GRACEに於ける二次記憶系の構成

4P-3

伏見 信也　喜連川 優　田中 英彦　元岡 達
(東京大学　工学部)

0. はじめに

現在までに提案され、或いは試作されてきた関係型データベースマシンに於ては、関係代数処理の際にはリレーションのフルスキャンを基本としているものが多く、インデックス等の補助データ構造に対しては考慮を払うことが少なかった。一方で、単純な selection処理やjoin時の検索空間の絞り込み等、二次記憶系の設計如何によりマシン全体の性能が大きく左右される場合も考えられる。本稿では、これらの事実をふまえ、GRACEに於ける二次記憶系の設計について、特に多次元clusteringを中心に考察する。

1. データベースマシンに於ける二次記憶系

一般に、リレーションはページと呼ばれる単位に分割されて二次記憶系に保持される。一方で、二次記憶系の動作速度は処理系のそれに比較して極めて低速であり、関係代数処理の際には1)アクセスするページ数を極力押えること、及び2)アクセスすべきページが二次記憶内で連続して配置されていること、が必要となる。1)は(joinの前処理を含めた) selection 処理を行なう際、そのpredicate を満足するレコードがページ空間に広く分散しない制御、即ちレコードの論理的なclustering問題であり、2)は可動ヘッドディスクのような二次記憶デバイスの特性に起因する物理的な連続配置問題である。ここでは1)についてのみ考える。

現在用いられている二次記憶系設計の手法は、リレーションの key attribute のみについてハッシュやインデックスを用いるものが多い。この場合、non-key attribute については alternateインデックスを用いるしかなく、一般に non-key attributeによる検索には全ページのフルスキャンが必要となる。これは key attribute のみclusteringを行なった結果、 non-key attribute に対する検索特性が"ぼけて"しまったと考えられる(図1)。一方、図2のように non-key attribute に関してもclusteringを施せば、key-attribute の検索特性は多少ぼけるものの、全体としてのclustering効果は大きく、上記1)の解決に大きく寄与するものと考えられる。尚、このような多次元のclusteringを行なったリレーションに対しても、 k-d tree [1] 等を用いたインデックス構成が可能である。

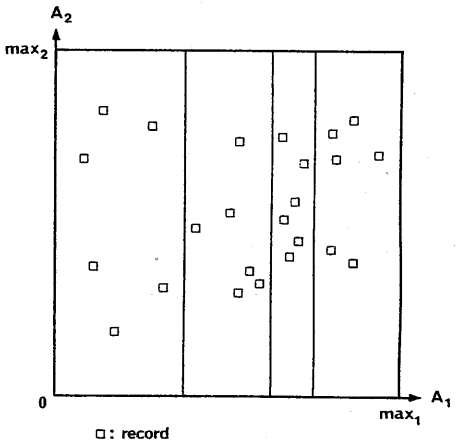

Fig.1 One-Dimensional Clustering

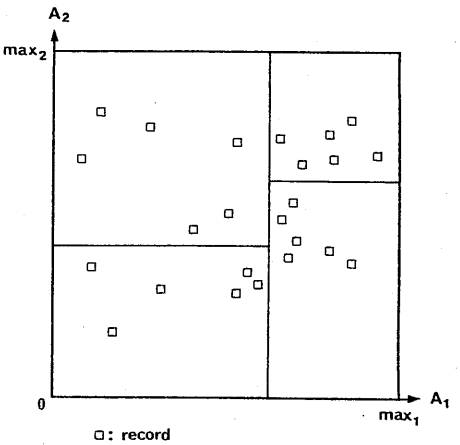

Fig.2 Multi-Dimensional Clustering

2. 多次元clusteringのコスト評価

以下、多次元clusteringを行なう際のページ分割のコスト評価、及びそれに関するいくつかの結果について述べる。

2.1 コスト評価式

リレーションR(A_1, …, A_k), D_i = dom(A_i), $0 \leq x_i \in D_i \leq max_i$と仮定する。一般にselection に用いられるpredicate c (ここでは単

