分割ビットスライストシグネチャファイルの提案と集合値検索への適用

渡辺悟康[†,☆] 北川博之[††]

今日，データベースシステムには，文書，画像，音声など多様なデータや複雑な構造を持つデータを効率良く格納し，検索することが求められている．シグネチャファイルは，従来主にテキストデータ検索の効率化を目的として研究開発されてきたが，近年レコード検索，画像検索，集合値検索などへの適用が検討されている．本論文では，シグネチャファイルの検索コストの低減を目的とし，分割シグネチャファイルにおける水平分割の考え方と，ビットスライストシグネチャファイルにおける垂直分割の考え方を組み合わせた新しいシグネチャファイル構成法である，分割ビットスライストシグネチャファイルの提案を行う．また，分割ビットスライストシグネチャファイルの検索コスト，更新コストおよび格納コストについて，集合値検索を対象として見積りを行い，その有効性を検討する．さらにまた，データの挿入，削除が頻繁な動的な環境により適合したビットスライストクイックフィルタを提案し，そのコスト評価をあわせて行う．

Partitioned Bit-Sliced Signature File and Its Application to Set-valued Object Retrieval

NORIYASU WATANABE[†,☆] and HIROYUKI KITAGAWA[††]

Modern database systems have to support storage and retrieval of a variety of data objects including documents, image, audio, and so on. The signature file is one of promising access methods to attain efficient manipulation of such data objects. In this paper, we propose new signature file organizations, named *Partitioned Bit-Sliced Signature File* (P-BSSF), combining the horizontal decomposition scheme of the partitioned signature file and the vertical decomposition scheme of the bit-sliced signature file. We estimate retrieval, update and storage costs for P-BSSF, and show that P-BSSF is very effective signature file organization. We also propose *Bit-Sliced Quick Filter* (BSQF), which is more suitable to dynamic environments where data objects are often inserted and deleted.

1. はじめに

今日，データベースシステムには，文書，画像，音声など多様なデータや複雑な構造を持つデータを効率良く格納し，検索することが求められている．シグネチャファイルは，従来主にテキストデータ検索の効率化を目的として研究開発されてきたが[3]，近年レコード検索[12]，画像検索[11]，集合値検索[4),6),17]，オブジェクト指向データベースにおけるデータナビゲーション支援[8),14]などへの適用が検討されている．

シグネチャとは，対象データオブジェクト O_i の特徴情報を表現したあるビット長 F のビット列 $S_i = b_{i1} \cdots b_{iF}$ であり，対象オブジェクト群 $O_1, \cdots, O_N$ に対するシグネチャ群 $S_1, \cdots, S_N$ をまとめたものがシグネチャファイルである．

シグネチャファイルによるデータベース検索では，シグネチャファイルをスキャンするコストが大きなコスト要因となる．そのため，そのコストを低減するためにこれまでいくつかのシグネチャファイル構成法が提案されてきた．それらの構成法は，次の 2 つのグループに分類することができる．

(1) 各シグネチャを先頭から s ビットずつの部分シグネチャに分割し，各部分シグネチャごとにまとめて格納する方法

(2) シグネチャをその一部のビットパターンに基づ

† 筑波大学工学研究科
Doctoral Degree Program in Engineering, University of Tsukuba
☆ 現在，NTT データ通信株式会社
Presently with NTT DATA Corporation
†† 筑波大学電子・情報工学系
Institute of Information Sciences and Electronics, University of Tsukuba

いて，パーティションと呼ばれる単位に分割して格納する方法

本論文では，(1)のような構成法を**垂直分割**を行う構成法，(2)のような構成法を**水平分割**を行う構成法と呼ぶ．垂直分割を行う構成法としては，**フレームスライストシグネチャファイル**（*Frame-Sliced Signature File*: FSSF）[9]や，その特殊な場合であるビットスライストシグネチャファイル（*Bit-Sliced Signature File*: BSSF）[12]がある．また，水平分割を行う構成法としては，**分割シグネチャファイル**（*Partitioned Signature File*）[7]，**クイックフィルタ**（*Quick Filter*）[16]等がこれまでに提案されている．これらの具体的構成法は2.3節で述べるが，いずれもシグネチャファイル中の全 $N \cdot F$ ビットを読むのではなく，その一部のみを読むことで検索を効率化することを目的としたものである．

本論文では，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，水平分割を行う構成法である分割シグネチャファイルと，垂直分割を行う構成法であるBSSFを組み合わせた新しいシグネチャファイル構成法である，**分割ビットスライストシグネチャファイル**（*Partitioned Bit-Sliced Signature File*: P-BSSF）を提案する．そして，これまで我々が主に研究の対象としてきた集合値検索[4),6),17)]を対象として，分割ビットスライストシグネチャファイルによる検索コスト，更新コストおよび格納コストの見積りを行う．そしてそれら見積り式を用いて，従来のシグネチャファイル構成法とコスト比較を行うことにより，その有効性の評価を行う．特に，検索コストについては集合の包含関係（$\supseteq$）に関する2種類の問合せ（$T \supseteq Q$, $T \subseteq Q$）☆について検討を行う．またさらに，パーティションの充足率が検索コストに与える影響について考察を行い，データオブジェクトの挿入や削除の多い動的な環境により適合した，**ビットスライストクイックフィルタ**（*Bit-Sliced Quick Filter*: BSQF）の提案と評価を行う．

筆者らは文献18), 19)において，分割ビットスライストシグネチャファイルの提案とその基本コストの評価を示した．我々の研究以外に水平分割の概念と垂直分割の概念を組み合わせたシグネチャファイル構成法の提案としては，拡張ハッシング[2]とFSSFを組み合わせた，*Two-dimensional Dynamic Signature File* (TDSF)[5]の提案がある．また，筆者らの提案[18),19)]とは独立に，Zezulaらは分割シグネチャファイルとBSSFを組み合わせた，*Key-Based Bit-Sliced Signature File* の提案を行っている[15]．これらの従来研究と対比した本研究の特徴を以下に示す．

(1) TDSFとKey-Based Bit-Sliced Signature Fileでは，直接元のシグネチャの一部に基づいて水平分割を行う．このため，シグネチャを均等にパーティションに分割するためには，各シグネチャの半分のビットが"1"にセットされていなければならない．一方P-BSSFでは，水平分割を均等に行うためのシグネチャを元のシグネチャとは別に作成する．これにより，P-BSSFではより適用対象に適したパラメータ設定をとることができる．たとえば文献4), 17)ではある種の問合せの場合，"1"をセットするビットを少なくする方がコスト的に有利になることが示されている．TDSFやKey-Based Bit-Sliced Signature Fileではこのような目的に応じたパラメータ設定を行うことはできない．この点に関する定量的比較も本論文中に示す．

