半構造資料部分検索のための演算法の一考察

胡　　進† 清光　英成†† 大月　一弘†† 森下　淳也††

† 神戸大学大学院総合人間科学研究科　〒 657-8501 兵庫県神戸市灘区鶴甲 1-2-1
†† 神戸大学国際文化学部　〒 657-8501 兵庫県神戸市灘区鶴甲 1-2-1
E-mail: †058f007f@y05.kobe-u.ac.jp, ††{kiyomitu, ohtsuki, jm}@kobe-u.ac.jp

あらまし　さまざまな様式の資料を電子化して蓄積する事業が盛んに行われ，デジタルアーカイブなどに蓄積された資料を効率的に検索する要求が高まっている．従来の資料検索は検索結果の単位が資料であった．しかしながら，ユーザが必要とするのは資料中のある部分のみであることが考えられるため，本研究は，緩く構造を与えられた資料中からユーザが必要とするであろう部分のみを検索結果として提供することを目的とする．特に，複数のキーワードに対して，その間の関連を簡単に指定したときに有効な検索結果を求める関係と演算とを整理する．さらに，検索式を解釈する方法を議論する．

Operations for Retrieving a Potion from Semi-Structured Resources

Jin HU†, Hidenari KIYOMITSU††, Kazuhiro OHTSUKI††, and Junya MORISHITA††

† Graduate School of Human Science and Cultural Studies,Kobe University
†† Faculty of Cross-Cultural Studies,Kobe University
E-mail: †058f007f@y05.kobe-u.ac.jp, ††{kiyomitu, ohtsuki, jm}@kobe-u.ac.jp

Abstract　Digital Archive is a collection of documents that contains books, serials, extracts from newspapers, magazines, journals, leaflets and so on. Recently, it begins to collect non-paper media that are continuous media data such that audio, video and their compositions. Our major objective is: retrieve a sub-resource from the archives and provide a view of archives to our users. In this sense, we divide a document to some sub-resources in its logical structure, and give some meta data to each sub-resource. Here, we have to resolve a problem that corresponding meta data are scattered around sub-resources in a document. In this paper, we propose augmented AND operations for providing an effective method to retrieve a sub-resource from archives.

1. はじめに

近年，計算機技術や通信技術の長足の進歩により，大量の文書・静止画・映像などを高品質で劣化しないデジタル情報として保存することが可能になっている．これに伴い，歴史・文化資産をデジタル化し保存・蓄積・活用するデジタルアーカイブへの注目が高まっている．

デジタルアーカイブに格納される資料は，あるテーマに関するあらゆるものを網羅的に収集するためその形態が多様性に富んでいる．また，ひとつの資料の中に個別の資料として扱うことの可能な構成要素を重層的に含むような複合型資料が多いという特徴がある．そこで，我々はこのような不定形複合型コンテンツに適したデータ格納手法ならびにその検索手法の開発を行ってきた[1]〜[4].

図版や映像などは，デジタル化したビットストリームから意味を抽出することが容易ではない．各資料に一つ以上のメタデータツリーを作成し，オリジナルな資料の全文検索ではなくメタデータツリーの各ノードが保有するキーワードを検索の対象とする．本方式で扱おうとしているデータは，

- 資料媒体・形態が多様で映像・音声等のマルチメディア資料はもちろん，紙媒体資料も書籍，シリーズもの，広報紙・チラシ・レジュメなど著しく量が異なる資料を同時に含む
- 一部分のみが収集対象である場合，また抜刷・切抜等で収集する場合など，資料となる単位が一定しない
- 記録を目的としない生の資料や，ミニコミ出版物などを十分な量含み，内容・編集の側面でも多様性に富んでいる

という特徴を持つ．このような多様な資料からなるアーカイブに対しては，利用者の要求が様々である．まず求められることは，資料の種類をその都度意識せずとも統一的に検索ができることである．資料媒体に関わりなく，あるテーマに関する情

報を横断的に検索できなくてはならない．テーマの広狭，必要とされる網羅性の度合いなど，それぞれの要求に対応できることも重要である．特に留意すべきは，様々な粒度の情報が求められるということである．利用者があるテーマの情報を網羅的に得たい場合，またごく狭いテーマの情報を得たい場合には，図書の書名など「資料」として物理的に独立した情報単位だけではなく，記事・論文・章・節といった構成要素の単位まで検索が行えなければ，満足な結果は得られない．場合によっては，広報紙上の小さなコラムや，1 枚の写真・図表などで，利用者の問題解決が果たされることもある．内容全文がテキストデータ化されている資料もあるが，検索の対象は入力されたメタデータである．ある視点に特化して収集される資料アーカイブは，学術論文など一定の均質性を備えた資料群のみを扱うのとは異なり，全文検索で実用的な精度を得ることは困難であると考えたからである．

