
# 11. 2分探索木，B-木，k-d 木による見出し探索†

星　守††　弓場敏嗣†††

## 1. はじめに

情報処理における基本操作として情報の**探索・挿入・削除**・変更などがある．これらの操作の対象となる情報の基本単位を**レコード**(record)とよび，レコードの集合を**データベース** (database) または**ファイル** (file) とよぶ．レコードは複数の属性の組からなる定形化された共通の表現形式をもつ．

例として **表-1** のデータベースを考えよう．レコードを構成する属性としては県名，人口，所得額，農業粗生産額を用いる．表の各行が1つの県（レコード）に対応する．

属性のうちレコードを探索するときに用いられる属性を見出し (key) という．(第2節以下では，混乱のおそれがない限り，見出し（属性）と見出しの値（属性値）を区別しないで見出しと記す．) 見出しのうちレコードを一意に参照するものを**主見出し**，その他の見出しを**副見出し**とよぶ．以下では簡単のためレコードを $k$ 個の見出しの組として表現し，探索に用いない属性は除く．

レコードの探索にあたって用いる質問にはいくつかの型がある．主見出しを用いる型と，複数の見出しを用いる型とに大別される．前者を**主見出し探索**，後者を**副見出し探索**とよぶ[Knuth 83]．質問 $Q$ は一般に $Q=(q_1, q_2, \cdots, q_i, \cdots, q_k)$ という形で表わされる．ここで $q_i$ は $i$ 番目の見出しの値に関する条件を示す．

例えば県名を主見出し，農業粗生産額を副見出しとすると，主見出し探索では $Q=$(TOKYO)，副見出し探索では $Q=$(TOKYO, 4) $Q=(*, 30)$ ($*$はその属性値は何でもよいことを示す) などの質問が許される．($Q=(*, 30)$ に対しては (TOCHIGI, 30) (GUNMA, 30) の2つが求めるものとなる．)

表-1　県別人口，所得，農業生産額

| | 属性 | | | |
|---|---|---|---|---|
| | 県名 | 人口 | 県民所得 | 農業粗生産額 |
| | | (万人) | (千億) | (百億) |
| 茨城 | IBARAKI | 259 | 36 | 50 |
| 栃木 | TOCHIGI | 181 | 26 | 30 |
| 群馬 | GUNMA | 186 | 26 | 30 |
| 埼玉 | SAITAMA | 534 | 79 | 28 |
| 千葉 | CHIBA | 477 | 69 | 45 |
| 東京 | TOKYO | 1,136 | 282 | 4 |
| 神奈川 | KANAGAWA | 693 | 122 | 11 |
| 新潟 | NIIGATA | 246 | 33 | 37 |
| 富山 | TOYAMA | 111 | 17 | 12 |

(注)　朝日新聞社「'82民力」より作成

レコードの格納に際して，データベースに対する諸操作に対して効率的となるように，レコードの配置にある構造・規則性を持たせる．そのような構造としては(1)順配置を用いた線形リスト(表)，(2)順配置を用いたハッシュ表，(3)つなぎ配置を用いた木構造（木構造探索法）が代表的なものである．

**木構造探索法**は2分探索木 (binary search tree [Knuth 73]) に代表される**探索木法**と Trie [Fredkin 60] 等に代表される**桁探索木法**の2つに大別される（**図-1** 参照）．本稿では前者に属する**2分探索木**，2分探索木の多分木 (multiway tree) への拡張の一種である**B-木**[Bayer 72]，多次元見出しへの拡張である**多次元2分探索木** (multidimensional binary search tree [Bentley 75]) における探索・挿入・削除のアルゴリズムについて解説する．

木構造を用いた場合の見出し探索は，根から始めて，各節において，ある“判定操作”を行い，その結

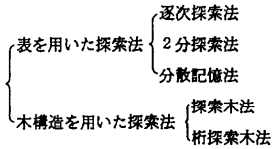



図-1　見出し探索法の類別

† Searching in binary search trees, B-trees and K-d trees by Mamoru HOSHI (Faculty of Engineering, Chiba University) and Toshitsugu YUBA (Electrotechnical Laboratory).

