一般化KD木を用いたデータベースマシンGRACEの二次記憶系の設計

正　員　伏見　信也[†]　　正　員　喜連川　優[††]
正　員　田中　英彦[†††]　　正　員　元岡　達[†††]

Design of Secondary Storage System of Database Machine GRACE Using Generalized KD-Tree

Shinya FUSHIMI[†], Masaru KITSUREGAWA[††], Hidehiko TANAKA[†††] *and* Tohru MOTO-OKA[†††], *Members*

あらまし　関係データベースマシンGRACEは，従来関係データベース処理の隘路となっていたjoin等の処理負荷の重い演算を高速に実行することができる．従って，GRACEでは低速大容量の二次記憶系に対するアクセスが処理の隘路となることが予想され，適切なクラスタリングを行うことによりこれを高速化して，更に高い性能を得ることができる．この為に，一般化KD木法と呼ばれる新しい多次元クラスタリング技法を開発した．本技法はリレーション中のタプルの分布，問合せ中の検索述語の分布に適応したページ分割を行うことができ，その性能評価の結果，従来のアクセス法，更には多次元クラスタリング技法に比較して，二次記憶への平均アクセス回数を大幅に減少させることが確認された．

1. まえがき

関係データベースマシンGRACE[(1)]は，従来関係データベース処理の隘路となっていたjoin等の処理負荷の重い演算を高速に実行することができる．従ってGRACEに於ける問合せ処理の隘路は，低速大容量の二次記憶系（磁気ディスクを主体とするディスクモジュール）に対するアクセス，即ちselection演算処理に移行することが予想され，これを高速化することにより更に高い性能を実現することが可能である．selection演算はクラスタリングにより検索述語を満たすタプルを可能な限り少数のディスクページに収め，ディスクアクセス回数を減少させることにより高速化される．二次記憶系設計の為のクラスタリング技法として，従来KD木法[(2),(3)]，拡張KD木法[(4)]，多次元trie法[(5),(6)]等の所謂多次元クラスタリング技法が提案されているが，これらはリレーション中のタプルの分布，更にそれらタプルに対する検索述語の発生分布に対し，充分な最適化を行っているとは言えず，GRACEの様な大規模データベースマシンの二次記憶系の設計技法としては不十分である．本稿ではリレーションのタプルの分布，問合せ中の検索述語の分布を考慮した多次元クラスタリングアルゴリズムを提案し，二次記憶へのアクセス時間に於いて支配的な，ディスクに対する平均アクセスページ数を大幅に減少させることを試みる．

† 東京大学大学院工学系研究科，東京都
Graduate School of Engineering, The University of Tokyo, Tokyo, 113 Japan
†† 東京大学生産技術研究所，東京都
Institute of Industrial Science, The University of Tokyo, Tokyo, 106 Japan
††† 東京大学工学部電気工学科，東京都
Faculty of Engineering, The University of Tokyo, Tokyo, 113 Japan

2. データベースマシンGRACEとその二次記憶系

GRACEはhashとsortを用いた並列アルゴリズムにより，joinすべき2つのリレーションのタプル数を各々T_1，T_2，処理の並列度をmとすると，$O((T_1+T_2)/m)$時間でjoinを実行することができる[(1)]．GRACEの二次記憶系は大容量可動ヘッドディスクを有するディスクモジュールであり，ここにベースリレーションが格納される．GRACEではjoin等の演算がタプル数に比例した時間で処理され，ディスクモジュ

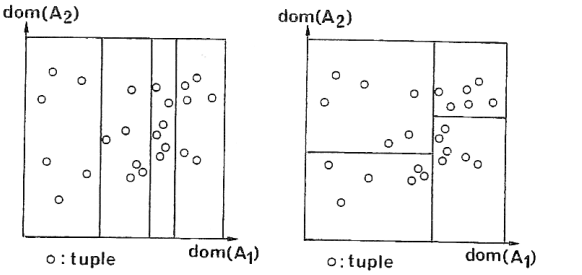
ールから送られてくるタプルのストリームに沿った関係代数処理が実現されている．従ってGRACEに於ける関係代数演算の処理性能は基本的にディスクモジュールに於けるタプルストリーム生成の効率，即ち当該演算に必要なタプルを得る為のディスクへのアクセス回数，1アクセスに要する時間，及びディスクからのデータの転送レートで定められる．ディスクからのデータ転送レートはディスクやディスク制御装置自身の物理的特性により規定されるが，一方ディスクへのアクセス回数，アクセス時間は各々タプルを適切にページ分割すること，更にそれらページをディスク空間に適切に配置することにより大幅に減少させることが可能である．即ち，一般に二次記憶系の設計問題は，(1)リレーション中のタプルをディスクのページ容量の大きさに分割するタプルのページ分割問題と，(2)これにより得られたページをディスク内にその機械的特性を考慮しつつ配置するページの物理的配置問題の2つに分類される．ディスクへのアクセス時間を支配するのは検索述語を満たすタプルを得るために必要なディスクページへのアクセス回数であり，これは基本的に(1)のみによって定まる．特にその平均値（平均ディスクページアクセス回数）はGRACEの性能を大きく左右するものであり，本稿ではこの最小化について議論する．尚，本稿を通じて問合せとは問合せ中に存在するselection述語を指すものとする．

