LE_002

クメール文字TrueTypeフォントのレガシ符号化方式とその自動識別可能性

Encodings of Non-Unicode Khmer TrueType Fonts and Auto-Detection

鈴木 俊哉 † 佐藤 大 ‡
suzuki toshiya Sato Dai

1. 背景

ブラフミ文字を起源とするインド系の文字は，基底文字となる子音文字に対し上下左右に母音記号や声調記号を配置し，複雑な合字規則を持ち，さらに配置の順序は音韻的な順序と一致しない．この複雑な合成規則のため，インド系の文字の符号化はローマ字，アラビア文字，日中韓の文字に比べて困難である．現時点では，国際標準準拠という観点ではインド系の文字はISO 10646文字集合 [1] をUnicode符号化方式 [2] で符号化するのが望ましいとされる．Unicodeの符号化方式はインドの国家企画であるIS 13194:1991を基本としている．これは，情報交換用のデータは音素を指示する抽象的な文字を音韻順に符号化しておき，入出力にはレイアウトエンジンを用いる処理を想定していた．インドでは，このようなシステムが1983年以降作成されてきた．東南アジアのインド系文字についても，早期に独自の国家規格を定義したタイを除けば [3]，Unicodeでは同様の符号化を規定されているものが多い [2, p. 265 - 287]．しかし，東南アジアの国々ではインドのようなテキストレイアウトエンジンを実装する技術的蓄積は不足しており，このような符号化方式では自国の文字を扱うことが困難であった．これに対応するため，具体的な字形の表示順に符号化する欧米用ソフトウェアで自国の文字を使う目的で，文字の表示における図形的な要素を符号化したフォントが広く用いられた [4, 5]．本稿では，このようなレイアウトエンジンを用いない符号化方式をレガシ符号と呼ぶ．これらの符号化方式のほとんどは各フォント製品に個別に実装されたもので，符号化方式の名前も無く，規格として整理されていない．従ってレガシ符号フォントを用いて作成された文書データは，フォントと共に配布することで見読性を提供することはできても，用いられる符号化方式に関する情報が欠けており，可搬性のあるテキストデータとして扱えない．本稿では，このような符号化方式が混乱した状況にある文字としてカンボジアのクメール文字をとり，インターネット上で配布されているフォントの符号化方式について調査し，符号化方式の自動識別の可能性について明らかにする．フォントの数は文書の数に比べて圧倒的に少ないので，文書を図形認識的処理によりテキスト化したり，あるいは，符号化文字列のビットパターン推測よりも高い精度でテキスト抽出が可能になると考えられる．

2. クメール文字の符号化の状況

クメール文字はカンボジアの公用語文字であるが，カンボジアの国家規格がない状態で，国外のクメール語専門家によってISO 10646およびUnicodeへの規格化が行なわれた [6]．この文字集合および符号化方式もISCIIに準じたものであるが，当時のISO 10646のテキストレンダリングの実装は，Yannisによるクメール語TeXだけであり [7]，ISO 10646に基いてWebブラウザやオフィススイートにクメール語サポートを追加した例はなかった．従って，前述のレガシ符号によるクメール文字サポートが広く用いられた．しかし，レガシ符号の設計の際に参照すべき標準が無いため，製品間で互換性が無い．また，レガシ符号のフォントの利用は，ユーザがフォントを容易に追加できるシステム (MacOS, Windowsなど) の普及により広がったが，これらのシステムの標準フォント形式であるTrueTypeでは，規格上はMacOSのクメール語レガシ符号しか宣言できない．そのため，レガシ符号フォントはUnicodeやRomanなどの標準的な符号化方式を宣言しているが，実際にはレガシ符号を用いているという構造になっている．

符号化方式	N_f	N_{cp}	N_{glyph}	備考
ABC	46	152	152-204	ABC派生
Zero Space ABC	33	153	153-201	
Theodore Rith Heng	19	151	151-155	Theodore Rith Heng派生
XIMPLEX	26	169	169-214	
KHEK	2	219	219	KHEK派生
KHEK1	2	219	219	
KHEK US	1	160	160	
KSCII	1	226	226	Mac用
Kh	7	407	407	Unicode含む
KSW	23	169	169-227	
Limon	18	152	152-155	
CDT	7	156	156-160	
Cambodia Portal	3	154	154-220	
SEA	1	125	125	言語学教材

