データベースシステム 107－7
（1996. 3. 11）

分割シグネチャファイル構成法とそれに基づく効率的検索手法の提案と評価

渡辺 悟康* 北川 博之†

*筑波大学 工学研究科
†筑波大学 電子・情報工学系

概要

集合は複雑なデータ構造を支援するデータベース中において、頻繁に現れる基本的なデータ構造である。そのため、各種のデータベース応用において集合値を効率良く支援する索引機構が要求される。我々は、従来テキスト検索に用いられてきたシグネチャファイルを集合値検索機構として取り上げ、検索コストなど各種コストの評価を行なってきた。

本論文では大規模データベースを対象とした、水平分割と垂直分割の概念に基づく一般化された分割シグネチャファイル構成法である、*Partitioned Frame-Sliced Signature File*(P-FSSF)の提案を行ない、集合値検索における検索、更新、格納の各コストの評価を行なう。さらに、シグネチャファイル作成時のパラメタ設定について議論を行ない、P-FSSFの特殊な場合である*Partitioned Bit-Sliced Signature File*(P-BSSF)が多くの場合で最適になることを示す。

Design and Evaluation of Partitioned Signature File Organizations

Noriyasu Watanabe* Hiroyuki Kitagawa†

*Doctoral Degree Program in Engineering, Univ. of Tsukuba
†Institute of Information Sciences and Electronics, Univ. of Tsukuba

abstract

Sets are primitive data objects and often appear in advanced databases which support complex data structures. Therefore, it is desirable to have access facilities which support set-valued object retrieval efficiently. In this paper, we propose the *Partitioned Frame-Sliced Signature File*(P-FSSF), which is based on horizontal and vertical partitioning, and estimate its retrieval, update and storage costs for set-valued object retrieval. Furthermore, we show that the *Partitioned Bit-Sliced Signature File*(P-BSSF), which is a special case of the P-FSSF, is an appropriate organization in general investigating the optimal selections of parameter values.

1 はじめに

近年データベース適用分野の拡大に伴い、単純なレコードだけでなく階層構造などを持つような複雑な対象をデータベース化する要求が高まっている。そのような要求に応えるために、入れ子型リレーショナルモデルや、オブジェクト指向モデル等の研究が進められてきた。これら複雑なデータモデルの操作を支援するためには、それらのデータ構造中に頻繁に現れるデータ型である集合データを効率良く扱うことが重要となる。これまでに入れ子型索引等、階層構造を意識した索引機構がいくつか提案されているが、それらは様々な集合値検索を意図したものではない。

我々は、従来テキスト検索で用いられてきたシグネチャファイル [1] を、集合値検索に適用することを考え、その処理方法やコスト評価についての研究を行なってきた。

シグネチャファイルの物理的構成法としては、*Sequential Signature File*(SSF)、*Bit-Sliced Signature File*(BSSF)、*Frame-Sliced Signature File* (FSSF)[2]、*Quick Filter*[3] 等がこれまでに提案されている。我々はそれらの内、ある種の条件を満たす BSSF が集合値検索において特に有効であることを示し [4]、さらに *Fixed-Prefix* 法 (FP 法)[5] と BSSF を組み合わせた *Partitioned Bit-Sliced Signature File* (P-BSSF) を [6] で提案し、その集合値検索に対する有効性を示した。

本論文では検索コストの低減を目的とし、水平分割と垂直分割の概念に基づく分割シグネチャファイル構成法として、P-BSSF をより一般化した *Partitioned Frame-Sliced Signature File*(P-FSSF) を提案する。そして P-FSSF の集合値検索における検索、更新、格納の各コストの見積りと評価を行なう。

さらに、検索コストと更新コストを考慮に入れたシグネチャファイル作成時のパラメタの設定法について議論を行ない、P-BSSF が多くの場合で最適になることを示す。

2 シグネチャファイルの概要

2.1 シグネチャファイルによる集合値検索

シグネチャ (*signature*) とは、個々のデータオブジェクトから作成される固定長のビット列のことである。本研究で対象とする集合値検索のためのシグネチャの作成法は以下による。

1. 集合データオブジェクトの各要素オブジェクトから、長さが F ビットで、その内 m ビットが "1" にセットされた**要素シグネチャ** (*element signature*) を作成する。またこの時 m をウェイトと呼ぶ。