(2) 従来研究では問合せ条件として，$T \supseteq Q$ のみを対象としている．しかし，本研究ではより一般的な集合値検索を考慮しているため，$T \supseteq Q$ のみでなく，$T \subseteq Q$ の場合についても検討を行っている．

(3) 本研究では，パーティションの充足率が検索コストに与える影響について考察を行い，その結果に基づきビットスライストクイックフィルタの提案を行っている．充足率についての検討は従来研究では行われていない．

(4) BSSFにおける検索効率化の一手法として，**スマート検索**（*smart retrieval*）を文献4), 17)で提案しており，本論文でもP-BSSFにスマート検索を適用した場合についての考察を行っている．スマート検索は従来研究では考慮されていない．

本論文の以下の部分の構成は次のとおりである．2章では，シグネチャファイルを用いた集合値検索と，これまでに提案されているシグネチャファイル構成法のうち，本研究のベースとなるBSSFと分割シグネチャファイル構成法の一種であるFP分割シグネチャファイルについて述べる．3章では，本論文で提案するP-BSSFの構成と各処理について述べる．4章では，性能評価のためのコストモデルについて説明する．5章では，通常検索とスマート検索の2通りの検索処理方法のコスト比較，P-BSSFの改良型のBSQFの提案，P-BSSFとKey-Based Bit-Sliced Signature File

☆ T はデータベース中の集合，Q は問合せ条件で与える集合を示す．これらの詳細については後述する．

のコスト比較を行う．6章では更新コスト，7章では格納コストの面からBSSF，P-BSSF，BSQF，Key-Based Bit-Sliced Signature Fileの比較評価を行う．8章は本論文のまとめである．

2. 基本概念

2.1 集合値検索

例として図1の学生リレーションを考える．図1では履修科目の属性値が集合値となっている．この学生リレーションに対する問合せとして，次の問合せQuery1を考える．

Query1：

select 名前
from 学生
where 履修科目 $\supseteq$ { "OS", "画像処理" }

Query1は，"OS" と "画像処理" の両方を履修科目に含む学生の名前を求める問合せである．問合せ条件中に現れる集合 { "OS", "画像処理" } を**問合せ集合**（*query set*, Q）と呼び，データベース中に格納され問合せ集合との比較の対象となる集合を**ターゲット集合**（*target set*, T）と呼ぶ．Query1は，問合せ集合を部分集合として含むターゲット集合を持つデータオブジェクトを検索する問合せである．このような問合せを $T \supseteq Q$（*has-subset*）の問合せと呼ぶ．これに対し，問合せ集合の部分集合となるターゲット集合に対応するオブジェクトを検索する問合せを，$T \subseteq Q$（*is-subset*）の問合せと呼ぶ．

本研究では，シグネチャファイルを用いた検索処理として，ここに述べたような集合値検索を対象とする．従来シグネチャファイルの主な適用対象として考えられてきたテキストオブジェクトに対するキーワード検索[3]は，$T \supseteq Q$ の集合値検索の一種と見なすことができる．

2.2 シグネチャファイルを用いた集合値検索

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 F のビット列のことであり，シグネチャを対応するデータオブジェクトのOIDとともに格納したものが**シグネチャファイル**（*Signature File*）[3]である．本論文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[4),6),17)]．

(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，そのうち 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．またこのとき m を**ウェイト**（*weight*）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*）を行い，**集合シグネチャ**（*set signature*）を作成する．

学生		
OID	名前	履修科目
oid1	田中	"DBMS" "OS" "画像処理"
oid2	佐藤	"OS" "情報通信" "信号処理"

図1 集合値を含むデータベースの例
Fig. 1 Sample database having set attribute values.

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2 集合シグネチャの作成
Fig. 2 Creation of a set signature.

図2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．問合せ集合，ターゲット集合，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．

$T \supseteq Q$，$T \subseteq Q$，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す．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，該当する候補オブジェクト自身を取り出してその判定を行う必要がある．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3に $T \supseteq Q$ に対する問合せ処理例を示す．

2.3 検索効率を向上を目的とした既存のシグネチャファイル構成法

全シグネチャ群を逐次ファイルに格納した**シーケンシャルシグネチャファイル**（*Sequential Signature File*: SSF）では，検索処理の際にシグネチャファイル全体をスキャンする必要があり，それが検索における大きなコスト要因となる．そのため，そのコスト

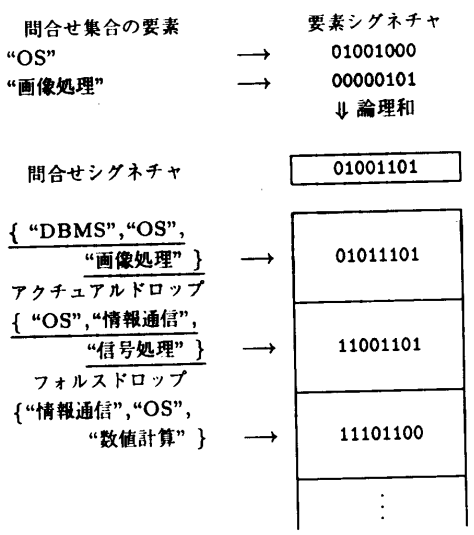

図3 問合せ処理 ($T \supseteq Q$)
Fig. 3 Query processing ($T \supseteq Q$).

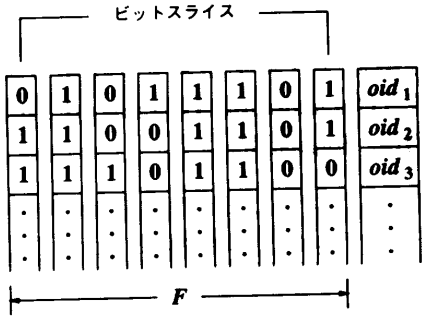

図4 BSSF の構成
Fig. 4 Structure of BSSF.

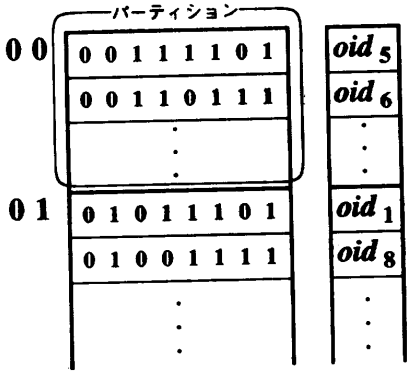

図5 FP 分割シグネチャファイルの構成
Fig. 5 Structure of FP-partitioned signature file.

