『広島平和科学』10 (1987) pp. 59-82 ISSN 0386-3565
Hiroshima Peace Science 10 (1987)

パーソナルコンピュータによる日本語テクストの用例検索のために：被爆手記を例として

松 尾 雅 嗣
広島大学平和科学研究センター

Towards the Personal Computer Asssisted Citation Retrieval of Japanese Texts: Hibakusha's Memoirs as an Example

Masatsugu MATSUO
Institute for Peace Science Hiroshima University

SUMMARY

The present paper is an interim report on the development of a citation retrieval system for Japanese texts on the so-called personal computer. It first critically examines the nature of the citation retrieval of Japanese texts, that is, texts containing Chinese characters. Next, on the basis of the examination, it discusses the desirable nature of the citation retrieval system and proposes a simplified test version of a citation retrieval program realized on an NEC PC-9801 machine.

目次

はじめに

文学作品，諸文書，新聞雑誌記事，論説，演説，手記，回顧録，インタビュー等の所謂テクストデータの分析に際して，コンコーダンスあるいはKWIC索引と呼ばれる，用語の用例付き索引がきわめて有用な研究の補助手段であることは多言を要しない。このような用例索引は聖書や文学研究の分野では，早くから利用されているし，コンピュータの普及とルーンによるKWIC形式の提唱（Luhn 1960）とともに，コンピュータを利用した多くの用例索引が作成され，公刊されていることも周知の事実である。このような用例索引が様々な分野の研究者に大きな福音であったことは疑うべくもない。しかし，ルーンのKWICの提唱以来4半世紀を閲した今日，その一方で，新たな隘路が出現しつつあることも看過できない。本稿は，パーソナルコンピュータ（以下，パソコンと略称）を利用することにより，このような従来の冊子体の刊本によるテクストの用例検索に代わる方法を探るものである。

1　目的と対象

議論に先立って，本稿で考察の対象とする範囲を予め限定しておく。

第一に，用例検索に使用される日本語テクストは，言うまでもなく，用例検索のみならず，テクストの語彙の計量分析，内容分析（content analysis）など多くの目的のためのデータとして利用可能である。パソコンであれ，汎用大型機であれ，用例検索システムは，このような目的をも射程に収めることが，より正確には，このようなデータ処理を含むシステムの一環として位置づけられることが，望ましいことは論を俟たないであろう。しかし，本稿では，このような分野へのデータ処理の拡張は念頭に置くにとどめ，差当り，日本語テクストの用例検索の議論に専念することとする。

第二に，純粋に国語学，言語学あるいは文学的研究を目的とする日本語テクストの用例検索システムは，最近の報告によれば，伊藤雅光による優れたシステムの開発が進行中であるが(伊藤　1987)，少なくとも筆者の研究関心に関する限り，

用例検索の目的は，むしろ，内容分析に代表されるような人文科学，社会科学的なものである。それ故，国語学あるいは言語学的な研究を目的とする用例検索システムは本稿の対象としない。

第三に，コンピュータ利用の用例検索に際しては，パソコンのみならず，大型汎用機も当然考慮に入れるべきであるが，ここでは，研究者個人あるいは小人数の研究者集団が常時利用できるという観点を重視して，汎用大型機の利用は考慮に入れない。実際，テクストが数百万字を越え，利用できる大型機の利用者サービス体制が余程よいといった場合を除き，利便性，経済性は，パソコンが優り，後述のように索引方式を用いれば，パソコンによる検索の速度も十分実用に耐える。ましてや，パソコンの主流が32ビット機へ移行しつつあり，ハードディスクの利用が一般化している今日においては，特別の事情がない限り，敢えて大型機を選ぶ理由は皆無である。

対象をこのように限定して，パソコンを利用した日本語テクストの用例検索について考えるとき，まず考慮すべき問題がふたつある。ひとつは，用例検索そのものの問題であり，他のひとつは日本語テクストの問題である。ここでは，まず用例索引そのものの問題を取り上げる。

2　日本語テクストの用例検索の方式

2．1　一括処理と選択処理

上述のように，従来のテクスト検索は，コンピュータを利用した場合でも，検索というよりむしろ一括処理により冊子体の印刷物を出力することのみを目的とするものであった。このような冊子体形式の用例索引はいまだに跡を絶たないが，二つの大きな問題を孕んでいる。

第一に，用例索引は，その定義からして，テクストに出現するすべての語彙，もしくはすべての重要な語彙についてその用例を網羅することを原則とする。それ故，必然的に大部にならざるをえない。例えば，スペバックのシェークスピアコンコーダンス（Spevack　1968－75）のように何巻かの文冊として刊行せざる

をえないのがごく普通である。たかだか数種類のテクストの用例索引を手元に置くとしても，大多数の研究者は近い将来大部な用例索引の山に埋もれることになろう。加えて，このような大部な用例索引の出版，刊行が，用例索引を作成する側でもまた参考書物として購入する側でも，経済的に非常に困難になりつつあるという事情もある。

第二に，仮にこのような，言わば非学問的な事情が改善されたとしても，従来の冊子体形式の用例索引に関しては，実はより深刻な学問的問題が残るのである。それは，用例に付される文脈の固定性，硬直性の問題である。用例索引は，個々の語ないしは句に関して，その前後の文脈を与えることを以てその存在理由とする。しかるに，刊行された用例索引における文脈はもはや利用者にとって変更不可能の所与であり，しかも，大多数の場合，出版，印刷時の経費とスペースという，およそ研究目的とは無関係な要因によって定められる。実際，用例に付される文脈は印刷の1行を越えることはまずないのである。