2. すべての要素シグネチャのビットごとの論理和をとる**スーパーインポーズドコーディング** (*superimposed coding*) を行ない、**集合シグネチャ** (*set signature*) を作成する。

このようにして作成されたシグネチャと、各集合データオブジェクトの識別子 (OID) の組を格納したものを**シグネチャファイル** (*signature file*) とする。図 1に集合 {"DBMS", "OS", "画像処理"} から集合シグネチャが作成される例を示す。

集合の要素		要素シグネチャ
"DBMS"	⟶	00010001
"OS"	⟶	01001000
"画像処理"	⟶	00000101
		⇓ 論理和
集合シグネチャ	⟶	01011101

図 1: 集合シグネチャの作成

問合せが与えられた際に、問合せ条件中に現れる集合を**問合せ集合** (*query set*, Q)、データベース中の検索対象の集合を**ターゲット集合** (*target set*, T) と呼ぶ。また、それぞれから作成される集合シグネチャを、**問合せシグネチャ** (*query signature*, S_Q)、**ターゲットシグネチャ** (*target signature*, S_T) と呼ぶ。

集合値検索の問合せ条件には様々なものが考えられるが、本論文では、T が Q を包含する場合 ($T \supseteq Q$) と、逆に Q が T を包含する場合 ($T \subseteq Q$) を対象とする。それぞれの問合せ条件について、以下の条件を満たすオブジェクトが問合せ条件を満たす候補となる。

$T \supseteq Q$: $S_Q \wedge S_T \equiv S_Q$

$T \subseteq Q$: $S_Q \wedge S_T \equiv S_T$

ここで、"$\wedge$" はビット列の論理積を、"$\equiv$" はビット列が等しいことを示している。

しかし、これらの候補の中には、実際に問合せ条件を満たす**アクチュアルドロップ** (*actual drop*) と、実際には条件を満たさない**フォルスドロップ** (*false drop*) があるので、該当するオブジェクト自身をを取り出してその判定を行なう必要がある。この操作を**フォルスドロップレゾリューション** (*false drop resolution*) と呼ぶ。図 2に $T \supseteq Q$ に対する問合せ処理例を示す。

2.2 検索効率を考慮したシグネチャファイル構成法

もっとも単純なシグネチャファイル構成法である SSF では、シグネチャファイル全体をスキャンする必要があり、そのコストが検索コストに大きな影響を与える。そのため検索効率を考慮したシグネチャファイル構成法がこれまでにも提案されている。ここでは *Bit-Sliced Signature File*(BSSF) と分割シグネチャファイル構成法の一つである Fixed Prefix 法 (FP 法) の二種類について述べる。

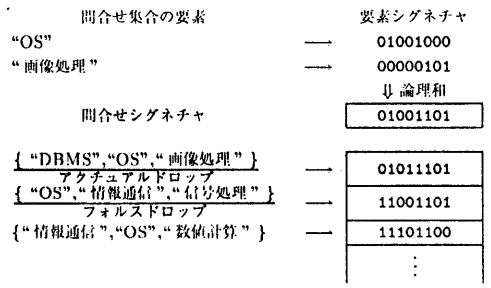

図 2: 問合せ処理 ($T \supseteq Q$)

2.2.1 BSSF

BSSF では、集合シグネチャを 1 ビットずつビットスライスファイル (*bit-slice file*) と呼ばれる別々のファイルに格納する。検索は問合せ条件によって次のように処理される。

1. 与えられた問合せから問合せシグネチャを作成する。
2. $T \supseteq Q$: 問合せシグネチャで "1" の立っているビット位置に対応するビットスライスファイルのみをスキャンする。

   $T \subseteq Q$: 問合せシグネチャで "0" になっているビット位置に対応するビットスライスファイルのみをスキャンする。
3. フォルスドロップレゾリューションを行なう。

長所は、問合せシグネチャのウェイトによっては、検索時のコストがかなり低減できることがあげられる。問題点としては、一般に SSF と比較して更新コストが大きくなってしまうことがあげられる。