3. 多次元クラスタリングと二次記憶系の設計

3.1 多次元クラスタリング

一般に，ページ化された二次記憶系に於いて，問合せにヒットするタプルをできる限り少数のページに集積し，記憶することによって平均アクセスページ数を減少させる手法をクラスタリングと呼ぶ．従ってクラスタリングの基本戦略は『同時に，しかも頻繁にアクセスされるタプルをできる限り少数のページに格納すること』である．

ここでクラスタリングの観点から従来の二次記憶系設計の設計技法を振り返ってみると，2分木，B木，ハッシュ等に基づく従来の技法は，基本的に主キーのみに関するページ分割，即ち主キーに関する一次元クラスタリングと見なすことができる（図1(a)）．従って，主キーに関する問合せに対しては，単純な値指定のもの（$v=c$）から，更に2分木，B木等に於いてはrange query（$c_1 \leqq v \leqq c_2$）に対しても原理的に最小のページアクセス数を実現することができる．一方，



(a) One-dimensional clustering　(b) Multidimensional clustering

図1　一次元クラスタリングと多次元クラスタリング
Fig.1 One-dimensional clustering and multidimensional clustering.

非キーに関する問合せに対しては，主キーのみに対するページ分割を行った結果，タプルが非キー値に関してページ空間に分散してしまい，極めて多数のページアクセスが必要とされる．このことは非キーに対するアクセスパスが交代インデックス等の密な木に限られることと対応する．この様に，従来の設計手法は主キーに対するクラスタリング特性を重視しすぎたページ分割を行うことによって，全体として平均アクセスページ数を著しく増大させている．

多次元クラスタリングとは主キーのみでなく非キーを含めた複数の属性に対してページ分割を行うことによって，平均アクセスページ数の減少を図るクラスタリング技法である（図1(b)）．多次元クラスタリングは各属性の軸に沿ったページ分割しか許さず，ページはリレーションの原空間（各属性のドメインの直積空間）で必ず超直方体となる点でクラスタリング技法全体の内の比較的小さな部分集合であるが，(1)無秩序なクラスタリングを行って得られたページ空間に対し，効率の良いアクセスパスが存在しない，(2) range queryに対し，非常に良好なクラスタリング特性を与える，等の点から二次記憶系設計に於いて好ましい性質を持つ．更にページ分割を再帰的に行う制約を加えた多次元クラスタリングのクラスに対しては，(3)タプルの挿入，削除等によって生じたページのオーバフロー／アンダフローに対し，ページの分割／併合によって局所的，動的に対処できる利点を有する．ここではこの再帰的な多次元クラスタリングを行うクラスを単に多次元クラスタリングと呼び，このクラスの中での平均アクセスページ数の最小化について検討する．

3.2 多次元クラスタリングの技法

多次元クラスタリング技法は，現在までにいくつか提案が行われている．KD木（K-Dimensional tree）法[2],[3]はリレーションRの原空間$\boldsymbol{D}$とRのタプル分布

が与えられた時，先ずRのタプルの中からそのA_1属性の値が値の大小関係に従ってRのタプル数を2分する様なものを選び，これにより$\boldsymbol{D}$を2分割する．得られた2つの部分空間$\boldsymbol{D}_l$, $\boldsymbol{D}_r$に対し，各々更にタプル数を2分するタプルを今度は属性A_2に関して選択し，これらを各々更に2分する．この操作を分割に用いる属性（分割属性）を巡回的に変化させながら再帰的に繰り返す．ページ容量分のタプルを含む部分空間が得られた時点で各再帰操作は終了する（図2）．得られたページ分割に対応してKD木と呼ばれるインデックスが生成される．アルゴリズムから明らかな様に，本技法はタプル分布のみに応じたページ分割を行うものであり，Rに対する問合せ分布を全く無視している．従って多次元にわたるクラスタリングにもかかわらず平均アクセスページ数は比較的大きな値となることが予想される．拡張KD木（Extended KD-tree）法[4]はKD木が問合せの分布を全く無視したページ分割を行う点を改良し，木の各レベルに於ける分割属性はレベル毎に共通であるものの，レベルに沿った分割属性は固定された巡回的なシークエンスではなく，問合せ分布を或る程度考慮したアルゴリズムにより決定する技法である．拡張KD木法は，タプル分布に加え，問合せ分布を考慮して平均アクセスページ数の減少を図るものであるが，アルゴリズムは問合せ分布がpartial match queryの場合に限られており，更にそこでは属性間の確率的独立性が仮定されている．また各レベル内では

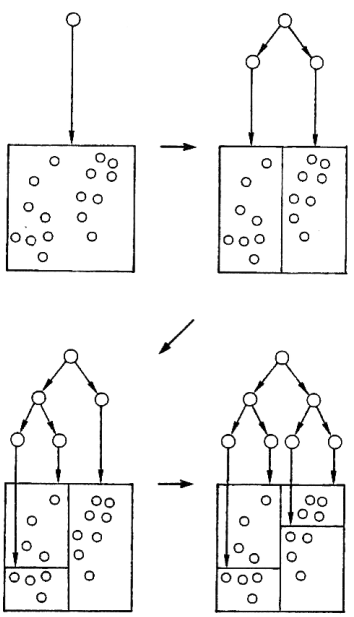


図2　KD木法によるページ分割
Fig.2 Page partitioning process of KD-tree method.

