情報学基礎 8－3
（1988. 2. 22）

文字コード論の試み

宮澤　彰
学術情報センター

最近、コンピュータの応用で多文字種や多言語処理の分野が拡大し、文字セットの標準の改訂、拡張など、文字に関する関心が高まってきている。本稿は、コンピュータで文字を扱うための基礎的な概念を提唱し、その枠の上で文字情報の種々の問題を検討する「文字コード論」の展開を呼びかける。まず、文字コードをめぐる最近の動向、ISO，JIS，アジアやアメリカ、また、TRONでの新しい考え方などを簡単に紹介する。次いで、文字の同定の問題、文字セットの環境や用途、文字セットの運用、漢字の異体字の問題など、文字コードに関係する種々の問題をあらいだす。最後に言語の世界からシステムの内部処理までの文字コードや文字セットの世界の枠組みを示し、その枠組みの上で文字コードについての問題を検討する文字コード論を展開すべきであるとする。

On Character Coding Systems

Akira Miyazawa

National Center for Science Information System
3-29-1 Otsuka, Bunkyo-ku, TOKYO, JAPAN 112

Multiscript and multilingual information processing is spreading recently. And, interests for character codes and character sets are growing accordingly. This paper shows a general frame work of character code and sets. First, recent trends for standardization, such as in ISO, JIS, other Asian national standards, US trends, and new concepts in TRON, are introduced. Next, problems and current issues of character code/set systems are discussed. Finally, a general framework for character code/sets world, which covers from linguistical field to information processing field, is shown.

１．はじめに

最近のコンピュータの応用の世界では、文字や言語に対する応用が拡大し、新しい文字セットの標準化等の動きが相次いでいる。しかしながら文字コードについての理論的な基礎付けは十分なされているとは言い難い。このため新しい文字セットやその応用をめぐって種々の問題点が明らかになりつつあるというのが現状である。

これらの問題点に解決を与えるためには、文字とそのコンピュータの世界での表現である文字コードについて考察する「文字コード論」を展開し、文字セットの選定、応用を基礎づける必要があろう。

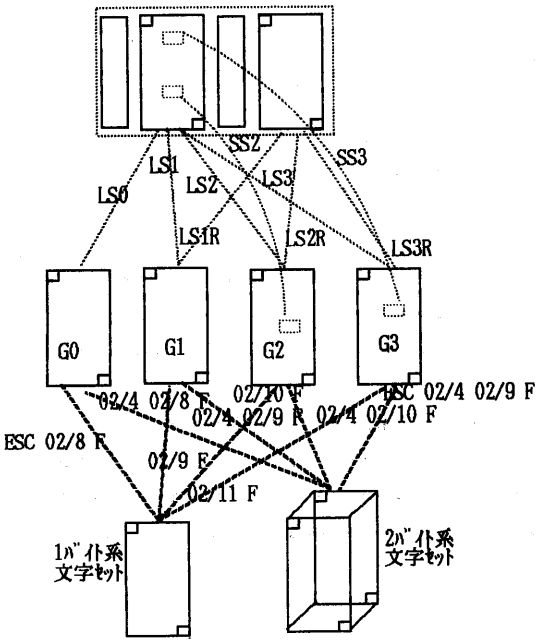
本稿は「文字コード論」展開の手がかりとして、以下の節でまず、最近の文字コードをめぐる世界の動向のいくつかを簡単に紹介し、次いで、その問題点をあらいだす。最後に概念の整理と、今後の展望について述べる。

２．最近の動向

２．１　ＩＳＯ

２．１．１　２０２２

ISO 2022は文字コードの拡張について定めた規格であり、多くの文字セットの基礎となっている。またJIS X0202（旧C6228）は2022の翻訳規格である。2022自体は古くからの７ビットの殻を捨て切れないなどの点で問題は残しているが、広く認められているものである。2022自体についてその概念や歴史を述べるのは本稿の目的ではないが、出発点としてその概念について略述しておく。



図－１　ISO 2022の指示と呼び出し（部分）

図－１に示すように、2022はいくつかの文字セットを平行して使用するための規格である。図形文字セットは94(96)文字、または94×94[×…](96×96[×…])文字の文字位置をもつ集合であり、これらの中から４つまでを選んでG0～G3として指示(designation)することができる。さらにその内の２つを8ビットの中に呼び出し(invocation)て同時に使用することができる。

