全文検索システムのための 複数転置ファイルを用いた登録高速化とランキング検索 (Extended Abstract)

小川泰嗣† 山本研策†
真野博子† 伊東秀夫†

1. はじめに

文書電子化とコンピュータネットワークの進展に伴い、蓄積された大量の文書群から所望の文書を見つけ出す文書検索へのニーズはこれまで以上に高まっている。われわれは、これまで n-gram 索引を用いた検索の高速化[9)12)13)] および高精度化[10)11)] の研究を行なってきた。これらの成果はサーバークライアント型の検索エンジン $\mathcal{FTS}$ として実用化され、図書管理システム LIMEDIO[7)]、オフィス文書管理システムの Ridoc シリーズ[14)] 等で使用されている。

$\mathcal{FTS}$ は n-gram 索引（n 文字組を索引単位とする転置ファイル）を用いている。転置ファイルでは、文書量の増大とともにその登録時間が増大することが問題である。ひとたび索引を作成した後には登録／削除がない静的な環境向けには、複数の文書における索引単位の出現情報を大量のメモリバッファに蓄えることでディスクアクセスを減らすバッチ登録が有効だが[1)]、大量文書が登録されている索引に対する少数の追加登録にはバッチ登録は有効ではない。しかし、今日的なアプリケーションでは文書の追加登録も頻繁に起こっており、文書を登録した直後から検索できる必要があるため、追加登録の高速化が重要である。

本発表では、まず、複数の転置ファイルを用いることで文書の追加登録を高速化する方法を提案する。複数転置ファイルを用いた場合、文書検索で一般的なランキング検索の検索精度／速度が低下する。そこで、その問題を解決するものとして転置ファイルが単一の場合と同じスコアを効率的に計算する方法を提案する。最後に、NTCIR-2 コレクションを用いた評価実験結果を報告する。

† (株) リコー ソフトウェア研究所
Software Research Center, RICOH Co., Ltd.

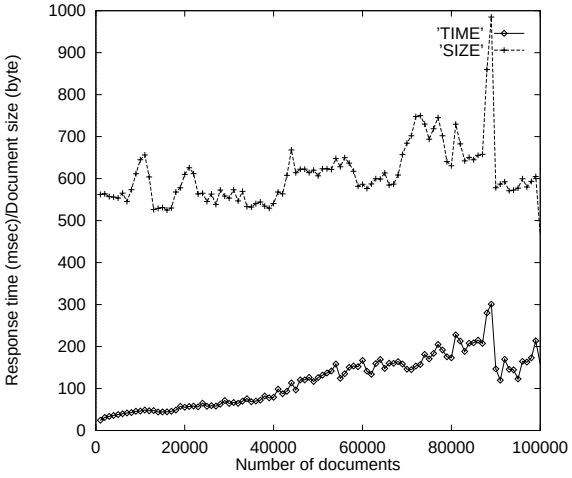

図 1 累積文書数と登録時間の関係

2. 複数転置ファイルを用いた登録高速化

2.1 登録文書数と登録時間の関係

転置ファイルは登録文書群から抽出した索引単位（単語索引であれば単語であり n-gram 索引であれば n-gram）の出現情報を索引単位ごとにまとめて記録するものである。1つの文書から抽出された索引単位の出現情報は、累積登録文書数が少ない間はディスク上の少ないページにまとまって記録されているが、累積文書数の増大にともなって多くのページに分散されるようになる。したがって、1件当たりの登録時間も累積文書数に応じて増大する。

NTCIR-2 コレクションを用いた登録実験（詳細は 4.1 節参照）における1件当たりの登録時間`TIME` および文書サイズ`SIZE` を累積の登録文書数ごとにプロットしたものを**図 1** に示す。文書サイズに応じて多少の変動はあるものの、累積文書数に応じて登録時間が単調に増加していることが確認できる。

