大規模テキスト並列検索エンジンRetrievalExpress (1) 並列検索方式

4N－7

福島 俊一† 赤峯 享† 会森 清† 田村 美保子‡ 柴田 茂明‡ 中塚 敏之‡
NEC† NEC情報システムズ‡

1 はじめに

今日、インターネットを介して極めて膨大なテキスト情報にアクセスすることが可能になり、また、新聞・特許などの大規模なテキスト情報をデジタル出版(CD-ROMなど)の形態で利用することも可能になっている。このようなテキスト情報を有効活用するためには、高速な全文検索技術が不可欠である[1]。

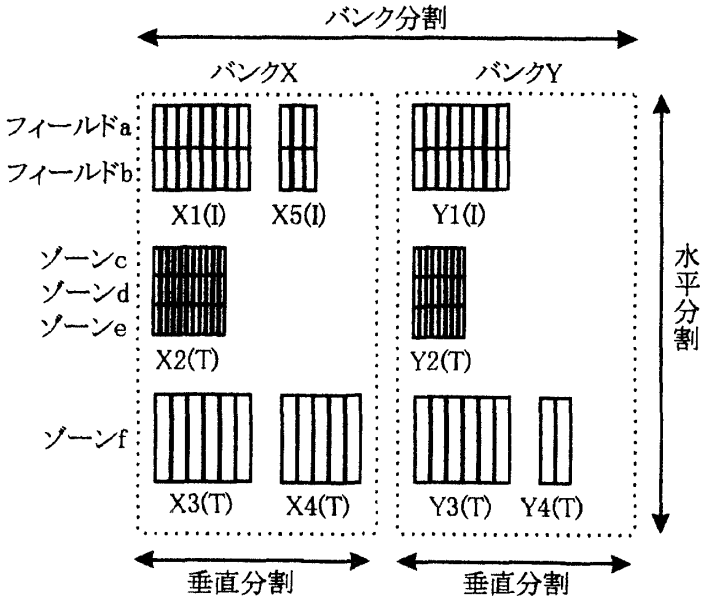
高速全文検索の要となるのは、インデックス(インバーテッドファイル)の形式とそのマッチングアルゴリズムである。日本語のようなべた書きテキストを対象として高速かつ洩れのない全文検索を実現するには、文字組をキーとして、その出現位置(テキストID＋オフセット)を記録するインデックス形式が有効なことが知られている[2][3]。筆者らの開発したフレキシブル文字列インバージョン法では、字種別キー文字列長、縮退文脈、高頻度キー文字列用サブインデックスなどを、検索対象テキストの文字列統計に適合させることで、検索レスポンスを改善した[3]。

このようなインデックス形式/マッチングアルゴリズムの工夫に加え、検索プロセスあるいはマシン構成を並列化することでスケーラブルな検索性能を提供することが、応用システム構築においては非常に重要である。すなわち、次第に増加していく検索対象テキスト量や、同時アクセスするユーザ数などに応じて、高速な検索レスポンスを確保できるような拡張性のあるアーキテクチャが必要になる。

全文検索における並列化のアプローチは、既にインターネット上のディレクトリサービスなどで実績があるようだが[4]、並列化の単位・方式・効果などに関する議論は必ずしも十分ではない。そこで、本稿では、フレキシブル文字列インバージョン法をベースとした検索エンジンRetrievalExpressについて、並列検索方式(インデックス分割による並列化、多重処理による並列化)とその評価結果を報告する。

2 インデックス分割による並列化

フレキシブル文字列インバージョン法のように、文字組をキーとして位置情報(テキストID＋オフセット)を対応させるインデックス形式では、位置情報の読み出し量が検索レスポンスを決定する最大の要因になる。また、位置情報の読み出し量は、登録テキスト容量とおよそ線形の関係になる。このことから、登録テキスト容量に応じてインデックスを分割し、分割した各インデックスを同時検索するような並列化によって、テキスト容量の増加に対しても高速な検索レスポンスを確保することが可能になる。



(I)は事項検索インデックス(フィールド検索用)
(T)は全文検索インデックス(ゾーン検索用)

図 1: インデックスの水平分割と垂直分割

そこで、RetrievalExpressでは、次のような3レベルのインデックス分割を行なうことにした(図1参照)。

バンク分割　登録テキストを、テキスト容量がほぼ同程度になるような複数のグループ(ここではバンクと呼ぶ)に分ける。

水平分割　RetrievalExpressでは、構造化テキストに対して題名・抄録・本文などのゾーンや日付・分類コードなどのフィールドを定義した検索が可能である[5]。このゾーン/フィールドを単位に、各バンクのテキストを切り分ける。

垂直分割　各バンクの各水平分割単位に対して最大インデックス容量を設定しておく。登録テキストが増加し、この最大容量を超えそうになったら、新たなインデックスに登録先を切り換える。

このように分割したインデックスを並列検索するために、次のようにプロセスやスレッドを割り当てて検索処理を実行する。まず、バンクの単位に検索プロセスを1つずつ割り当てる。さらに、その各バンクの検索プロセスは、水平分割&垂直分割の結果の各インデックスに対して、スレッドを1つずつ起動する。また、バンクごとの検索プロセスの全体を制御するプロセス(制御プロセス)も設ける。

Large-scale text parallel search engine RetrievalExpress (1) parallel search methods

Toshikazu Fukushima, Susumu Akamine, Kiyoshi Emori, Mihoko Tamura, Shigeaki Shibata and Toshiyuki Nakatsuka
NEC Corp., NEC Informatic Systems Ltd.