分割属性は固定されている点などから，range queryを含み，確率的に属性間に強い相関を持つ実際の使用環境下では大きな性能改善は期待できないと考えられる．多次元trie法[5],[6]は通常の（binary）trieをKD木様に多次元化した技法である．KD木法，拡張KD木法がページ分割の際にタプル数を2分する値を分割値として選ぶのに対し，多次元trie法では分割値として分割しようとする空間自身を2等分する値を選択する．多次元trieの各ノードは分割属性とその属性に対するビットポジションを持つ．分割に使用するビットは各属性につき最上位から順に用いられる．多次元trie法は，そのアルゴリズムから明らかな様に生成される各ページの境界は2ベキ値に限られ，この精度でしかタプル分布に対応出来ない為，一般に100%のロードファクタは得られない．一方，アクセスパスに必要な補助記憶の容量は非常に小さく保つことが出来る．(6)では一様なpartial match queryに対する分割属性選択のアルゴリズムが与えられているが，一般の問合せ分布に対して，平均アクセスページ数の最小化を考えた多次元trieによるページ分割のアルゴリズムはまだ提案されていない．

4. コスト評価式

本章では二次記憶系設計の為のコスト評価式として，（多次元）クラスタリングによって生成されたページ空間に対する平均アクセスページ数を与える一般式を導出する．以下，本稿を通じて対象とするリレーションを$R(A_1, \cdots, A_k)$（属性総数k），Rのタプル数をT，ページ数をN，ページ容量をV（タプル）とする．加えて，ここではRの原空間$\boldsymbol{D}=\coprod_{i=1}^{k}\mathrm{dom}(A_i)$（$\mathrm{dom}(A_i)$は属性$A_i$のドメイン）に於けるタプルの分布関数として$D(t)$，問合せ分布関数として$Q(q)$を用いる．

1. タプル分布関数　$D(t)$：

$D(t)=n \iff t=(v_1, \cdots, v_k)$なるタプルが$R$中に$n$個存在する．リレーションがトランスポーズされていなければ一般に$n=0$ or 1である．即ち，

$$\int_{t\in\boldsymbol{D}} D(t)\,dt = T$$

であり，$D(t)$は一定のアルゴリズムが生成可能なページ分割の総数に関与する．

2. 問合せ分布関数　$Q(q)$：

$Q(q)$は問合せqに対する正規化された分布関数とする．即ち，

$$Q:\boldsymbol{Q}\rightarrow[0,1]\quad(\boldsymbol{Q}\text{は}q\text{の全体})$$

$$\int_{q\in\boldsymbol{Q}}Q(q)dq=1$$

一般にリレーションの原空間$\boldsymbol{D}$がN個のページ$\{P_1,\cdots,P_N\}$に分割された時，問合せ分布$Q(q)$に対する平均アクセスページ数$\bar{\pi}$は，各qにヒットするページ数$\pi(q)$にqの重み$Q(q)$を乗じ，qについてこれを加え合わせることにより得られ，

$$\bar{\pi}=\int_{q\in\boldsymbol{Q}}\pi(q)Q(q)dq$$

と表される．ここで問合せqによりページP_jがアクセスされることを$\Omega(P_j,q)$と表せば，

$$\pi(q)=\#\{P_j\mid\Omega(P_j,q)\}$$

（#は集合のcardinality）

であり，

$$\delta(P_j,q)=\begin{cases}1 & \Omega(P_j,q)\text{が真}\\ 0 & \Omega(P_j,q)\text{が偽}\end{cases}$$

と定義すれば，

$$\pi(q)=\sum_{j=1}^{N}\delta(P_j,q)$$

従ってこれにより$\bar{\pi}$を次の様に書き換えることが出来る．

$$\left.\begin{aligned}\bar{\pi}&=\int_{q\in\boldsymbol{Q}}\pi(q)Q(q)dq\\&=\int_{q\in\boldsymbol{Q}}\left(\sum_{j=1}^{N}\delta(P_j,q)\right)Q(q)dq\\&=\sum_{j=1}^{N}\int_{q\in\boldsymbol{Q}}\delta(P_j,q)Q(q)dq\\&=\sum_{j=1}^{N}\int_{q\in\Gamma(P_j)}Q(q)dq\\&\left[\Gamma(P_j)=\{q\in Q\mid\Omega(P_j,q)\}\right]\\&=\sum_{j=1}^{N}H(P_j)\end{aligned}\right\}\quad(1)$$

但し，$H(P_j)=\int_{q\in\Gamma(P_j)}Q(q)dq$

即ち，$\bar{\pi}$はN個のページP_j各々についてP_jにヒットする問合せqの重み$Q(q)$を加えることによって得られる．この式はクラスタリングのクラスや問合せのクラスに依らず一般的に成立する．ここで，$\bar{\pi}$は$Q(q)$と$\{P_j\}$のみによって定まることに注意すべきである．