表1: 調査したフォントの各符号化方式，フォント数 (N_f)，コードポイント数 (N_{cp})，グリフ数 (N_{glyph})．

3. 調査

3.1 調査方法

調査はインターネット上で無償配布されているWindows用のクメール文字フォント189個を対象に，各コードポイントのグリフを印字した符号表を作成して行なった．符号化方式の同定については，フォント内のWindows用UCS2コードポイントと文字描画データ (グリフ) を実際に印字して符号表を作成し，グリフが割り当てられているコードポイント (使用済コードポイント) について同一グリフがわりあてられているものを同一の符号表とした．同一ベンダのフォント製品でも空きコードポイントに文字を追加している場合がありグリフ数 (N_{glyph}) にはばらつきがあるが，フォント間では外字として扱い，符号表コードポイント (N_{cp}) としては各フォントで一致

† Information Media Center, Hiroshima Univ.
‡ Strategic Informatics Center, Tohoku Univ. Hospital

しているものを取った．レガシ符号フォントでは，描画データ内に基底文字の位置を示す点線の○などの記号は一切含まれないが，本稿では Unicode 符号表との対比のため特に記入した．符号表内でこれが記入されている文字は，合字形成のためにメトリクスが調整してあり，基底文字との重ね打ちができるようにしていた．これにより，合字が印字が可能となる．重ね打ちのために描画位置が移動しないようにメトリクスを調整した文字をノン・スペーシング・キャラクタ (NSC) と呼び，通常の文字のように印字毎に描画位置が移動するスペーシング・キャラクタ (SC) と区別する．

3.2 調査対象フォントから得られた符号化方式

調査結果を表 1 に示した．符号化方式の名称が不明なものは，開発者または配布者を符号化方式名とした．クメール文字の合成規則をもとにすると，理論上区別されるべき合字は 535060 個ありえるが，日常的に使われている合字は 2821 個にすぎず，この程度の数を提供できれば実用的と見積られている [7]．ISCII のように，合字は符号化せず音素を指示する文字まで分解した場合には子音胴文字 (基底文字となる子音文字の形状) 35 個，子音脚文字 (基底文字に付加して複合子音を表記する形状) 33 個，母音記号 (胴文字に付加して母音を修正する形状) 16 個，孤立形母音文字 (単体で母音を表記する形状) 17 個，ASCII 記号 22 個，その他記号 15 個，の合計 138 個が必要とされている [8]．ISO 10646 では胴文字と脚文字を同一コードポイントにすることでコードポイント数を 114 個に抑えたが，これは OpenType フォントのように一つのコードポイントに対して複数のグリフを割り当てるような仕組みが無ければ処理できない．従って，そのような機能がないレガシ符号フォントでは 138 個，記号類を除いても少なくとも 101 個のコードポイントが必要となる．

Windows 用 UCS2 符号は 16 ビットで記述されるので 2821 個のグリフを収容可能であるが，Kh 符号を除く全てのフォントでの使用済コードポイントは 0x0000-0x00FF の ASCII/Latin-1 領域だけで，収録されているグリフの数は上記の 101 個よりは多いが 256 個以下であった．Kh 符号フォントは，0x1780-0x17FF(ISO 10646 におけるクメール文字の領域) に Unicode 符号のクメール文字を割り当てているが，ASCII/Latin-1 領域にもレガシ符号でのクメール文字を 178 個割り当て，クメール文字レガシ符号フォントとしても使えるようにしたものであった．この結果から，レガシ符号化は全て 8 ビット符号化方式であることが分かった．コードポイント空間を 16 ビットにとりながらも 8 ビットで符号化していたことから，インプットメソッドによる支援が必要なマルチバイト符号化を避ける強い動機が窺われる．本稿ではレガシ符号として 0x0000-0x00FF のコードポイントに注目するので，下位 8 ビットの 0x00-0xFF の符号表を示す．紙面の都合上，全ての符号表を示さないが，G0(0x21-0x7E), G1(0xA1-0xFE) のコードポイントが主に使用され，C0(0x00-0x1F), C1(0x80-9F) はあまり使用されない傾向が見られた．