BSSF はシグネチャファイルを垂直方向に分割することによって検索コストを削減する構成法といえる。

2.2.2 FP 法

FP 法は、シグネチャの先頭数ビットをキーとして、シグネチャを複数のパーティションに分割格納する、分割シグネチャファイル構成法の一手法である。検索の時は、検索対象のパーティションの絞り込みが可能なためすべてのパーティションをスキャンする必要がないので、検索処理が高速になる。しかし問合せ集合の要素数が、$T \supseteq Q$ では小さいと、$T \subseteq Q$ では大きいとパーティションの絞り込みが十分できず、ほぼすべてのシグネチャをスキャンする必要が生じる。

FP 法はシグネチャファイルを水平方向に分割することによって検索コストを削減する構成法といえる。

3 Partitioned Frame-Sliced Signature File (P-FSSF)

Partitioned Frame-Sliced Signature File(P-FSSF) は FP 法と FSSF を組み合わせ、水平分割、垂直分割共に行なうことによって、検索コストの低減を図ったシグネチャファイル構成法である。

P-FSSF の構成と、$T \supseteq Q$ の問合せの際にスキャンするシグネチャファイルを図 3に示す。

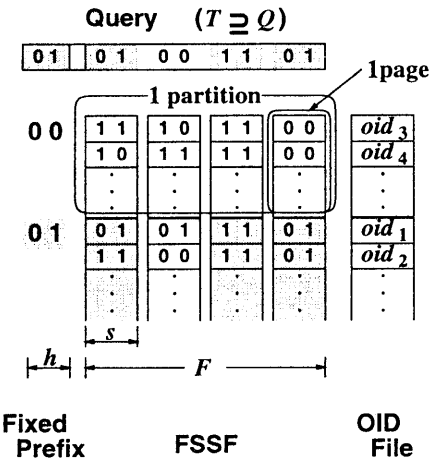

図 3: P-FSSF の構成と $T \supseteq Q$ の問合せ処理例

3.1 P-FSSF の作成法

P-FSSF は次のように作成される。

1. 長さ f ビット、ウェイトをターゲットシグネチャ中の "0" と "1" の生じる確率がそれぞれ 0.5 となる $m_{opt}(= \frac{f}{D_t} \ln 2)$ ビットとして、ターゲットシグネチャを作成し、その先頭 h ビットのビットパターン (Prefix, $S_{T\{FP\}}$) によりシグネチャを格納するパーティションを決定する。ここでウェイトを m_{opt} とするのは、各パーティションへの割り当ての均衡を図るためである。
2. s ビットのフレーム k 個の内、n フレームに m ビットづつビットをセットする。それをターゲットシグネチャ $S_{T\{FSSF\}}$ として、対応する FSSF 部に格納する。
3. 対応する OID を OID ファイルに格納する。

一つの集合値に対して Fixed Prefix 部用と FSSF 部用の二種類のターゲットシグネチャを作成するのが一つの特徴である。また、FSSF 部の各フレームスライスファイルの第 i 物理ページからなる k ページの集まりをパーティションと呼ぶ。

3.2 検索処理

問合せ集合が与えられると次の処理を行なう。

1. ターゲットシグネチャと同様に、FSSF 部用シグネチャ ($S_{Q\{FSSF\}}$) と Prefix 用のシグネチャ ($S_{Q\{FP\}}$) の二種類の問合せシグネチャを作成する。

2. $S_{Q\{FP\}}$ を用いて、パーティションの絞り込みを行なう。次の条件を満たすパーティションが検索対象となる。

$$T \supseteq Q \ : \ S_{T\{FP\}} \wedge S_{Q\{FP\}} \equiv S_{Q\{FP\}}$$
$$T \subseteq Q \ : \ S_{T\{FP\}} \wedge S_{Q\{FP\}} \equiv S_{T\{FP\}}$$

3. 検索しなければならないパーティション中の FSSF 部のターゲットシグネチャ ($S_{T\{FSSF\}}$) と、$S_{Q\{FSSF\}}$ とのマッチングを行ない、問合せ条件を満たす候補を取り出す。その時に候補となる $S_{T\{FSSF\}}$ は、問合せ条件によって次のようになる。

$$T \supseteq Q \ : \ S_{T\{FSSF\}} \wedge S_{Q\{FSSF\}} \equiv S_{Q\{FSSF\}}$$
$$T \subseteq Q \ : \ S_{T\{FSSF\}} \wedge S_{Q\{FSSF\}} \equiv S_{Q\{FSSF\}}$$