本稿では問合せのクラスとしてrange queryの全体$\left\{\bigwedge_{i=1}^{k}x_i\leqq v_i\leqq y_i\right\}$を考察の対象とする．$x_i\leqq y_i$を考慮すれば，range query qは$\coprod_{i=1}^{k}[x_i,y_i]$と同一視され，各$q$の第$i$成分は図3に示す直角三角形状の区域中の一点として表現される．ここで各属性のドメインを有界な全順序集合，即ち$\mathrm{MIN}_i\leqq v_i\in\mathrm{dom}(A_i)\leqq\mathrm{MAX}_i$ for all iと仮定する．図中，A点は一般のrange predicateに相当し，一方B点，C点は各々predicateが与えられていない場合，値が指定されている場合（exact match）に対応する．従って，このクラスはrange queryのみならず，値指定の問合せ，partial match queryをも含む一般的なクラスである．

さて上記の同一視によりqは$\boldsymbol{D}$の部分集合と見なされるから，$\Omega(P_j,q)\equiv P_j\cap q\neq\phi$が成立し，更に各ページ$P_j$を$\boldsymbol{D}$に於ける超直方体に限ると（即ち，多次元クラスタリングのクラスに属する技法が生成するページ分割に制限すれば），$P_j=\coprod_{i=1}^{k}[\alpha_{ij},\beta_{ij}]$となり，$N$個のページは各々$q$と同型の空間内の点として表現される．従って，各$P_j$を$\boldsymbol{D}$内の一点として扱うことが出来る（図3）．この時，

$\Omega(P_j,q)\equiv P_j\cap q\neq\phi\equiv x_i\leqq\beta_{ij}\wedge y_i\geqq\alpha_{ij}$ for all i

から，ページP_jにヒットする問合せは各属性について図3中，斜線で示される範囲に属するものとなる．

式(1)の積分値を解析的に求めることは一般に不可能であるが，幾つかの簡単なページ空間，問合せ分布に対しては計算が可能であり，多次元クラスタリングと従来の一次元クラスタリングとの性能比較を行うことができる[7]．例えば，各属性のドメインを各々n_i区間（$i=1,\cdots,k$）に分割することにより原空間をメッシ

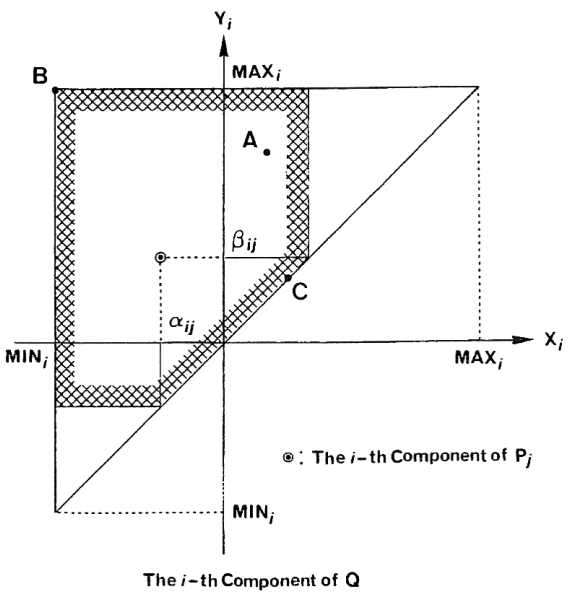


図3　問合せとページの$\boldsymbol{Q}_i$に於ける表現
Fig.3 Representation of queries and page in $\boldsymbol{Q}_i$.

ュ状にページ分割して得られたページ空間に対し，$Q(q)$ として Q 上一様な分布を仮定すれば，

$$\bar{\pi}=\prod_{i=1}^{k}(n_i/3+1-1/3n_i)$$

$$\fallingdotseq\prod_{i=1}^{k}(n_i/3)=O(N/3^k)$$

が得られ，この場合平均アクセスページ数 $\bar{\pi}$ は属性数の3のベキ乗に反比例して減少することがわかる．これに対し，従来の一次元クラスタリングによる平均アクセスページ数 $\bar{\pi}'$ は，$n_1=N$，$n_2=\cdots=n_k=1$ として