3.3 符号化方式の概観

まず，カンボジア政府が使用している 2 種の符号化方式，Zero Space ABC 符号化と Limon 符号化について図 1

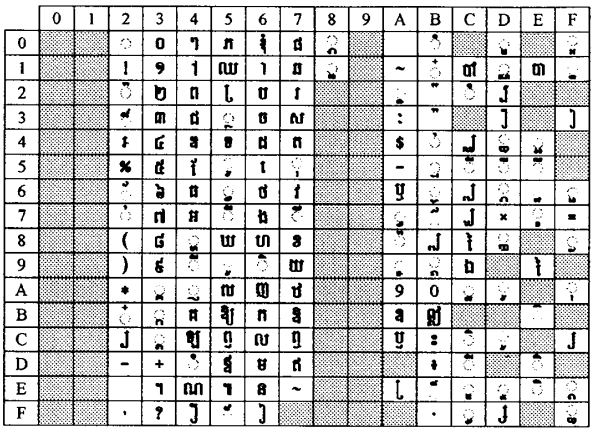

図 1: ABC 符号 (使用フォント: text02a.ttf)

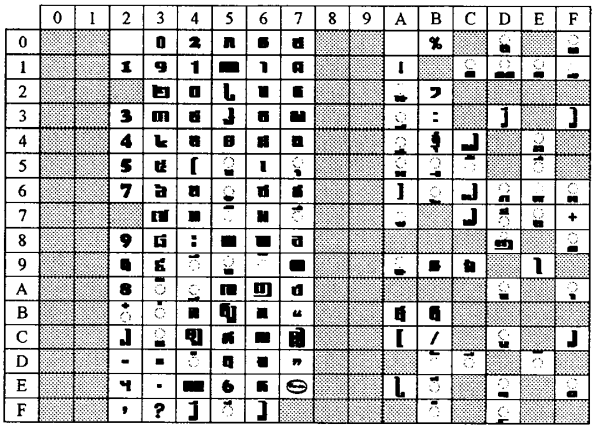

図 2: Limon 符号 (使用フォント: Limon_F4.ttf)

と図 2 に符号表を示した．Zero Space ABC 符号に含まれる NSC は 65 個，Limon 符号に含まれる NSC は 50 個であった．この数は脚文字と母音記号の合計 49 個に近いが，基底文字の右に配置される脚文字や母音記号は SC として符号化されるので，実際に必要な NSC の数はこれよりも少ない．むしろ 0x49 と 0xCD, 0x57 と 0xC5, 0x69 と 0xED, 0x77 と 0xE5 のように同一の母音記号についてグリフの微妙な位置の違いを使い分けるために複数のコードポイントを用いたことによって NSC の数が増えていた．これらの位置調整は TrueType フォントの kern テーブルでペア・カーニングを定義することでも可能であるが，調査したフォント 189 個中，kern テーブルを用いて調整している例はなく，全て符号レベルで位置を調整していた．

符号表を比較すると，Limon 符号の文字集合は算用数字を含んでいたが Zero Space ABC はこれを含まないなど，両符号化方式は，文字集合のレベルで互換性がなかった．しかし，ASCII 符号においてローマ字がわりあてられていたコードポイント (0x41-0x5A, 0x61-7A) では 52 個中 50 個の文字が一致していた．また，算用数字用のコードポイント (0x30-0x39) はクメール数字が割り当てられており，これも両符号表で一致していた．G1 で文字が一致しているコードポイントは 75 個中 33 個しか一致しなかったので，算用数字およびローマ字用のコードポイント上では両符号表は良く一致していたと言える．

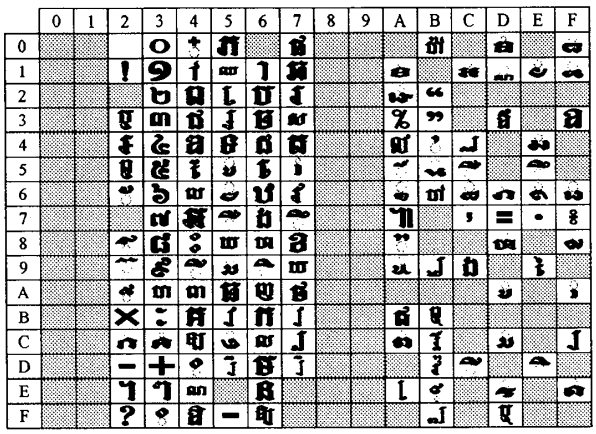

図 3: Theodore Rith Heng 符号 (使用フォント: BARAY_.TTF)