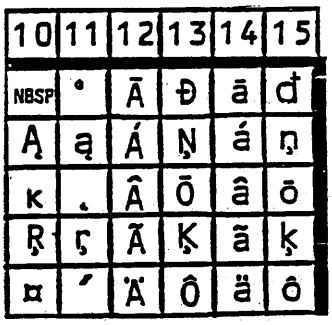
これらの指示にはエスケープシーケンスが使われ、また、呼出しには１バイトの制御文字やエスケープシーケンスが使われる。どの文字セットを指示するかはエスケープシーケンスの終端文字（図－１のF）で識別される。文字セットは国際登録することによっ終端文字を割当てられる。例えばJIS X 0208（旧C6226）は4/02が終端文字である。

2022の指示、呼出しという２段階の概念は、わかりやすいものとは言い難い。また、実際のハードウェアでは2022に完全に従って文字セットの切り替え可能なものは殆どないと思う。

にもかかわらず、ほとんどの文字セットはこの規格に従う形で設定されており、その意味で出発点となる概念といってよい。

２．１．２　８８５９，６９３７

現在ISOではDIS 8859が投票にはいっている。内容はヨーロッパの言語で使用するアルファベット、特にアクサンやウムラウトなどダイアクリティカルマークの着いたアルファベットを中心としている。図－２にその一部を示すがダイアクリティカルマークのついたものを１文字として１ポジションを割当てている。この方法ではラテンアルファベットの文字セットは１つでは収まらず、複数の文字セットに分かれる。



図－２ ISO 8859(部分)

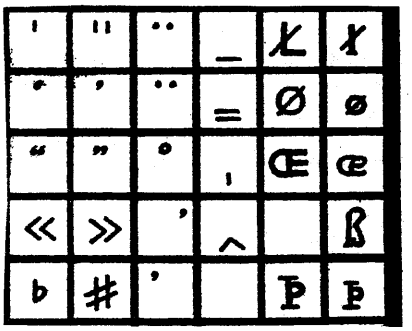
実は、同種のものとして6937がある。これも対象となっている文字は同じであるが、ダイアクリティカルマークはノンスペーシングキャラクタとして扱われ、アルファベットと組合せて用いられる方式である。

8859，6937のどちらも複数の文字セットからなり、ギリシャアルファベット、キリルアルファベットを含んだパートを持っている。またこれら２つの文字セットは、配置とダイアクリティカルマークの扱いの点では異なるが、文字としてはほぼ同じものを含んでいるといってよい。

２．１．３　書誌用文字セット

ＩＳＯで文字セットを扱っているのはTC97 SC2であ

るが、ドキュメンテーション分野の TC46 も「書誌用」(bibliographic use)文字セットをいくつか決めている。例えば5426，5427，5428などラテン、キリル拡張、ギリシャアルファベットである。図－３は5426の一部であるが、8859や6937とは文字の種類の上でも一致しない。



図－３ ISO 5426 (部分)

これらの文字セットは少なくとも現在の所普及しているとは言い難い。特にTC46系の文字セットは全く使用されていないといっても過言ではないと思う。

２．１．４　２オクテット図形文字

現在コンピュータの内部処理は殆どの場合８ビット系で行われている。7ビット系との互換性を考えた2022の場合、特に２バイトで使用すると、左と右との組合せでの２バイトが使用できないため無駄を生じる。２バイトでの表現は日本語や中国語のみならず、１バイトの文字セットを３つ以上同時に固定長のコードとして扱うためにも必要である。この問題を解決するためにＳＣ２ではＷＧ２を設けて２オクテット図形文字の検討を行っている。今年10月頃DIS化の予定ということである。[1]

これは2022での７ビットの殻を破るものであり、また多言語処理の一般化を促進することが期待される。現在の案では2022からの拡張性を重視した案となるということである。

２．２　ＪＩＳ

JISで文字コードに関連した規格としては、前述のコード拡張法Ｘ0202、１バイトの文字セットＸ0201(旧Ｃ6220)、漢字コードＸ0208などがある。

特にＸ0208は交換用としてのみならず、日本の殆どのハードウェアで内部処理用としても採用されている。さらに、その構成法の面で中国の標準にも影響を与えるなど、大きな影響力を持った標準であった。

近年のコンピュータの応用の範囲の拡大はＸ0208の文字（6353漢字＋524非漢字）だけでは不足するという分野を増加させ、拡張を望む声が大きくなってきた。例えば印刷分野、人名や地名を処理する応用分野などからである。