このようなアーカイブに蓄積される資料は，様々な要素を重層的に含んでいる．例えば，記事，章・節，写真，図などで構成される資料は情報単位となり得る構成要素を重層的に包含して成り立っている．そこで，各構成要素を独立した情報単位として取り扱い，この情報単位を検索・操作の対象とすることによって，資料中の該当する部分を検索結果として提示する．以下，独立した情報単位として取り扱える構成要素を部分資料と呼ぶ．

グラフの構造を利用して資料の部分を検索する研究はいくつか報告されている．田島らは Web 情報に対する検索手法を提案している [5]．Web のリンクをグラフと捉え，そのネットワーク構造の中からある情報単位を抽出するが，情報間の距離を基に検索ノイズを除去しようとしている．絹谷らは XML 文書の構造を解析することで検索単位となりえる部分文書を決定し，それらを検索単位として適合する部分文書を検索結果としようとしている [6]．

我々は結果となる情報単位を，該当するキーワードを保有するメタデータツリーを素に部分資料として得ようとしている．つまり，オリジナルのデータにまったく触れずに外部注釈として用意されるメタデータを対象として検索を行おうとしている．さらに，従来の検索機構で複数の検索キーワードを指定したときの解釈は AND,OR であったが，特に AND を細分化してユーザが意図するキーワード間の関係を簡単に指定できる方法を議論しようとしている．

2. データの構造

様式がまちまちなデータを予め整理して構造を与えることは困難である．また，そのようなデータの構造を機械的に把握することは容易でない．このため，各資料の二次情報として資料を任意の部分に分割してメタデータを与え，部分間の階層関係に基づいて構造を与えることを想定している．例えば，

- 文書資料をその論理構造で章・節・項などの部分に分割して木構造を構成 (図 1)
- ビデオデータをカット・ショット・シーンなどの部分に分割して木構造を構成，あるいは，時間的・空間的に重なりを許し構造をとらえて木構造を構成
- 画像データを領域・副領域などの部分に分割して木構造を構成

し，各ノードにノードを代表するキーワードなどのメタデータを与えることなどが考えられる．

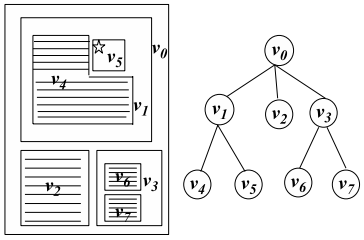

図 1　文書の構造例
Fig. 1　An example structure of a document

報告書や論文の類はある程度様式が統一されているので論理構造に基づいて分割することが妥当であるが，芸術作品や整理されていない記録など解釈の方法を複数許すような資料は見方が定まるまでの期間複数種類の構造を許す必要がある．また，どのように区切っても情報単位となり得る連続メディアデータを時間的あるいは空間的領域の重なりを排して表現することは厳密すぎるので，重なりを許したい．このため，一次情報の外部にメタデータを記述するとともに，一資料一構造木という構造記述よりも，一解釈一構造木というアプローチを採用している．つまり，解釈の数だけ表現が存在することを許そうというアイディアである．

3. 諸 定 義

ユーザが発行する検索式を解釈して，対応する部分資料を検索結果として提供することを目的とする．検索式に二つ以上のキーワードを入力した場合，従来は各キーワードを含む資料（或いはページ）を単純な AND または OR という論理演算に基づいた処理を行っている．それは前述のとおり，検索対象が単一の資料で結果の単位も資料であるからに他ならない．我々は，キーワード間の関係を考慮して AND 演算を細分化している．例えば，キーワード k_1 と k_2 が「k_1 と k_2」という意図で入力されているのか，それとも「k_1 の k_2」という意図で入力されるのかを区別したいのとともに，ユーザがキーワード間の関係を明示的に指定できれば有用と考えたからである．