4. フォルスドロップレゾリューションを行ない、アクチュアルドロップを返す。

3.3 更新処理

更新処理として、挿入処理と削除処理について述べる。その際、更新するオブジェクトのターゲットシグネチャ、$S_{T\{FSSF\}}$ 及び $S_{T\{FP\}}$ と OID が分かっているものとする。

3.3.1 挿入処理

P-FSSF に新しいターゲットシグネチャを挿入する場合、以下の手順で処理を行なう。

1. $S_{T\{FP\}}$ により、挿入すべきパーティションを決定する。

2. そのパーティションの OID ファイルをスキャンし、削除ビット[1]が立っている場所を見つけ、そこにオブジェクトの OID を挿入する。

3. 対応する削除ビットをリセットする。

4. $S_{T\{FSSF\}}$ で "1" の立っているフレームについて、パーティション中の FSSF 部に "1" を立てる。

[1]OID ファイル中にあるビットで、その位置に有効なオブジェクトが存在するかしないかの情報を持っている。

3.3.2 削除処理

P-FSSF からオブジェクトを削除する場合、以下の手順で処理を行なう。

1. $S_{T\{FP\}}$ により、そのシグネチャが含まれているパーティションを決定する。

2. そのパーティション中の対応する OID ファイルをスキャンし該当する OID があれば削除ビットを立てる。

3. FSSF 部のビットの立っているフレーム中のビットを "0" にリセットする。

4 コストモデル

P-FSSF の検索コスト RC、更新コスト UC、格納コスト SC のコストモデルを作成する。検索コスト、更新コストは物理ページの I/O の回数を、格納コストは格納するのに必要な物理ページ数をコストとする。コストモデルの作成の際に使う記号の定義を表1に示す。

記号	定義
D_t	ターゲット集合の要素数
D_q	問合せ集合の要素数
N	オブジェクトの総数
P	1 ページのバイト数
b	1 バイトのビット数
oid	OID のバイト長
P_o	1 オブジェクトを取り出すためのページアクセス数
F, f	シグネチャのビット長
A	アクチュアルドロップ数
Fd	フォルスドロップ確率
pl	パーティションの充足率
h	FP 部のキーのビット長
n	FSSF 部でビットを立てるフレーム数
s	FSSF 部のフレームのビット長
k	FSSF 部のフレーム数 ($k \cdot s = F$)
m	FSSF 部で 1 フレーム中に立てるビットの数

表 1: 記号の定義

4.1 フォルスドロップ確率

シグネチャファイルを用いた検索ではフォルスドロップが発生する。これは異なる要素を同じ要素シグネチャにハッシングしたり、スーパーインポーズドコーディングを行なうことによって生じる。

このフォルスドロップ確率は、(1) 式で定義される [1]。

$$Fd = \frac{\text{フォルスドロップ数}}{N - \text{アクチュアルドロップ数}} \qquad (1)$$

以下でFSSFのフォルスドロップ確率の見積りを行なう。

あるフレームにビットが立っている確率は次のようになる。

$$1-\left(1-\frac{n}{k}\right)^{D_t}$$

問合せ、ターゲット各シグネチャでビットの立つフレームの期待値は、それぞれ(2)、(3)式で表される。

$$n_q = k\left(1-\left(1-\frac{n}{k}\right)^{D_q}\right) \quad (2)$$

$$n_t = k\left(1-\left(1-\frac{n}{k}\right)^{D_t}\right) \quad (3)$$

各シグネチャにおいて、ビットが立っているフレームを選んだ要素数の期待値は、それぞれ(4)、(5)式で表される。

$$D'_q = \frac{nD_q}{n_q} \quad (4)$$

$$D'_t = \frac{nD_t}{n_t} \quad (5)$$

各シグネチャにおいて、ビットの立っているフレーム中のウェイトの期待値は、それぞれ(6)、(7)式で表される。

$$m_q = s\left(1-\left(1-\frac{m}{s}\right)^{D'_q}\right) \quad (6)$$

$$m_t = s\left(1-\left(1-\frac{m}{s}\right)^{D'_t}\right) \quad (7)$$

以上より、$T \supseteq Q, T \subseteq Q$のフォルスドロップ確率はそれぞれ(8), (9)式で表される。

$$Fd_{\{T\supseteq Q\}} = \left\{\left(1-\left(1-\frac{n}{k}\right)^{D_t}\right)\cdot\left(1-\left(1-\frac{m}{s}\right)^{D'_t}\right)^{m_q}\right\}^{n_q} \quad (8)$$

$$Fd_{\{T\subseteq Q\}} = \left\{\left(1-\left(1-\frac{n}{k}\right)^{D_q}\right)\cdot\left(1-\left(1-\frac{m}{s}\right)^{D'_q}\right)^{m_t}\right\}^{n_t} \quad (9)$$