に queryと呼ぶ）は

$$c=\prod_{i=1}^{k}[x_i, y_i]\in Q$$

$$((Q=\{\prod_{i=1}^{k}[x_i, y_i] \mid x_i, y_i\in D_i,\ x_i\leq y_i\}))$$

と書ける。ここに$x_i=y_i$ならA_iについての値を指定した queryとなり、$x_i=0$ 、$y_i=\max x_i$ならA_iについては何も指定しない queryとなる。ページ分割の問題は、データベースの分布 d（x）（$x\in\Pi D_i$）と queryの分布 q（c）（$c\in Q$）が与えられた時、query 空間に対する平均アクセスページ数の最小化問題として定式化できる。1ページの容量をVとすればデータベースを格納するのに必要なページ数 nは、

$$n=\int_{x\in\Pi D_i} d(x)\,dx/V$$

であり、 n個のページを各々、

$$p_j=\prod_{i=1}^{k}[\alpha_{ij}, \beta_{ij}]\quad (j=1, \cdots, n)$$

と表わすと、

$$\sum_{j=1}^{n} p=\prod_{i=1}^{k} D_i$$ （ΠD_i はp_jのdisjoint union）。

一方、あるquery $c=\prod_i[x_i, y_i]$にヒットするページ$p_j=\prod_i[\alpha_{ij}, \beta_{ij}]$が満たすべき条件G（$p_j$, c）は、

$$G(p_j, c)\Leftrightarrow p_j\cap c\neq\phi$$
$$\Leftrightarrow\ x_i\leq\beta_{ij}\wedge y_i\geq\alpha_{ij}\ \text{for all } i$$

であり、従って、card（K）を集合Kの濃度とすれば、c にヒットするページ数 fは、

$$f(c)=\mathrm{card}(\{p_j \mid G(p_j, c)\})$$

となる。ここで、

$$\delta(p_j, c)=\begin{cases}1 & G(p_j, c)\\ 0 & \text{otherwise}\end{cases}$$

と定義すれば、

$$f(c)=\sum_{j=1}^{n}\delta(p_j, c)。$$

従って fの平均値 $\bar{f}$として、

$$\bar{f}=\int_{c\in Q} f(c)\cdot q(c)\,dc$$
$$=\int_{c\in Q}\sum_j \delta(p_j, c)\cdot q(c)\,dc$$
$$=\sum_j\int_{c\in Q}\delta(p_j, c)\cdot q(c)\,dc$$
$$=\sum_j\int_{c\in W(p_j)} q(c)\,dc\quad\cdots\quad(*)$$

$$(W(p_j)=\prod_i\{(x_i, y_i) \mid G(p_j, \prod_i[x_i, y_i])\})$$

なる式が得られる。

2.2 多次元clusteringの評価

一般に、（*）の最小値を与えるページ分割を求めるのは困難であるが、簡単な場合として q（c）, d（x）が共に一様な分布の時の1．で述べた多次元clusteringの効果について考える。この時、 m（$1\leq m\leq k$）個の attributeを用いて多次元clustering した場合、（*）の積分を実行すれば、

$$\bar{f}=\prod_{i=1}^{m}(\sqrt[m]{n}/3+1-1/(3\sqrt[m]{n}))$$
$$\sim o(n/3^m)$$

を得る。 即ち、clustering attributeを増すほど平均アクセスページ数は指数的に減少し、この場合、多次元clusteringの効果は非常に大きい。

3． おわりに

GRACEは、 transpose（縦割り）されたリレーションに対しても効率の良い処理が可能である。この場合の $\bar{f}$の評価は、各サブリレーションについての d（x）, q（c）の周辺分布を用いて（*）を評価した値の和となるが、この際 attribute間の相関の大小により効果的な transpositionが期待される。今後、 $\bar{f}$を最小化するページ分割を行なう効率の良いアルゴリズム、多次元clusteringを管理するハードウェア・アーキテクチャと共に、この点についても研究を進めていく予定である。

［参考文献］

［1］Bentley, J. L, 「Multidimensional Binary Search Trees Used for Associative Searching」, CACM, no. 9, 1975