そのため，入力される検索式に対して以下の検索結果を返すことを目的とする．

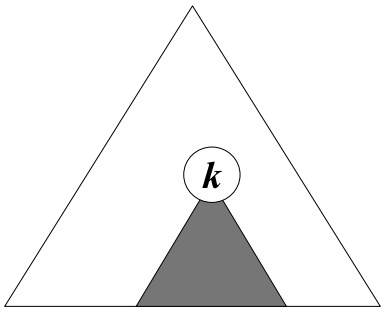

図 2　k の検索
Fig. 2　A quering by key word k

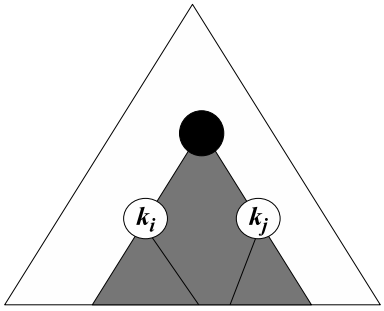

図 3　k_i と k_j の検索
Fig. 3　A quering by k_i and k_j

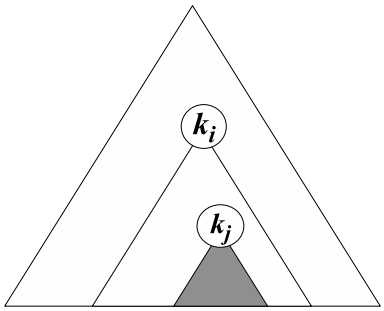

図 4　k_i の k_j の検索
Fig. 4　A quering by k_j of k_i

k　：キーワード k を含む部分資料 (図 2)
k_i と k_j　：キーワード k_i, k_j をともに含む最小の部分資料 (図 3)
k_i の k_j　：キーワード k_i を含む部分資料と k_j を含む部分資料のうち，他方に包含される部分資料 (図 4)

図中の k, k_i, k_j はそれぞれキーワード k, k_i, k_j を含むノードを表し，それを頂点とする三角形はそのノードを root とする部分資料を表す．濃く塗られた三角形が検索結果となる部分資料を表している．一番外側の三角形が資料全体を表している．

これらの検索を実現するために，関係と演算ならびに関数を次項より定義する．

3.1　関　　係

ここで，v_i が存在するツリーのルートから v_i への経路上に存在するすべてのノードの集合を path(v_i) とする．これは，部分資料に対応する部分木の root ノードを元資料の root からたどれる経路上に存在するノードの集合で代表させて議論を簡潔にするためである．

親戚関係 *(relatives)*

v_i, v_j が同一の tree 上に存在すれば，v_i, v_j 間に親戚関係が成立．つまり，

$$path(v_i) \cap path(v_j) \neq \phi \Rightarrow v_i \ relatives \ v_j$$

ここで，relatives は親戚関係を表す述語 (predicate) である．

直系関係 *(direct-descendants)*

v_i, v_j が同一の tree 上に存在し，tree の root から一方への経路上に他方が存在れば，v_i, v_j 間に直系関係が成立．つまり，

$$path(v_i) \supset path(v_j) \vee path(v_i) \subset path(v_j)$$
$$\Rightarrow v_i \ direct-descendants \ v_j$$

ここで，direct-descendants は親戚関係を表す述語である．

3.2　演　　算

親戚演算子 *RAND*

RAND 演算は，キーワード間の並立関係「と」に対応し，親戚関係にある v_i, v_j の共通で最も近い祖先を解とする．

$$v_i \ RAND \ v_j = v_k$$
(ただし，$path(v_k) = path(v_i) \cap path(v_j)$)

n 項親戚演算

n 個のノード $v_1, \cdots, v_n$ の RAND 演算は，親戚関係にある $v_1, \cdots, v_n$ の共通で最も近い祖先を解とする．

$$v_1 \ RAND \cdots RAND \ v_n = v_k$$
(ただし，$path(v_k) = path(v_1) \cap \cdots \cap path(v_n)$)

RAND 演算は各ノードへの経路上に存在するノード集合の積をとるので，入力されるノードの順に依存しない．

直系演算子 *DAND*

DAND 演算は，キーワード間の修飾関係「の」に対応し，直系関係にある v_i, v_j の下階層側を解とする．

$$v_i \ DAND \ v_j = \begin{cases} v_i, & \text{path}(v_i) \supseteqq \text{path}(v_j) \\ v_j, & \text{path}(v_i) \subseteqq \text{path}(v_j) \end{cases}$$

ここで，解を下階層側としたのは，構造の下階層に下るにしたがって内容が特化していくことを考慮したからで，細項目側となる下階層側を解とするのが無難と考えたからである．従って，検索式は「v_i の v_j」であっても，「v_j の v_i」であっても，解を

下階層側とするので，v_i と v_j の出現順序を問わず，検索結果は同じものとする．

n 項直系演算

n 個のノード $v_1, \cdots, v_n$ の DAND 演算は，直系関係にある $v_1, \cdots, v_n$ の最も下層のノードを解とする．

$$v_1\ DAND \cdots DAND\ v_n = v_k$$
$$(\text{ただし，}\ path(v_k) = path(v_1) \cup \cdots \cup path(v_n))$$