4.2 検索コストモデル

検索コストRCは、Prefixで絞り込まれたパーティション中のFSSF部をスキャンするコスト(LC_{sig})、OIDファイルにアクセスするコスト(LC_{OID})、フォルスドロップレゾリューションを行なうコストの和になり、(10)式で表される。

$$RC = LC_{sig} + LC_{OID} + P_o(A + PAR \cdot Fd(N-A)) \quad (10)$$

LC_{sig}は問合せ条件によって(11), (12)式で表される。

$$LC_{sig\{T\supseteq Q\}} = n_q \cdot PAR_{\{T\supseteq Q\}} \cdot \left\lceil \frac{N}{\lfloor \frac{Pb\cdot pl}{s} \rfloor} \right\rceil \quad (11)$$

$$LC_{sig\{T\subseteq Q\}} = (k - n_q \cdot p_q) \cdot PAR_{\{T\subseteq Q\}} \cdot \left\lceil \frac{N}{\lfloor \frac{Pb\cdot pl}{s} \rfloor} \right\rceil \quad (12)$$

ここで、p_qはそのフレーム中のビットがすべて"1"になる確率を表している。

PARは検索する必要のあるパーティションの割合(*Partition Activation Ratio*)を表し、$w(S_Q)$を問合せシグネチャのウェイトとすると(13)、(14)式で表される。

$$PAR_{\{T\supseteq Q\}} \approx \left(1-\frac{w(S_Q)}{2f}\right)^h \quad (13)$$

$$PAR_{\{T\subseteq Q\}} \approx \left(\frac{f+w(S_Q)}{2f}\right)^h \quad (14)$$

LC_{OID}は、SC_{OID}をOIDファイルの格納総ページ数とすると、Yaoの関数[7]を用いて(15)式で表される。

$$LC_{OID} = SC_{OID} \cdot \left[1-\prod_{i=1}^{A+Fd(N-A)} \frac{N\cdot(1-1/SC_{OID})-i+1}{N-i+1}\right] \quad (15)$$

4.3 更新コストモデル

更新は挿入、削除ともにまずPrefixを用いて、更新するパーティションを決定する。次に、そのパーティションに対応するOIDファイルを探索し、最後にFSSF部にビットをセット、またはリセットする。よって更新コストは、(16)式で表される。

$$UC = 2(n_t + 1) \quad (16)$$

4.4 格納コストモデル

P-FSSFの格納コストは、FSSF部の格納コストとOIDファイルの格納コストの和となり、(17)式で表される。

$$SC = \left\lceil \frac{N}{\lfloor \frac{Pb\cdot pl}{s} \rfloor} \right\rceil + \left\lceil \frac{N}{\lfloor \frac{P\cdot pl}{oid} \rfloor} \right\rceil \quad (17)$$

5 コスト解析

前章で示した見積り式を基に、検索、更新、格納の各コストの解析を行なっていく。表2にパラメタの値を示す。

記号	値
D_t	ターゲット集合の要素数 (=100)
N	オブジェクトの総数 (=800,000)
P	1 ページのバイト数 (=4096)
b	1 バイトのビット数 (=8)
oid	OID のバイト長 (= 8)
P_o	1 オブジェクトを取り出すためのページアクセス数 (=1)
F	シグネチャのビット長 (=1024)
pl	パーティションの充足率 (= 0.75)