2.2 複数転置ファイルを用いた登録処理

累積文書数にともなって登録時間が増大するという

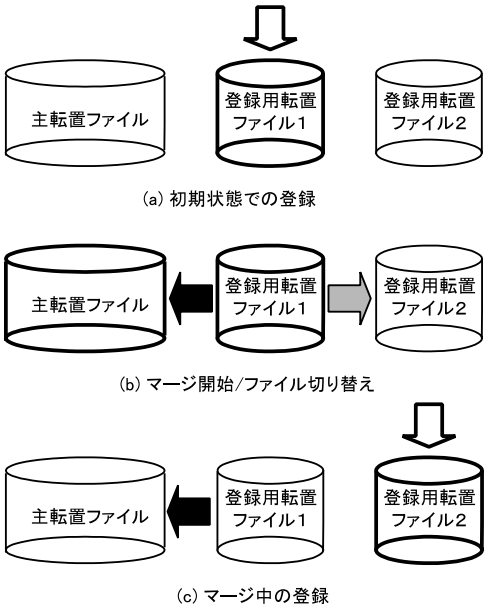

図 2 転置ファイルの切り替え

転置ファイルの問題点は、登録用に登録が高速な索引を用意しておくことで解決できる。登録用索引としては、転置したデータ構造をメモリ上[2)15)] あるいはファイル上[3)] に持つことが考えられる。いずれの方式でも、登録用の索引は一時的なものなので、登録用索引に記録した文書をそのまま放置できず、その大きさが閾値を超えた時点で登録用索引の内容を主要な転置ファイル（以下、主転置ファイル）に反映する必要がある。

$\mathcal{FTS}$ は、以下の点を考慮し、登録用索引として主転置ファイルと全く同じ構造の転置ファイルを採用することとした。

- 使用メモリ量を抑えられる
- 登録直後から検索できる
- トランザクション処理等をそのまま利用できる

登録用転置ファイルが一定の大きさになると、その内容を主転置ファイルにバックグラウンドでコピーする（この操作をマージ処理と呼ぶ）。マージ処理の間にも登録処理を受け付けることができるように登録用転置ファイルは２つ用意しておき、一方がマージしている間には残りの登録用転置ファイルを登録に使用する。また、マージし終えた転置ファイルは空にして、つぎの順番に備える。このようすを**図 2** に示す。

なお、マージ処理は主転置ファイルに対する更新なので排他制御の必要があるが、ファイル単位に排他制御したのではマージ中に検索できなくなる。そこで、マージ処理では索引単位ごとに排他制御することとし、マージ中でも検索可能とした。

3. 複数転置ファイルでのランキング検索

3.1 ランキング検索

文書検索におけるランキング検索では、検索文字列の頻度情報に基づいて文書のスコアを計算する。$\mathcal{FTS}$ は確率モデル[1)] を採用しており、文書頻度（検索語 t を含む文書数）f_t、文書内頻度（文書 d における t の出現回数）$f_{d,t}$、検索要求内頻度（検索要求 q における t の出現回数）$f_{q,t}$ とすると、q に対する d のスコアを下式で計算する[10)11)]。

$$score(d,q) = \sum_{t \in q} \alpha_t \cdot \left(k_t \cdot \log \frac{N}{f_t} + 1 \right) \cdot \frac{f_{d,t}}{K + f_{d,t}} \cdot \frac{f_{q,t}}{k_q + f_{q,t}}$$

$$K = k_d \cdot \left((1-b) + b \frac{l_d}{l_{ave}} \right)$$

ここで、N は全文書数、l_d は d の文書長、l_{ave} は登録文書に対する平均の文書長、k_t, k_d, k_q は各頻度情報の影響の調整パラメータ、b は文書長の調整パラメータ、α_t は t の品詞等によって定まる定数である。