†† 千葉大学工学部
††† 電子技術総合研究所

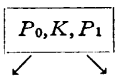
果に従ってその節の部分木（必ずしも1つとは限らない）を探索することを繰り返す．

2分探索木法では，見出しの集合に全順序関係があり（ないときは適当に導入する），各節で見出しの順序（大小）比較を行って順次探索範囲を狭めていく．桁探索木法では，見出しがいくつかの桁に分割されているものと考え，各節では見出しの桁の値を調べ順次探索領域を狭めていく．なお各節に見出しを置く場合を**密モデル**とよび，端節のみに見出しを置く場合を**疎モデル**とよぶ．以下では密モデルの例を示す．

## 2. 2分探索木

2分探索木は木構造を用いた主見出し探索法の代表的なものであり，最もよく研究されている．これは主見出しの集合のもつ（あるいは導入された）順序関係に従ってレコードを並べた順配置の線形リスト（表）上での2分探索法を，つなぎ配置のデータ構造で実現したものと考えてよい．

**図-2** に 表-1 のレコードを上から順番に挿入して構成した2分探索木を示す．節は○印で表わされている．内側のローマ字列はそこに置かれている見出しである．以下では見出し集合の順序関係はアルファベットによる辞書式順序を用いる．2分探索木の各節は，見出し$K$と左右の部分木を指す2つのポインタとから成る：




2分探索木は次の性質を満たしている：

性質 1：任意の節において，その節に置かれている見出しは，その節の左（右）部分木中にあるすべての見出しより大きい（小さい）．

この性質から，与えられた見出し$X$を探索・挿入するアルゴリズムは次のようになる．

[探索・挿入手順]

A 1　根から始める．根の見出し$K$と与えられた見出し$X$とを比較する．一致すれば探索は成功で（成功探索）手順は終了．

A 2　$X<K(>K)$ ならば左（右）部分木の根を調べる；左（右）部分木が空ならば探索は不成功（不成功探索）で，左（右）部分木として新しい節を作り，そこに$X$を置く（挿入）．空でなければ左（右）部分木へ進み，探索を繰り返す．

図-2 で $X$=NIIGATA を探索すると，IBARAKI, TOCHIGI, SAITAMA と進み，KANAGAWA の右部分木を調べると，部分木は空なので$X$が存在しないことが判明する．追加するときはKANAGAWA の右子節として新しい節を作り，そこに見出し$X$を置き，2つのポインタを ∧(空の部分木を示す) として手順を終了する．

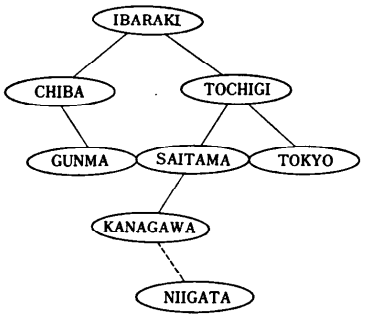



**図-2**　2分探索木

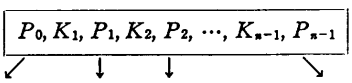
見出し$X$を削除する場合は，まず探索手順によって$X$の置かれている節を見つける．その節が端節の場合はその節を切り離して終了．端節でない場合はその見出しを除去し，その節を根とする部分木中の見出しの中で(見出し集合の順序関係において)，$X$の直前または直後の見出し$Y$を$X$の代わりに置く．このとき$Y$は同様の手続きで削除される．この操作は端節に至って終了する．例えば図-2で $X$=IBARAKI を削除すると，直前（後）の見出し GUNMA (KANAGAWA) が IBARAKI の代わりに置かれ，端節 GUNMA (KANAGAWA) は切り離されて終了する．

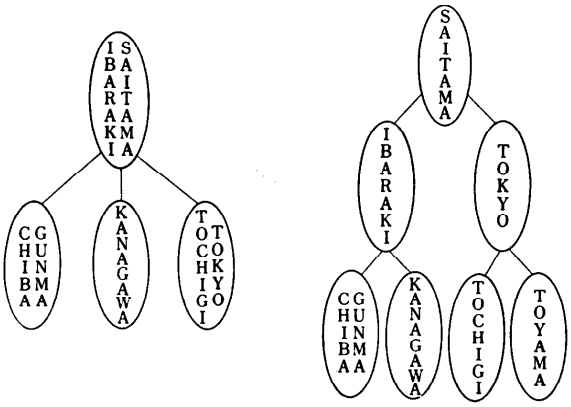
## 3. B-木

B-木は平衡した2分探索木を$m$分木 ($m\geq 3$) に拡張したもので，多分平衡木 (multiway balanced tree) とよびうるものである．平衡という意味は，木のどの節においても左右の部分木の高さの差が，ある一定値以内であるという意味である．

B-木は，レコードの挿入・削除を繰り返しても，呼出し時間および記憶空間利用率が著しく低下することがなく安定しているので，2次記憶を用いるファイル管理に使用される．

