組込みＤＢＭＳ向け全文検索インデクス更新方法の開発と評価

井上 尚樹† 田原 靖大‡ 福島 大雅‡

†（株）日立製作所 ソフトウェア事業部 〒212-8567 川崎市幸区鹿島田 890
‡（株）日立製作所 ソフトウェア事業部 〒244-8555 神奈川県横浜市戸塚区戸塚町 5030
E-mail: †naoki.inoue.gg@hitachi.com, ‡{yasuhiro.tahara.bp ,taiga.fukushima.yq}@hitachi.com

あらまし 組込み DBMS 向け全文検索機能におけるインデクス更新方法の開発と評価を行った。組込みシステムに適用するため，リアルタイム性を満たし，使用メモリ量が指定値を超えず，かつ，高速に処理できるインデクス更新方式として「差分インデクス分割反映方式」を提案し，実装した。差分インデクス分割反映方式のインデクス更新処理は，複数文書をあらかじめ与えたデータ量のバッファ領域にまとめた後，インデクスの索引見出し単位の更新処理に分割して実行する。検索処理は，インデクスの更新処理における分割単位で割り込んで実行することにより，リアルタイム性を保証する。また，登録可能な文書量を制御することで，バッファ領域があふれないことを保証する。

本研究では，組込み DBMS において，差分インデクス分割反映方式を実装し，組込み特有の制約を満たしていることを検証した。

キーワード 組込み DBMS，全文検索

Text Search Index Updating on Embedded DBMS

Naoki INOUE† Yasuhiro TAHARA‡ and Taiga FUKUSHIMA‡

†Hitachi, Ltd., Software Division 890 Kashimada, Saiwai, Kawasaki, Kanagawa, 212-8567 Japan
‡Hitachi, Ltd., Software Division 5030 Totsukacho,Totsukaku, Yokohama, Kanagawa,244-8555 Japan
E-mail: †naoki.inoue.gg@hitachi.com, ‡{yasuhiro.tahara.bp ,taiga.fukushima.yq}@hitachi.com

Abstract We developed and evaluated on text search index updating on embedded DBMS. We proposed a new updating method, "Gradually Updating of Differential Index", that meets restrictions on process time and resource in embedded systems. "Gradually Updating of Differential Index" extracts a buffer area of fixed size from documents and updates the index on each element of the differential index gradually. The process time restriction was satisfied by updating the index element by element and the resource restriction was satisfied by controlling the amount of the registered documents depending on the progress of the index updating.

We implemented "Gradually Updating of Differential Index" on embedded DBMS, and verified the efficacy of the method.

Keyword Embedded Database, Full-Text Search

1. はじめに

近年，アプリケーションの複雑化・肥大化，製品出荷サイクルの短縮化傾向から，組込みシステムにおいても，ミドルウェア適用の動きが活発化している。その流れの中で DBMS 適用の可能性も検討されている。

組込みシステムへの DBMS の適用例としては，カーナビゲーションシステムにおける施設名データ管理がある。カーナビゲーションシステムの施設名検索では，応答時間が問題となるため，検索処理の高速化が求められる。また，文字の検索はカーナビゲーションシステムに限らず携帯電話などへも応用できる。このため，全文検索機能を提供する組込みシステムが増加することが予想される。従って，DBMS が全文検索機能を提供することで，組込みシステムの開発効率の向上と開発工数の低減が可能であると考えられる。

DBMS を組込みシステムに適用する場合には，リアルタイム性とリソース制約を満たす必要がある。

本研究では，リアルタイム性を確保し，かつ，リソース制約を満たす,全文検索機能の実装方式を検討し，実装及び評価を行った。

論文構成としては，２章で関連研究，３章で組込みシステム向け全文検索機能処理方式とインデクス更新の問題点，４章で問題点の解決案の検討，５章で検証結果，６章でまとめと今後の課題を述べる。

2. 関連研究

高速な全文検索機能は，n-gram インデクス等の転置ファイルを用いることで実現できる[1]。しかし，転置ファイルを使用する場合，インデクス更新により文書登録処理が非常に遅くなるという問題がある。

この問題の解決手段として，部分更新方式[2] や，差分インデクスを用いて後で更新する方式などが研究されている。しかし，これらの方法をそのまま組込みシステムに適用すると，リアルタイム性と，リソース制約を満たすことは難しい。