この点に関して言えば，ルーン以後用例索引の代名詞となった感のあるKWIC形式は，印刷の物理的1行という限られた範囲内に可能な限りの文脈を提供することと，それ以前の，コンコーダンスと称される形式に比べ，文脈中での検索対象となる用語の認識を格段に容易にした点で画期的なものであったが，今日においては，所詮過渡期の妥協の産物であったと見なすべきであろう。なぜなら，KWICの大前提は，用例の文脈を何らかの出力媒体の，例えば，プリンタやコンソールの，1行の範囲に収めることであり，このような前提は用例とその文脈をパソコンの画面に，例えば，一つの用例を表示するとき最大25行まで利用できる画面に，表示できるときには，何ら意味をもたないからである。そのうえ，KWICのもうひとつの効能である検索対象となった用語の文脈中での認識の容易さも，パソコンの場合問題ではない。カラー画面であれ，モノクロ画面であれ，問題の用語の表示を変えれば，例えば白黒反転するか別の色で表示すれば，事足りるからである。

この意味において，KWIC形式はもはや用例検索の主要形式たる地位を失ったと考えるべきである。このことは，勿論，KWIC形式がその利用価値をすべて喪失したことを意味しない。大量の用例をざっと一瞥したいときなど，限られ

た範囲内では，今日でもきわめて有用な方式であることは言うまでもない。

従来の冊子体用例索引のこのような文脈の固定性，硬直性にもかかわらず，特定の用語の用例の最適な文脈は，研究目的，研究関心により多種多様であり，到底一義的には定まらない。この点に関しては文学作品を中心に多くの議論がなされたが（Burton 1982），これらの議論から仮に結論を引き出せるとするならば，個々の用例の最適な文脈など予め定めようがないということだけである。実際，最悪の場合には，刊行された用例索引によって用語の出現箇所だけを確認し，改めて原典に当たるということも珍しくない。このような場合，用例索引は，単に用語索引（word index）としての機能を果たすのみであり，その名に値しない代物である。

冊子体形式の用例索引に関しては，このように大きな問題点に加えて，更に検索結果の再利用が難しいという難点を指摘できる。単純に引用する場合にも当然労力の問題はあるが，検索結果を対象として何らかの計量分析を行なうとき，テクストを再入力しなければならないようでは，研究の効率は大幅に低下する。

このような従来型の用例検索の問題点を克服するの唯一の現実的な方策は，従来のように用例索引を冊子体の形で一括出力するのでなく，その時々に必要な用例だけを，選択的にかつ必要な文脈を利用者の目的に応じて出力する方法を採用することである。前者は，何も目新しいことではなく，情報検索，就中科学技術文献情報検索の分野では夙に行なわれていることである。そして，このような選択的検索を可能にするのが，端末やパソコンである。また，パソコンを利用することにより，第二の問題である文脈の選択に関する自由度も格段に改善することができるし，検索結果の再利用も必要に応じディスク出力することにより，きわめて容易になる。例えば，本稿執筆に当たって筆者が行なったように，検索した用例をディスクに出力しておき，日本語ワープロで執筆している論文中に引用として取り込むことなどもごく容易である。

2．2　探索方式

このように，選択的でかつ文脈を相当程度自由に設定できる用例検索システム

をパソコン上に実現する際，検索対象となる用語をテクスト中で探索する方法に関し，理論的にはふたつの選択肢がある。

ひとつは，テクスト中で検索すべき用語（以下，検索キーと称する），具体的には単語（あるいは単語群，句，文字列等）をテクストの先頭から順に探していくという，所謂シークエンシャルサーチの方法である。これは，有名なUNIXのgrepをはじめとして，メーカー提供のプログラム，ワープロソフトやエディタの付属プログラムあるいは内蔵の一機能として非常に多くのものがある。

この方式は，次に述べる索引方式と比較して，予め検索キーを設定する必要がなく，またどのような文字列でも検索キーとすることができるという利点をもつ。しかし，これら既成のプログラムは，通常，テクストやプログラムの入力，修正の補助手段として提供されているものであり，日本語テクストの用例検索を本来の目的としたものではない。それ故，用例のテクスト中の出現位置（節，頁，行などの番号）に関する情報の出力が不十分（一般には，ファイル中の行番号だけ）であること，文脈として出力する範囲を自由に設定できないことなど，日本語テクストの用例検索システムとして見た場合，機能がきわめて限定されているという欠点がある。そのうえ，決定的に重要なことだが，検索速度が致命的に遅い。勿論，200－300Kバイトのテクストで，しかも頻繁に検索しないのであれば，このようなプログラムの中で適当なものを利用すれば当座の目的には十分である場合も決して少なくない。また，日本語テクストの用例検索に特化したプログラムを開発すれば，機能の不備は大部分補うこともできよう。しかしながら，この場合でも，検索速度が決定的に遅いという欠陥を正す余地はほとんどない。

これに対して，他のひとつの選択肢は，予めテクストを処理して検索キーの索引を作成しておき，この索引を利用して検索を行なうという方式である。この方式では，索引作成の処理が必要であるが，ひとたび索引が作成されれば，検索はきわめて高速に，ほとんど瞬時に行なうことができる。また，検索時の様々な条件も，利用者との対話方式を採用することにより自由に設定することができる。