RAND 演算と同様に DAND 演算は各ノードへの経路上に存在するノード集合の和をとるので，入力されるノードの順に依存しない．

3.3 関　　数

RAND 演算を実行する rand() 関数と DAND 演算を実行する dand() 関数を以下のように定義する．

$rand()$ 関数

$rand()$ 関数は，任意のノード v_i, v_j を引数とし，v_i, v_j が親戚関係にあるとき，$path(v_i) \cap path(v_j) = path(v_k)$ であるようなノード v_k を戻り値とする．v_i, v_j が親戚関係にないときと，空値であるとき ϕ を戻り値とする．

n 項の RAND 演算

$v_1\ RAND\ v_2\ RAND \cdots RAND\ v_n$

を $rand()$ 関数で表現すと，

$$rand(\cdots(rand(v_1, v_2)\ldots), v_n)$$

である．これは，RAND 演算を左から順に実行することを意味し，戻り値と次の引数とを引数として $rand()$ 関数の適用を $n-1$ 回行う．

$dand()$ 関数

$dand()$ 関数は，任意のノード v_i, v_j を引数とし，v_i, v_j が直系関係にあるとき，$path(v_i) \cup path(v_j) = path(v_k)$ であるようなノード v_k を戻り値とする．v_i, v_j が直系関係にないときと，空値であるとき ϕ を戻り値とする．

n 項の DAND 演算

$v_1\ DAND\ v_2\ DAND \cdots DAND\ v_n$

を $dand()$ 関数で表現すると，

$dand(\cdots(dand(v_1, v_2)\ldots), v_n)$

である．これは，DAND 演算を左から順に実行することを意味し，戻り値と次の引数とを引数とした $dand()$ 関数の適用を $n-1$ 回行う．

4. 検索式の解釈

入力される検索式を内部処理するために，検索式の解釈を定義する．ここで，

- 検索式の解釈の関数表現を $Q()$
- ノード v のキーワード集合を $key(v)$

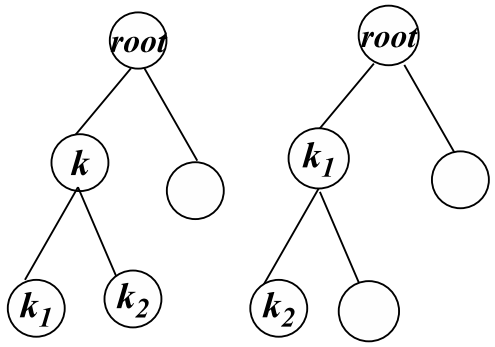

図 5　path から見る RAND と DAND
Fig. 5　RAND and DAND seen from path

とする．$Q()$ の戻り値はノードの集合である．また，見やすさを考慮して，検索式を [ ] でくくることとする．

まず，検索式 $[k]$ の解釈は，

$$Q(k) = \{v \mid k \in key(v)\}$$

である．

検索式 $[k_i\ と\ k_j]$ の解釈は，

$$Q(k_i\ と\ k_j) = \{rand(v_i,\ v_j) \mid v_i \in Q(k_i), v_j \in Q(k_j)\}$$

である．

検索式 $[k_i\ の\ k_j]$ の解釈は，

$$Q(k_i\ の\ k_j) = \{dand(v_i,\ v_j) \mid v_i \in Q(k_i), v_j \in Q(k_j)\}$$

である．

5. 複合検索

検索キーワード間の関係を複数指定したときの解釈を考察する．

$DAND$ 優先

本稿はここまで root ノードからキーワードを含むノードへの経路に着目して演算を定義して利用してきた．たとえば，

RAND の場合（図 5 左を参照），

$path(k_1) = root/k/k_1$

$path(k_2) = root/k/k_2$

$path(k_{1,2}) = path(k_1) \cap path(k_2) = root/k$

DAND の場合 (図 5 右を参照)，

$path(k_1) = root/k_1$

$path(k_2) = root/k_1/k_2$

$path(k_{1,2}) = path(k_1) \cup path(k_2) = root/k_1/k_2$

こういう解釈から RAND は積の性質，DAND は和の性質を持つという結論に至った．次，RAND と DAND の優先順位を考える時，視点を変えて，検索に出てきたノードをそのノードをルートノードとするサブツリー (或いは部分資料) と見なすことにする．検索式 $[k_1\ と\ k_2]$ は k_1 をキーワードとして持つ