を低減するためのシグネチャファイル構成法がこれまでに提案されている．ここでは垂直分割を行うビットスライストシグネチャファイル（*Bit-Sliced Signature File*: BSSF）と，水平分割を行う分割シグネチャファイル構成法のひとつである FP 分割シグネチャファイル（*FP-Partitioned Signature File*）[7]の 2 種類について述べる．

2.3.1 ビットスライストシグネチャファイル (Bit-Sliced Signature File: BSSF)

垂直分割を行う構成法である BSSF では，集合シグネチャをビット位置ごとにビットスライス（*bit-slice*）と呼ばれる単位で格納する．シグネチャをビットスライスに分割することによって，検索処理上読む必要のあるビット位置のみへのアクセスが可能となり，検索効率を向上させることができる．**図4** に BSSF の構成を示す．

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う．

BSSF では，上に述べた理由により，問合せシグネチャのウェイトが $T \supseteq Q$ では小さいほど，$T \subseteq Q$ では大きいほど，検索コストを低減することができる．問題点としては，1 つのシグネチャを更新するためには，一般に F 個のビットスライスにアクセスする必要があるので，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る．

2.3.2 FP 分割シグネチャファイル (FP-Partitioned Signature File)

水平分割を行う構成法である FP 分割シグネチャファイルは，シグネチャの先頭数ビット（プレフィクス）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．この分割の方法は，フィクスドプレフィクス（Fixed-Prefix）法と呼ばれる．FP 分割シグネチャファイルはシグネチャを各パーティションに分割することによって，検索処理上読む必要のあるパーティションのみへのアクセスが可能となり，検索効率を向上させることができる．**図5** に FP 分割シグネチャファイルの構成を示す．

検索のときは，問合せシグネチャのプレフィクスを用いることで検索対象のパーティションの絞り込みが可能であり，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．特に問合せシグネチャのウェイトが，$T \supseteq Q$ では大きいほど，$T \subseteq Q$ では小さいほど，パーティションの絞り込みを行うこ

とができる．

3. 分割ビットスライストシグネチャファイル（Partitioned Bit-Sliced Signature File: P-BSSF）

前章で述べたように，BSSF は問合せシグネチャのウェイトが，$T \supseteq Q$ では小さいとき，$T \subseteq Q$ では大きいとき有効である．一方，FP 分割シグネチャファイルは，逆に問合せシグネチャのウェイトが，$T \supseteq Q$ では大きいとき，$T \subseteq Q$ では小さいとき有効である．したがって，これらの両者を組み合わせることでより効率的なシグネチャファイル構成法とすることができると考えられる[18),19)]．

以下では，前述の BSSF と FP 分割シグネチャファイルを組み合わせた，分割ビットスライストシグネチャファイル（*Partitioned Bit-Sliced Signature File*: P-BSSF）の提案と，その基本処理手順を示す．

3.1 P-BSSF の作成法

P-BSSF 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} ($= f/D_t \cdot \ln 2$, D_t はターゲット集合の要素数)[3)] ビットとして，ターゲットシグネチャ $S_{T\{FP\}}$ を作成し，その先頭 h ビットのビットパターン（プレフィクス，$S'_{T\{FP\}}$）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シグネチャの割当ての均衡を図るためである．

(2) 長さ F ビット，ウェイト m ビットでターゲットシグネチャ $S_{T\{BSSF\}}$ を作成し，対応する BSSF 部に格納する．

(3) 対応する OID を OID ファイルに格納する．

1 つの集合値に対してプレフィクス部用と BSSF 部用の 2 種類のターゲットシグネチャを作成するのがひとつの特徴である．**図 6** に P-BSSF の構成を示す（$h = 2$, $F = 8$）．ここで，BSSF 部の各ビットスライスファイルの第 i 物理ページからなる F ページの集まりを**パーティション**と呼ぶ．P-BSSF はあらかじめ決まった n ($= 2^h$) 個のパーティションからなる．

3.2 検索処理

P-BSSF における検索処理として通常の検索処理と，BSSF における検索効率化の手法である**スマート検索**（*smart retrieval*）[4),17)] を応用した検索処理を示す．

3.2.1 通常の検索処理

通常の検索処理では，問合せ集合が与えられると次の処理を行う．

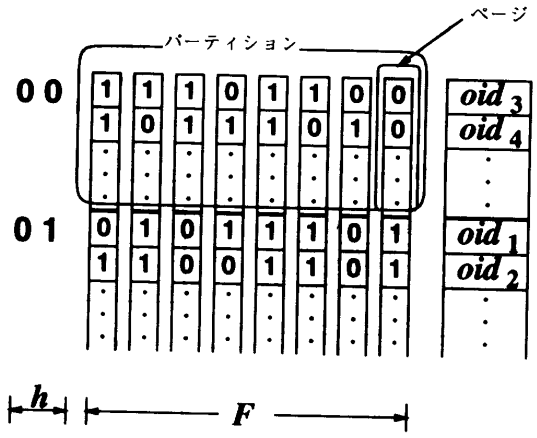

図 6 P-BSSF の構成
Fig. 6 Structure of P-BSSF.

(1) ターゲットシグネチャと同様に，BSSF 部用シグネチャ（$S_{Q\{BSSF\}}$）とプレフィクス部用のシグネチャ（$S'_{Q\{FP\}}$）の 2 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 BSSF 部のターゲットシグネチャと，BSSF 部用の問合せシグネチャとのマッチングを行い，問合せ条件を満たす候補を取り出す．そのときに候補となるシグネチャ $S_{T\{BSSF\}}$ は，問合せ条件によって次のようになる．

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

(4) フォルスドロップレゾリューションを行い，アクチュアルドロップを返す．

図 7 に $T \supseteq Q$ の問合せが与えられた際に探索しなければならない領域を示す．$S'_{Q\{FP\}}$ が "01" であるので，2 番目と 4 番目のパーティションのみ探索する必要がある．また $S_{Q\{BSSF\}}$ で，2, 5, 6, 8 番目のビット位置に "1" がセットされているため，その位置に対応するビットスライスのみスキャンする必要がある．

3.2.2 スマート検索による検索処理

スマート検索は，文献 4), 17) において提案した BSSF における検索効率化の一手法である．シグネ

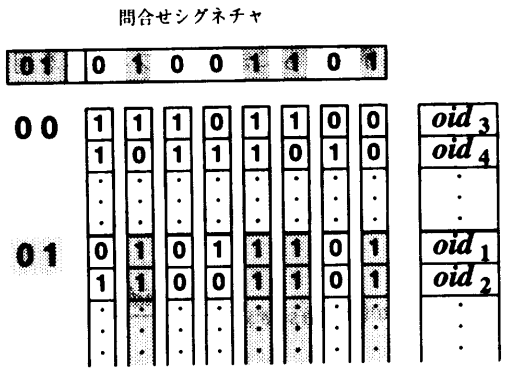

図7 P-BSSF の問合せ処理 ($T \supseteq Q$)
Fig. 7 Query processing with P-BSSF ($T \supseteq Q$).