以上の比較からして，用例検索システムとして考えるならば，索引方式を採らざるをえないことは明らかであろう。従って，本稿では，議論の対象を索引方式に限定する。但し，索引方式を採用することにより，検索キーをどのように設定

するかという新たな問題が浮かび上がってくるが，これに関しては，日本語テクスト固有の問題との関わりも大きいので後に取り上げる。

2．3　日本語テクストの問題

日本語テクストの用例検索システムを構想するに際して考慮すべき第二の点は，日本語データないしは日本語テクストの問題である。これには，多くの側面があり，ここでそのすべてを論ずることはできないが，用例検索に関わる主要な問題としては，以下のものが挙げられよう。

利用者の側の日本語入力の問題には，ふたつの局面がある。ひとつは，テクストの作成という局面であり，他のひとつは検索時の検索対象となる用語の入力である。テクスト作成に関しては，利用者が自身で入力しないときには問題は生じないし，利用者が入力する際には，適当な日本語ワープロソフトあるいはフロントエンドプロセッサを利用すればよいので，特に問題はないものとして，ここでは触れない。後に示す試験的検索システムで利用できる条件は，テクストがMS－DOSのテクストファイルでなければならないということだけである。

これに対して，何らかのフロントエンドプロセッサは，検索時に検索する用語を入力するために不可欠である。そして，検索システムは，利用可能なフロントエンドプロセッサに関しては，特別の制約を課さないことが望ましい。周知のように，現行のフロントエンドプロセッサは仮名漢字変換システムと辞書から成っており，検索時の入力量はそう多くはないことからして，変換システムに関しては特に問題はないと言ってよい。他方，辞書についてはふたつの問題がある。ひとつは辞書の容量の問題であり，他のひとつは内容の問題である。現行のフロントエンドプロセッサの辞書は数百Kバイトのものが普通であり，フロッピィディスクで用例検索システムと併用する場合，扱えるテクストの長さを著しく制約する。それ故，日本語テクストの用例検索システムでは，RAMディスクやハードディスクの利用を前提とすることになろう。また，このような辞書は本来ビジネス向けであり，我々が研究対象とする多様なテクストの処理に決して便利なものではない。しかし，どれを使用するにせよ，日本語テクストの用例検索のために

は，「広辞苑」並みとまではいかなくても，せめて第２水準漢字や熟語の変換が容易なフロントエンドプロセッサが望ましい。

3　用例検索システムの構造

3．1　検索キーの設定

日本語テクストの用例検索システムを上述のような索引方式で作成する際には，索引作成時に大きな問題が生ずる。用例検索システムである限り，索引により検索可能な単語あるいは検索キーは可能な限り網羅的であることが要請される。このとき，日本語テクストから，どのようにして検索キーとなる単語を決定するかという問題が生ずる。単語以外の要素を主たる検索のキーとするとしても，問題の本質は変わらない。例えば英語の場合を考えれば容易に想像できるように，ヨーロッパ系の言語のテクストの場合，複合語や分離綴りといった問題がないわけではないが，基本的には単語の分離はごく容易であり，単語単位で網羅的な索引を作成することはさして困難ではない。これに対して，日本語を始めとする単語を分ち書きしない言語では，コンピュータに単語あるいは検索のキーとなる要素を分離，抽出させるためには複雑なアルゴリズムを用意する必要がある。実際，粗野なものは別としてこのようなアルゴリズムはまだ確立されていない。その意味で，現実の日本語テクストを入力して，自動的に検索キーを決定させることは現状では不可能に近いと言えよう。

これに対処する方法は幾つか考えられる。第一は，辞書方式とも称すべき方法である。これは，用語のリストや，文献情報検索の分野でシソーラスと呼ばれている同義，包含などの階層関係を有する用語のリストを予め作成しておき，リストに出現する用語に関してのみ索引を作成するという方法である。この方法は，問題のリストが既に作成されている場合には，非常に効率のよい方法であるが，そうでないときには，非常な労力を要する。一般的には，個々の研究者の時々の研究関心を十全に満足させる辞書は存在しないと考えておく方が安全であろう。

第二は，文字キー方式と呼ぶべき方法であり，用語ではなく，文字単位で索引

を作成し，検索時には与えられた単語や用語を構成する文字とその索引を利用するという方法である。この方法は索引作成が容易であり，テクスト中の如何なる文字列でも検索可能であるという利点をもつ一方，索引が異様に肥大化するという欠陥を有する。例えば，テクストが漢詩あるいは漢文のみから成る場合には，これが最も有効な方法であろうが，一般のテクストでは，そのままでは索引部分の無駄が大き過ぎよう。

第三は，利用者指定方式とも称すべき方法である。検索システムは検索キーの内容には原則的に関与しないという方法である。システムは，利用者がテクスト中に何らかの形で指定した文字列，あるいは対話形式で指定した文字列を検索キーと見なしそれについてのみ索引を作成するというのがこれである。この方式では，プログラムによる検索キーの抽出という難問は回避できるが，キーの網羅性，首尾一貫性はすべて利用者の責任であり，その意味で利用者に過大な負担を強いるものと言わねばなるまい。