このため「漢字符号系調査検討委員会」が設けられ、議論が続けられてきた。８８年２月現在、未だ最終結論は出ていないが、相当数の漢字とダイアクリティカルマーク付アルファベットを中心に、新しい文字セットが作られている。この新しい文字セットはＸ0208で足りないような応用においての交換用文字コードを定めるものとして位置付けられることとなろう。

２．３　中国

中国では文字セットの国家標準としてGB 2312-80が存在している。この94×94の２バイト文字セット（終端文字4/01として国際登録されている)には漢字約6763文字と非漢字662文字（ラテン、ギリシャ、ロシアアルファベット、日本語かな、記号など）が含まれている。

しかしながら、この文字セットは基本集として位置付けられており、補助集が当初から計画されていた。補助集は未だ制定されていないが、補助第１集は基本集の繁体字、第２集、第４集は基本集にない簡体字で、第３集、第５集がその繁体字の予定ということである。[2]

なお、中国では漢字を扱えるハードウェアが日本ほど普及している訳ではなく、またGB 2312時体も近年の者であるから、ようやくGB 2312準拠のハードウェアが普及しはじめた所という印象である。

台湾では、コンピュータ、ワープロなどがかなり普及しているが種々の内部コードが使われており、統一コードや交換用コードのないことが問題であった。しかし1986年、CNS 11643が制定されている。CNS 11643は94×94の２文字セットに分かれていて、第１セットには651非漢字（記号、ラテン、ギリシャアルファベット、注音符号、部首など）と常用字5401漢字が含まれ、第２セットには次常用字7650漢字が含まれる。

ＣＮＳも現段階では普及は未だしの感があるが、今後の普及が期待されている。

台湾では CCCII (Chinese Character Code for Information Interchange)という文字セットもあった。台湾の国立図書館では、そのMARCに後述するアメリカとの関係もあって、このCCCIIを使用する予定であるという。CCCIIは３バイトを使用し、94×94×6の層が15層あるものとする。この構造のうち第１層に正体字（台湾での）を置き、対応する簡体字を第２層に、その他の異体字を第３層以下対応する場所に置くという構成をとっている。

２．４　韓国

韓国では1987年に国家標準の文字セットKSC-5601が改定された。この文字セットも94×94の２バイト文字セットで、2350ハングルおよび、4888漢字が含まれている。

ハングルはよく知られているように音素の組み合わせで形づくる文字のため、音素を１文字とみてコード付けする方法（この場合１バイトですむ）と、組合せられたものを１文字とみてコード付けする方法（２バイト必要）とがあり、この２つの方法の間での議論があった。また、後者の場合、現代韓国語で使用されているもの、古く使われていた形、理論的にありうる形などで数が大きく異なり、この点での議論もあったという。[3]

また、漢字は現代の韓国語ではあまり多くは用いられないが、古くはかなりの数が用いられていた。どの程度の文字が必要かについては丁度日本でと同様の問題が起こる。

韓国ではコンピュータ、ワープロなどがかなり普及しており、その間での文字コードの不統一はやはり問題であった。新しいKSCの普及が望まれている。

２．５　アメリカ合衆国

英語がいわゆるASCIIで十分な言語であるため、文字や文字セットに対する関心は、アメリカでは工業レベルになりにくいようである。もっともXEROX社のSTARや、Apple社のMacintoshのように多国語対応を意識したシステムもある。

文字セットの点では図書館の世界から動きが出てきた。いわゆる書誌ユティリティの１つ RLG (Research Libraries Group)は、中国語、日本語、韓国語の資料をカタログするための“CJK Terminal”を1982年にインプリメントした。

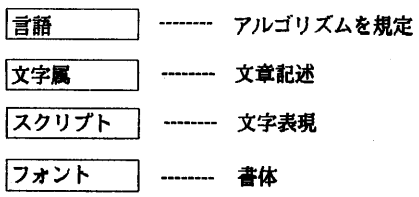
このシステムで使われている文字コード系 REACC （RLIN East Asian Character Code)は台湾のCCCIIをベースとして、GB 2312、JIS X 0208、韓国のKIPSというシステムで使用されていた文字セット、をマージしたものである。CCCIIは前述のように異体字－正体字関係をコード空間上の位置関係にマッピングしているため、その構造を温存しながら拡張しているようである。