$$\bar{\pi}'=N/3+1-1/3N$$
$$=O(N/3)$$

これから，この様な場合には多次元クラスタリングによって $\bar{\pi}$ は大きく改善される．

5. 一般化KD-treeによる二次記憶系の設計

5.1 一般化KD-tree

3.2で見た様に，多次元クラスタリングに基づく二次記憶系設計に対してはいくつかの技法が提案されているが，それらの技法はページ分割の際の分割属性及び分割値選択に関する自由度の点から生成可能なページ分割の全体を制約していることがわかる．簡単の為，タプルの分布 $D(t)$ は一様であるとすると，リレーション R を N ページに分割する時，KD木法では分割属性，分割値の選択に関して自由度が無い（アルゴリズムによってアプリオリに定まっている）為，生成可能なページ分割の総数は1となる．拡張KD木法では木の各レベル毎に分割属性選択の自由度があり，ページ分割全体の総数は $k^{\log N}$（$D(t)$ の一様性から木は平衡する）である．多次元 trie 法は木の各ノードについて k 通りの自由度がある為，ページ分割全体の総数は k^{N-1} となる．逆に言えば，各々のページ分割アルゴリズムは，この生成可能なページ分割全体の集合の中から出来るだけ小さな $\bar{\pi}$ を与える1つのページ分割を選択するアルゴリズムと考えられる．ページ分割の自由度の点から考えれば，木の各ノードに於いて分割属性，分割値共に最大限の自由度を与え，クラスタリングによって生成可能なページ分割全体の集合を拡大することによって実現可能な $\bar{\pi}$ の最小値の下限を大幅に小さくすることが可能である．

一般に n ページのタプルを含む空間の分割に対して100％のロードファクタを保証しようとすると，k 個の各属性に対し $n-1$ 通りの分割値候補点が存在する．ページ分割の際の分割値選択に関し，この $k\cdot(n-1)$ の自由度を実現する多次元クラスタリングのクラスをここでは一般化KD木法（Generalized KD-tree，略してGKD木）と呼ぶ．GKD木法によるページ分割の概略は次の様になる．N ページ分のタプルを含むリレーション R のページ分割に対し，一定のアルゴリズムに従って $k\cdot(N-1)$ 個の分割値候補点の中から1つを選ぶ．これによって決定されるページ容量 n に従って原空間 $\boldsymbol{D}$ を各々 n ページ，$N-n$ ページ分のタプルを有する2つの部分空間 $\boldsymbol{D}_l, \boldsymbol{D}_r$ に分割する．次に得られた部分空間 $\boldsymbol{D}_l, \boldsymbol{D}_r$ に対し，N を各々 n，$N-n$ として手続きを再帰的に繰り返す．分割すべき空間に含まれるタプル数が1ページ容量に達した時点で各再帰手続きは終了する．GKD-tree によって生成可能なページ分割全体の集合の大きさ $C(N)$ は漸化式

$$C(N)=\sum_{i=1}^{N-1}k\cdot C(i)\cdot C(N-i)$$

によって定められ，その値は

$$C(N)=(1/N)\cdot\binom{2N-2}{N-1}\cdot k^{N-1}\quad(N\geqq 1)$$
$$\fallingdotseq(4k)^{N-1}/N\sqrt{\pi\cdot(N-1)}$$

と求められる．この値は上述のいずれの技法よりも大きく，問合せ分布 $Q(q)$ を考慮し，ページ分割の際の分割属性，分割値を決定するアルゴリズムを適切に設定してこの自由度を活かすことにより，他の技法に比較して非常に小さな $\bar{\pi}$ を実現するページ分割を得ることが可能となる．

5.2 ページ分割アルゴリズム

GKD木を用いたクラスタリングを行うにはページの再帰的な分割に際して分割属性，分割値を決定するアルゴリズムが必要である．

さて，前述の様に二次記憶系設計に於けるクラスタリング技法の基本的戦略は『頻繁にアクセスされるタプルは出来る限り同一ページに収める』ことである．即ち，この様なタプルを同一ページに収めることが出来れば，アクセス頻度の少ないタプルがページ空間に分散し，多くのページアクセスを必要としても全体として平均アクセスページ数 $\bar{\pi}$ は大きく減少する．これを逆に考えれば問合せ分布の peak にヒットするタプルは出来る限り複数ページに分離しないページ分割，即ち $Q(q)$ の値が小さい q にヒットするタプルから順にページに分割して行くアルゴリズムが考えられる．

ここで，ページ $P_j=\prod_{i=1}^{k}[\alpha_{ij},\beta_{ij}]$ の A_i 属性に関し値 r_{ij}（$\alpha_{ij}\leqq r_{ij}\leqq\beta_{ij}$）でページ分割を行うと，$P_j^l(r_{ij})$

$=[\alpha_{1j}, \beta_{1j}]\times\cdots\times[\alpha_{ij}, r_{ij}]\times\cdots\times[\alpha_{kj}, \beta_{kj}]$ 及び $P_j^r(r_{ij})=[\alpha_{1j}, \beta_{1j}]\times\cdots\times[r_{ij}, \beta_{ij}]\times\cdots\times[\alpha_{kj}, \beta_{kj}]$ なる2つのページが生成される．この時次の定理が成立する．

〔定理〕

$$H(P_j)+H(P_j^l(r_{ij})\cap P_j^r(r_{ij}))=H(P_j^l(r_{ij}))+H(P_j^r(r_{ij})) \quad (2)$$