以上幾つかの可能性を挙げたが，いずれも一長一短があり，この段階でひとつを選ぶことは不可能である。実際，現実に何ほどかの一般性を有する用例検索システムを構築するとすれば，上記３方式のすべてをなんらかの形で併用するものとなろう。しかし，本稿では，既成辞書の無いことと，次に議論する検索キーの階層化を考慮に入れ，利用者指定方式を検討の対象とする。

３．２　検索キーの階層化とコーディング

辞書方式であれ，利用者指定方式であれ，なんらかの検索キーを設定し，検索を行なおうとするとき，本稿でテストデータとした被爆手記におけるように「わからない」，「分らない」，「解らない」，「判らない」といったキーを一括して検索する必要は常に生ずる。日本語の用言の活用形であれば，語幹のみを検索キーとすることで，問題はある程度解決できようが，このような多様性に関しては，何らかの対策が必要である。他方，「日米安全保障条約」，「日米安保」，「安保」といったキーを一つのグループとして扱う必要も生ずるであろう。このような状況に対処するため，検索対象とするテクスト中で問題の語句を，予め，「判らない」や「日

米安全保障条約」に変えておくことも考えられないわけではない。しかし，この方法では，検索結果を見て，実際に使用されている語句をもう一度原典で確認しなければならないうえ，「日米安全保障条約」を「防衛問題」，「日米関係」，「平和問題」といった異なったキーとともに扱うことがきわめて困難になる。

ここには，「わからない」の例や「國民」と「国民」の例に見られる表記の多様性と，「日米安全保障条約」，「防衛問題」の例に見られる概念カテゴリーと用語の一対多対応というふたつの異質な問題が含まれている。後者は，内容分析の観点からすれば，所謂コーディングの問題である。しかし，内容分析に限らず，一般の用例検索においても，表記であれ意味であれ，何らかの関連を有する用語をグルーピングする必要は常に生ずるものであり，用例検索システムはこのような要請に十分応えられるものでなければならない。

このような要請を満たすものとしてひとつの示唆を与えるのは，内容分析を目的として開発された General Inquirer System（GI と略称）(Arp et al, 1970) である。GI は，上述の辞書方式と利用者指定方式を併用してテクスト中の語句をキー（GI ではタグと呼ばれる）に変換し，キーを利用，操作して内容分析的処理を行なう。GI では，個々のキーに対応する原文は，当然のことながらキーを抽出，設定するための素材に過ぎない。これに対し，用例検索では，原文における形や語句を無視することは，用例検索という定義そのものに反する。用例検索では，テクストに出現する実際の形，具体的には，文字や語幹や単語や句をキーとして無視することは不可能である。と同時に，上述のように，キーのグルーピング，コーディングの要請も満たさねばならない。この問題を解決するには，検索キーを一次元的に捉えるのではなく，検索キーに複数のレベルあるいは階層を設ける必要が生ずる。それ故，上述の問題を解決するため，検索キーに以下の階層を定める。

まず，テクストに実際に出現する形を，第１次キーあるいはタイプとする。上例で言えば，「国民」，「國民」，「わからない」，「判らない」，「日米安全保障条約」，「日米安保」等は，当該のテクストに出現する限りにおいて，すべて第１次キーである。これに対し，任意の第１次キーの集合を第２次キーもしくは GI に倣ってタグとする。第２次キーは特定の第１次キーと同一の形であってもよい。例え

ば,「判らない」という第2次キーは4個の第1次キー「わからない」,「分らない」,「解らない」,「判らない」の集合であってもよい。この段階では，第1次キーはＧＩにおけると同様ただひとつの第2次キーに属するものとする。それ故，第1次キーと第2次キーの関係は常に静的であり，動的なグルーピング，コーディングを可能にするために，任意の第1次キーと第2次キーの集合として第3次キーなるレベルを導入する必要がある。任意の第1次キー，第2次キーは，複数の第3次キーの要素となることが可能である。

実用に耐える用例検索システムを構築するには，検索キーの問題の他に，考慮すべき多くの問題があるが，ここでは，そのうちひとつだけを取り上げる。それは，検索された用例に付ける索引項目の処理である。出力された用例が元のテクストの何処に出現しているかを示す索引項目がなければ，用例索引の利用価値は半減するといっても過言ではない。それ故，用例検索システムでは，多様で柔軟な索引項目の出力ができるよう十分な配慮が必要である。用例検索システムは，少なくとも次のような処理能力を有する必要があろう。

複数の，可能な限り多くの索引項目を利用できること。
索引項目の入力あるいは作成に関して利用者の負担を最小限にとどめること。
（テクストの各行毎に，巻，章，頁，行，参照番号などをすべて付けることを要求するようなシステムは問題外である。）
実際の出力に付ける索引項目を取捨でき，出力順序も指定できること。

このような要請を満たすにはどのような方法が可能であるかについては，次節で報告する試作版がその解決のひとつの試みである。

索引項目の他，考慮すべき問題は多々あるが，ここでは，割愛する。

4　日本語用例検索システムの試作版

4．1　試作版の実行環境

前2節の検討にもとづいて日本語テクスト用例検索システムを試作した。ここでは,前2節の結論をどのように実現したかを中心にシステムの概略を報告する。まず，実行環境と機器構成は以下のとおりである。