本論文では，高速なインデクス更新が可能で，リアルタイム性を確保し，かつ，リソース制約内で動作可能な方式を検討・実装し，その効果の検証を行う。

3. 組込みシステム向け全文検索機能実現方式とその課題

3.1. 全文検索実現方式

本研究では，高速な全文検索機能の実現方式として，n-gram インデクス方式を採用した。組込みシステムにおける全文検索機能は，検索処理が高速で，検索ノイズが発生してはいけない。また，インデクス作成のためのキーワードや辞書は不要であることが望ましい。このため，これらの条件を満たす n-gram インデクス方式を採用した。

n-gram インデクス方式とは，文書を１文字ずつずらして文字列(n-gram)を切り出し，切り出した文字列(索引見出し）に対する文書番号と文字位置（出現位置情報）をインデクス要素として n-gram インデクスに記録し，このインデクスを用いて検索処理を実行する方式である。これにより，インデクスを用いた高速な検索が可能になる。また，検索時にはヒット文書に文字列が含まれることを保証できるため，検索ノイズは発生しない。また，キーワードや辞書は使用しない。

3.2. n-gram インデクス方式の問題点

n-gram インデクス方式には，文書登録・削除時にリアルタイム性を保証できないという問題と，インデクスが巨大化するという問題がある。これは、n-gram インデクス方式では，一つの文書登録に対し，登録文書に存在する文字数に比例した数のデータをインデクスに書き込む必要があるためである。

本論文では，n-gram インデクスへの文書登録・削除時にリアルタイム性が保証できないという問題の解決方法を詳細に述べる。これは，組込みシステムで DBMS を使用する利点として，データ管理，特に文書の登録・削除によるインデクス更新が容易になることが挙げられるためである。なお，n-gram インデクスが巨大化することに対しては，格納するデータを圧縮することで対応した。この方式に関しては，本論文では言及しない。

n-gram インデクス方式を組込みシステムに適用する場合，文書の登録･削除処理及び検索処理において，リアルタイム性を保証する必要がある。ここでの，リアルタイム性は，人が利用する場合のレスポンス時間として，理想的には 150 ミリ秒以下，通常は 1 秒以下，遅くても 4 秒以内に終了するのが望ましい[3]。

また，n-gram インデクス方式は，文書の登録・削除処理及び検索処理において，限られたリソース制約内で動作する必要がある。本方式を用いる文書蓄積用の領域サイズは一定サイズ以下にする必要がある。

4. n-gram インデクスの更新方式

本章では，前章で説明した組込みシステム特有の課題を解決するため，メモリ領域の使用方法を工夫したインデクス更新方式を提案する。上述した 1 点目の課題であるリアルタイム性の確保に対しては，インデクス更新用のバッファ領域を二つの領域に分割し，文書登録・削除処理とインデクス更新処理を分離することで対応する。２点目の課題であるリソース制約に対しては，バッファ領域の使用量に対する制御を行い，規定したリソース量の範囲内での更新処理を実現する。

4.1. リアルタイム性の確保方法

本研究では，文書の登録・削除処理に対するリアルタイム性を確保するため，バッファ領域を二つに分割して使用する方法を提案する。本方法を適用することにより，更新処理に関しては，分割した二つの領域で，文書登録・削除処理とインデクス更新処理を分割することで文書登録・削除におけるレスポンス時間を短縮し，リアルタイム性を確保することができるようになる。また、検索処理に関しては，二つに分割したバッファ領域に対し，従来技術であるインデクス要素単位の割り込み処理をベースにした対策を検討し，対応した。以下の項では，本方法を適用した場合の登録・削除処理と検索処理の詳細をそれぞれ説明する。

4.1.1. 文書登録・削除処理に対するアプローチ

本研究では、文書の登録・削除処理に対するリアルタイム性を確保するため，バッファ領域を入力用領域とマージ領域の二つに分割して、文書の登録・削除処理とインデクス更新処理を分離する方式を提案する。

入力用領域は，登録・削除を行う文書を最初に蓄積するための領域である。また，マージ領域は，登録・削除する文書に対応する差分インデクスを蓄積しておく領域である。また，両領域をまとめてバッファ領域と呼ぶ。