或いは，$H(P_j)$ を $H(\alpha_{1j}, \beta_{1j}, \cdots, \alpha_{kj}, \beta_{kj})$ と書くと，

$$H(\alpha_{1j}, \beta_{1j}, \cdots, \alpha_{ij}, \beta_{ij}, \cdots, \alpha_{kj}, \beta_{kj})+H(\alpha_{1j}, \beta_{1j}, \cdots, r_{ij}, r_{ij}, \cdots, \alpha_{kj}, \beta_{kj})=H(\alpha_{1j}, \beta_{1j}, \cdots, \alpha_{ij}, r_{ij}, \cdots, \alpha_{kj}, \beta_{kj})+H(\alpha_{1j}, \beta_{1j}, \cdots, r_{ij}, \beta_{ij}, \cdots, \alpha_{kj}, \beta_{kj}) \quad (\alpha_{ij}\leqq r_{ij}\leqq\beta_{ij}) \quad (3)$$

（証明） 属性 A_i に関する $\boldsymbol{Q}$ の射影空間 $\boldsymbol{Q}_i$ を考える（図4）．r_{ij} で分割を行うと，図に示される様に A_i に関しては $[\alpha_{ij}, \beta_{ij}]$ が2つのページ境界 $[\alpha_{ij}, r_{ij}]$ 及び $[r_{ij}, \beta_{ij}]$ に分離される．この分割によって生成された2つのページに対し，A_i 以外の属性に関しての計算の積分区域に変更は無く，A_i のみに対して各々積分区域が図中三，|||| で示される範囲に変更される．2つの積分区域は図中囲で示される部分が重複しており，この部分は仮想的にページ $P_j^A(r_{ij})=[\alpha_{1j}, \beta_{1j}]\times\cdots\times[r_{ij}, r_{ij}]\times\cdots\times[\alpha_{kj}, \beta_{kj}]$ に対応する．従って，積分の積分区域に関する加法性により式(2)，(3)が成立する． （証明終）

上式から P_j のページ分割前の平均アクセスページ数

$$\bar{\pi}=\sum_{j=1}^{N}H(P_j)$$

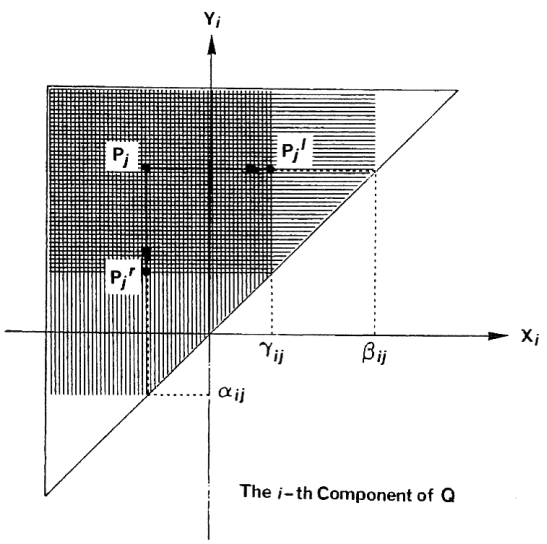


図4 ページ分割によるページ境界の変化
Fig.4 Page boundary changes caused by page split.

は P_j の r_{ij} に関するページ分割によって

$$\bar{\pi}'=H(P_1)+\cdots+H(P_{j-1})+H(P_j^l(r_{ij}))+H(P_j^r(r_{ij}))+H(P_{j+1})+\cdots+H(P_N)=H+H(P_j^A(r_{ij}))$$

に変化することがわかる．即ち，このページ分割によって $\bar{\pi}$ は $H(P_j^A(r_{ij}))$ だけ増加する．頻繁にアクセスされるタプルを横切って P_j を分割するとこの値は大きくなることから，先の抽象的アルゴリズムは次の様に具体化される．N ページ分のタプルを含むリレーション R の分割に対し，$k\cdot(N-1)$ 個の分割値候補点の内，値 $H(P^A(r_i))$ が最小となる r_i を選ぶ．r_i によって $\boldsymbol{D}$ が各々 n，$N-n$ 個のタプルを含む2つの部分空間 $\boldsymbol{D}_l=[\mathrm{MIN}_1, \mathrm{MAX}_1]\times\cdots\times[\mathrm{MIN}_i, r_i]\times\cdots\times[\mathrm{MIN}_k, \mathrm{MAX}_k]$ 及び $\boldsymbol{D}_r=[\mathrm{MIN}_1, \mathrm{MAX}_1]\times\cdots\times[r_i, \mathrm{MAX}_i]\times\cdots\times[\mathrm{MIN}_k, \mathrm{MAX}_k]$ に分けられたとすると，$\boldsymbol{D}$ を $\boldsymbol{D}_l(\boldsymbol{D}_r)$，$N$ を $n(N-n)$ としてこの操作を繰り返す．各再帰ステップは分割すべき空間がページ容量分のタプルを含む場合に終了する．

本アルゴリズムはバックトラックを行わない直線的アルゴリズムであり，個々のページ分割のステップで局所的に最適な分割値を選ぶものであるから，生成されたページ分割に対する $\bar{\pi}$ は必ずしも最小とは限らない．しかし，本アルゴリズムはGKD木のページ分割に際しての分割属性，分割値の選択に関しての大きな自由度を活かし，『頻繁にアクセスされるタプルは可能な限り複数ページに分割しない』という点を保証しており，平均アクセスページ数を大きく減少させることが期待出来る．