この文字セットは議会図書館のいわゆるLC-MARCに採用され、OCLC,RLINの２大書誌ネットワークで使用されるようになった。現在議会図書館はこの文字コードをEACC(RLINが抜けてEast Asian Character Code) として米国内の標準とし、また国際標準にもしたいという意向をもっている。[4]

２．６　ＴＲＯＮの多国語対応

ワークステーションにおける多国語対応ではXEROX社のSTARやApple社のMacintoshのものがあるが、TRONではより一般化された方式を提唱している。その中には文字コードについてもいくつかの注目すべき考え方が含まれているので、簡単に紹介する。

TRONの規約の１つである TAD (TRON Application Databus)言語環境は図－4のような階層構成をもつ。[5]



図－４ TAD言語環境の階層

言語層は英語、日本語といった言語の層である。言語ごとに切り替えられるアルゴリズム、例えば入力アルゴリズム、ソーティングアルゴリズムなどがこの層に付随する。フォント層は書体を規定する階層である。

文字属層は「文章記述」の層、スクリプト層は文字の印刷・表示に関する「文字表現」の層であるとする。このスクリプトと文字属の分離が新しい考え方である。

これまでのいわゆる「文字セット」はほぼ「文字属」に対応する。スクリプトとはラテンスクリプト（ローマ字とそのダイアクリティカルマーク付のアルファベットなど）、キリルスクリプト、漢字などである。 ISO 8859でみたように、ラテンスクリプト全体では１バイトで表現できる数は超えるが、１つの言語で使用するのは一般にそれらのうち一部である。文字属はスクリプトのサブセットで１つまたは複数の言語で使用される文字を１バイトまたは２バイトの範囲にはいるようきりだしたものであると考える。スクリプトと文字属の写像は言語層で指定される。なお、スクリプトには出力用文字、例えばｆとｉのリゲチャなども存在する。

なお、TAD言語環境では８ビットの１バイト及び２バイトを使用し、2022とはやや異なるが、やはり文字属を切り替えて使用する方式である。

３．問題点のあらいだし

３．１　何が「文字」か？

(1)前説に述べたようにISO 6937ではダイアクリティカルマークをノンスペーシングキャラクタとして扱うが、8859ではダイアクリティカルマークとアルファベットの組で１「文字」である。同様の問題は日本語にもあり、JIS X 0201ではガをｶﾞで表現するが、X 0208ではガが１「文字」として存在する。

(2)これとよく似た問題で、リゲチャligatureの問題がある。欧文印刷ではfとiはくっついたかたちで印刷される。ラテン語などに使われる　や　もこれに似ているが、単なる印刷上のみならず、２重母音としての意味も持っている。さらにスペイン語のＣＨは印刷上は２文字とかわりないが、辞書では１「文字」とし独立した見だし項目となっている。

(3)ギリシャ文字のσとςはともにΣの小文字であるが、語尾にくるときはς、語中ではσの字形を用いる。これは同じ「文字」か？このような状況はアラビア文字ではさらによく起こる。似たケースでも一般にｇととの差は書体の違いで「文字」の違いとは言わない。

(4)漢字には「異体字」の問題がある。これにもレベルがあって、秋と穐、島と嶋が異体字関係にあるというのから、尭と堯、草と艸、あるいは主と主と主も異体字関係にあるといえる。この最後の例は同じ「文字」であるとされる方が多いが、秋と穐は同じ「文字」といわれることが少ない

(5)ISOの6937と5426ではダイアクリティカルマークの種類が異なる。例えばラトビア語の　の文字に対し、前者はGとcedillaとしているが、後者はGとhook to leftとし cedilla と区別している。また 542 6ではdiaeresisとumlautを区別しているが、6937では区別されない。

(6)JIS X 0208では“と”とは別の「文字」である。JIS X 0201では”は１種類である。”と”とは同じ「文字」か？

(7)ローマ数字Ⅰ，Ⅱはアルファベットｉと同じ「文字」か？日本のワープロでは独立した文字としている例が多い。欧米ではアルファベットを使って表現することが多い。

(8)ISOの8859は分数½や¼を「文字」としていれている。また上付数字も「文字」として採用している。

(9)数式のコード法と出力システムで知られるＴｅＸでは、括弧記号だけでもその断片など入れて数十字はある。

(10)あるデータベース中で温度の単位℃を上付の句点「。」とCとで表していた。

3．2　使用用途に関して

(1)前節にみたようにISO　TC46系では「書誌用」の文字セットを決めている。処理用や交換用、書誌用、テキスト交換用などの目的によって多くの文字セットができてしまうのは好ましくないと思われる。