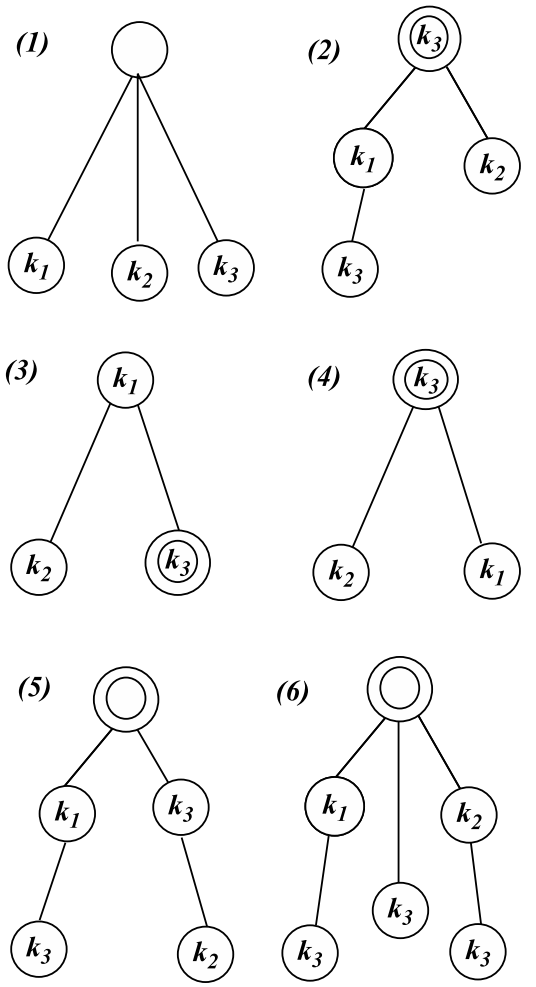

図 6　検索式 [(k1 と k2) の k3] の分配則的解釈例
Fig. 6　Interpret as Distribution

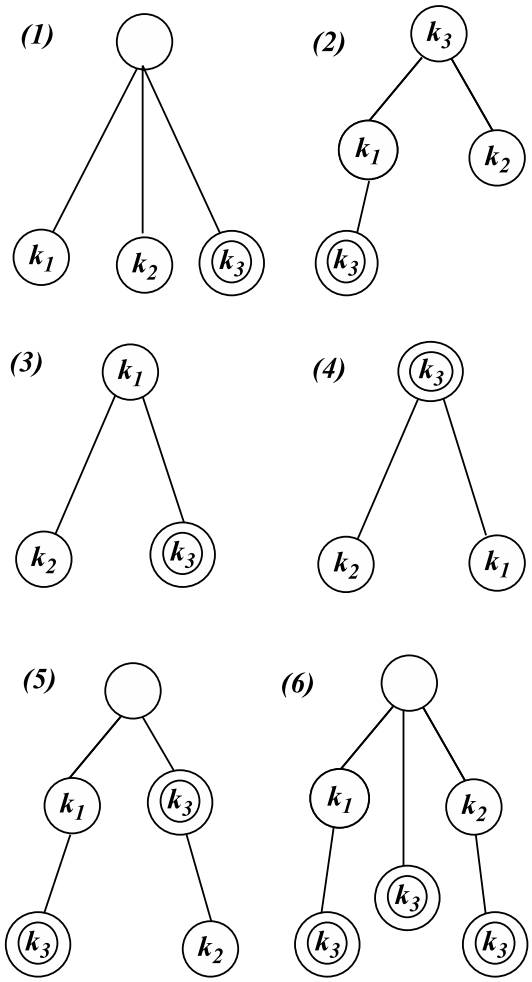

図 7　検索式 [(k1 と k2) の k3] の算数的解釈例
Fig. 7　Interpret as Numbers

ノードを root とするサブツリーと，k_2 をキーワードとして持つノードを root とするサブツリーとを含む部分資料を求める．それぞれのサブツリーが含むノードの集合を $V_{k_1}, V_{k_2}, V_{k_1,k_2}$ とすると，

$V_{k_1} \cup V_{k_2} \subseteqq V_{k_1,k_2}$

となっていなければならないはずである．同様に，検索式 [k_1 の k_2] は k_1 をキーワードとして持つノードを root とするサブツリーと，k_2 をキーワードとして持つノードを root とするサブツリーで包含される方の部分資料を求める．それぞれの部分資料が含むノードの集合を $V_{k_1}, V_{k_2}, V_{k_1,k_2}$ とすると，