本方式の概念図を図 1に示す。

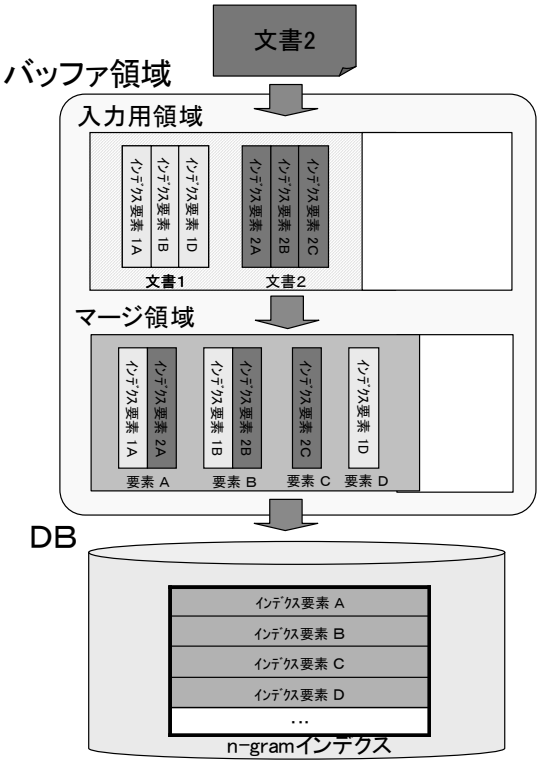

図1　文書の登録・削除処理

文書の登録・削除処理では，入力用領域とマージ領域からなるバッファ領域と，書き込み済みサイズを保存する一つの整数値を使用する。

文書の登録・削除処理では，まず，登録・削除する文書を入力用領域に格納する。ここで，入力用領域のデータ形式は，文書そのものでも良いし，文書から抽出した差分インデクスでも良い。以降，差分インデクス形式で格納した場合について説明する。入力用領域のデータはマージ領域の内容が全て書き込み済みになった時点で，マージ領域に一括して移動する。移動時には，マージ領域上で差分インデクスのインデクス要素順になるように並べ替える。マージ領域のデータは，マージ領域上の並び順に n-gram インデクスのインデクス要素毎に書き込み，書き込み済みサイズを更新する。

n-gram インデクス方式では，文書の登録・削除処理の延長で実行するインデクス更新処理によってレスポンス時間が長くなり，リアルタイム性を満たせないため，処理対象の文書を一旦バッファなどに蓄積し，文書の登録・削除後にインデクスを更新する方式を用いることとした。これにより，文書の登録・削除時のレスポンスが低下せず，リアルタイム性を満たすことができる。また，一旦蓄積した文書を n-gram インデクスに書き込む処理の処理時間が問題となるため，文書の登録・削除処理の延長で実行する書き込み処理での書き込みサイズを制限する方式を考えた。詳細については次節4.2「リソース制約内での動作保証」で説明する。

また，蓄積した複数の文書が同じインデクス要素を生成する場合，n-gram インデクスの更新処理をインデクス要素毎にまとめて行うことができ，ディスクアクセスの合計回数が減り，インデクスの更新高速化が期待できる。この結果として，更新のリアルタイム性を確保することができる。

4.1.2. 検索処理に対するアプローチ

本研究では，検索処理のリアルタイム性を確保するため，インデクス更新処理に対してインデクス要素単位での割り込みを行う方式を採用した。インデクス更新処理中の，文書の検索処理においては，インデクス更新処理への割り込みを高速に行うため，一時ファイルや文書情報からの書き込み中に割り込みを禁止するのではなく，インデクス要素単位で割り込みを許可する方式とした。これにより一時ファイルや文書情報からの書き込み中に割り込みを禁止する場合に比べ，より短い時間で検索を開始することができる。インデクス更新処理実行中の検索割り込みタイミングの例を図2に示す。