OS　　：　MS−DOS
ソースプログラム：　Lattice C j 3.0 D モデル
サンプルテクスト：　広島市民政局社会教育課編（1950)「原爆体験記」(廣島平和協会発行）の1−72頁（但し，本文中の「ぬきがき」欄と「被爆状況」欄は除く）

機器構成
本体　　：　NEC PC−9801 F2　メモリ　640K
周辺機器：　ハードディスク，CRT，プリンタ
（周辺機器の内ハードディスクは必須ではないが，日本語処理とデータ容量を考慮するならば，フロッピィディスクより望ましい）

4．2　テクスト処理の手順

テクスト処理は次の4ステップに分割した。
ステップ1：テクスト作成。
ステップ2：テクストの定義
ステップ3：第1次キーの定義と索引作成
ステップ4：第2次キーの決定

ステップ1のテクスト作成の実際には，システムは直接関与しないが，作成されるテクストに関し，以下の制約が課される。

第一に，テクストはMS－DOSのテクストファイルでなくてはならない。このテクストファイルの各行が，検索結果出力の１行として扱われる。本稿のサンプルデータでは，１文をテクストファイル１行とした。それ故，用例は元のテクストの文単位で出力される。また，このファイルの１行の長さは1024バイト（漢字にして512文字）を越えてはならない。なお，テクストの作成に関しては，元のテクストを幾つかのファイルに分割して作成してもよい。このときには，ファイルの連結処理などを行う必要もない。

第二に，入力テクスト中には，索引項目の値，つまり，用例に付けるテクスト中での位置を示す何らかの値（以下，これを識別値と称する）を与えなければならない。本稿で取り上げた被爆手記集を例に取れば，手記の表題，頁番号などがこれに当たる。これは，別の観点からすれば，テクストがどのような構成要素から成るかを決定し，構成要素の個々の現れに何らかの値を与えることに他ならない。ここで，テクストの構成要素を，テクストユニットと呼ぶことにすると，現在の試作版では，テクストユニットに関し，以下の定義を与えている。

テクストユニットはすべて連続した入力行を構成要素とする。
最大４種類までテクストユニットを定義できる。（例，巻，章，節,頁等）
テクストユニット関の階層関係は任意的である。
テクストは，論理的には，任意のテクストユニットの連鎖として定義される。
テクストユニットには，名称を与えなければならない。（例，頁）

利用者は，設定されたすべてのテクストユニットの，原則としてすべての現れについて，その境界を示し，それを識別する値もしくは名称を，テクスト中に与えなければならない。例えば，本稿のサンプルテクストは，幾つかの手記から成っているが，手記の境界と名称を与えることになる。このようなテクストユニットの境界と識別値は，概略以下の規則に従ってテクストファイル中に与える。

テクストユニットは，固有の識別記号，即ち固有の半角記号によって識別される。（巻の識別記号は“＊”，章は“＃”等）この記号は，利用者が自由に決定し

てよい。

識別記号は，テクストユニットの個々の現れの境界を示すと同時に，これに続く文字や数値がテクストそのものではなく，テクストの特定の部分を識別するための参照用の要素，即ち識別値であることを示す。

識別値は識別記号に空白を置かずに続ける。

システムではすべての行に識別値を付けるが，利用者はすべての行に識別値を与える必要はない。テクストユニットの境界部分にのみ，別の言い方をすれば，識別値の変更があるときにのみ，例えば，新たな手記が始まるときや頁が変わるときにだけ，与えればよい。

テクスト中では，一種類のテクストユニットに関して，例えば，頁内での行番号に関して，同一の値が何度与えられてもよい。

テクストユニットは，次の３つのうちいずれかのモード（識別モードと称する）を有する。かつ，このモードは，各テクストユニットごとに利用者によって決定され，以後変更できない。

行モード：行が変わる毎に番号を１増加する。このモードでは，テクストの各行に識別値を与える必要はない。テクストの最初に初期値を与え，番号が不連続になる場合にのみ，識別記号に続けて新たな番号を与えればよい。

文字モード：識別値は，数値として解釈しない。このモードでは識別記号の後ろに何も与えない形，即ち識別値がないという指定も可能である。

数値モード：識別記号に続けて半角の数値を与える。数値が与えられていないときには，直前の値を１増やす。（従って，頁番号が１ずつ増えるときには，識

別記号だけ与えればよい。)

テクストユニットの識別記号と識別値は組は，当該のユニットが始まる入力行より前の入力行になければならない。このことは，必ずしも，識別記号と識別値の組がテクスト本文と別の行になくてはならないということを意味しないが，ここでは単純化して，識別記号と識別値の組は，テクスト本文とは別の行にあるものとする。

一つの入力行には，ひとつもしくは複数のテクストユニットの識別記号と識別値の組を与えてもよい。また，そのとき与える順序は任意である。

このように規則という形式で表現すると煩雑に見えるが，実際には，次の例1に見られるようにむしろ単純であるし，索引項目なり識別値を与える際の利用者の労力と負担を大幅に軽減するものである。