5.3 ページ分割アルゴリズムの時間計算量

$H(P^A(r))$ の値を計算する手間を1とすると，N ページのページ分割に対し前節で提案したアルゴリズムの時間計算量 $TC(N)$ は次の様に求められる．

［最良の時］ ページ分割により空間が常に2等分される場合に時間計算量は最小となり，

$$TC(N)=k\cdot(N-1)+2\cdot TC(N/2)$$

即ち，

$$TC(N)=k\cdot((\log N-1)\cdot N+1)=O(k\cdot N\cdot\log N)$$

この場合，対応して生成されるGKD木は完全に平衡する．

［最悪の時］ ページ分割により生成される空間の一方が常に1ページ容量のタプルを含む場合，最も長い時間が必要とされ，この時，

$$TC(N)=k\cdot(N-1)+TC(N-1)$$

即ち,

$$TC(N)=k\cdot N\cdot(N-1)/2$$
$$=O(k\cdot N^2)$$

対応して生成されるGKD木は直線状のものとなり，木の探索の時間も増大する．

5.4 GKD木を用いたページ分割アルゴリズムの評価

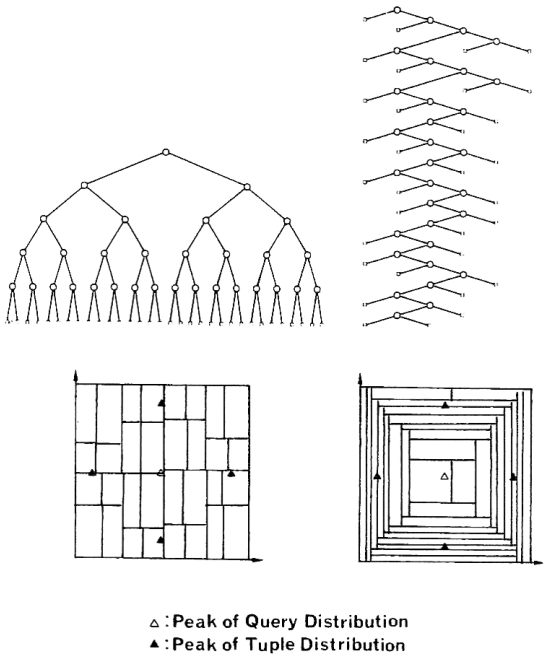
GKD木を用いたページ分割アルゴリズムは種々の分布に対して評価を終えているが，ここではその一例を示す．評価に用いた$Q(q)$，$D(t)$はそのpeakにずれがある場合であり，図5（$k=2$の場合）に示す様に，$Q(q)$は一つのpeakを$\boldsymbol{D}$の中心に，また$D(t)$は$2k$個のpeakを$\boldsymbol{D}$の周辺に持つ．この分布に対し，KD木法及びGKD木法によりページ分割を行った結果を図5(a)，(b)に示す．図に示される様にKD木法ではタプル数を常に2分するページ分割を行う為，$Q(q)$の分布のpeakを横切ってしまうページ分割が生成される．一方，GKD木法は$Q(q)$のpeakを認識し，それを避けつつ，これを囲む様にしてページ分割を行っており，$Q(q)$のpeakはほぼ2ページに収められている．ここで，GKD木法では一つの$Q(q)$のpeakを検出すると，これを避けながらページ分割を行う為，図5(b)に見られる様に生成されるGKD木が平衡しない．しかし，一般には問合せの分布は複数のpeakを有し，この様な場合にはGKD木のページ分割アルゴリズムはこれらを避けながらページ分割を行い，結果として大局的には平衡し，局所的に一次元状に成長したGKD木が得られる．$k=4$の時の同様な分布に対する平均アクセスページ数$\bar{\pi}$とページ数Nとの関係を図6に示す．図6からGKD木法によるページ分割はKD木法に比して大きな性能改善を与えていることがわかる．

我々は以下の様な点からGRACEの二次記憶系設計技法としてGKD木法を採用することとした．

(1) GKD木法はページ分割に対し分割属性，分割値選択に関する自由度を可能な限り許すことによって，平均アクセスページ数を大幅に小さくすることができ，GRACEに於いては極めて効率の良いデータ流の生成が可能となる．

(2) GKD木法は**3.1**(2)，(3)で述べた多次元クラスタリング技法が有する利点を保存する．

(3) 前述の様に，GKD木法では生成されるGKD木が平衡しない場合があり，一般にこれは木探索の時間を増大させる．しかしGRACEの如き大規模関係データベースマシンに於ては，アクセスパスとして生成されたGKD木を専用に設けた半導体記憶中に保持することとしても，コスト的にそれ程問題とならない．これに加えて，インデックス木検索の専用ハードウェア等の付加が可能であり，木の平衡化はさほど問題とならない．



(a) Page partitioning by KD-tree method　(b) Page partitioning by GKD-tree method

図5 KD木法とGKD木法によるページ分割
Fig.5 Page partitionings produced by KD-tree method and GKD-tree method.