表 2: 各パラメタの値

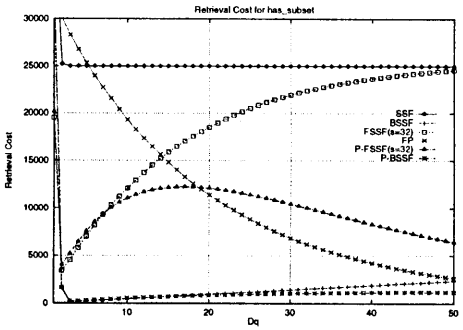

図 4: $T \supseteq Q$ の場合の検索コスト

5.1 検索コスト

5.1.1 $T \supseteq Q$ の検索コスト

$T \supseteq Q$ の検索コストを図 4に示す。

図 4で D_q が 1 ~ 2 のときコストが大きくなっているのは、フォルスドロップが多いためにフォルスドロップレゾリューションを行なうコストが高くなるためである。逆に D_q がそれ以上の時は、シグネチャファイルをスキャンするコストが主要なコストとなっている。

$D_q \geq 2$ では BSSF, P-BSSF が最もコストが低くなっていることが分かる。

5.1.2 $T \subseteq Q$ の検索コスト

$T \subseteq Q$ の検索コストを図 5に示す。

FSSF, P-FSSF で D_q が 600 以上でコストが急増しているのは、フォルスドロップが増加したためである。D_q が 600 以下ではフォルスドロップは少なく、コストのほとんどはシグネチャファイルのスキャンコストとなる。

D_q が 100 ~ 300 では、FP 法, P-FSSF, P-BSSF がコストが低くなる。しかし、D_q が 600 以上では FP 法, P-FSSF のコストは大きくなってしまう。それに

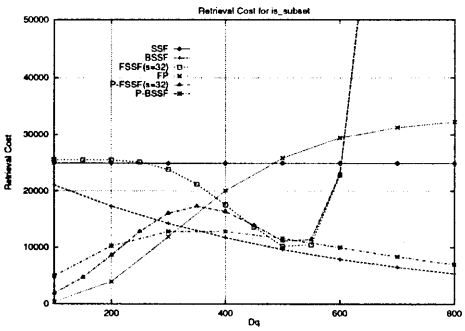

図 5: $T \subseteq Q$ の場合の検索コスト

対し、BSSF, P-BSSF のコストが低くなる。

5.2 更新コスト

仮定として、SSF, BSSF, FSSF は新しいシグネチャの挿入をファイルの最後尾に行なうものとする。その際の、各シグネチャファイル構成法の更新コストを表 3に示す。

構成法	挿入コスト	削除コスト
SSF	4	783
BSSF	366	1147
FSSF($n=2, s=32, m=4$)	64	847
FP 法	4	2
P-FSSF($n=2, s=32, m=4$)	64	64
P-BSSF	366	366

表 3: 更新コスト

挿入コストは SSF, FP 法が最も低くなる。削除コストは FP 法が最も低くなる。これは削除するシグネチャが、そのキーによってどのパーティションに存在するのかが分かるからである。同様に、P-FSSF も FP 部で削除するシグネチャの場所が特定できるので、FP 法についでコストが低くなっている。P-BSSF でもパーティションは特定できるが、BSSF を用いているためコストは大きくなってしまう。

5.3 格納コスト

各シグネチャファイル構成法の格納コストを表 4に示す。

SSF, BSSF, FSSF が格納コストは小さくなる。それに対し、FP 法, P-FSSF, P-BSSF は充足率 (pl) を 75% としているためにその分コストが大きくなっている。

構成法	格納コスト
SSF	26563
BSSF	27163
FSSF($n=2, s=32, m=4$)	26587
FP 法	34897
P-FSSF($n=2, s=32, m=4$)	35428
P-BSSF	35355

表 4: 格納コスト

6 シグネチャファイル作成時のパラメタ設定

前章より、検索コストは BSSF, P-BSSF が、更新コストは SSF, FP 法が最も低くなることが分かった。しかし実際のデータベースアプリケーションは検索または、更新のみ発生することは少ない。そこで本章では、シグネチャファイルを作成する際に、どのシグネチャファイルを作成するのが望ましいか、また作成時のパラメタはどのようにすればよいのか、について議論していく。

本章では、あらかじめ決められたシグネチャファイルの大きさ内で、検索コストと更新コストを考慮した総コストが最適になるようなパラメタ設定について述べていく。総コスト C を (18) 式のように定義する。

$$C = (1-\delta)RC + \delta \cdot UC \qquad (18)$$

ここで、RC, UC はそれぞれ検索、更新コストを表す。また、δ は更新処理の発生する頻度を表す。

$\delta = 0$ は検索のみで更新は行なわれない場合を、$\delta = 0.5$ は検索と更新が等しい頻度で発生する場合を表している。

まず FSSF と P-FSSF では、P-FSSF の方が図 4, 5、表 3より検索、更新両コストともに低くなる。よって以下では、P-FSSF のパラメタ設定についてのみ検討を行なう。