チャファイルを用いた検索処理では，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われる．よって，フォルスドロップが多少増加しても，読み出す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，スマート検索方式である．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うことが可能である．具体的な処理については 5.2 節で述べる．

3.3 更新処理

更新処理として，データオブジェクトの挿入処理，削除処理について述べる．その際，更新するデータオブジェクトのターゲットシグネチャ $S_{T\{BSSF\}}$ およびプレフィクス $S'_{T\{FP\}}$ と OID が与えられるものとする．また，パーティションのオーバフローは生じないものとする．

3.3.1 挿入処理

P-BSSF 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う．

(1) プレフィクス $S'_{T\{FP\}}$ により，挿入すべきパーティションを決定する．
(2) OID ファイル中のそのパーティションに対応する部分を調べ，空きビット☆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) 対応する空きビットをリセットする．
(4) $S_{T\{BSSF\}}$ で "1" の立っている位置について，パーティション中の BSSF 部に "1" を立てる．

3.3.2 削除処理

P-BSSF からターゲットシグネチャを削除する場合，以下の手順で処理を行う．

(1) プレフィクス $S'_{T\{FP\}}$ により，そのターゲット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) OID ファイル中のそのパーティションに対応する部分を調べ，該当する OID があれば空きビットを立てる．
(3) BSSF 部の立っているビットを "0" にリセットする．

☆ 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．

4. コストモデル

P-BSSF の性能を評価するために検索コスト RC，更新コスト UC，格納コスト SC のコスト式を導出する．検索コストと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．コスト式を導出する際，以下を仮定する．

- D_t はすべてのオブジェクトで一定
- データベース中には各集合値は均等に出現

4.1 検索コストモデル

総検索コスト RC は，プレフィクスで絞り込まれたパーティション中の BSSF 部の検索コスト，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うコストの和になり，式 (1) で表される．

$$RC = n \cdot PAR \cdot LC_{BSSF} + LC_{OID} + P_o(A + PAR \cdot Fd(N - A)) \quad (1)$$

ここで，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，式 (2) および式 (3) で表される．

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

$$PAR_{\{T \subseteq Q\}} \approx \left(\frac{f + w(S_{Q\{FP\}})}{2f}\right)^h \quad (3)$$

ここで，$w(S_{Q\{FP\}})$ は $S_{Q\{FP\}}$ の平均ウェイトを示し，式 (4) で表される．

$$w(S_{Q\{FP\}}) = f\left(1 - \left(1 - \frac{m_{opt}}{f}\right)^{D_q}\right) \approx f\left(1 - \left(\frac{1}{2}\right)^{\frac{D_q}{D_t}}\right) \quad (4)$$

$PAR_{\{T \supseteq Q\}}$ の導出方法は文献 1) に示されている．また，$PAR_{\{T \subseteq Q\}}$ の場合も同様に導出することができる．

表1 記号の定義
Table 1 Symbol definition.

記号	定義
N	オブジェクトの総数
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)
D_t	ターゲット集合の要素数
D_q	問合せ集合の要素数
α	各パーティションの充足率
F, f	シグネチャのビット長
m	BSSF部の要素シグネチャのウェイト
m_{opt}	プレフィクス部の要素シグネチャのウェイト (= $f/D_t \cdot \ln 2$)
h	プレフィクス部のビット長
n	プレフィクス部に基づく分割により得られるパーティションの数 (= 2^h)
A	アクチュアルドロップ数
Fd	フォルスドロップ確率
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)

LC_{BSSF} は1パーティション中のBSSFの検索コストを表し，次の式(5)および式(6)で表される．

$$LC_{BSSF\{T\supseteq Q\}} = w(S_{Q\{BSSF\}}) \quad (5)$$

$$LC_{BSSF\{T\subseteq Q\}} = F - w(S_{Q\{BSSF\}}) \quad (6)$$

ここで，$w(S_{Q\{BSSF\}})$ は $S_{Q\{BSSF\}}$ の平均ウェイトを示し，式(7)で表される[4),6),17)]．

$$w(S_{Q\{BSSF\}}) = F\left(1-\left(1-\frac{m}{F}\right)^{D_q}\right) \approx F(1-e^{-\frac{m}{F}D_q}) \quad (7)$$

LC_{OID} は，Yaoの関数[13)]を用いて式(8)で表される．

$$LC_{OID} = SC_{OID}\cdot\left[1-\prod_{i=1}^{A+Fd(N-A)}\frac{N\cdot(1-1/SC_{OID})-i+1}{N-i+1}\right] \quad (8)$$

ここで，SC_{OID} はOIDファイルを格納するコストであり，式(9)で表される．

$$SC_{OID} = \left\lceil \frac{N}{\left\lfloor \frac{P\cdot\alpha}{oid}\right\rfloor}\right\rceil \quad (9)$$

Fd はBSSF部でフォルスドロップの生じる確率で，式(10)および式(11)で表される[4),6),17)]．

$$Fd_{\{T\supseteq Q\}} \approx (1-e^{-\frac{m}{F}D_t})^{mD_q} \quad (10)$$

$$Fd_{\{T\subseteq Q\}} \approx (1-e^{-\frac{m}{F}D_q})^{mD_t} \quad (11)$$

4.2 更新コストモデル

更新は挿入，削除ともにまずプレフィクスを用いて，更新の対象となるパーティションを決定する．次に，OIDファイル中のそのパーティションに対応する部分を調べ，最後にBSSF部にビットをセット，またはリセットする．よって，更新コストは式(12)で表される．

$$UC = 2\cdot w(S_{T\{BSSF\}}) + \frac{b\cdot oid}{2} + 1 \quad (12)$$

ここで，$w(S_{T\{BSSF\}})$ は $S_{T\{BSSF\}}$ の平均ウェイトを示し，式(13)で表される[4),6),17)]．

$$w(S_{T\{BSSF\}}) = F\left(1-\left(1-\frac{m}{F}\right)^{D_t}\right) \approx F(1-e^{-\frac{m}{F}D_t}) \quad (13)$$

4.3 格納コストモデル

総格納コスト SC は，BSSF部の格納コストとOIDファイルの格納コストの和になり，式(14)で表される．

$$SC = F\left\lceil\frac{N}{Pb\cdot\alpha}\right\rceil + \left\lceil\frac{N}{\left\lfloor\frac{P\cdot\alpha}{oid}\right\rfloor}\right\rceil \quad (14)$$