$V_{k_1} \cap V_{k_2} = V_{k_1,k_2}$

となっていなければならないはずである．つまり， RAND に和の性質があり，DAND に積の性質があることがわかる．視点を変えることによって和と積の性質が逆になっていることが注意してほしい．優先順位を考える時，この後者の視点に基づいて考えるので，検索式の解釈に DAND 優先を採用する．

たとえば，検索式 [k_1 と k_2 の k_3] の解釈は

$$
\begin{aligned}
&Q(k_1 \text{ と } k_2 \text{ の } k_3) \\
&\quad = \{rand(v_1, dand(v_2, v_3)) \\
&\qquad\qquad \mid v_1 \in Q(k_1), v_2 \in Q(k_2), v_3 \in Q(k_3)\}
\end{aligned}
$$

というように，DAND の処理を行った後に RAND の処理を行うというように解釈する．

ところで，ユーザが「k_1 と k_2 の k_3」を探そうとしたとき，「k_1」と「k_2 の k_3」を探している場合と「k_1 の k_3」と「k_2 の k_3」を探している場合の二種類が考えられるが，本稿では検索式の自然言語的解釈は行わず，単に「と」を RAND ,「の」を DAND と読むことにする．

また，引数に空値（ϕ）を含む場合，RAND に和の性質があるため，この空値を無視し，他方の引数の結果を戻り値とする．DAND に積の性質があるため，結果として，ϕ を戻り値とする．