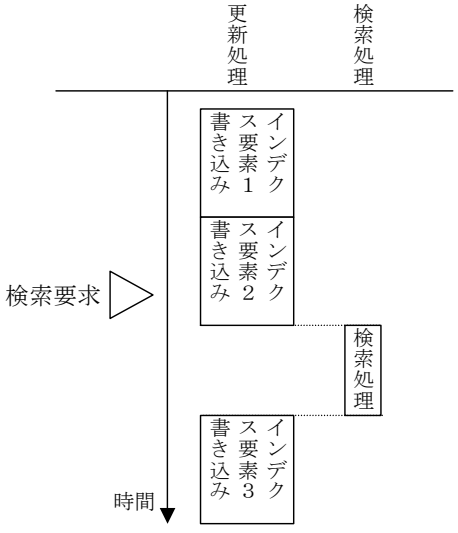

図2　検索実行のタイミングの例

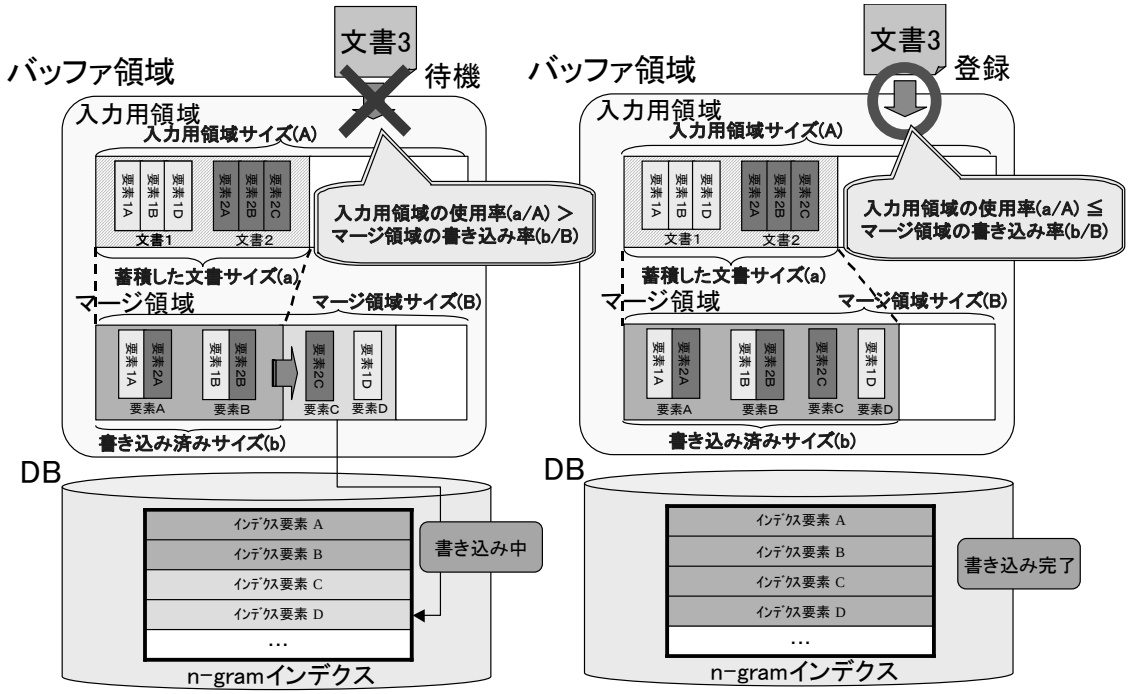

図3　書き込み待ちの調整方式

図 2は，更新処理によるインデクス要素２の書き込み中に，検索要求が発生した例である。インデクス要素２の書き込みが完了した時点で更新処理を一時停止し，検索処理を開始する。検索処理完了後にインデクス要素３の書き込みに遷移し，更新処理を再開する。

文書の検索処理では，インデクス更新中の場合は，インデクス要素単位で書き込み，インデクス要素単位での書き込みが終了したタイミングで書き込みを一旦中断して検索を実行する。インデクス要素単位の出現位置情報はそれぞれ独立であり，他のインデクス要素の出現位置情報には影響を与えない。このため，インデクス要素単位での書き込みが終了していれば n-gram インデクスが破損していないことが保証できるため n-gram インデクスの検索結果の整合性を保つことが容易になる。

バッファ領域中に n-gram インデクスに反映されていない情報が格納されている場合に，検索結果の整合性を保つためには，n-gram インデクスだけでなく，マージ領域の未書き込み部分と入力用領域も検索すればよい。

上記の方法を用いることで，インデクス更新中の検索処理において，リアルタイム性を満たすことができる。

4.2. リソース制約内での動作保証

本研究では，リソース制約内での更新処理を実現するため,入力用領域及びマージ領域の使用量を計測し，規定のバッファ領域サイズ内で更新処理が行なえるよう制御する方法を提案する。本方式の概念図を図 3に示す。