$m$次の B-木の各節は，$(n-1)$ 個の見出し $K_1<K_2<\cdots<K_{n-1}$ と$n$個のポインタ $P_0, P_1, \cdots, P_{n-1}$ とから構成されている（ただし $\lceil m/2\rceil\leq n\leq m$）：






(a) 人口順に挿入　(b) TOYAMA を追加挿入

**図-3** 2-3 木

さらに根はそれ自身が端節の場合を除いて2つ以上の子節を持ち，すべての端節のレベルは同一であるという木の形状に関する条件を満たす．

B-木は性質1の自然な拡張である性質2を満たしている：

性質 2：任意の節においてその節の $n-1$ 個の見出しを $K_1<K_2<\cdots<K_{n-1}$，$n$ 個のポインタを $P_0, P_1, \cdots, P_{n-1}$ とすると，ポインタ $P_i$ $(i=1\sim n-2)$ で指示される部分木中の見出しは，見出し $K_i$ と $K_{i+1}$ の間の見出しである．$P_0$ $(P_{n-1})$ の指す部分木中の見出しは $K_1$ $(K_n)$ より小さい（大きい）見出しである．

この性質から探索は次のようになる．

[探索手順]

B 1　根から始める．根が空なときは，根に見出し $X$ を置き，ポインタ $P_0, P_1$ を $\wedge$ とする．空でないときは，根中の見出し $K_1, \cdots, K_{n-1}$ の中で $X$ を探す．存在すれば終了（成功探索）．

B 2　$K_i<X<K_{i+1}$ ならばポインタ $P_i$ を見る（$X<K_1$ ならば $P_0$，$K_{n-1}<X$ ならば $P_{n-1}$）；$P_i$ が空ならば $X$ は存在しない（不成功探索）ので挿入手順を行う．空でないときは $P_i$ の指す部分木へ行き探索を繰り返す．

**図-3** に表-1の見出しを人口順に挿入した3次のB-木（**2-3木**または**3-2木**ともよばれる）を示す．

挿入・削除に関しては，B-木の条件を保つために，少し工夫が必要となる．

**図-3**（a）の例を用いて説明する．この木に $X=$ NIIGATA を挿入する場合は，$X$ の探索を行うと節 KANAGAWA の右部分木を指すポインタが空であることから，NIIGATA が存在しないことが判明する．この場合，この節には見出しが1つしかないので，そこに NIIGATA を置いて挿入は終了する．見出し $X=$TOYAMA を挿入する場合は，最右端の節で不成功探索となるが，そこには既に許容数の2 (=3−1) 個の見出しが置かれているので，そこに置くことができない．このような**オーバフロー**が生じる場合には，3個の見出しTOCHIGI, TOKYO, TOYAMA のうち，順序が中央の見出しTOKYOを親節に渡し，オーバフローを生じた節を2分割してそれぞれに残った見出しを置く．見出し TOKYO を受け取った節（この場合は根）でも再びオーバフローが生じるので，そこでも節の分割を行う．中央の見出し SAITAMA を新しく創製した根に渡しそこに置く（**図-3**（b））．この結果，木のレベルは1だけ増加する．オーバフローは端節から根の方向へ向かって伝播して行くが，根でそれが生じたときのみレベルの増加が生じる．

削除する場合は，削除する見出し $X$ の置かれている節を，前述の探索手順で求めて，その見出しを除去する．その節が端節でないときは，$X$ の直前または直後の見出しを $X$ の代わりに置き（そのような見出しは端節にある），その見出しを端節から除く．いずれの場合も端節での削除で終了する．このとき，端節内の見出しの数が $\lfloor m/2 \rfloor$ 以下となることがある（**アンダフロー**の発生）．そのときはすぐ隣りの兄弟節から見出しの移動を行う．あるいは兄弟節と兄弟節を分離している親節内の見出しを吸収合併して1つの節とする必要がある．この結果親節で再びアンダフローが生じる場合があるが，同様の操作を繰り返す．アンダフローが根まで伝播したときは，レベルが1だけ減少する．(この処理によって記憶空間利用効率が常に 50 % 以上に保たれる．)

図-3(b)で $X=$TOYAMA を削除するとアンダフローが根にまで伝播し，最後には図-3(a)の木となる．

## 4. 多次元2分探索木

B-木が2分探索木を $m$ 分木に拡張したものであるのに対して，多次元2分探索木は扱いうる見出しを複数にしたという意味で，2分探索木を多次元化したものといえる．多次元2分探索木は，副見出し探索における完全一致型 (exact match queries)，部分一致型 (partial match queries)，区間指定型 (region queries)，最近接型 (nearest neighbour queries) などの質問に