3.2 ランキング検索における問題点

複数転置ファイルを用いた場合、転置ファイルごとの文書頻度が検索対象全体に対する文書頻度と異なることが問題となる。単純には検索対象全体に対する大域的な文書頻度を索引単位ごとに別途記録しておけばよい[5)6)]。しかし、n-gram 索引においては、検索語が n 文字（n-gram）でない場合には複数の n-gram の出現位置から検索語の頻度情報を計算によって求める必要があるため[9)]、この方法は適用できない。

別の方法としては、大域的な文書頻度を利用せずに転置ファイル（あるいは検索サイト）ごとに局所的な文書頻度に基づいてランキング検索を行ない、最後にそれらの結果をマージする方法がある[4)5)]。この方法（以下、局所法）は n-gram 索引にも適用できるが、文書頻度の相違から検索精度が低下することが問題である。検索精度の低下はファイル数に応じて大きくなる傾向がある。登録高速化のための複数転置ファイル方式ではファイル数は最大でも３つと少ないが、転置ファイルのサイズに大きな偏りがあるので、精度低下の問題は残ると考えられる。

3.3 単一走査法

高い検索精度のため、大域的な文書頻度に基づいてスコア計算するには以下のようにすればよい。まず検索語の大域的な文書頻度を転置ファイルごとの文書頻度を合計して求め、つぎに転置ファイルごとに大域的な文書頻度を用いてランキング検索し、最後に転置ファイルごとの結果をマージする。

しかし、このような単純な方法を n-gram 索引に適用した場合、n 文字でない検索語の文書頻度を求める際に検索語を構成する n-gram の出現位置を走査しながら検索語の出現を判定しなければならないことが問題となる。文書スコアを計算する際にも、文書内頻度を求めるために n-gram の出現位置を走査するので、転置ファイルを２回に渡って走査しなければならず、検索速度が低下するからである。

この問題に対し、文書頻度を求める際に検索語が出現している文書については文書内頻度（全ての出現位置の個数）も求めることで対応する。このようにすることで、転置ファイルごとの文書頻度がわかった時点には、検索語が出現している文書の文書内頻度も求まっている状態になる。したがって、大域的な文書頻度を求めた後に各文書のスコアを計算する際には、前のステップで求めたメモリ上にある検索結果を走査するだけで良く、転置ファイルの走査が不要となるため検索が高速化できる。この方法を単一走査法と呼ぶ。

4. 評価実験

4.1 実験方法

実験には国立情報学研究所が作成した NTCIR-2 の随時検索タスク用のテストコレクション[8] を使用した。このコレクションの特性は以下の通りである。

- 検索対象：学会発表データベース、科学研究費補助金研究成果概要データベース
- 対象規模：736166 件
  配布されたデータからテキスト部分を抽出した。テキスト量は合計約 820MB、１件当たりでは約 1.1KB。ただし、検索対象によって１件当たりの文書サイズは異なっており、科研費データベースは学会発表データベースの約３倍ある。
- 検索要求：49 件
  各検索要求は複数のフィールドから構成されるが、今回は数語の名詞句で内容を簡潔に表現した title フィールドのみを使用した。

測定には、SUN Blade 1000 (CPU UltraSPARC-III 750MHz; メモリ 1GB) を用いた。

4.2 登録高速化の効果

NTCIR-2 コレクションを、従来の単一転置ファイル方式と今回提案した複数転置ファイル方式で登録した場合の１件当たりの登録時間を累積文書数ごとにプロットしたものを**図 3** に示す☆。この図では、`SINGLE` が単一転置ファイル方式、`MULTI` が複数転置ファイル方式である（`SINGLE` は図 1 の`TIME` と同一）。この図から明らかなように、`MULTI` では登録時間が累積登録文書数によらず一定の値を示しており、複数転置ファイル方式の有効性が確認できる。