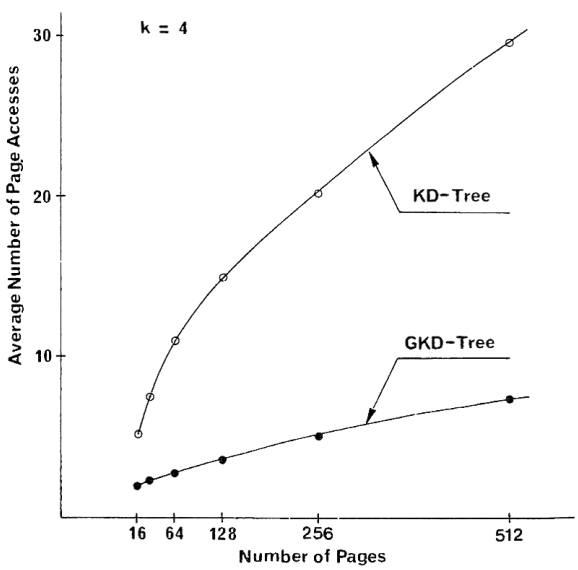


図6 KD木法とGKD木法の性能比較
Fig.6 Performance comparison between KD-tree method and GKD-tree method.

6. 実現手法

GKD木法は，GRACEの二次記憶系に用いることを前提としており，その実現に際してはデータベースマシンの実装環境を利用することができる．GKD木の探索に関しては，上記の様にGKD木をディスクモジュールに付加された半導体記憶中に保持することができ，基本的に探索時間は十分に小さいと考えることができる．また頻繁にアクセスされるページは必ず木の最深部のノードに対応する為，この事実を逆に利用して探索を高速化することも可能である．

一方，ページ分割の各再帰ステップに於いては，分割値候補点の生成，及び問合せ分布の管理と$\boldsymbol{D}$の部分空間に対するHの値の計算を効率良く実現することが必要である．前者に関しては，既に開発したハードウェアソータ[8]を用いた処理技法を，また後者に関しては縮退統計と呼ばれるアルゴリズムを提案し，その有効性を確認した[7]．

更に，タプルの挿入／削除が頻繁に行われる所謂動的ファイルの環境ではページのオーバフロー／アンダフローが多発するが，先に述べた様にGKD木は再帰的なページ分割を行うクラスタリングのクラスに属しており，これに局所的，動的に対処することができる．

7. むすび

関係データベースマシンGRACEの二次記憶系設計技法について考察した．提案する技法は一般化KD木（GKD木）による多次元クラスタリング技法であり，本技法はページ分割の際の分割属性，分割値選択に関する自由度を利用し，問合せ分布$Q(q)$，タプル分布$D(t)$を十分考慮したクラスタリングを実現し，従来の二次記憶系設計技法，更には多次元クラスタリング技法に比較し，大幅に平均アクセスページ数を減少させることができる．一方で一様な低次元の partial match query 等，リレーションをタプル方向に分割する限り平均アクセスページ数の減少をそれほど期待出来ない問合せ分布も存在する．この様な場合，リレーションの縦方向の分割（トランスポーズ，原空間の射影分解）とGKD木によるリレーションの横方向の分割を組み合せる技法が有用である[9]．また問合せ分布，タプル分布の管理／処理技法として各々縮退統計，ハードウェアソータを用いた手法を考案したが，これら分布は一般に時間と共に変化し，これに動的に対応して行く必要がある．これら問題については現在検討を進めており，本技法の更に詳細な評価，実装技法と共に，稿を改めて発表したい．

文　献

(1) M. Kitsuregawa, H. Tanaka and T. Moto-oka : "Application of Hash to Data Base Machine and Its Architectue", New Generation Computing, 1, 1, pp.63-74 (July 1983).

(2) J.L. Bentley : "Multidimensional Binary Search Trees Used for Associative Searching", Comm. ACM, 18, 9, pp.509-517 (Sept. 1975).

(3) J.L. Bentley : "Multidimensional Binary Search Trees in Database Applications", IEEE Trans. Software Eng., SE-5, 4, pp.333-340 (April 1979).

(4) J.M. Chang and S.K. Fu : "Extended KD-tree Database Organization : A Dynamic Multi-attribute Clustering Method", Proc. Very Large Data Base, pp.39-43 (1979).

(5) J.A. Orenstein : "Multidimensional Tries Used for Associative Searching", Info. Proc. Let., 14, 4, pp.150-157 (April 1981).

(6) Y. Tanaka : "Adaptive Segmentation Schemes for Relational Database Machines", in Database Machines, pp.293-318, Springer-Verlag (1983).

(7) 伏見，喜連川，田中，元岡："多次元クラスタリング技法に基づくGRACE二次記憶系の設計と評価"，信学技報，EC83-49 (1984-02).

(8) 喜連川，伏見，桑原，田中，元岡："パイプラインマージソータの構成"，信学論(D), J66-D, 3, pp.332-339 (昭58-03).

(9) 伏見，喜連川，田中，元岡："GRACE二次記憶系に於ける拡張多次元クラスタリング技法"，情報処理学会第27回全国大会，2K-2, pp.669-670 (1983).

（昭和59年11月8日受付，60年2月1日再受付）