括弧と展開

検索式 [k_1 の k_3 と k_2 の k_3] の解釈は

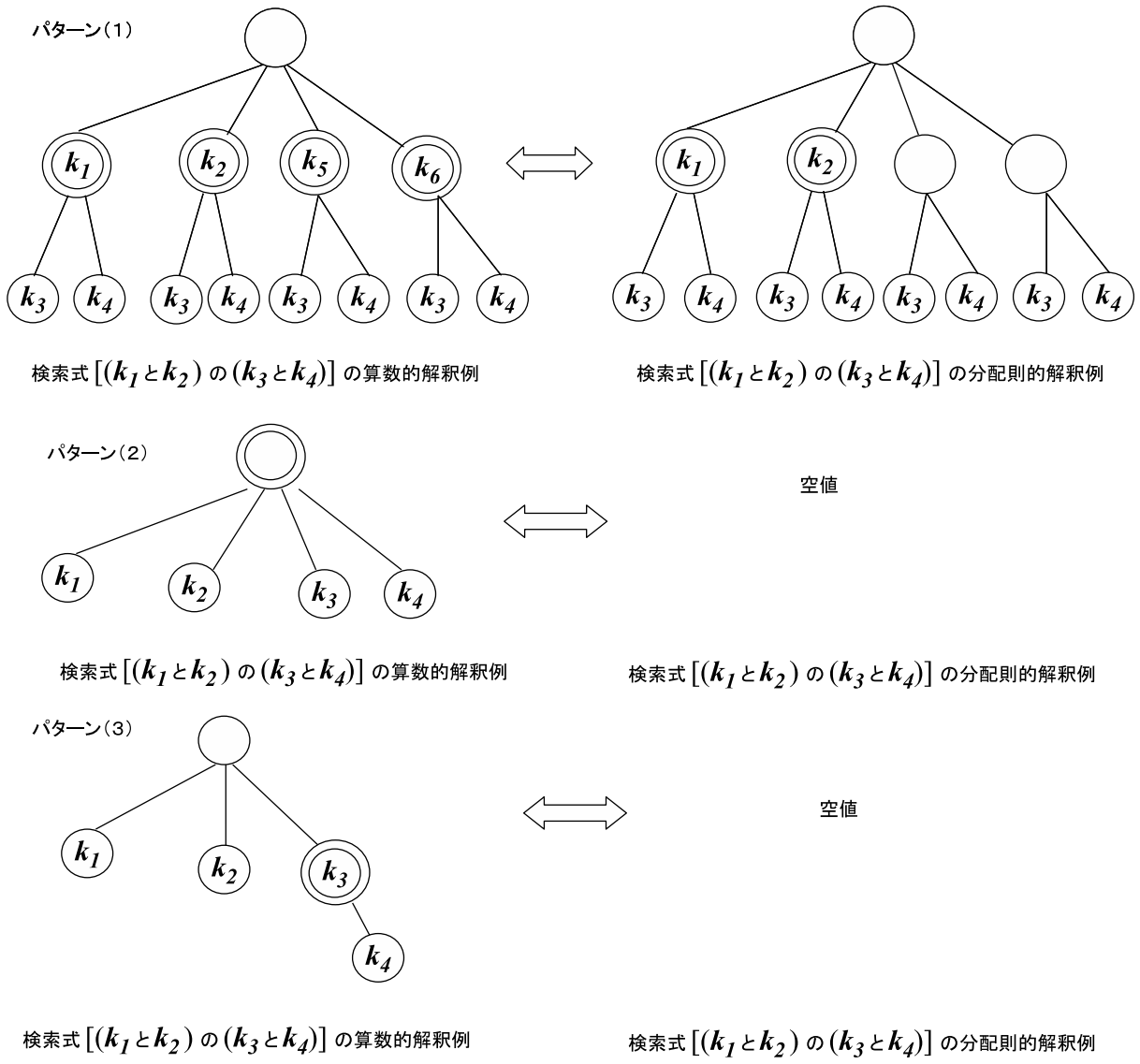

図 8　検索式 [(k_1 と k_2) の (k_3 と k_4)] の算数的解釈と分配則的解釈例
Fig. 8　Difference between Interpretation as Numbers and as Distribution

$Q(k_1$ の k_3 と k_2 の $k_3)$
$= \{rand(dand(v_1, v_3), dand(v_2, v_3))$
$\mid v_1 \in Q(k_1), v_2 \in Q(k_2), v_3 \in Q(k_3)\}$

である．しかしながら，この検索式には k_3 が二度指定されている．そこで，括弧を用いて入力を簡略化する．上記はユーザが「k_1 の k_3」と「k_2 の k_3」を探していると考えられるので検索式に括弧を許し，検索式 [(k_1 と k_2) の k_3] を検索式 [k_1 の k_3 と k_2 の k_3] と展開することにする．これは，上述の RAND が持つ和の特徴と DAND が持つ積の特徴を利用して分配則を適用している．

ここで，検索式 [(k_1 と k_2) の k_3] を算数的に
$Q((k_1$ と $k_2)$ の $k_3)$
$= \{dand(rand(v_1, v_2), v_3))$
$\mid v_1 \in Q(k_1), v_2 \in Q(k_2), v_3 \in Q(k_3)\}$
と解釈したとしよう．算数的に解釈する場合，展開せず，k_1 と k_2 を rand 演算を行ってから，k_3 と dand 演算を行う方法と，[k_1 の k_3] と [k_2 の k_3] に一旦展開して，それぞれ dand 演算を行い，その結果を rand 演算を行う二通りの方法がある．どの方法でも，算数上では，正しい結果を得られそうであるが，検索の場合は，展開せず得た結果に間違いがある．ここでは，便宜上，展開せずに演算を行う方法を算数的解釈，展開してから演算を行う方法を分配則的解釈と呼ぶ．図 6 に検索式 [(k_1 と k_2) の k_3] を分配則に基づいて解釈したときの結果を，図 7 に検索式 [(k_1 と k_2) の k_3] を算数的に解釈したときの結果をそれぞれ示した．図中の k_1, k_2, k_3 はキーワード k_1, k_2, k_3 をそれぞれ含むノードを示し，二重丸のノードが各解釈での結果である．図 6 と図 7 が全ての場合を網羅しているわけではないが，算数的に解釈した場合，ユーザが意図しない結果を含み，期待した結果を含まないことが考えられる．それは，図 7 で (3) と (4) 以外は k_1 あるいは k_2 をキーワードとしてもつノードを含まない部分木の root を結果の材料としているからである．従って，検索の場合，必ず展開して結果を求めなければならないこ

とが分かる．このルールの正しさはもっと複雑な検索式 [(k_1 と k_2) の (k_3 と k_4)] においても，成立する．

図 8 パターン（1）で，算数的解釈においては，k_1 と k_2 と全く関係のない k_5 と k_6 も検索結果に入っているのが分かる．これは，検索のノイズになり,「ユーザが必要とする資料中のある部分のみを検索」という本来の目的から外れるので，正しいとは思えない．それに対して，分配則的解釈を適用すると，検索結果が k_1 と k_2 に絞っているのが分かる．またパターン（2）においては，直観的に検索結果が空値（ϕ ）であるにもかかわらず，算数的解釈を適用すると，意図しない結果が出る．パターン（3）においても，検索結果が空値（ϕ）のはずであるが算数的解釈を適用すると，期待しない結果が出る．従って，現時点では分配則に基づく解釈を採用した．