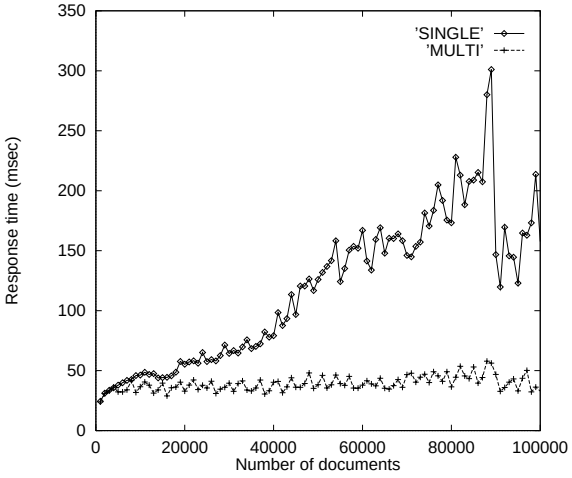

図 3 単一転置ファイル方式と複数転置ファイル方式での登録時間

表 1 検索精度の比較

	局所法	単一走査法
AveP	0.2277	0.2411
P@20	0.3286	0.3418

コレクション全体の登録時間は以下の通りで、提案した複数転置ファイル方式により全登録時間も大幅に短縮できた☆☆。

- 単一転置ファイル方式： 178.8 時間
- 複数転置ファイル方式： 22.3 時間

4.3 検索精度への影響

複数転置ファイル方式によりコレクション全体を登録した状態では主転置ファイルには 735899 件（全体の 99.964%）、登録用転置ファイルの１つには 267 件（全体の 0.036%）が分配されていた。この状態で、局所的な文書頻度に基づく局所法と、今回提案した単一走査法で検索を行なった。

まず、検索精度 — 代表的な指標である平均適合率（AveP）と上位 20 文書での適合率（P@20）[1] — への影響を調べた。49 件の検索要求に対する評価指標の平均値を**表 1** に示す。局所法に比べて単一走査法は平均適合率で 5.9%、上位 20 文書での適合率で 4.0% 向上しており、大域的な文書頻度に基づいてスコアを計算する必要性が確認できた。なお、単一走査法は大域的な文書頻度に基づいてスコア計算するので、検索結果は単一転置ファイルの場合と同じである。

☆ 正確には、先頭の 100,000 文書について、1000 件の登録時間と文書サイズの平均値を 1000 件ごとにプロットしたもの。

☆☆ バッチ登録はさらに高速であり、登録時間は 5.8 時間であった。なお、文書サイズは図 1 と同じなので省略した。

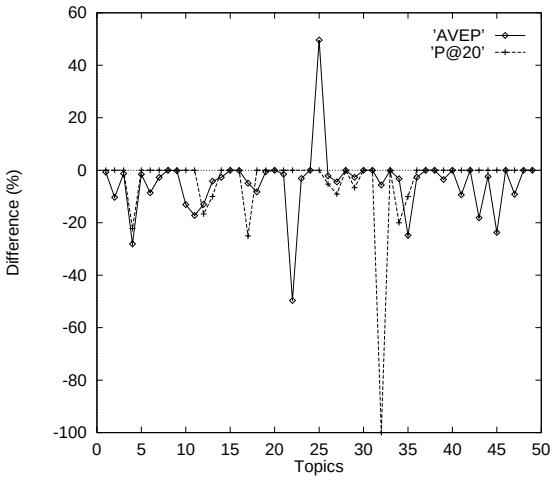

図 4 検索要求ごとの検索精度の変化の割合

つぎに 2 つの手法による評価指標の差異を検索要求ごとに示す（**図 4**；局所法の精度が低い場合にマイナスとなる）。両者で検索精度が変わらないこともあるが（上位 20 文書での適合率では影響のないものが多い）、局所法のほうが検索精度が良いのは 1 件しかないの対し、低下するものは多く、20%以上の大幅な低下をする検索要求も 6 件もある。この図からも局所法では検索精度面で問題があることわかる。