以上の規則にしたがって，識別記号と識別値を入力ファイル中に与えた実例を例1に示す。これは，サンプルテクストの冒頭の部分である。

＜例1＞　テクストファイルの例

(　"¥¥"の記号は行末を示すため，筆者が挿入した。また，／＊　＊／の部分は本来のファイルにはなく，この例の説明のために筆者が付け加えた説明である。)

```
＊爆心にあびる　　／＊　テクストユニット「手記表題」の識別記号と識別値　＊／
＃野村英三　　／＊　テクストユニット「著者」の識別記号と識別値　＊／
％1　　／＊　テクストユニット「頁」の識別記号と識別値頁　＊／
　　／＊　値を1から始める　＊／
　廣島市中島本町，丁度元安橋南詰に現在燃料會舘がある。¥¥
當時廣島縣燃料配給統制組合の本部であった。¥¥
この建物は地上三階地下一階で鐵骨鐵筋コンクリ―ト建ての丈夫なもので爆心点から西南約百メ―トルに位置している。¥¥
　組合は當時毎朝八時に全員を二階に集めて國民儀禮をするのが例であった。¥¥
その朝も河合業務部長の音頭で濟まし全出勤者三十七名は各階各自の机にかえって仕事前の一服をやっていた。¥¥
```

さて「仕事」だと私は机上を見たところいつもの書類がまだ置いてない。¥¥
いつも課長が地下室から持ってくるのを今朝にかぎって忘れていたのだった。¥¥
そこで自分の隣の廣瀬女事務員に取りに行って貰うつもりでその方を見たら忙がしそうにしていたので私は二階を下りて地下室へ行った。¥¥
　下りる前に私はめがねを外し財布をズボンのポケットから出しそしてズボンのバンドに巻いてある鎖を解いて懐中時計を出し机上にこの三点を揃えて地下室へ下りて行った。¥¥
この品は勿論みな焼いてしまったが何故そんなことをしたのかは五年後の今日どうしてもわからない。¥¥
%　　／＊　新しい頁　1増えるから識別値は不要　＊／
　地下室は建物の三分の一位の廣さで十坪餘りの狭まいもので常に電灯が灯してある。¥¥
書類が見あたらないのであちこち探して階段下の倉庫の處へ來た。¥¥
その時だった。ド—ンという可なり大きな音が聞えた。¥¥
とたんにパッと電灯が消え真暗になった。¥¥
同時に二三ケ所，固い小石の切片のようなものが當った。¥¥
おかしい！兩手でそっとさはってみた。¥¥
　痛い！と手を頭にやってみたらねっとりしたものが流れている，血だ！何んだろう？¥¥
何事が起こったのだ？ 暫くして解らないまゝ頭の外にどこか傷をしてはいないかと上半身，兩腕，兩足その他を調べてみたが別に異常はないらしい。¥¥
室内は真暗がりで何にも見えぬ。¥¥
　私は階段の直ぐ下に立っていた。¥¥
上ろうと思って足を階段にかけた。¥¥

ステップ2はごく短いステップであり，検索対象となるテクストの名称，略称，テクストユニットの名称，識別記号，識別モード等を対話方式で定義し，システムに知らせる。

ステップ3は，テクストを読み込んで，テクスト中の第1次キーを決定しながら，索引を作成する。このときの処理手順は，次のようになる。まず，テクストの1行を読み込み，それを画面に表示する。そして，この行中に既に登録されている第1次キーに一致する文字列があれば，その部分を反転表示し，先頭の1文字は赤で示す。既存のキーとの一致は，最長一致による。この状態で，利用者は以下の処理が可能である。

新たにキーとして登録すべき単語，文字列があれば，カーソルを移動してその

先頭と終端を指定する。このとき新たに指定された部分は先頭の１文字を赤とする反転表示に変わる。

キーとされている単語，文字列を，即ち反転表示されている単語，文字列をキーとして扱うことをやめる。このとき，その部分は通常の白色表示となる。

表示されている行のキー登録を最初からやり直す。

表示されている行に関するキー登録を終了する。

キー登録を終了するという指示が与えられると，利用者に次の行を処理するかを問う。利用者はいつでもノーと応答できる。ノーの応答があったときには，必要な情報をすべて書き出した後，プログラムを終了する。次にこのステップを再開したときには，プログラムは自動的にそれ以前に処理の終わった行の次の行からキー登録処理を始める。

ステップ４では，ステップ３で決定された第１次キーにもとづいて第２次キーを決定する。この部分のプログラムは未完成であり，詳細は，未定である。現在の試作版では，このステップをバイパスしても，検索自体は可能であるし，第１次キーにもとづくキーのグルーピングもある程度まで可能である。なお，前節で提案した第３次キーについては，現在のところ検索時の機能として実現する予定である。

４．３　コマンドとオプション

検索は，検索プログラムを起動し，必要なファイルを読み込んだ後可能となる。起動とファイルの入力にはさして時間を要しない。検索はメニュー方式でなくコマンド方式とした。検索のためのコマンドは現在以下のとおりである。

C　用例検索

K　出力をKWIC形式とする用例検索

Q　検索の終了

このほか，ヘルプ，キーの一覧，第3次キー作成，マクロ文字列の定義，画面表示パラメータの設定のためのコマンドを追加予定である。プログラムは，ひとつのコマンドの処理が正常終了すると再びコマンドの入力を待ち，Qコマンドが入力されると終了する。実行中に何らかのエラーが検出されたときには，まずエラーメッセージを表示し，エラーが回復可能であれば，実行を継続する。回復不可能であれば，その旨のメッセージを出して異常終了する。