文書の登録・削除では，入力用領域の領域サイズに対する蓄積した文書サイズの比率（以下，入力用領域の使用率と呼ぶ）がマージ領域の領域サイズに対する n-gram インデクスへの書き込み済みデータのサイズの比率（以下，マージ領域の書き込み率と呼ぶ）より大きい場合は，文書の登録・削除処理を待機させる。待機中にバッファ領域から n-gram インデクスへの書き込み処理を実行し，入力用領域の使用率がマージ領域の書き込み率以下になった時点で，文書の登録・削除を実行する。これにより，入力用領域があふれる前にマージ領域中の全インデクス要素の n-gram インデクスへの書き込みが完了する事となり，バッファ領域のあふれを回避できる。

上記の方法を用いることで，一時的に使用する入力用領域とマージ領域のリソース所要量を固定にすることができる。

4.3. 差分インデクス分割反映方式

4.1及び4.2で説明したインデクス更新方式を，差分インデクス分割反映方式と呼ぶ。差分インデクス分割反映方式は，複数の登録・削除対象の文書をまとめて保持するために，固定サイズの入力用領域とマージ領域の二つの領域を用い，マージ領域から特定サイズずつ n-gram インデクスに差分を反映するインデクス更

新方式である。文書の登録・削除時には，マージ領域の書き込み率が入力用領域の使用率と等しいかより大きくなるまで，処理を一旦待機させる。また，検索処理では，n-gram インデクスだけでなく，マージ領域と入力用領域の情報も参照して検索する。

差分インデクス分割反映方式を採用し，複数文書のインデクス情報を差分インデクスにまとめて更新することで，更新に要する時間を削減することができる。また，インデクス要素単位での書き込み中断が可能となり，検索のリアルタイム性を満たすことができる。リソース制約に関しては，リソース所要量を固定することで，満足することができる。

5. 検証

本章では，まず検証方法を説明し，次にリアルタイム性とリソース所要量の観点で n-gram インデクスを用いた検索の有効性と差分インデクス分割反映の有効性，差分インデクス分割反映の検索性能への影響の検証結果を説明する。

5.1. 検証方法

検証に用いたハードウェア及びソフトウェアを表 1 に示す。今回の検証では，検討した差分インデクス分割反映方式の効果を確かめることを主目的とし，ＰＣ環境にて検証を行った。

表1　検証環境

#	分類	項目	環境
1	ハードウェア	CPU	Pentium(R)[1] 4 2.80GHz
2		メモリ	1024MB
3		HDD	ディスク容量：40GB ディスク回転数：5,400 回転/分 平均シーク時間：11ms データ転送時間：133MB/s(最大値) データバッファ：2MB
4	ソフトウェア	OS	Microsoft(R) Windows(R)[2] 2000 5.00.2195 Service Pack 4

検索及び文書の登録・削除の処理性能の検証において，使用したデータベースのテーブル構造とインデクスの概要を表 2 に示す。

表2　検証用データベース概要

#	項目	説明	補足
1	テーブル構造	文字列格納用カラム 1 列のみ	最大容量 256 バイト
2	インデクス	n-gram インデクス	文字列格納用カラムに対し，n-gram インデクスを作成

検証用データとしては，ゆうびんホームページで公開されている，平成 18 年 11 月 30 日更新版「事業所の個別郵便番号」のデータ 20,923 件から，事業所名のみを抜き出したものを使用した。本データは，全文検索機能の適用先の一つと考えられる施設名称のデータであり，すべて全角文字で構成され，かな・漢字・記号が含まれている。

今回の検証では，ディスク使用量を増やすことに主眼を置き､対象としたテキストの登録を 50 回繰り返して，1,045,150 件のデータを登録した状態から測定を開始した。これにより、登録済みデータの総量が検証に十分な量になり、n-gram インデクスの見出しが大きな場合であっても、効果が見込めるかを確認することができる。

5.2. 検証結果

検証では，n-gram インデクスの検索時間と，文書登録に必要な時間を測定する。この測定により，本研究の目的である，リアルタイム性の確保とリソース制約内での動作保証を検証する。

n-gram インデクスを用いた検索において，検索のリアルタイム性に対する検証のため，組込み DBMS 上で n-gram インデクスを使用した場合と使用しない場合について，検索に要する時間を測定した。その結果を表 3に示す。表中の値は，n-gram インデクスを使用した検索処理時間を基準とした相対値を示している。なお，基準値には，1 を用いた。