(2)現代の日本語文を書くうえでJISX0208(以下JIS)はほぼ十分な文字セットといえる。（第2水準のかなりはほとんど不要でさえある）。しかしながら人名、地名などの固有名詞が出てくるとこれで足りない漢字もある。仮に、異体字について研究しようと思ったら、JISでコードされたデータを対象にしても無意味なことは明らかであろう。目的によって文字セットが異なることは必要性があるとも言える。

(3)目的と似たものに処理用途がある。例えばワープロは殆どの場合美しい出力を得る目的で使用され、其のデータは出力処理と蓄積用である。書誌データベース中では蓄積、出力の他にキーワード自動抽出などの処理や、ソーティング処理の対象となる。一般にコンピュータにはいっている文書は、蓄積、出力、交換の他、自動翻訳など言語処理の対象となる。

(4)内部処理用コードと交換用コードという考え方がある。現在のハードウェアでは1バイトの文字コードの場合、文字の対応さえはっきりしていれば、内部と交換用コードが異なってもとくに問題なく変換される。これに対して2バイトの文字コードでは、内部コードと交換用コードを一致させるか、計算によって変換するようにしている場合が多い。変換テーブルの大きさ、作成工数、性能などの問題によるものと思われる。このための内部コードと交換用コードの分離の悪さが問題を複雑にしている面もある。

3．3　文字セットに関して

(1)文字セットの安定性stabilityという言葉がある。ある文字セットがいっさい変更されなければもちろん安定である。さらに、ある文字セットに新らしい文字を割当てることをリビジョンと呼んで安定の範疇に入れる考え方がある。これに対し割当てられた文字を変更したり入れ換えたりすることは、この範疇にはいらない。

文字セットが変更になると、其の文字セットを使って蓄積されていたデータ全体に影響が及ぶ。データベースの膨大な蓄積全体にわたって新しい文字セットに更新する仕事は莫大なコストがかかり、不可能に近くなってきている。

ISO 2022は文字セットのリビジョンに対して新しい終端文字を与えず、リビジョンシーケンスをつける方法に改定された。

(2)この文字セットの変更に関してはJISの1978版と1983版の問題がある。この文字セットの「変更」は大きな影響を及ぼした。「変更」内容は3つあり、符号の空きへの追加、「字形」の変更、「字形」の入れ換えである。字形の変更や入れ換えは全く別の「文字」になるものではなく、些末な差から最大異体字関係の範囲である。

この「変更」に対しては批判の声が強いが、同時に、3．1(4)に述べた何が「文字」かの問題を浮き彫りにしたという要素もある。

(3)JISの1983年「変更」による混乱にもう1つ、「符号の空き」の使用方法の問題がある。1978年にJISができたとき、ユーザはその後の追加があるとは想像せず、その符号の空きに「外字」を割当てた。たまたま1987版では83区のちょうど最後まで文字が埋まっていたため、84区から「外字」を割当てた例が多かったという事情もある。ところが1983版で84区の1～4点に追加が行われたため、それらのユーザは困ってしまった。これに対しては内部コードと交換コードとの分離ができればよかったはずという考え方もある。

これらのことから、符号の空き領域について、ユーザ独自に文字を割当てられる自由領域と、今後の追加のための保留領域とはっきり示すべき出あるという考え方が支配的となってきた。韓国のKSC-5601には自由領域が設定されている。

(4)しかしながら、ユーザの私的な利用を許す自由領域は、交換のためには有害であるという考えもある。現にJIS拡張要求はそのような使用を行っている分野から声があがってきた。TRONでは、自由領域は認めず、TRONユーザ全体の中で文字セット管理を行い交換性を保証しようという考え方である。

(5)文字セットが複数あるとき、その間に同じ「文字」が存在する場合がある。例えばJISX0208とX0201はともにＡＢＣ，０１２などを含んでいる。前者は2バイト後者は1バイトであるから処理上は全く異なるコードである。

これに対していくつかの考えかたがある。2バイト＝全角、1バイト＝半角という対応を行って別の「文字」と考える方法は比較的安易に行われている。学術情報センターの応用システムではこれらを同じ「文字」と考え、入力された文字列を正規化表現することによってこの問題に対応している。(2バイトでコマンドを書いても良い).TRONでは、日本語文字属のアルファベットはA)…B) …といった使用のための文字で、英語を書くためのアルファベットではなく別「文字」としている。