パタン	並列度	検索対象インデックス (*1)	マシン数 (*2)	マシンあたり検索プロセス数(*3)	検索要求発行ユーザ数
(ア)	1	A	1	1	1
(イ)	2	A,B	2	1	1
(ウ)	3	A,B,C	3	1	1
(エ)	2	D,E	2	1	1
(オ)	3	F,G,H	3	1	1
(カ)	1	A	1	1	10
(キ)	2	A	1	2	10
(ク)	3	A	1	3	10

(*1) インデックスは各々別マシンにおく。
特許抄録テキスト約260万件を3分割→A,B,C。
Aを2分割→D,E。Aを3分割→F,G,H。
(*2)NEC EWS4800/460(R10000×2CPU、200MHz)。
(*3) これら検索プロセスのほかに制御プロセスが動く(全体に対して1個)。

図 2: 実験対象とした並列化構成のパタン

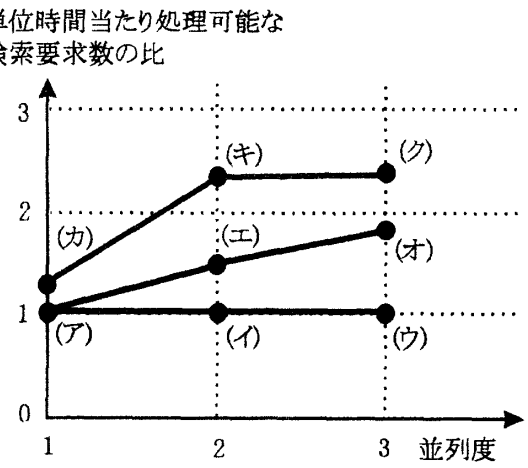


図 3: 並列化による検索レスポンスの改善効果

制御プロセスは、ユーザからの検索要求をすべての検索プロセスにばらまき、それらの検索結果を統合する(テキストID集合をOR結合する)。各検索プロセス内では、水平分割方向については、検索要求をフィールド/ゾーンごとに分割し、検索結果はフィールド/ゾーン間の演算(AND/OR)にしたがって統合する。垂直分割方向についてはOR結合する。

3 多重処理による並列化

第2章で述べたようなテキスト容量増加に対するスケーラビリティに加えて、ユーザからの同時アクセス数の増加に対するスケーラビリティも必要である。

そこで、RetrievalExpressでは、第2章で述べた各バンクに対応する検索プロセスをあらかじめ定めた並列度で複数個起動しておき、多重処理できるような構成にした。同一のバンクに対応する複数の検索プロセスが、同一のインデックスセットを、異なる検索要求で同時並行的に検索することになる。このような並列化によって、前の検索要求に対する処理が完了していなくとも、次の検索要求の処理を開始し得るため、スループットが向上する。

4 性能評価

第2章・第3章で述べた並列検索方式の効果を確認するために、図2に示す8通りの並列化構成について、特許公開公報の抄録テキスト(約260万件で2GB弱)を用いて評価した。検索要求には特許遡及検索に実際に用いられたキーワード条件式(1877件)を用い、すべての検索要求を連続実行するのに要した時間を測定して、単位時間あたりに処理できた検索要求数の比を比較した。その結果を図3に示す。

(ア)～(オ)はインデックス分割による並列化に対応する(ただし、今回はバンク分割のみを比較評価した)。(ア)(イ)(ウ)の結果より、登録テキスト容量が増加しても、増加分を別インデックスにバンク分割して並列検索することによって、ほぼ一定の検索レスポンスが保てることがわかる。また、(ア)(エ)(オ)の結果より、インデックスを複数個のバンクに分割して並列検索することで、検索レスポンスをほぼ線形オーダーで改善できていることがわかる。ただし、検索プロセスと制御プロセス間の通信などのオーバヘッドが現状ではかなり大きく、並列度の増加に比較した検索速度の向上は4割程度にとどまっている。

(カ)～(ク)は多重処理による並列化に対応する。1877件の検索要求を10分割し、10ユーザから検索を実行した。2CPUマシンで実行したために並列度2で頭打ちにはなったが、スループットの改善効果は確認できた。

5 おわりに

大規模テキスト並列検索エンジンRetrievalExpressに実装した並列検索方式を述べた。インデックス分割による並列化と多重処理による並列化により、検索対象テキスト量および同時アクセスユーザ数に対するスケーラブルな検索性能が実現できることを確認した。ただし、現状では、並列検索の制御に関わるオーバヘッドがかなり大きいと思われ、今後はその分析や改良を進める予定である。また、マルチスレッド化の効果についても評価する予定である。

参考文献

[1] 道本ほか、高速全文検索の威力(特集)、日経バイト、1996年10月号。
[2] 菊池、日本語文書用高速全文検索の一手法、信学論、第J75-D-I巻、第9号、1992年。
[3] 赤峯ほか、高速全文検索のためのフレキシブル文字列インバージョン法、情処ADBS、1996年。
[4] 丹波ほか、AltaVistaにおける大規模検索、情処ADBS、1996年。
[5] 赤峯ほか、大規模テキスト並列検索エンジンRetrievalExpress (2) 構造化テキスト検索方式、情処55全大、4N-8、1997年。