5. 検索コスト

本章では，P-BSSFとBSSFの検索コストを比較する．FP分割シグネチャは，$T\supseteq Q$ では問合せ集合の要素数 D_q が非常に大きい場合，$T\subseteq Q$ では非常に小さい場合以外は，P-BSSFやBSSFに比べ検索コストが非常に大きくなってしまうので，以下の議論では比較の対象から除外する．ここでは文献4), 6), 17)と同様に，$D_t = 100$, $F = 1024$ として解析を行う．検索コストに影響を与える要因は各種あるが，各パーティションの平均充足率 α はその中でも重要なコスト要因と考えられる．通常，P-BSSFはフィクスドプレフィクス法を用いているためパーティションの充足率は100%にはならない．そこで，以下では充足率が100%，80%，60%の場合について検討を行う．また，充足率 α によって，オブジェクトの総数 N (= $Pb\cdot n\cdot\alpha$) が変化するので，検索コスト RC を N で割った相対的な検索コスト RC/N を比較する．パーティション数 n を32としたときの，P-BSSFにおける充足率とオブジェクトの総数の関係を**表2**に示す．また，比較対象とするBSSFのオブジェクトの総数は800,000とする．

5.1 通常の検索処理によるコスト

通常の検索処理における検索コストを，**図8**，**図9**

表2 充足率とオブジェクトの総数
Table 2 Partition load factor and total number of objects.

充足率 (α)	オブジェクトの総数 (N)
60%	629,146
80%	838,861
100%	1,048,576

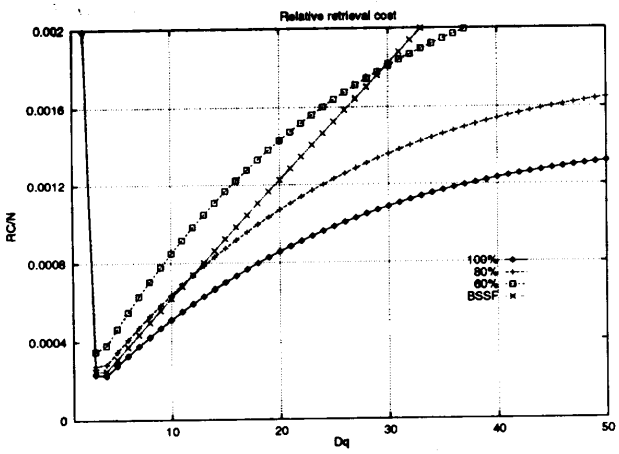

図8 $T \supseteq Q$ の検索コスト
Fig. 8 Retrieval costs for $T \supseteq Q$.

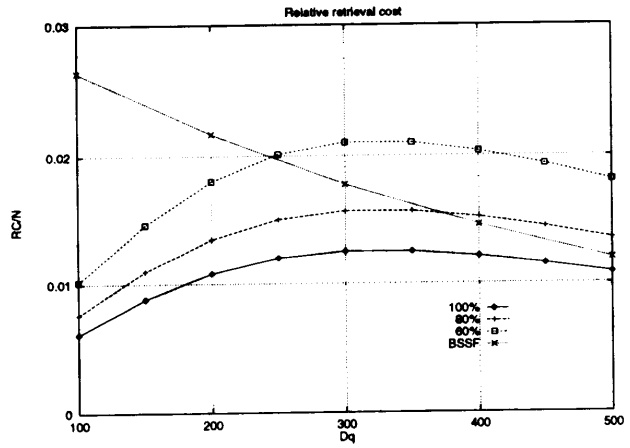

図9 $T \subseteq Q$ の検索コスト
Fig. 9 Retrieval costs for $T \subseteq Q$.

に示す．図8，図9より，$T \supseteq Q$ では D_q が大きい場合，$T \subseteq Q$ では D_q が小さい場合，BSSF と比較して P-BSSF が有効であることが分かる．これは，フィクスドプレフィクス法に基づく水平分割の効果によるものである．また，充足率が検索コストに影響を与えることが分かる．充足率100%の場合が最も効率良く，充足率の低下にともない検索コストは増加する．これは充足率が低下することにより，各パーティション中にシグネチャが存在しない領域が増加し，アクセスしなければならないページが増加するためである．これに対して，BSSF はその構成上充足率はほぼ100%となる．そのため充足率が低下した場合には，$T \supseteq Q$ では D_q が小さい場合，$T \subseteq Q$ では D_q が大きい場合，BSSF の方が検索コストで有利な領域が生じることが分かる．

5.2 スマート検索処理によるコスト

図8より，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，フォルスドロップの発生頻度は少なく，検索コストの大部分はシグネチャのスキャンに費やされる．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，BSSF に対するスマート検索の処理は以下のようになる．

(1) 実際の問合せが出されたとき，$D_q \leq 4$ であれば，通常どおりの検索を行う．
(2) $D_q > 4$ ならば，問合せ集合中の任意の4つの要素を選んで，問合せシグネチャを作成し，検索を行う．

このスマート検索を P-BSSF に応用すると次のようになる．

(1) $D_q \leq 4$ のときは通常どおりの検索を行う．
(2) $D_q > 4$ のときは，全要素を用いた問合せシグネチャからプレフィクスを作成し，パーティションを決定する．続けて任意の4つの要素を選んで BSSF 部のシグネチャを作成し，検索を行う．

$T \subseteq Q$ の場合も同様にスマート検索は適用可能である．スマート検索方式を適用したときの検索コストを，**図10**，**図11** に示す．スマート検索によるコストは，図10，図11のように BSSF で一定になっている範囲についても，P-BSSF では減少を続けており，$T \supseteq Q$ では D_q が大きいほど，$T \subseteq Q$ では D_q が小さいほど P-BSSF の方が有利になることが分かる．これは，上記の範囲について P-BSSF の水平分割の効果が大きくなるためである．しかし通常の検索と同様，充足率が影響を与え，充足率が低くなるとコストが悪化する領域が生じる．

5.3 ビットスライストクイックフィルタ (Bit-Sliced Quick Filter: BSQF)

前節で述べたように P-BSSF では，充足率が検索コストに影響を与える．したがって，データオブジェクトの追加や削除が比較的多い動的な環境では，この点が問題となる．そこでこのような場合でも，ある程度の充足率を保てるように，パーティションの数を前もっ

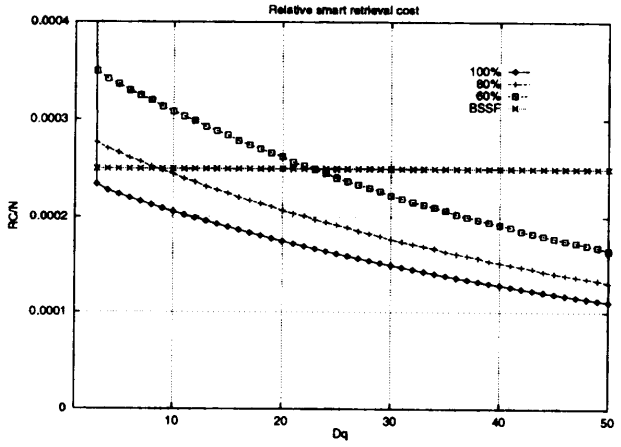

図10 $T \supseteq Q$ のスマート検索コスト
Fig. 10 Smart retrieval costs for $T \supseteq Q$.