3．4　漢字の問題

(1)漢字は其の数の多さのために特別な問題を引き起こす。例えば、日本語にせよ中国語にせよ、これで全国で10年間十分という漢字のセットを作ることはおそらく不可能である。例えば、大漢和辞典の約5万字でさえその役は果たせない。印刷産業連合会のJIS外字の調査によれば大漢和外の字が約2千5百ないし4千種出現しているのに対し、大漢和内のJIS外字は1万2千ないし1万4千種程度という比率であった。[6]

(2)最近の国際化は、中国語や韓国語、ことにその人名、地名などの固有名詞の日本語文章中への登場機会を増やしている。そのかなりは、例えば鄧小平の鄧など、JISには含まれない。またその簡体字が別の「文字」であるとするならば数はさらに増えるであろう。これらのうち日本語で使われるであろう上位いくつかの予測をするとしても、鄧小平の後継者の予測より難しそうである。

(3)中国語の漢字と日本語の漢字は同じスクリプトであるかどうかの議論がある。結論はスクリプトという

述語の定義による。少なくとも、ロシア語のアルファベットと英語のアルファベットとの関係よりは近く、フランス語のアルファベットとドイツ語のアルファベットとの関係よりは遠いといえるだろう。

4．文字コード論のために

図－6に文字コードをめぐる世界の枠組みを示す。

4．1　言語の世界における文字

これまでの文字を介した情報伝達はすべて、書かれた文字を媒介としていた。手稿であれ印刷であれ、受け取った人間は其の上にある図形を認識し、この図形をある文字として認識するという過程を経る。このような認識が可能であるのは、送り手と受け手が文字の集合を共有しているからであると考えられる。この文字という集合とその上にある構造は言語という枠組みの中に抽象的に存在するが、例えば字典のような形で固定され、具体的に表示される。このような文字の集合とその構造を各言語について明らかにするのは、各言語学の役割であろう。

コンピュータを利用した情報伝達も最終的な受取の形は変わっていないが、情報の蓄積や交換の段階で図形となる以前の形の媒体を用いる。この媒体の世界は人間の認識する文字の世界と一致していることが望ましい。こういった観点からの議論は言語学では新しいものであり、今後の検討を待つところも多い。

4．2　スクリプト

表記される言語の世界における文字を、情報処理の枠に載せるために具体的なものにする必要がある。そのためには例えば字典や文字表のような形で文字を示さなければならない。いくつかの言語の文字表で、その歴史からも同一の文字と認められる文字を殆ど共通に含んでいるとき、それらは同一のスクリプトに属すると認められる。同一のスクリプトに属する文字表はマージすることができる。

スクリプトは「文字」の集合である。個々の言語の中で何が「文字」であるかを決めること、また、どの言語表記がどのスクリプトにまとめられるかを決めることは言語学の分野の仕事である。（ここにいうスクリプトはTRONにおけるスクリプト[5]とは異なる）。

文字コード論の枠組みからは、スクリプトの世界は「文字」の集合であるスクリプトが、いくつかある世界であると捉える。表記された言語の中で個々の文字を識別するための属性、また、言語の中での其の文字の使われかたなどがスクリプト中での文字の属性である。また、文字の間の関係にどのようなものがあるかは文字のｎ個と、関係属性の組で表現できる。

なお、言語は生きているものであるから、スクリプト集合は永久に固定できるものではなく。常にメンテナンスされるべきものである。

4．3　文字コード空間

文字コード空間は、文字をコード付けするための順序を持った「場所」の集合である。この場所はグループ化して幾つかのサブセットに分けられる。例えばISO 2022では2/1～7/14で識別される94個の場所を持った集合が79個と、94×94［×…］(96×96［×…］)の場所を持った集合が同じく79個とが文字コード空間を構成しているわけである。

文字コード空間のサブセットをどのような構成にするかは、次に述べる処理コード空間との関係で決められるべきことであり、情報処理側の課題である。一般に１バイトまたは２バイトでおさまるようなサブセットを考えて、それらがいくつかあるという構成にする。処理コード空間とのマッピングの便のためである。