これまでに [2] で、FSSF のパラメタ設定方法として山登り法を用いた手法が提案されている。そこで本章では P-FSSF のパラメタ設定について、山登り法を適用し総コスト C を評価することにする。

今回検討を行なった範囲を以下に示す。

- $N = 800,000, pl = 0.75$
- $D_t = 10, F = 256$ と $D_t = 100, F = 1024$
- 検索条件は、$T \supseteq Q$ と $T \subseteq Q$
- δ は、0, 0.25, 0.5 の 3 通り

その結果、P-FSSF の最適なパラメタは多くの場合で $s = 1$ となった。つまり、P-BSSF が最適な構成法となることが分かった。表 5に P-BSSF が最適になる場合の割合を示す。

		$\delta = 0.0$	$\delta = 0.25$	$\delta = 0.5$
$D_t = 10$	$T \supseteq Q$	100%	100%	100%
$F = 256$	$T \subseteq Q$	70%	70%	70%
$D_t = 100$	$T \supseteq Q$	100%	100%	100%
$F = 1024$	$T \subseteq Q$	85%	85%	60%

表 5: 最適な構成法が P-BSSF になる場合の割合

表 5より検索条件が $T \supseteq Q$ の場合は、検討した範囲では、すべて P-BSSF が最適になることが分かる。$T \subseteq Q$ の場合でも多くの場合で P-BSSF が最適になることが分かった。

7 スマート検索

スマート検索 (*smart retrieval*) とは、[4] において提案された BSSF における検索効率化の一手法である。

シグネチャファイルを用いた検索では、候補となったオブジェクトが問合せ条件を満たすかどうかという、最終的な判定はフォルスドロップレゾリューションの段階で行なわれる。よって、フォルスドロップが多少増加しても、スキャンするビットスライスファイルの数を減らした方が、検索コスト全体から見れば有利になることがある。この性質を利用した手法がスマート検索である。

P-BSSF の中には BSSF 部が存在するので、P-BSSF でも同様にスマート検索を行なうことが可能である。

P-BSSF の $T \supseteq Q$ における検索コストについて検討する。まず、図 4より P-BSSF の検索コストは $D_q = 4$ のときに最小となることが分かる。この場合の検索コストは、$D_q \leq 4$ ではフォルスドロップレゾリューションのコスト、$D_q > 4$ ではシグネチャファイルをスキャンするコストが中心となる。特に $D_q > 4$ では、フォルスドロップはほとんど発生せず、フォルスドロップレゾリューションのコストは無視できる。したがって、$D_q > 4$ でのコストの増加は無駄なシグネチャのスキャンを削減することで低減できる。

この場合、スマート検索の処理手順は以下のようになる。

1. 実際の問合せの要素数が $D_q \leq 4$ であれば、通常通りの検索を行なう。
2. $D_q > 4$ ならば、問合せ集合の全要素から Prefix を作成し、スキャンするパーティションを決定する。続けて任意の 4 要素から BSSF 部の問合せシグネチャを作成し、検索を行なう。