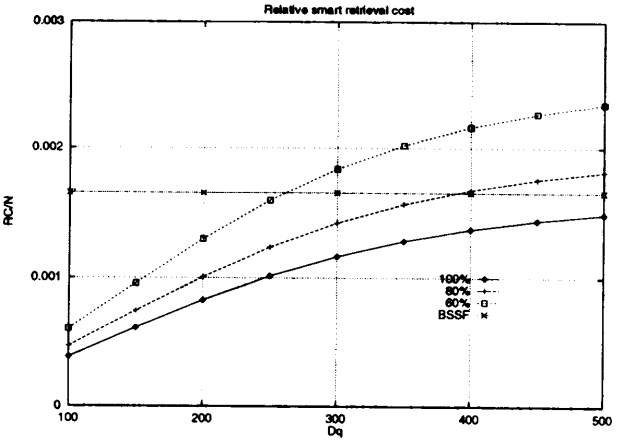

図11 $T \subseteq Q$ のスマート検索コスト
Fig. 11 Smart retrieval costs for $T \subseteq Q$.

てあらかじめ決定するフィクスドプレフィクス法の代わりに，リニアハッシュ法[10]を用いたクイックフィルタ（Quick Filter）[16]の方式と組み合わせた，**ビットスライストクイックフィルタ**（***Bit-Sliced Quick Filter***: BSQF）を提案する．リニアハッシュ法はハッシングのレベルを変化させることで，パーティションの数を動的に決めることができる．

リニアハッシュ法では，パーティションの分割，併合が動的に生じる．BSQF ではこの際，各シグネチャのパーティションへの配分を決定するために用いるプレフィクスの再計算を避けるため，あらかじめ h_{MAX} ビットのプレフィクス用のターゲットシグネチャを格納しておくことにする．

図12 に BSQF の構成と $T \supseteq Q$ の問合せが与えられた際に読み出す必要のあるシグネチャを示す．検索処理，更新処理の方法は，更新時にパーティションの分割，併合が生じる可能性があることを除いては，ほぼ P-BSSF と同様である．**図13**，**図14** に BSQF と

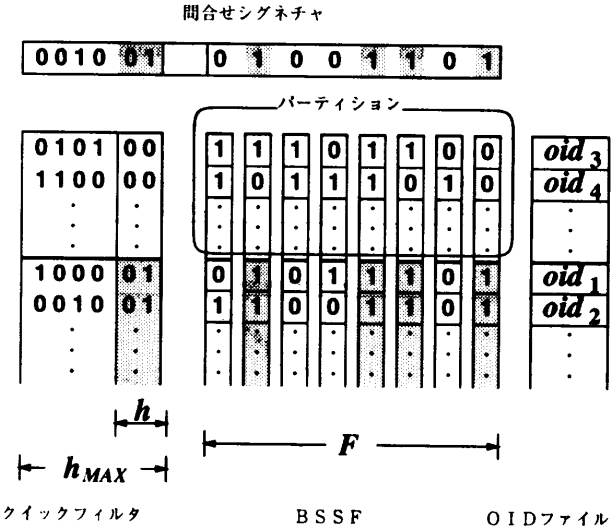

図12 BSQF の構成と問合せ処理（$T \supseteq Q$）
Fig. 12 Structure of BSQF and query processing ($T \supseteq Q$).

BSSF のスマート検索による検索コスト（$D_t = 100$, $F = 1024$, $N = 800,000$, $\alpha \approx 75\%$）を示す．

5.4 Key-Based Bit-Sliced Signature File との比較

Zezula らの提案する Key-Based Bit-Sliced Signature File[15]（以下，KS）は，ターゲットシグネチャをパーティションに均等に分割するために，各集合シグネチャの半分のビットが "1" にセットされるよう，m を m_{opt} に制限している．しかし，P-BSSF では水平分割を行うための集合シグネチャを元の集合シグネチャとは別に作成するため，このような制限はない．KS は P-BSSF で $m = m_{opt}$ という制限を加えたものと考えることができる．

図15 と**図16** に m に制限のない P-BSSF と KS のスマート検索に基づく検索コストを示す（$D_t = 100$, $F = 1024$, $N = 800,00$, $\alpha \approx 75\%$, $m_{opt} = 7$）．KS に対してもスマート検索方式が P-BSSF と同様に適用可能である．ただし，P-BSSF については $T \supseteq Q$（図15）では $m = 2$，$T \subseteq Q$（図16）では $m = 10$ である．

図15より，$T \supseteq Q$ の場合 m_{opt} より小さい m の値（$m = 2$）をとることによって，検索コストは P-BSSF の方が有利になっているのが分かる．$T \supseteq Q$ では，問合せシグネチャで "1" の立っている位置に対応するビットスライスを読み出す必要がある．よって，m を小さくすることで，問合せシグネチャの "1" の立っている位置を減少させ，読み出すビットスライスを削減できる．しかし，m を極端に小さくしてしまうと，D_q が小さい場合のフォルスドロップが非常に多くなるた

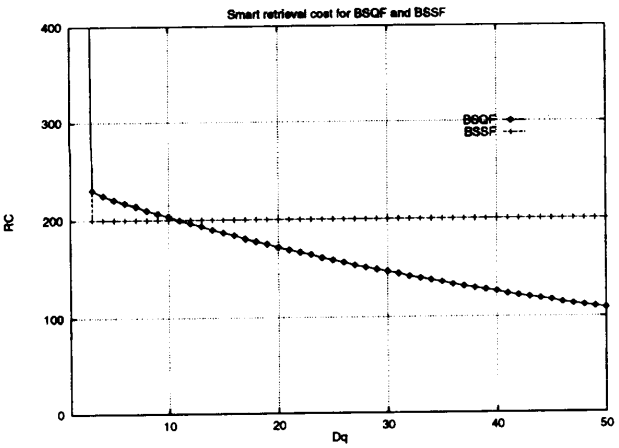

図13 BSQF と BSSF のスマート検索コスト ($T \supseteq Q$)
Fig. 13 Smart retrieval costs of BSQF and BSSF ($T \supseteq Q$).