コマンドは，「コマンド=」というプロンプトが表示されている状態で，半角で入力する。大文字，小文字いずれでもよい。Qコマンドを除き，各コマンドには，MS−DOSのコマンドの場合と同様，定められたオプションをコマンドラインに与えることができる。オプションは半角の“−”に続く半角のアルファベット1文字で指定する。これは，大文字でも，小文字でもよい。コマンドとオプション，オプション間の区切りは半角の空白である。オプションの解釈に関しては，オプションが与えられていないとき，自動的にシステム既定値を使用する方法と，利用者に問い合わせる方法を随時切り替えることができる。また，マクロ文字列を定義し，コマンドラインに与えることもできる。マクロ文字列は，接頭記号（半角の“#”）で始まること以外，事実上の制限はなく，入れ子も可能である。勿論，漢字で定義することも可能である。これにより，コマンドやオプションの入力は格段に容易になる。

4．4　検索の実際

Cコマンドを例として，以下，検索の実際を紹介する。まず，このコマンドには，現在，次のオプションがある。

−S　　検索キーの候補の選択。

デフォルトは入力された文字列に一致するキーすべての用例を出力。

−T　タグを指定した検索。デフォルトは，第1次キーの検索

−L　複数の文字列（文字列のリスト）の入力による検索。
入力媒体はキーボードまたはディスクファイル。
デフォルトは1個の文字列の入力。

−C　用例の出現する行の前後に文脈として前後の行を出力。
このオプションに続けて数値を入力すればそれを出力行数とする。
数値が与えられていなければ，1と仮定する。
デフォルトは0，即ち用例の出現する行のみ出力。

−R　複数行出力の際，行を追い込み出力。
デフォルトは，1行毎に改行。

−D　追込み出力の際，行の区切りを示す。デフォルトは，示さない。

−O　出力媒体。デフォルトはコンソール。
Pが続けばプリンタ。
Dまたは何も続かないときには，ディスク。
Dに続けてファイル名を与えれば，それを出力ファイルとする。
Dに続けてファイル名がなければ，システムの設定した標準出力ファイルに出力する。

−M　与えられた文字列に一致するキーが複数個ある時すべてのキーのすべての用例をテクストでの出現順に並べ替えて出力。
デフォルトは，キーの順にキー毎に出力。

デフォルトは，オプションが与えられなかったときの処理を示すものであるが，前述のようにオプション処理のモードを切り替えておけば，その都度処理の詳細を問い合わせる。

このコマンドは，ひとつまたは複数の文字列を与えてそれに一致するキーの用例を検索し出力する。文字列にはMS-DOSのワイルドカード類似のワイルドカード，任意のゼロ文字以上の文字列を示す半角のアステリスク“*”を使用できる。文字列をひとつだけ与えるときには，コマンドに続くコマンドラインの任

意の位置に与える。複数個与える場合には，Lオプションを指定し，入力を促すプロンプトが出てからリストを入力する。この文字列は，第1次キーでも第2次キーであってもよい。第2次キーのときには，予めコマンドラインにTオプションを与える。入力された文字列と一致するキーが複数個ある場合，一致するキー画面に表示してそののうち必要なもののみを選ぶこともできる。更に，キーが第2次キーであるときには，その要素のうちから適当なものだけを，同様に選択することができる。このような機能は，第3次キーを一時的に設定するものに他ならない。また，以上の説明から明かなように，すべての検索を第1次キーに還元して行なうものである。

4．5　検索結果の出力

検索された結果は，指定された媒体に出力される。コンソール以外の出力は一括出力である。ここでは，一致した1次キーが1個の場合，あるいは複数個でも，Mオプションにより一括処理する場合のコンソール出力の実例をみてみよう。

まず，画面には，テクスト中での最初の用例が表示される。用例の表示は，例2に示すように，その出現個所を示す識別値（手記名，頁等），用例とその文脈の順で行なわれる。このとき，キー自体は，地の文とは異なった状態，例えば，黄色の反転状態，で表示される。(例2では，網をかけて示してある)。利用者は，ここで次のいずれかの処理を選択することができる。

（あれば）次の用例の表示
（あれば）直前の用例の表示
テクスト中の最初の用例の表示
テクスト中での最後の用例の表示
表示中の用例の印刷
表示中の用例のディスク出力（出力先は，標準出力ファイル）
表示の終了とコマンド入力待ち状態への復帰

＜例２＞　検索結果の画面出力例（「火」の用例のひとつ）

爆心にあびる　野村英三　5

脱出して救援隊に知らせて來て呉れないかと会計の宍戸君がいった。　／　行手の模様が全然判らないこの炎の中をくぐることは死を意味する。　／　出るからには再びここへは歸られないことを覺悟しなければといって，大型の焼トタン板を一枚手にして再び相生橋に立ってどの方面に救援を求めに行かうかと見渡した。

ここに挙げた選択肢を組み合わせれば，１回の検索処理において，用例を十分に検討できるはずである。これに付け加えるとすれば，検索されたすべての用例を対象として，任意の文字列を検索するという，一種の絞り込み検索の機能，特定の用例に直接ジャンプする機能，用例のマーク機能，第３次キーとしての登録機能などであろう。

４．６　アルゴリズム

以上は，利用者の側から見たプログラムの動きであるが，ここで，プログラムがどのように検索を行なっているかを，アルゴリズムというと些か大げさだが，Ｃコマンドを例として簡単に述べておく。なお，以下の処理手順は，本質的には他のプログラムで筆者が用いたものと同一である。詳細は松尾・鈴木（1987）を参照されたい。