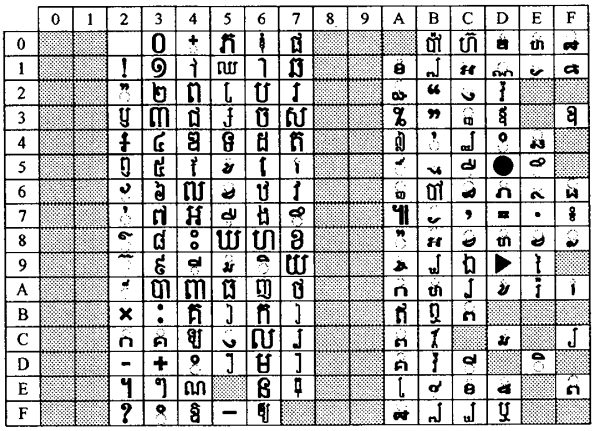

図 4: XIMPLEX 符号 (使用フォント: Ekreach_V3.ttf)

3.4 文字集合拡張の傾向

本節では，レガシ符号において，文字集合を拡張した場合に，どのような文字に新たにコードポイントを割り当てるかについて整理する．ここで，簡単な比較のため，符号化方式が非常に近い Theodore Rith Heng 符号と XIMPLEX 符号を比較し，使用コードポイントが増えた場合にどう利用されるかを見る．

Theodore Rith Heng 符号化方式の符号表を図 3 に，XIMPLEX 符号を図 4 に示した．Ekreach_V3.ttf に追加されている 22 文字の内訳は子音文字が 18 個 (うち脚文字 15 個，結合済胴文字 3 個)，母音記号 1 個，記号 2 個，不明 1 個である．脚文字の追加が大半を占めているが，このうち，Theodore Rith Heng 符号にまったく含まれていない文字は 2 文字にすぎず，13 個は微妙に異なるメトリクスを調整するためにコードポイントをわりあてられている．文字によっては最大 3 個のコードポイントを用いている．この結果から，レガシ符号化のフォントは，文字を追加する際には結合字形の収録ではなく，異なるメトリクスの文字の収録が行なわれると考えられる．

3.5 G0 コードポイントの使用傾向について

ここまでに示した符号表は，符号化方式に互換性が無いにも関わらず G0 の符号はよく一致している．これは US-ASCII キーボードをクメール文字タイプライタとして使おうとした場合のローマ字-クメール文字の対応づけに由来すると考えられる．情報処理上は，キーボード配列と符号化方式は別のものであるので，情報処理用に再設計した場合には異なる符号化になる場合がある．その具体例として，MacOS 用に再設計された KSCII 符号を図 5 に示した．KSCII 符号の G0 は，図 1-4 と比較すると，算用数字以外はまったく異なる．

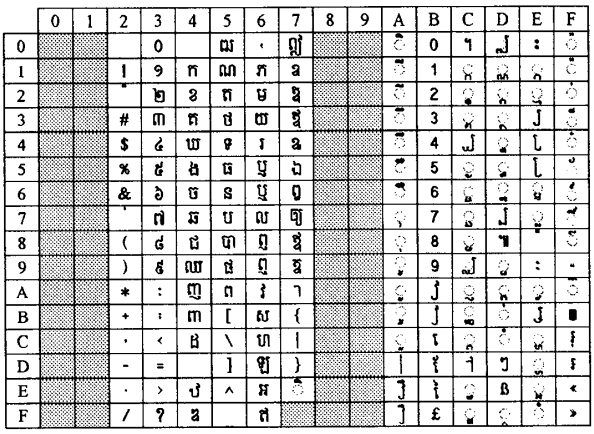

図 5: KSCII 符号 (使用フォント: KMon40K.ttf)