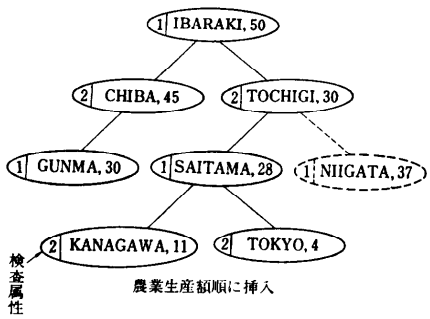
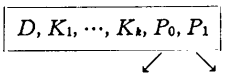



図-4　2-d 木

対して探索効率が良いデータ構造であるばかりでなく，他のさまざまな問題においても有効なデータ構造として用いられる．

$k$ 次元2分探索木の各節は，**検査属性** $D$，$k$ 個の見出し $K_1, \cdots, K_k$ および2つのポインタ $P_0, P_1$ から構成される：




検査属性 $D$ は，その節での見出し比較に何番目の見出しを用いるかを指示している．各節での検査属性の与え方には自由度がある．J.L. Bentley が提案した ***k*-dimensional tree**（又は ***k*-d tree**）では，レベル $i$（根のレベルは0）での見出し比較に，$i \pmod k + 1$ 番目の属性を用いる．

表-1 のレコードを，県名と農業粗生産額とを2つの ($k=2$) 見出しとして用いて，生産額順に挿入した結果の 2-d 木を 図-4 に示す．

質問 $Q=(X_1, X_2, \cdots, X_k)$（$X$ は $*$ でもよい）に対する探索の手順は，基本的に2分探索木と同じである．

[探索手順]

C1　根から始める．各節において，その節の検査属性で指定される属性を用いて，見出しの大小比較を行う．検査属性が $i$ のときは，質問 $(X_1, X_2, \cdots, X_i, \cdots, X_k)$ 中の $X_i$ と節中のレコードの第 $i$ 属性値 $K_i$ とを比較する．

C2　$X_i < K_i$ ($X_i > K_i$) のときには，2分探索木と同様に左（右）部木へと探索を進める．$X_i = K_i$ のときは，レコード $(K_1, \cdots, K_k)$ が質問を満たすか否かを調べて，満たせば1つレコードが求まる．その後左（又は右のどちらでもよいが一方に定めておく）部分木へ進む．$X_i = *$ のときは，左右の部分木を共に探索する．

C3　端節に至って終了する．

挿入手順も2分探索木と同じであるが，新しく創製する節の検査属性を適当に与える必要がある．

例えば，図-4 の木に対して質問 (NIIGATA, 37) を満たすレコードを探す．根では検査属性が1であるから，第1番目の属性である県名を見出し比較に用いる．NIIGATA > IBARAKI なので右部分木（の根）へ進む．その節での検査属性は2なので生産額を比較に用いると，37 > 30 なので右部分木（の根）へ進む．そこで右部分木が空なので (NIIGATA, 37) は存在しないことが判明する．挿入するときは右子節を作り，検査属性を1とし，見出し NIIGATA, 37 を置いて終了する（挿入）．

削除する場合は，まず探索手順で削除するレコードの置かれている節を求める．次いで，そのレコードを除去し，代わりにその節の検査属性に関して，その左部分木（又は右部分木）内で属性値の大きい（小さい）レコードを持って来る．この操作を端節に至るまで再帰的に繰り返す．

## 5. おわりに

木構造を用いた見出し探索法は．1950年代に考案された2分探索法に始まり，1960年代に入ってその研究は盛んになった．以下では，その後の木構造探索に関する研究の足取りを概観する．

新しい見出し探索法としては，疎な桁探索木 Trie [Fredkin 60]，d-c 木[Sussenguth 63]，PATRICIA 木[Morrison 68] などが1960年代に発表されている．1970年代前半には，密な桁探索木 sequence hash tree[Coffman 70] の他，B-木[Bayer 72] などが発表された．70年代の中頃からは副見出し探索の研究が盛んになり，Quad tree[Finkel 74]，k-d 木[Bentley 75]，Quintary tree[Lee 80] などが提案された．また最近は，B-木の多次元化[Ouksel 81]，Trie の多次元化[Orenstein 82] などの研究もある．