測定用の検索タームには，本方式適用先のターゲットの１つとして想定した施設名管理データに関係するものを選定した。また，検索結果の表示等の検索以外にかかる時間を無視できるように，検索結果のヒット件数が 0 件より多く，かつ，最もヒット件数が少なくなることも考慮して「ソフトウェア事業部」を今回の測定に用いる検索タームとした。なお，検索のヒット件数は 50 件である。

表 3の結果は,検索 1 回あたりに要した時間を示す。この時間は，検索を 100 回実行するのに要する時間から 1 回あたりの平均を求めたものである。また，計測はインデクス作成直後の，入力用領域とマージ領域の

[1] Pentium は，Intel Corporation のアメリカ合衆国及びその他の国における登録商標です。

[2] Microsoft は，米国およびその他の国における米国 Microsoft Corp.の登録商標です。Windows は，米国およびその他の国における米国 Microsoft Corp.の登録商標です。

両方が空の状態で行った。なお，検索時間は検索文の解析等による誤差を無くすため，構文解析後から，最終結果を取得するまでの時間，すなわち中間一致の処理のみに必要な時間を測定した。

表3　n-gram インデクス使用別検索時間比

#	n-gram インデクス使用の有無	処理時間比
1	使用	1
2	未使用	245

この結果から，n-gram インデクスを用いない場合は検索のリアルタイム性を満たしていない。n-gram インデクスを用いることで，検索処理時間が約 250 分の 1 に短縮でき，検索のリアルタイム性を満たしていることが分かる。

次に，文書の登録・削除処理に対するリアルタイム性の検証のため，差分インデクス分割反映方式を使用した場合と使用しない場合について n-gram インデクスの更新時間を測定した。ここで測定した時間は，バッファ領域から n-gram インデクスへの書き込みが発生した場合の処理時間である。これは，差分インデクス分割反映方式では，最初の入力用領域のみに書き込んでいる間は，n-gram インデクスの更新処理が実行されないためである。

n-gram インデクスの更新時間の平均値と最大値の測定結果を表 4に示す。表中の値は，バッファ領域サイズ 0Kの平均時間を基準とした相対値を示している。なお，基準値には，1.00 を用いた。バッファ領域サイズについては，8k バイト，16k バイト，32k バイト，64k バイト，128k バイトについて測定した。なお，バッファ領域サイズとは，差分インデクス分割反映方式が使用する入力用領域のサイズとマージ領域のサイズの合計である。n-gram インデクスを更新するために追加登録する文書としては，平成 18 年 11 月 30 日更新版「事業所の個別郵便番号」のデータ 20,923 件からランダムに抽出した 1000 件の事業所名を用いた。

表4　n-gram インデクスの更新時間の比較

#	測定項目	バッファ領域サイズ					
		0 K	8K	16K	32K	64K	128K
1	平均時間	1.00	0.89	0.67	0.62	0.62	0.62
2	最長時間	2.32	2.91	2.53	2.16	1.68	1.80

この結果から，バッファ領域サイズを特定サイズ以上とし，差分インデクス分割反映方式を用いることで，n-gram インデクス更新処理の平均時間，最長時間ともに短縮できることが分かる。ただし，平均時間はバッファ領域サイズを大きくするに従い短くなるが，最長時間はあるサイズを境に長くなる。これは，バッファ領域サイズを大きくすると，より多くの文書の出現位置情報がまとめられるために平均時間が短縮され，一方，入力用領域からマージ領域への書き込みに要する時間が長くなる分，最長時間が長くなるためである。リアルタイム性については，ＰＣ環境ではあるが，実測値での平均時間は 0.7 秒から 1 秒程度であり，特にバッファ領域サイズが 16K 以上の場合には 0.7 秒程度を確保しており，組込みシステム環境においても，適用効果を期待できる見込みを得た。しかし，最長時間は 2 秒近くかかっており，通常のタスクとしての要求を満たすのは困難との結果となった。

最後に，差分インデクス分割反映方式を使用することによる検索のリアルタイム性への影響を検証するために，文書登録後の n-gram インデクスの検索時間を測定した。差分インデクス分割反映方式を使用しない場合と，差分インデクス分割反映方式を使用した場合において、追加登録文書の総件数と検索所要時間の相関関係を図 4に示す。図 4の横軸は追加登録文書の総件数を示す。縦軸は，各総件数分登録した場合のバッファ領域サイズ 0Kの検索時間を基準とした相対値を示している。なお，基準値には，1.00 を用いた。実行した検索は，表 3で示す，n-gram インデクスを使用した検索と同一である。