プログラムは，まず，コマンドとオプションを処理する。このときキーボードからの入力が必要であれば，適当なプロンプトを出して，入力を促す。

次に，与えられた文字列が，何個であれ，それと一致するキーが存在するかどうかを調べる。第２次キーの場合処理が２段階になるので，ここでは，与えられた文字列が第１次キーとであるとしておく。第１次キーはソートされて，ファイルに記憶され，プログラム起動時にメモリに読み込まれているので，これをサーチすることになる。与えられた文字列がワイルドカードを含まなければ，バイナリサーチを行なう。ワイルドカードを含み，一致するキーがあるときには，利用者がキーの選択を指定していれば（Ｓオプションが与えられていれば），一致し

たキーを画面に表示して利用者の選択を待つ。このようにして与えられたすべの文字列についてサーチが完了した時点で，用例を表示すべき第1次キーが確定する。このとき，条件を満たすキーがひとつもなければ，勿論，処理はここで終了し，エラーメッセージを出して次のコマンド入力を待つ。

次に，確定したキーのすべてについて，(第1次) キー情報ファイルから，キー固有の情報を読み込む。具体的には，キーのテクストでの出現度数，キー索引ファイル中の出現位置リストの開始位置などである。キー索引ファイルには，キーの個々の用例が出現する行番号と行中での開始カラム位置から成る索引が，キー毎にまとめて連続して記憶されている。従って，キー索引ファイルでの索引開始位置と出現度数が得られれば，索引を読み込むことができる。ひとつのキーについては，行番号と開始カラム位置から成る索引はテクストでの出現順に配置されている。このようにして，キー固有の情報を読み込み，それにもとづいて用例の索引を読み込んだとき，用例をキー毎に表示せず，すべてのキーのすべての用例をテクストでの出現順に並べ替えるという指定があれば，行番号と開始カラム位置のデータを昇順にソートする。

次の段階では，読み込まれた出現位置を示す索引のそれぞれについて，そのうちの行番号にもとづいて行情報ファイルから行固有の情報を読み込む。行情報ファイルは，テクストの各行に固有の情報を記憶したテーブルであり，すべての行について，長さ，識別値，行ファイルでの開始位置が記憶されている。問題の行の情報が得られ，行ファイルの開始位置から長さ分読み込めば，問題の行そのものが得られる。文脈として前後の行が必要な場合も，同様である。これが終われば，用例の出現する行そのもの，行中での位置，その行のテクスト中での位置を示す頁番号などの識別値等必要な情報はすべて読み込まれたことになる。後は，利用者の指示に従ってこれらの情報を順次出力すればよい。

5 あとがき

前節で報告した試作版は，あくまで試作版に過ぎず大幅な機能の拡張と洗練が必要であることは言うまでもない。機能の拡張としては，内容分析的な計量処理

機能を付加するという問題もあるが，用例検索に限っても早急に実現すべき機能は多い。ひとつは，ふたつのキーあるいはキーの集合の共出現の検索である。被爆手記を例に取れば，「悔恨」，「罪の意識」といった意味のキーの集合と「家族，身内の死」といった意味のキーの集合とがともに出現する用例を検索できるようにすることである。また，検索の補助手段として，どのようなキーがあるかを，画面表示により直ちに知ることができる機能も必要であろう。利用者の便を考えれば，必要に応じ，コマンドの使用法等を画面で確認できる，所謂マニュアルレスな利用法も配慮する必要がある。そのほか，検索範囲の限定，テクストの切り替え機能等多くの改良，拡張が考えられる。そして現在の16ビット機に代わり32ビット機が主流となれば，更に多くの機能を追加することも可能であろう。

ということは，むしろ現在の試作版は，本来備えるべき機能のごく一部しか実現していないということに他ならない。このような，未完成のシステムを公表するのは，知的怠慢の譏りを免れまいが，敢えてこのような試みを行なったのは，時代錯誤的とも筆者には思われる冊子体の刊本偏重の風潮に一石を投じ，パソコンによる用例検索の有効性を明かにせんがためである。

引用文献

Arp, Dennis J. et al (1970) "An Introduction to the Inquirer II System of Content Analysis", *Computer Studies in the Humanities and Verbal Behavior*, Vol.3, No.1, pp. 40–45

Burton, Dolores M. (1982) "Automated Concordances and Word-Indexes: Machine Decisions and Editorial Revisions", *Computers and the Humanities*, vo.16, no.$, pp. 194–218

伊藤雅光 (1987) "パソコンによる漢字仮名混じり文用KWIC索引作成システム－「こもれび」－", 計量国語学, Vol. 16, No. 3, pp. 113－127

Luhn, H. P. (1960) "Keyword-in-Context Index for Technical Literature (KWIC index)", *American Documentation* 11 (4) 288–95. Reprinted in Hays, David G. (ed) (1966) *Readings in Automatic Language Processing*, New York, American Elsevier

松尾雅嗣・鈴木重樹 (1987) "パソコン用簡易用例検索プログラム：PC－KWIC", 計量国語学, Vol. 15, No. 4, pp. 130－139

Spevack, Marvin (1968–75) *A Complete and Systematic Concordance to the Works of Shakespeare*, Hilbesheim, Georg Olms Verlag