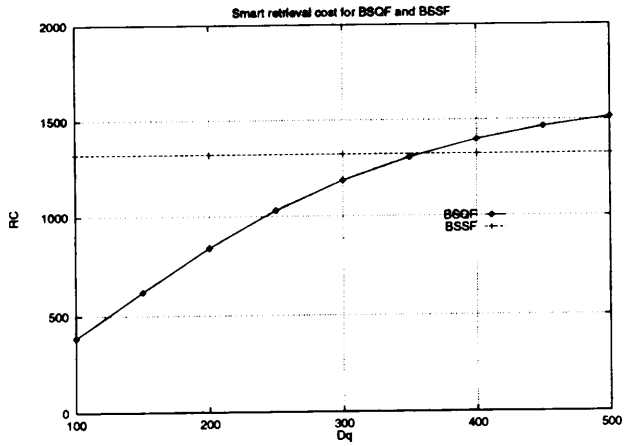

図14 BSQF と BSSF のスマート検索コスト ($T \subseteq Q$)
Fig. 14 Smart retrieval costs of BSQF and BSSF ($T \subseteq Q$).

め，検索効率が悪くなる．一方，図16より，$T \subseteq Q$ の場合 m_{opt} より大きい m の値 ($m = 10$) をとることによって，P-BSSF の検索コストが有利になっていることが分かる．$T \subseteq Q$ の場合，$T \supseteq Q$ と逆に問合せシグネチャで "0" となっている位置に対応するビットスライスを読み出すことになる．よって，m を大きくすることで，問合せシグネチャの "0" となっている位置を減少させ，読み出すビットスライスを削減できる．しかし，$T \supseteq Q$ の場合と同様，m を極端に大きくしてしまうと，フォルスドロップが増加してしまい，やはり検索効率が悪くなる．このように，P-BSSF では適用対象と検索パターンにより適合したパラメータ設定をとることにより，検索効率を向上することができる．

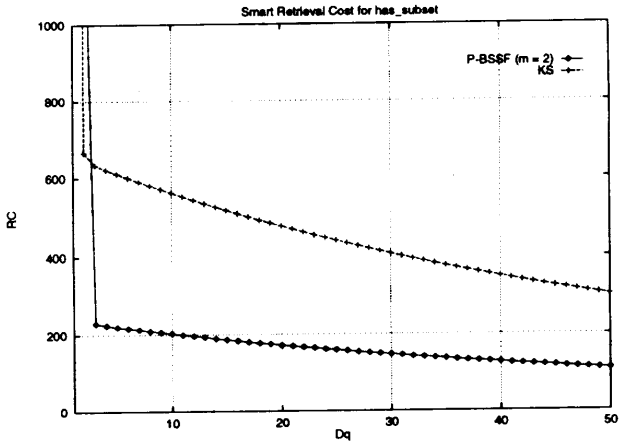

図15 P-BSSF と KS のスマート検索コスト ($T \supseteq Q$)
Fig. 15 Smart retrieval costs of P-BSSF and KS ($T \supseteq Q$).

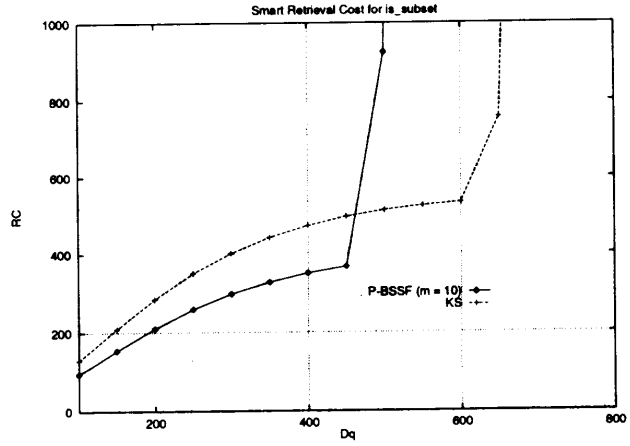

図16 P-BSSF と KS のスマート検索コスト ($T \subseteq Q$)
Fig. 16 Smart retrieval costs of P-BSSF and KS ($T \subseteq Q$).

6. 更新コスト

これまでに述べた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 ($F = 1024$, $N = 800,000$, $\alpha \approx 75\%$)．ただし，BSQF についてはパーティションの分割，併合が生じる確率は非常に低いためそのコストは考慮していない．また，BSSF への新しいターゲットシグネチャの挿入はファイルの最後尾に行われるものとする．

$m = 2$ の P-BSSF，BSQF の削除コストが BSSF に比べかなり低くなっていることが分かる．これは，P-BSSF，BSQF では削除するシグネチャを含むパーティションをプレフィクスを用いることによって特定できるが，BSSF では OID ファイルをスキャンする必要があるためである．一方，KS や $m = 10$ の P-BSSF，BSQF は BSSF 部のウェイトが高いため，挿入，削除両コストともに大きくなってしまう．

表3 更新コスト
Table 3 Update costs.

ファイル	挿入コスト	削除コスト
P-BSSF, BSQF ($m = 2$)	397	397
P-BSSF, BSQF ($m = 10$)	1310	1310
KS	1049	1049
BSSF	366	1147

表4 格納コスト
Table 4 Storage costs.

ファイル	SC_{BSSF}	SC_{OID}	SC
P-BSSF, BSQF ($m = 2$)	32,768	2,048	34,816
P-BSSF, BSQF ($m = 10$)	32,768	2,048	34,816
KS	32,768	2,048	34,816
BSSF	25,600	1,563	27,163

7. 格納コスト

$F = 1024$, $N = 800,000$, $\alpha \approx 75\%$ のときの，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．BSQFの格納コストは，P-BSSFのコストにクイックフィルタ部のコストを加えたものとなるが，BSSF部の格納コストに比べ，非常に小さいので無視できる．表4より，BSSFが他の方法に比べ格納コストが低いことが分かる．これは，BSSF以外の方法では新しいシグネチャの挿入に備えてあらかじめ確保すべき空き領域を考慮して，充足率 α を75%としていることが理由である．

8. まとめ

本論文では，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，水平分割を行う分割シグネチャファイルのアプローチと，垂直分割を行うビットスライス方式のアプローチを組み合わせた分割ビットスライストシグネチャファイル構成法を提案し，その検索，更新，格納の各コストの見積りとBSSFとの比較評価を行った．特に検索については，集合値検索を対象として，$T \supseteq Q$ と $T \subseteq Q$ の場合，更新については挿入と削除の場合について解析を行った．また，水平分割と垂直分割の概念を組み合わせた手法であるKey-Based Bit-Sliced Signature File（KS）との比較も行った．