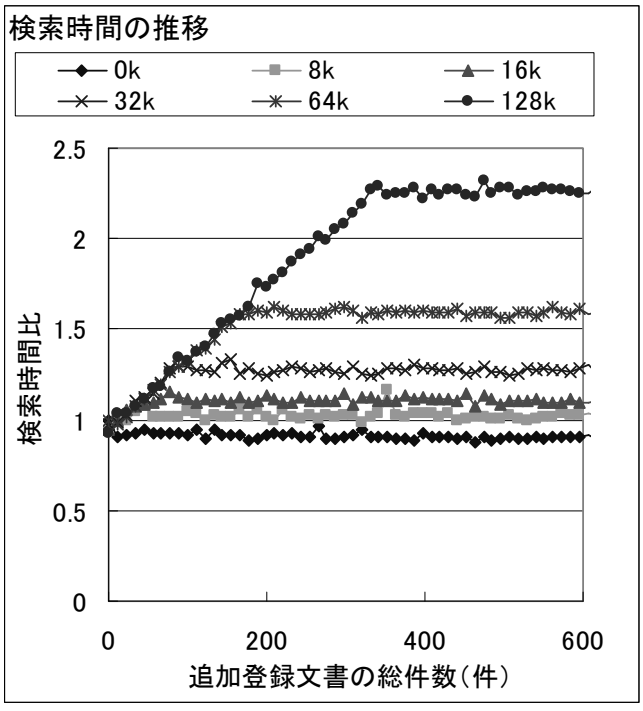

図4　文書登録件数に対する検索時間の変化

この結果から，追加登録文書総件数がある件数に達するまでは検索時間が長くなるが，それ以降はほぼ一定となることが分かる。また，バッファ領域サイズが大きいほど，検索時間の最大値が大きくなることも分かる。

差分インデクス分割反映方式を使用しない場合との検索時間の差に関しては，バッファ領域サイズを 2 倍にした場合に約 2 倍となり，バッファ領域サイズに比例することが分かる。これは，入力用領域とマージ領域を検索するための時間の影響であると考えられる。なお，バッファ領域サイズが 128K バイトの場合であっても，最長検索時間はリアルタイム性の指標である 150 ミリ秒に対し，十分に小さく無視できる範囲内である。従ってバッファ領域の検索による検索リアルタイム性への影響はほとんど無いことが分かる。

以上の結果から，検索時のリアルタイム性は確保できていることが分かる。一方，文書の登録・削除時のリアルタイム性については，改良の余地がある。リソース制約に関しては，バッファ領域の所要量を固定にでき,リソース制約内での動作を保証できると言える。

6. まとめと今後の課題

差分インデクス分割反映方式により全文検索処理におけるリアルタイム性を確保できることが確認できた。また，文書登録・削除処理に関しても，処理時間が短縮でき，かつ，検索性能への影響もほとんどないことが確認できた。ただし，今回の検証では，差分インデクス分割反映方式の効果の早期見極めを主目的にしたため，ＰＣ環境を用いた。今後は、今回の検証結果の妥当性を高めるために，組込みシステム環境での測定，実験データの拡充，処理時間の測定に使用する検索タームの種類数の増強などに取り組んでいく必要がある。n-gram インデクスの更新処理性能については，実用として使用するにはまだ不十分である。更新時間のほとんどはディスクアクセス時間であるため，今後は，バッファ管理や，データのクラスタリングなどの方式検討が必要である。また，n-gram インデクスの更新時に出力されるログ量が多いという問題もあるため，ログ量削減方式の検討も必要である。

文　献

[1] 原田昌紀，風間一洋，佐藤進也，“unicode を用いた N－gram 索引の一実現方式とその評価，”情報処理学会研究報告，Vol.2000，No.29，pp.127-134，2000

[2] 吉原潤，加藤和彦，“WWW 検索エンジンのためのインクリメンタルな全文検索インデックス更新方式，”情報処理学会論文誌：データベース，Vol.40 No. SIG8(TOD4)，pp.112-125，1999

[3] Ben Shneiderman,東 基衛，井関 治 監訳，“ユーザーインタフェースの設計―やさしい対話型システムへの指針，日経 BP 1995.