最後に検索時間（49 個の要求文の検索時間の平均値）への影響を調査する。複数転置ファイルにデータが登録された状態で局所法および単一走査法を適用した場合、登録用転置ファイルを主転置ファイルにマージした後に検索した場合（このとき局所法と単一走査法の区別に意味がない）の 3 つの場合について測定した。結果は以下の通りである。

- 局所法： 1.53 秒
- 単一走査法： 1.54 秒
- マージ後： 1.47 秒

複数転置ファイルの場合の局所法と単一走査法の差はわずかであり、検索精度を考慮すると、複数転置ファイルに対するランキング検索法として単一走査法が有効であると考えられる。マージして 1 つの転置ファイルとした方が複数転置ファイルに対する検索よりも 4.1% 高速である。

5. おわりに

文書の追加登録を高速化するために複数の転置ファイルを切り替えながら利用する複数転置ファイル方式を提案した。さらに、複数転置ファイルに対しても検索速度／検索精度を低下させないランキング検索として単一走査法も提案した。NTCIR-2 コレクションを用いた評価により、提案手法の有効性を確認できた。

参 考 文 献

1) Baeza-Yates, R. and Ribeiro-Neto, B.: *Modern Information Retrieval*, ACM Press (1999).
2) Barbara, D., Mehrotra, S. and Vallabhaneni, P.: The Gold Text Indexing Engine, *Proc. of Int. Conf. on Data Engineering*, pp. 172–179 (1996).
3) Brown, E., Callen, J. and W.B.Croft: Fast Incremental Indexing for Full-Text Information Retrieval, *Proc. of 20th VLDB Conf.*, pp. 192–202 (1994).
4) Callan, J. P., Lu, Z. and B., C. W.: Searching Distributed Collections With Inferenece Networks, *Proc. of 18th ACM SIGIR Conf.*, pp. 21–28 (1995).
5) Gravano, L., Chen-Chuan, K. C. and Garcia-Molina, H.: STARTS: Sanford Proposal for Internet Meta-Searching, *Proc. of ACM SIGMOD Conf.*, pp. 207–218 (1997).
6) Harman, D., McCoy, W., Toense, R. and Candela, G.: Prototyping a Distributed Information Retrieval System Using Statistical Ranking, *Information Processing and Management*, Vol. 27, No. 5, pp. 449–460 (1991).
7) LIMEDIO: http://www.ricoh.co.jp/limedio/.
8) NTCIR Workshop: http://research.nii.ac.jp/ntcir/ntcir-ws2/work-ja.html.
9) 小川泰嗣: 擬似頻度法: n-gram 索引のための高速な日本語文書のランキング検索法, 電子情報通信学会論文誌, Vol. J83-D-I, No. 10, pp. 1043–1054 (2000).
10) Ogawa, Y. and Mano, H.: RICOH at NTCIR-2, *Proc. of 2nd NTCIR Workshop*, pp. 227–229 (2001).
11) Ogawa, Y., Mano, H., Narita, M. and Honma, S.: Structuring and expanding queries in the probabilistic model, *Proc. of 8th TREC*, pp. 541–548 (2000).
12) 小川泰嗣, 松田透: N-gram 索引を用いた効率的な文書検索法, 電子情報通信学会論文誌, Vol. J82-D-I, No. 1, pp. 121–129 (1999).
13) 小川泰嗣, 松田透, 橋本信次: N-gram 索引における複合条件の効率的な処理方法, 情報処理学会論文誌データベース, Vol. 40, No. SIG5 (TOD2), pp. 43–53 (1999).
14) Ridoc Document System: http://www.ricoh.co.jp/ridoc_ds/rds/.
15) Tomasic, A., Garicia-Molina, H. and Shoens, K.: Incremental Updates of Inverted Lists for Text Document Retrieval, *Proc. of 1994 ACM SIGMOD Conf.*, pp. 289–300 (1994).