4. 符号化方式の自動識別

本章では，調査結果をもとに，与えられたレガシ符号化の TrueType フォントについて，符号化方式を識別する可能性について検討する．自動識別の方法として，TrueType フォントの規格に基いて演繹的に識別する方法と，フォントが含む文字の分布から発見的に識別する方法が考えられる．しかし，上に述べたように，レガシ符号化フォントでは符号化方式を Unicode と宣言しながら内部でレガシ符号を用いていたため，演繹的な識別は困難である．一方，グリフに対する図形認識処理は実装の負荷が大きい．特にクメール文字とクメール文字においては，同一文字の書体間の違いが同一書体の文字間の違いより大きい場合が多く，OCR の精度が向上しない問題が知られている [9]．従って，図形認識処理により文字を特定して符号化方式を識別することも困難である．そこで，符号化方式の識別の手掛りとして，フォント内部の独立したテーブルに格納されているメトリクス情報に注目した．これにより，図形認識を行なわずにグリフが SC であるか NSC であるかを判別することができる．つまり，未使用コードポイントか，また，使用済コードポイントについても SC と NSC のどちらが配置されているか (属性) から符号化方式の識別を試みるというものである．SC, NSC の識別は文字の図形の特徴を識別するものであるが，図形認識のような複雑な処理は必要なく，また，書体間の差異も小さい．

ここで注意しなければならないことは，クメール文字のレガシ符号フォントでは，未使用コードポイントがしばしば外字を追加する目的で使用されるということである．すなわち，C0, C1 のように，ISO 8859 では制御コード領域とされていて透過性が保証されない領域以外では，特定のコードポイントを未使用であると想定して比較することは望ましくない．よって，符号化方式の比較の際には，未使用コードポイントの分布よりも，SC と NSC

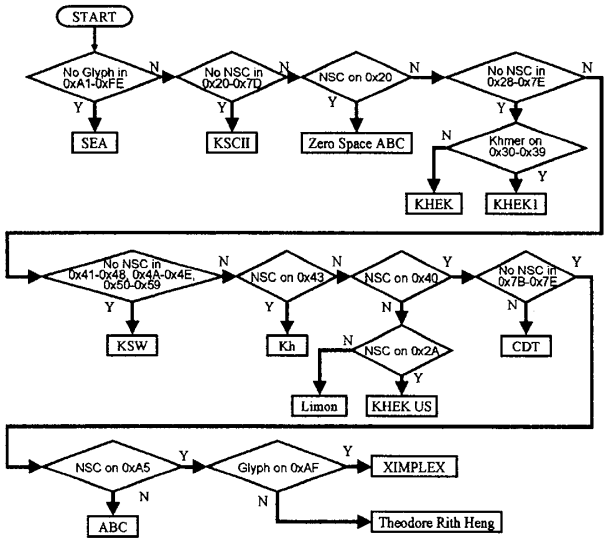

図 6: レガシ符号表の識別アルゴリズム

の分布の比較を優先させるべきである．

さて，図 1-5 の比較から，レガシ符号化では G0 がほとんど使用済みとなっており，また G0 内部のコードポイントではメトリクス調整のための同一文字に対する重複した符号化も行なわれていないことがわかっている．さらに，G0 のコードポイント入力にはキー配列が大きく影響するため，フォント製品の差別化において文字を追加する際に使われるのは G1 と予想される．この予想に基き，まず G0 内の使用済みコードポイントの属性から符号表を識別することが望ましい．しかし，Theodore Rith Heng 符号と，それから派生したと考えられる XIMPLEX 符号のように，使用済コードポイントが完全に同一の場合は，未使用コードポイントについて比較せざるを得ない．以上の方針をもとに，収集したレガシ符号化フォントの符号化方式を識別するアルゴリズムの一例を図 6 に示す．