文字コード空間にスクリプト中の文字を割当てるのも情報処理側の課題である。スクリプトと文字コード空間のサブセットが１：１に対応し、スクリプト中の文字と文字コード空間中との場所とが１：１に対応することがもっとも美しくはあるかもしれないが、実際には不可能である。実用的見地から決められることであり、スクリプト中の１文字が、文字コード空間上の２つ以上の場所に割当てられることもある。

一般に文字コード空間上のサブセットとそこへの文字の割当てを文字セットといっている。JIS X 0208はこの意味での文字セットである。TRONの文字属もこれにあたる。他にコード表といった言葉、例えばJISコード表、でこれらを表すこともある。

4．4　処理コード空間

処理コード空間はオペレーティングシステム、あるいは一連のソフトウェアシステムで図形文字を表現する属性と値の集合である。この中に文字コード空間のサブセットである文字セットをマッピングして（呼び出して）使用するわけであるが、そのマッピングは一般に固定的または半固定的である場合が多い。いわゆる外字処理は、処理コード空間の一部を動的にマッピング可能にしたものが多い。

例えば、日本のメインフレーム系OS富士通の日本語情報システムJEFでは、１バイト系の$(40)_{16}$～$(FF)_{16}$、及び、２バイト系の第１バイト$(41)_{16}$～$(FE)_{16}$・第２バイト$(A1)_{16}$～$(FE)_{16}$である。この２つの空間の切り替えは制御文字で行う。文字セットは１バイト系空間はEBCDICまたはEBCDIK、２バイト系空間の$(A1A1)_{16}$～$(FEFE)_{16}$にJIS X 0208を固定的にマッピング、$(80A1)_{16}$～$(A0FE)_{16}$はユーザ定義、$(41A1)_{16}$～$(7FFE)_{16}$はJEFの独自文字セットとしている。[7]

もう一つの例として、いわゆるシフトJISの処理空間は１バイト系の$(21)_{16}$～$(7E)_{16}$∪$(A0)_{16}$～$(DF)_{16}$∪$(F0)_{16}$～$(FE)_{16}$、及び、２バイト系の第１バイト$(80)_{16}$～$(9F)_{16}$∪$(E0)_{16}$～$(EF)_{16}$・第２バイト$(40)_{16}$～$(7E)_{16}$∪$(80)_{16}$～$(FC)_{16}$である。この２つの空間の識別は第１バイトで行える。したがって切り換え制御文字を必要としない。JIS X 0208は２バイトの空間にマッピングされる。

UNIXのEUC(Extende UNIX Code)のJAE (Japanese Application Environment)に使われている処理コード空間は、１バイト系で$(21)_{16}$～$(7E)_{16}$∪$(A1)_{16}$～$(FE)_{16}$、２バイト系で第１第２バイトとも$(A1)_{16}$～$(FE)_{16}$の２つの空間からなる。１バイト系の$(21)_{16}$～$(7E)_{16}$にはISO 646(ASCII)、２バイト系にはJIS X 0208がマッピングされている。１バイト系の$(A1)_{16}$～$(FE)_{16}$はシングルシフト（SS2）の後１文字のみ現れ、JIS X 0201のカナ文字セットがマッピングされている。別のシングルシフト（SS3）により一時的に「外字」セットを２バイ

ト系にマッピングすることもできる。シングルシフトの後ろを除けば１バイト系か２バイト系かは最上位ビットだけで判断できる。[8]

この方法はほぼISO 2022に準拠した方法である、制限はあるものの文字コード空間と処理コード空間との動的なマッピングを実現している例である。

EUC JAEにはもう１つの特徴がある。上に述べた処理コード空間は「ファイルコード」と呼ばれるものであり、「処理コード」と呼ばれるもう１つの処理コード空間を持っている。（まぎらわしいが！）。「処理コード」の処理コード空間は２バイトで第１第２バイトとも$(21)_{16}$～$(7E)_{16}$U$(A1)_{16}$～$(FE)_{16}$である。文字コード空間とのマッピングは固定的で、ビットパターンのみによって文字セットを識別できるようになっている。「処理コード」はプログラム中の処理に主として使われる。この方法は漢字処理を入れたＰＡＳＣＡＬ系言語ＰＬＡＫ[9]で実現された方法と同種のものである。

