3, is that if we see a character with an IDC {K} or {L}, or an IDC of a character that can be transformed to IDC {K} or {L}, we will try to use its components to form characters.

```

Let x be a Chinese component (x = cc);
Let LCsub be the list of sub-components c;
IF x[structure] = IDC{K} THEN
  LCsub = c : c[structure] = IDC{A} AND
  c[component(1)] = x[component(1)] AND
  c[component(2)] = x[component(2)] or
  c[component(2)] = x[component(2)] AND
  c[component(3)] = x[component(3)] or
  c[component(1)] = x[component(1)] AND
  c[component(3)] = x[component(3)]
END
**the same algorithm works when x[structure] =
IDC{L}, then the result c[structure] will become
IDC{B}

```

Figure 4 Pseudo-code for handling a character with three components

```

Let s be the seed (s = cc);
Let r be the result component;
if s[structure] = IDC{A}
  if s[component(1)][structure] = IDC{A} then
    r = IDC{K} +
    s[component(1)][component(1)] +
    s[component(1)][component(2)] +
    s[component(2)]
  else if s[component(2)][structure] = IDC{A}
  then
    r = IDC{K} + s[component(1)] +
    s[component(2)][component(1)] +
    s[component(2)][component(2)]
  end
end
**this algorithm also works when s[structure] =
IDC{B}, then the result structure will become
IDC{L}

```

Figure 4 Pseudo-code of For the Split Step

In many cases, we still want to maintain the characters in the right decomposition, e.g, to decompose them into two components first and then further decompose if needed. Take another character “樹” as an example. Suppose it is only decomposed into two components (“木” and “對”). This makes the search more complex. In order to simplify the search, we need to go through an internal step which we call the *Split Step* to decompose the character into three

components before we allow for component to character search. The pseudo code for the *Split Step* is shown in Figure 4. The generated result is shown in Table 5.

Character	IDC	Comp1	Comp2	Comp3
樹	A	木	封	
封	A	壹	寸	
樹	K	木	壹	寸

Table 5. An example Output of the *Split Step*

For some characters like “衡”, the Split Step must consider the component “重” in the middle as an insertion into the character “行”. We use similar handling to decompose “衡” into “彳”, “重” and “亍”, with an IDC {K}. In order to find a character with the component “行” such as “衡”, we need additional algorithm to locate components that are potentially being split to the two sides with an inserted component. We try to decompose a component into two sub-components if their IDC is “A” or “B”. Once we get the two sub-components, we try to make different combinations to see if there are any characters with an IDC {K} or {L} that contain the two sub-components as shown in Figure 5.

```

Let x be a Chinese character (x = cc);
Let Clst be the list of results c;
if x[structure] = IDC{A} then
  Clst = c : c[structure] = IDC{K} and
  ((c[component(1)] = x[component(1)] and
  c[component(2)] = x[component(2)]) or
  (c[component(2)] = x[component(1)] and
  c[component(3)] = x[component(2)]) or
  (c[component(1)] = x[component(1)] and
  c[component(3)] = x[component(2)]))
end
**this algorithm also works when x[structure] =
IDC{B}, then the result structure will become
IDC{L}

```

Figure 5. Pseudo-code of finding inserted component

### 3. Performance Evaluation

Since the algorithms have to do excessive search for many combinations in many levels recursively, performance becomes a very important issue especially if we want to make this for public access through the internet. However, since the decomposition is static, it

does not need to be done in real time. as the search doesn't need to be done online, In other words, searching of the same data will always give the same result unless the decomposition rules or algorithms are changed. Consequently, we built two pre-searched tables to store the results of both “*Compnt-to-Char*” algorithm and the “*Char-to-Compnt*” algorithm. Once we have the pre-searched tables, we can totally avoid the recursive search. Instead, the search result can be directly retrieved in a single tuple. This results in much better performance both in terms of usage of CPU time and I/O usage.

Character	Pre-searched result
銅	銅 銀 則 釧 金 貝
語	語 言 吾 五 口
.....	

Table 6. Examples of pre-searched results of “*Cha-to-Compnt*” Algorithm

Character	Pre-searched result
口	口 言 吾 語 谷 鯨 靈 ... (total 5481 characters)
五	五 吾 語 瘡 迺 齧 ... (total 44 characters)
.....	

Table 7. Examples of pre-searched results of “*Component to Character*”

Table 6 and table 7 shows some samples of the pre-searched tables for the downward search and the upward search, respectively.

Although the advanced control algorithms can retrieve most Chinese characters correctly, they also return some components that do not make much sense. For example, the character “章” has a structure of IDC{B}, and components “立” and “早”. However, when it is eventually decomposed into “立”, “日” and “十”. Using the algorithm “*Char-to-Compnt*”, the component “辛” will also be returned, even though “辛” has no cognate relationship with the character “章”. We can take into consideration of only a subset of characters that can be split in character formation, such as “行” and “衣”. This way, the insertion components will only be considered for these characters.

### 4. Conclusion

In this paper, we focus on the algorithms of character decomposition and formation. The results can be used for the standardization of

computer fonts, glyphs, or relevant language resources. We have implemented a Chinese Character Search System based on the result of this standardization work. We can use this search system to look for character decomposition or formation results. The system comes with many handy and useful features. It provides a lot of useful information on Chinese characters, such as the code for various encodings, and pronunciations. A stand-alone version is also built. The actual implementation of these algorithms and of the database helps people to get information about Chinese characters very quickly. It further facilitates researchers' work in related areas. For more information on the system, please visit the website <http://www.iso10646hk.net>.

### Acknowledgement

The project is partially supported by the Hong Kong SAR Government (Project code: ITF AF/212/99) and the Hong Kong Polytechnic University (Project Code: Z044).

### References

- [1] ISO/IEC, "ISO/IEC 10646-1 Information Technology-Universal Multiple-Octet Coded Character Set - Part 1", ISO/IEC, 2000
- [2] ISO/IEC, "ISO/IEC 10646-2 Information Technology-Universal Multiple-Octet Coded Character Set - Part 1", ISO/IEC, 2001
- [3] John Jenkins, "New Ideographs in Unicode 3.0 and Beyond", Proceedings of the 15<sup>th</sup> International Unicode Conference C15, San Jose, California, Sept. 1-2, 1999
- [4] Dept. of Education (Taiwan), "Dictionary of Chinese Character Variants Version 2", Dept. of Education, Taiwan, 2000
- [5] 李學銘 (主編), 《常用字字形表》, (二零零零年修訂本), 香港教育學院出版, 二零零零年 (LEE Hok-ming as Chief Editor, Common Character Glyph Table 2<sup>nd</sup> Edition, Hong Kong Institute of Education, 2000)
- [6] GF3001-1997 《國家語言文字工作委員會語言文字規範 信息處理用 GB13000.1 字符集漢字部件規範》, 國家語言文字工作委員會, 1997 年 12 月.
- [7] Lu Qin, The Hong Kong Glyph Specifications for ISO 10646's Ideographic Characters. 21<sup>st</sup> International Unicode Conference, Dublin, Ireland, May 2002
- [8] Hong Kong Special Administrative Region Government, "Hong Kong Supplementary Character Set", HKSARG, September 28, 1999
- [9] Hong Kong Special Administrative Region Government, "Hong Kong Supplementary Character Set - 2001", HKSARG, December 31, 2001