検索コストについては，特にパーティションの充足率の影響を考慮して検討を行った．その結果，一定以上の充足率が確保され，特に $T \supseteq Q$ では D_q が大きいとき，$T \subseteq Q$ では D_q が小さいときは，P-BSSFはBSSFに比べきわめて有効であることを示した．しかし，一定以上の充足率が確保されなかった場合には，特に $T \supseteq Q$ では D_q が小さいとき，$T \subseteq Q$ では D_q が大きいとき，P-BSSFの検索効率がBSSFに比べ悪くなることがあることが分かった．一方，スマート検索を応用することによって，通常の検索コストを大きく削減可能で，スマート検索を行ったBSSFよりもP-BSSFではスマート検索による効果が大きいことを示した．しかし，通常の検索処理の場合と同様，充足率や D_q の値によっては効率が悪くなる場合があることが分かった．この問題を緩和する手法として，パーティションの数を動的に決めることができるBSQFの提案を行い，その有効性を検討した．また，KSとの比較では，P-BSSFの方が適用対象と検索パターンにより適合したパラメータ設定をとることが可能であることを示した．

更新コストについては，一般的にBSSFは不利であるといわれている．これは，集合シグネチャを作成する際に“0”と“1”が同じ確率で出現するように，m の値として m_{opt} を用いる場合が多いからである．P-BSSFとBSQFではプレフィクスを用いることにより，削除対象のターゲットシグネチャシグネチャを含むパーティションを特定することができ，ほぼ同じパラメータ値を持つBSSFと比較して削除コストを削減可能である．また，m の値を小さくとったP-BSSF，BSQFではBSSF部のターゲットシグネチャのウェイトが小さくなるために，更新コストは小さくなる．

格納コストはBSSFが最も小さく，P-BSSF，BSQFのコストに充足率をかけたコストになる．

以上より，一定以上のパーティションの充足率が確保され，かつ挿入や削除の少ない静的な環境においては，BSSF等の従来の構成法よりもP-BSSFは有効なシグネチャファイル構成法であるということができる．一方，挿入や削除が多く，あらかじめ用意すべきパーティション数を見積もることが難しい動的な環境にはBSQFが適合するといえる．これらは，シグネチャファイルのスキャンコストが大きい大規模データベースにおいて，より効率的であると考えられる．

今後の課題としては，データの出現頻度に偏りがある場合の性能解析や，シグネチャファイルを有効に利用するための問合せ処理方式などがあげられる．

謝辞 本研究に関してご意見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いたします．なお，本研究の一部は文部省科学研究費重点領域研究「高度データベース」（課題番号08244101）の助成による．

参考文献

1) Ciaccia, P. and Zezula, P.: Estimating Ac-

cesses in Partitioned Signature File Organizations, *ACM Trans. Inf. Syst.*, Vol.11, No.2, pp.133–142 (1993).
2) Fagin, R., Nievergelt, J., Pippenger, N. and Strong, H.R.: Extendible Hashing – A Fast Access Method for Dynamic Files, *ACM Trans. Database Syst.*, Vol.4, No.3, pp.315–344 (1979).
3) Faloutsos, C. and Christodoulakis, S.: 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).
4) Ishikawa, Y., Kitagawa, H. and Ohbo, N.: Evaluation of Signature Files as Set Access Facilities in OODBs, *Proc. ACM SIGMOD*, pp.247–256 (1993).
5) Kim, J.K. and Chang, J.W.: A Two-dimensional Dynamic Signature File Method, *Proc. International Symposium on Advanced Database Technologies and Their Integration (ADTI '94)*, pp.63–70 (1994).
6) Kitagawa, H., Fukushima, Y., Ishikawa, Y. and Ohbo, N.: Estimation of False Drops in Set-valued Object Retrieval with Signature Files, *Proc. 4th International Conference on Foundations of Data Organization and Algorithms*, pp.146–163 (1993).
7) Lee, D.L. and Leng, C.: Partitioned Signature Files: Design Issues and Performance Evaluation, *ACM Trans. Off. Inf. Syst.*, Vol.7, No.2, pp.158–180 (1989).
8) Lee, W. and Lee, D.L.: Signature File Methods for Indexing Object-oriented Database Systems, *Proc. Intl. Computer Sciences Conf. (ISCS)*, pp.616–622 (1992).
9) Lin, Z. and Faloutsos, C.: Frame-Sliced Signature Files, *IEEE Trans. Knowl. and Data Eng.*, Vol.4, No.3, pp.281–289 (1992).
10) Litwin, W.: Linear Hashing: A New Tool for File and Table Addressing, *Proc. 6th International Conference on VLDB*, pp.212–223 (1980).
11) Rabitti, F. and Savino, P.: Image Query Processing Based on Multi-level Signatures, *Proc. 14th Annual Intl. ACM/SIGIR Conf. on Research and Development in Information Retrieval*, pp.305–314 (1991).
12) Roberts, C.S.: Partial-match Retrieval via the Method of Superimposed Codes, *Proc. IEEE*, Vol.67, No.12, pp.1624–1642 (1979).
13) Yao, S.: Approximating Block Accesses in Database Organizations, *Comm. ACM*, Vol.20, No.4, pp.260–261 (1977).
14) Yong, H.-S., Lee, S. and Kim, H.-J.: Applying Signatures for Forward Traversal Query Processing in Object-oriented Databases, *Proc. 10th Intl. Conf. on Data Engineering*, pp.518–525 (1994).
15) Zezula, P., Ciaccia, P. and Tiberio, P.: Key-based Partitioned Bit-sliced Signature File, *ACM SIGIR Forum*, Vol.29, No.2, pp.20–34 (1995).
16) Zezula, P., Rabitti, F. and Tiberio, P.: Dynamic Partitioning of Signature Files, *ACM Trans. Inf. Syst.*, Vol.9, No.4, pp.336–369 (1991).
17) 石川佳治，北川博之，大保信夫：シグネチャファイルによる集合値検索のコスト評価，情報処理学会論文誌，Vol.36, No.2, pp.383–395 (1995).
18) 渡辺悟康，北川博之：集合値検索を対象とした分割シグネチャファイル構成法の検討，第50回情報処理学会全国大会論文集，pp.4-27–4-28 (1995).
19) 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, pp.1–8 (1995).

(平成8年5月14日受付)
(平成8年9月12日採録)

渡辺 悟康（正会員）

1971年生．1994年筑波大学第三学群情報学類卒業．1996年同大学大学院工学研究科修士課程修了．同年NTTデータ通信株式会社入社．現在，同社技術開発本部オープンシステムセンタデータベース技術担当勤務．

北川 博之（正会員）

1955年生．1978年東京大学理学部物理学科卒業．1980年同大学大学院理学系研究科修士課程修了．日本電気（株）勤務を経て，1988年筑波大学電子・情報工学系講師．現在，同大学同学系助教授．理学博士（東京大学）．異種分散情報資源統合，時制データベース等に興味を持つ．著書「データベースシステム」（昭晃堂），「The Unnormalized Relational Data Model」（共著，Springer-Verlag）等．電子情報通信学会，日本ソフトウェア科学会，ACM，IEEE-CS各会員．