スマート検索を行なった場合の検索コストを、図 6, 7に示す。

図 6, 7より、$T \supseteq Q$ では D_q が大きいほど、$T \subseteq Q$ では D_q が小さいほど、スマート検索による効果

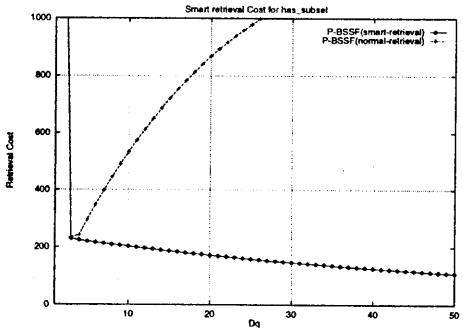

図 6: $T \supseteq Q$ の検索コスト

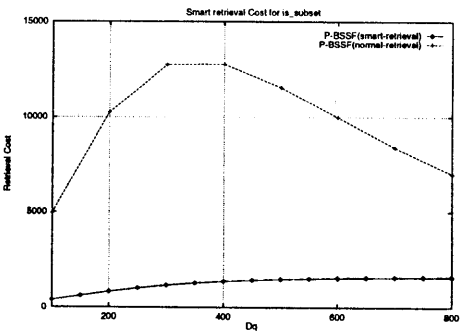

図 7: $T \subseteq Q$ の検索コスト

が大きく、検索コストが大幅に削減できているのが分かる。

8 まとめ

本論文では検索対象の多い大規模データベースにおいて、シグネチャファイルによる検索コストの低減を目的とし、分割シグネチャファイル構成法として、P-BSSF をより一般化した P-FSSF の提案を行なった。また、P-FSSF の検索、更新、格納の各コストの見積りと評価を行ない、その他のシグネチャファイル構成法との比較を行なった。特に検索については集合値検索を対象として、$T \supseteq Q$ と $T \subseteq Q$ の二種類の検索条件について解析を行なった。また更新コストは、挿入、削除の両コストについて検討を行ない、各コストを他のシグネチャファイル構成法と比較した。

その結果、$T \supseteq Q$ の検索コストは BSSF, P-BSSF が効率的であることが分かった。また、$T \subseteq Q$ の場合、D_q が 100 ~ 300 では FP 法, P-FSSF, P-BSSF が、D_q が 600 以上では、P-BSSF, BSSF が効率的であることが分かった。一方、挿入コストは SSF, FP 法が、削除コストは FP 法が最も低いコストとなり、逆に BSSF, P-BSSF は挿入、削除ともにコストが大きくなってしまうことが分かった。また格納コストは、SSF, FSSF, BSSF が小さくなり、FP 法, P-FSSF, P-BSSF は充足率を 75% としているためにその分大きくなることが分かった。

さらに本論文では、これら各コストの結果から、P-FSSF の最適なパラメタの設定について議論した。更新処理がある一定の頻度 δ で発生することを仮定し、総コスト C を山登り法を用いて評価した。その結果今回考察した範囲では、検索条件が $T \supseteq Q$ のときは 100%、$T \subseteq Q$ のときでも 60 ~ 80% の範囲で P-BSSF を用いるのが、最も効率的であることを示した。

また、P-BSSF はスマート検索を適用することによって、さらに検索コストを低減できることが分かった。

今回の見積りでは検索、更新コストはページアクセス数のみで評価を行なっているので、実時間を考慮した見積りを行なうことや、実データを対象とした有効性及び性能評価などが今後の課題としてあげられる。

謝辞

シグネチャファイルに関して共に御検討いただいた奈良先端科学技術大学院大学の石川佳治助手、ならびに筑波大学データベース研究室の諸氏に感謝致します。

参考文献

[1] C. Faloutsos and S. Christodoulakis. "Signature Files: An Access Method for Documents and Its Analytical Performance Evaluation". *ACM Trans. Off. Inf. Syst.*, Vol. 2, No. 4, pp. 267-288, 1984.

[2] Z. Lin and C. Faloutsos. "Frame-Sliced Signature Files". *IEEE Trans. Knowl. and Data Eng.*, Vol. 4, No. 3, pp. 281 289, 1992.

[3] P. Zezula, F. Rabitti, and P. Tiberio. "Dynamic Partitioning of Signature Files". *ACM Trans. Inf. Syst.*, Vol. 9, No. 4, pp. 336 369, 1991.

[4] Y. Ishikawa, H. Kitagawa, and N. Ohbo. "Evaluation of Signature Files as Set Access Facilities in OODBs". In *Proc. ACM SIGMOD*, pp. 247 256, 1993.

[5] D. L. Lee and C. Leng. "Partitioned Signature Files: Design Issues and Performance Evaluation". *ACM Trans. Off. Inf. Syst.*, Vol. 7, No. 2, pp. 158 180, 1989.

[6] 渡辺悟康, 北川博之. "ビットスライス方式に基づく分割シグネチャファイル構成法の提案と評価". 情報処理学会データベースシステム研究会報告, DBS-104, 1995.

[7] S.B. Yao. "Approximating Block Accesses in Database Organizations". *Communications of ACM*, Vol. 20, No. 4, pp. 260 261, 1977.