以上のように次々と新しい探索木が提案されてきたが，それらの木の性能評価，最適木・準最適木・自己組織木を構成する問題などに関する理論的・実験的研究が多くなされている（それらに関しては [弓場 80] を参照）．

最後に，木構造を用いた見出し探索に関する解説，展望論文を紹介する．[Knuth 73] の第6章は最も内容が豊かで含蓄に富んだものである．ACM Computing Survey 誌に掲載されたものとして [Nievergelt 74],

[Severance 74] の二つがある．前者は平衡2分探索木の解析結果を中心に，実際の応用局面への課題等について展望している．後者は，著者の提案する探索モデルに沿って，見出し探索法の諸問題を扱ったものである．

国内のものとしては，[宮川 75] は分散記憶法を含めて見出し探索全般について，平易にその技法を紹介している．また，[星 75] と [弓場 80] は著者等の分類法に基づいて木構造を用いた見出し探索法を分類し，関連論文を多数（それぞれ50, 144件）掲載している．B-木の解説には，[Comer 79], [溝口 80] がある．

## 参 考 文 献

[星 75] 星 守，弓場敏嗣：木構造と見出し探索—サーベイ，信学技法，EC 75-54, 21-30 (1975, 12).

[宮川 75] 宮川正弘，弓場敏嗣：計算機における見出し探索の技術，電学誌，Vol. 95, No. 8, pp. 11-18 (1975, 8).

[溝口 80] 溝口徹夫："B-tree" によるデータ管理，情報処理，Vol. 21, No. 7, pp. 769-776 (1980).

[弓場 80] 弓場敏嗣，星 守：木構造を用いた見出し探索の技法，情報処理，Vol. 21, No. 1, pp. 28-49 (1980).

[Bayer 72] Bayer, R. and McGreight, E.: Organization and Maintenance of Large Ordered Indexes, Acta Informatica, Vol. 1, pp. 173-189 (1972).

[Bentley 75] Bentley, J. L. : Multidimensional Binary Search Trees Used for Associative Searching, Commun. ACM, Vol. 18, No. 9, pp. 509-517 (Sep. 1975).

[Coffman 70] Coffman, E. G., Jr. and Eve, J.: File Structures Using Hashing Functions, Commun. ACM, Vol. 13, No. 7, pp. 427-436 (July 1970).

[Comer 79] Comer, D.: The Ubiquitous B-Tree, Comput. Surv., Vol. 11, No. 2, pp. 121-137 (June 1979).

[Finkel 74] Finkel, R. A. and Bentley, J. L.: Quad Trees a Data Structure for Retrieval on Composite Keys, Acta Informatica, Vol. 4, pp. 1-9 (1974).

[Fredkin 60] Fredkin, E.: Trie Memory, Commun. ACM, Vol. 3, No. 9, pp. 490-499 (Sep. 1960).

[Knuth 73] Kunth, D. E.: The Art of Computer Programming, Vol. 3, Sorting and Searching, Addison-Wesley (1973).

[Lee 80] Lee, D. T. and Wong, C. K.: Quintary Trees: A File Structure for Multidimensional Database Systems, ACM Trans. on Database Systems, Vol. 4, No. 3, pp. 339-353 (1980).

[Morrison 68] Morrison, D. R.: PATRICIA—Practical Algorithm to Retrieve Information Coded in Alphanumeric, J. ACM, Vol. 15, No. 4, pp. 514-534 (Oct. 1968).

[Nievergelt 74] Nievergelt, J.: Binary Search Trees and File Organization, Comput. Surv., Vol. 6, No. 3, pp. 195-207 (Sep. 1974).

[Orenstin 82] Orenstein, J. A.: Multidimensional Tries Used for Associative Searching, Information Processing Letters, Vol. 14, No. 4, pp. 150-157 (1982).

[Ouksel 81] Ouksel, M. and Scheuermann, P.: Multidimensional B-Trees: Analysis of Dynamic Behavior, BIT, Vol. 21, pp. 401-418 (1981).

[Severance 74] Severance, D. G.: Identifier Search Mechanisms: A Survey and Generalized Model, Comput. Surv., Vol. 6, No. 3, pp. 175-194 (Sep. 1974).

[Sussenguth 63] Sussenguth, E. H., Jr.: Use of Tree Structures for Processing Files, Commun. ACM, Vol. 6, No. 5, pp. 272-279 (May 1963).

[Wirth 75] Wirth, N.: Algorithms+Data Structures=Programs, Prentice Hall (1975).

(昭和57年12月3日受付)