ソフトウェアの都合からいえば処理コード空間は文字コード空間と１：１に対応していることが望ましい。しかしながら、多くのスクリプト、多くの文字の要求に対し１：１を保つためには処理コード空間は２バイト系を超えて拡大しなければならなくなる。このことはストレージの無駄、処理効率の低下、既存ハード・ソフトとの非互換性などの欠点をもたらす。EUC JAEの「ファイルコード」と「処理コード」との分離はこの点に関しての１つの解決法として優れている。

結局のところ、多文字種の要求に対応するには文字コード空間と処理コード空間との対応を、よりダイナミックに変えられるような方法にして行く他はないのではないだろうか。

４．５　文字コード

漢字処理や多スクリプト処理が一般化する以前は処理コード空間＝文字コード空間＝スクリプト空間という認識が普通であった。したがって文字コードという言葉はこれらのどれについても使われていた。

本稿のこれらを区別した枠組みの中では、処理コード空間の文字コード、文字コード空間の中での文字コード、スクリプト空間の中での文字コードは区別されるべきである。コードは各空間の位置につけられたアドレスである。例えば、文字‘宮’の文字コードはスクリプト空間ではどのスクリプトの何番目の文字（仮に字典をスクリプトの文字表とすれば検字番号など）、文字コード空間ではJIS X 0208の21区60点、そしてEUC JAEの処理コード空間（ファイルコード）にマッピングされると$(B5DC)_{16}$であるといった言い方となる。

文字コードをこのように区別する考え方は、それらの区別のない従来の考え方にくらべて、多くの問題に対する解決を与え、また、見通しをよくする。例えば３節にあげた問題点の幾つかは、スクリプト空間の確立がされていないために起こっているといったことが見えやすくなる。また、シフトJISとEUC JAEの処理コード空間の違い、それに対する文字セットのマッピングの違いなども見通しがよくなる。これらの点からしても、本節に述べた概念は文字コード論の枠組みとして有効なものと思う。

（なお、この枠組みを数学的に形式化することは可能であるが、本稿では自然言語で述べている。そのために、なお曖昧な点もあることは付記しておく）。

５．おわりに

４節で与えた枠組みは、３節の問題点を位置づけるが、多くの点について回答は与えない。これらの未解決の問題点について解決を与えていくことが、文字コード論の今後の課題である。この意味で、本稿の表題は「試み」としている。

文字コード論の全体は情報処理の分野から、言語学それも個別の言語学（国語学、中国語学など）にわたっている。したがって、これらの分野の多くの人々が協力していかなければ、解決は得られない。これまで決して近かったとは言えない分野同士だけに、この点での困難はあろう。しかし多くの情報システムが存在し、コンピュータ間通信が一般的な道具になりつつある現在、この問題も言語学、情報学の見地から検討されなければならないことは確かである。

そしてまた、この仕事は急がなければならない。情報システムが巨大化して、柔軟性を失う前に回答を与えておくべき問題だからである。

文献

[1] Wada, Eiiti. “Three Byte Code Considered Harmful and Standardization of the Two Octet Character Sets”. Comment paper for the First International Conference on Scholarly Information Network - East Asian Applications & International Cooperation - Tokyo, December, 1987.

[2] Zhu, Yan and Chen, Yaoxing. “Chinese MARC and its Processing”. Comment paper for the First International Conference on Scholarly Information Network - East Asian Applications & International Cooperation - Tokyo, December, 1987.

[3] Yu, Kyong Hee. “Korean Character Set/Code and its Control”. Position paper for the First International Conference on Scholarly Information Network - East Asian Applications & International Cooperation - Tokyo, December, 1987.

[4] Timlake, William P. et al. “East Asian Character Coding and Administration : An RLG Review of Basic Issues”. Position paper for the First International Conference on Scholarly Information Network - East Asian Applications & International Cooperation - Tokyo, December, 1987.

[5] 坂村健．TAD言語環境と多国語対応．第３回リアルタイムアーキテクチャーTRON研究会．東京，1987.

[6] 印刷システム高度化に関する研究．日本印刷産業連合会，東京，1987.

[7] 神田泰典他．ＪＥＦ漢字コード体系．ＦＵＪＩＴＳＵ．Vol.31, No.5, p823-832(1980).

[8] 小野芳彦．ＵＮＩＸの日本語化の実現方法．情報処理．Vol.27, No.12, p1393-1400(1986)

[9] Shimasaki, Masaaki, et al. “PLAK/R : Programming Language with Abstract data type for Kanji processing and Relational database operations”. Proceedings of ICTP’83, Tokyo, October, 1983.