図 6 に示した識別アルゴリズムは，大きな特徴を持っている符号表を先に候補から除外してゆくという順番で組みたてている．仮に誤認識が生じた場合でも，少なくとも先に除外された符号化方式ではないと考えることができる．本稿では全ての符号表を示さなかったので，注目した符号化方式の違いについて以下に説明する．SEA 符号は G0 にのみクメール文字を定義しており，G1 にはグリフをわりあてない．他のレガシ符号は G0，G1 の二つを利用しているので，この特徴はもっとも大きな違いである．これによりまず SEA 符号を除外する．次に，KSCII 符号は図 5 に示したように G0 が 0x7E を除いて全てスペーシング・キャラクタである．他のレガシ符号，特にタイプライタに由来すると考えられる符号化方式では，G0 にノン・スペーシング・キャラクタを含むので，検出する大きな特徴である．次に，Zero Space ABC 符号だけは 0x20 のコードポイントはゼロ幅のスペース文字がわりあてられているが，他のレガシ符号は 0x20 は通常の有限幅のスペース文字であるので，これにより Zero Space ABC 符号を検出できる．この 3 つは，符号化方式の設計に関わるものであるため，大きな特徴として現われた．これらを除外した後の符号化方式の識別は，設計方針を反映している部分が明らかではなく，十数個～数個のコードポイントでのノン・スペーシング・キャラクタの分布で識別する．これ以降の識別手順においては，未使用コードポイントに製品差別化のためにノン・スペーシング・キャラクタがわりあてられる可能性を考慮し，連続したコードポイント群にスペーシング・キャラクタが配置されていることを調べる．これにより除外できるのは，KHEK, KHEK1, Kh, KSW などである．ここで，KHEK と KHEK1 は同一ベンダによる符号化方式で，クメール数字を使う (KHEK) か算用数字を使う (KHEK1) かのみが異なるので，この識別には図形認識が必要である．ただし，クメール数字と算用数字の書法は同じであり，この違いがクメール語テキストの見読性に及ぼす影響は他の符号化方式との誤認にくらべれば小さい．これ以降の符号化方式の識別 (Limon, KHEK US, CDT, ABC, XIMPLEX, Theodore Rith Heng) は，数点のコードポイント上での属性から識別せざるを得ないので，ここで示したものとは異なる順序で除外してゆくアルゴリズムもありうる．

5. まとめ

本稿では，インド系文字のうち，Unicode 以外の規格化がされていない文字系の例としてクメール文字をとり，レガシ符号フォントについて図形認識によらずに符号化方式を識別する可能性について検討した．調査の結果，インターネット上で配布されているクメール文字 TrueType フォント 189 個から 14 種類の符号化方式が見つかり，同一ベンダにより ASCII 数字とクメール数字を置換したものを除き，文字のメトリクス種別により自動識別が可能であることを示した．

この結果から，レガシ符号化された文書データからテキスト抽出について，文字認識的な手法に比較してより効率的な方法の可能性が示されたが，文書データのレンダリング自体にはレガシ符号化のフォントが必要なことには変わりがない．レガシ符号化されたフォントを OpenType でエミュレートする可能性についても検討が必要である．

参考文献

[1] ISO/IEC JTC1/SC2, : *ISO Standards: Information Technology – ISO/IEC 10646:2003, Universal Multiple-Octet Coded Character Set (UCS)*, ISO (2003).

[2] The Unicode Consortium, : *The Unicode Standard, Version 4.0.0*, Addison-Wesley (2003), ISBN 0-321-18578-1.

[3] Karoonboonyanan, T.: *Thai vs Indic Encoding Scheme* (2004), http://linux.thai.net/~thep/thai-vs-indic.html.

[4] Mikami, Y. and Pavol, Z.: Writing Systems and Character Codes in the World (1), *IPSJ Magazine*, Vol. 46, No. 8, pp. 919–924 (2005).

[5] Mikami, Y. and Pavol, Z.: Writing Systems and Character Codes in the World (2), *IPSJ Magazine*, Vol. 46, No. 9, pp. 1046–1052 (2005).

[6] Daniels, A.: *Unicode Technical Report #1 - Burmese, Khmer, and Ethiopia* (1992), http://www.unicode.org/unicode/reports/tr1.html.

[7] Haralambous, Y.: Typesetting Khmer, *Electronic Publishing*, Vol. 7, No. 4, pp. 197–215 (1994).

[8] Cambodia, : *Report by Higher Education Task Force* (1996), http://www.bauhahnm.clara.net/Khmer/Page1.JPG ～ Page4.JPG.

[9] Cooper, D.: How to Read Less and Know More: Approximate OCR for Thai, *Proceedings of the 20th annual International ACM SIGIR Conference*, pp. 216–225 (1997).