6. ま と め

従来の資料検索は検索結果の単位が資料であったが，ユーザが必要とするのは資料中のある部分のみであると考えられるため，資料の任意の部分にメタデータを与え，ユーザが必要とするであろう部分のみを検索結果として提供するための検索方法を考察した．特に，複数のキーワードに対してその間の関連を簡単に指定したときに有効な検索結果を求める関係と演算を整理し，検索式の解釈方法について議論した．キーワード間の関係として並列関係を表す「と」と修飾関係を表す「の」を検索式に指定するため，従来の検索機構が概ね AND, OR の論理演算を実装するのに対して AND を細分化して RAND 演算と DAND 演算を導入した．それは，二つの検索キーワード k_1 と k_2 が「k_1 と k_2」という意図で入力されているのか，それとも「k_1 の k_2」という意図で入力されるのかを区別したいのとともに，ユーザがキーワード間の関係を明示的に指定できれば有用と考えたからである．さらに，検索キーワード間の関係を複数指定した場合，いわゆる複合検索において検索するノードをそのノードをルートノードとするサブツリーという視点から，DAND 演算は積の性質を備えるため，RAND 演算より，優先することを論じた．また，[(k_1 と k_2) の k_3] と [(k_1 と k_2) の (k_3 と k_4)] のような複雑な検索式の場合，展開せず演算を行う算数的解釈より展開してから演算を行う分配則的解釈の方がより正しい結果を得られることが分かった．

一般的な全文検索に対するデジタルアーカイブ検索の利点として，検索結果の単位を資料全体から利用者が必要とする部分資料に絞れることであるが，最も小さい適合単位のみを抽出するだけでは不十分である．なぜなら，利用者が欲するデータの量は一様でないからである．検索式を拡張することによって，利用者が欲するデータの情報をなるべく詳しく検索式に指定できれば有用である．より少ないヒントから所望のデータを抽出できることが理想的であるが，利用者とシステムとの重畳的な協調を探る取組みも重要である．特にデジタルアーカイブ検索の場合，検索結果の単位が資料から部分資料になっていく，つまり情報の粒度がより細かくなっていくという利点がある反面，検索結果のデータ量は増えることがある．絞込検索で検索結果の量が増えてしまうように見えるが，結果の件数と利用者の要求は絞込まれている．

図 9　神戸大学デジタルアーカイブ
Fig. 9　A Digital Archives at Kobe University

神戸大学付属図書館では，阪神・淡路大震災関係資料をデジタル化し，震災文庫デジタルアーカイブとして，1999 年 7 月から WWW 上で一般公開している（図 9）．本稿で提案した事案の一部は既にこのシステムに実装反映されている．この検索システムでは，五つまでの検索キーワードを指定することができ，そのキーワード間の関係を「直列 AND・親戚 AND・単純 AND・兄弟 AND」四種類に指定することができる．ただ，現状では利用者が入力したキーワードを順番に実行していく．今後，本稿で検討した検索の優先順位と複合検索を行う場合の分配則的解釈を実際のシステムに反映し，実験していきたいと思う．

＜謝辞＞

本研究の一部は，神戸大学国際文化学部「教育研究プロジェクト」の補助を受けている．

文　　献

[1] 依田平，大月一弘，森下淳也，清光英成，“ デジタルアーカイブに対する効率的な検索の提案　―神戸大学電子図書館システムを例として― ”，情報処理学会シンポジウムシリーズ 18 号　人文科学とコンピュータシンポジウム論文集，pp.259-266，2001.

[2] 依田平，小椋正道，大月一弘，森下淳也，清光英成，“ 電子図書館用デジタルアーカイブの検索方法の検討 ”，情報処理学会研究報告 70 号，pp.469-476，2001.

[3] 依田平，大月一弘，清光英成，森下淳也，“ ツリー型不定形文書からの部分文書の検索手法の検討 ”，第 14 回データ工学ワークショップ DEWS2003，2003.

[4] 依田平，渡邊隆弘，大月一弘，鳩野逸生，岩杉大輔，“ 多様な資料構造に対応したデジタルアーカイブシステムー神戸大学電子図書館アーカイブ検索システムー ”，情報処理学会研究報告，2003-FI-73，pp. 45-52，2003.

[5] K. Tajima, K. Hatano, T. Matsukura, R. Sana and K. Tanaka: Discovery and Retrieval of Logical Information Units in Web, Proc. of WOWS, pp. 13-23, Berkeley, CA, 1999.

[6] 絹谷弘子, 波多野賢治, 吉川正俊, 植村俊亮: 情報検索技術を用いた部分文書構造の自動抽出, 情報処理学会論文誌: データベース, Vol. 42, No. SIG8(TOD10), pp. 36-46, 2001.