D-001

大規模並列全文検索エンジンにおける多言語検索対応索引方式
Index Method for Multilingual Retrieval in Large Scale Parallel Full Text Search Engine

中村 隆顕† 山岸 義徳† 郡 光則†
Takaaki Nakamura Yoshinori Yamagishi Mitsunori Kori

1. はじめに

全文検索エンジンにおいて、多言語の文書の検索を可能にする多言語検索対応の必要性が高まっている。N-gram 索引を適用した全文検索エンジンを多言語に対応した場合、文字種の数の増加や、文字の出現頻度の偏りによって、検索速度が低下する課題がある。本稿では、その課題を解決するための索引方式を提案する。また、提案方式を実装し、検索速度性能を評価した結果についても報告する。

2. 従来方式の課題

2.1 多言語応文字コードの課題

全文検索エンジンにおいて、多言語検索対応するためには、その基本となる文字コードを、Unicode の様な多言語の文字を単一の文字コード体系で扱うことが出来る文字コードとする必要がある。しかし、その様な文字コードには、次の 2 つの特徴がある。

第一の特徴は、文字種の大幅な増加である。特に N-gram 索引方式の場合、N の値に応じて文字の組み合わせの数が大幅に増加する。

第二の特徴は、文字の出現頻度の偏りである。全文検索エンジンの実運用の場面では、少数の言語の文字が集中して使用されることが考えられる。Unicode では、その文字コード空間上に言語毎に文字が配置されているため、特定の言語の領域に割り付けられた文字の出現頻度だけが高くなる。

2.2 N-gram 索引方式の課題

N-gram 索引方式の索引は、N-gram 毎の位置情報を記録した索引データ部と、索引データ部上の N-gram 位置情報の格納位置情報を記録する管理情報からなる。管理情報には、高速に N-gram 位置情報の格納位置が参照できるために、木構造やトライ構造などのデータ構造が広く採用されている。しかし、木構造やトライ構造の管理情報では、文字種の数が多い場合に、管理情報のサイズが大きくなりメモリに収まらなくなると、索引データの参照効率が低下すると言う課題があった。

3. ブロック化 N-gram 索引方式

3.1 ブロック化 N-gram 索引方式の構成

我々は、大規模文書の高速検索を目的とした、日本語文字コードのN-gram方式による大規模並列全文検索エンジンを開発した[1][2][3]。本検索エンジンでは、1TBのメール相当のテキスト情報から 1 秒で検索することができる。我々はブロック化N-gram索引方式を開発することにより、大規模においても高速な検索エンジンを実現した。ブロック化N-gram索引の索引データ部は、以下の要素から構成される。

- 1 種の N-gram の位置情報を複数文書分まとめた N-gram 位置情報
- N-gram 位置情報を複数種の N-gram 分まとめた索引ブロック。索引ブロックの数は、管理情報や索引ブロックのサイズを考慮すると、256K 個程度が効率的である。
- 索引ブロックを複数まとめたブロックセット

そして、管理情報をハッシュにより直接検索する構造として、索引データを参照することにより、スケーラブルな検索性能を実現した。

3.2 ブロック化 N-gram 索引方式の課題

ブロック化 N-gram 索引方式では、ブロックセット毎に並列 I/O 処理を行う。このとき、索引のブロックのサイズが均等でないと、I/O の効率が低下し、検索速度が低下する課題がある。

特に Unicode において、N-gram 位置情報と索引ブロックとの対応関係を、単純に文字コード空間上の文字の配置に従って決定すると、同じ言語に属する文字に関する N-gram 位置情報が、少数の索引ブロックに集中して格納される。このとき、N-gram の種類の数が多く、文字の出現頻度の偏りがあるために、特定の索引ブロックのサイズが極端に大きくなる。これにより索引ブロックのサイズに偏りが生じ、検索速度が低下する。

4. 多言語検索対応索引方式

4.1 内部文字コードの定義

これらの課題を踏まえて、全文検索エンジンの高速性を維持しつつ、多言語対応文字コードに対応するためには、使用言語の組み合わせに依存せず、索引ブロックのサイズを均等化する索引生成方式が必要である。そこで、この条件を満たす適切なハッシュ関数を採用することにより、索引ブロックのサイズを均等化した。N-gram 位置情報と索引ブロックの対応関係を決定する手順は以下の通りである。

① 元の文字コードを元に、各 N-gram を一意に識別するための内部文字コードを算出する。
② ①で算出した内部文字コードの順に、N-gram 位置情報を索引ブロックに割り付ける。

①において、元の文字コード空間上で近接した文字であるほど、その文字に関する内部文字コードが分散されるように定義する。それにより、同一の言語に属する文字に関する N-gram 位置情報を、索引ブロック全体に渡って均等に分散させることが可能になる。

以下では、N-gramの内部文字コードの具体的な算出手順について説明する。まず、検索エンジンで使用する基本の文字コードをUnicodeとする。Unicodeの文字コードは最大で 21bitの数値（0x000000～0x1FFFFF、1,114,112 種）とす

る。図 1は 2-gram「携帯」を例にした内部文字コードの算出の例である。

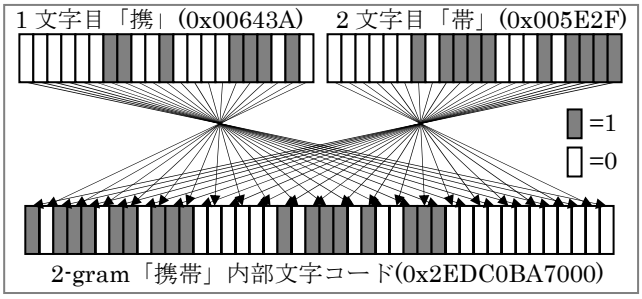

図1　2-gram「携帯」の内部文字コード

内部文字コード生成の基本アイデアは、「bit 転置」と「bit 交互配置」の組み合わせである。bit 転置は、文字コードの値の上位と下位のビット入れ替えるもので、近接した文字の値を分散させる効果が得られる。bit 交互配置は、2 以上の N-gram の場合に、そこに含まれる文字の値から、交互に 1bit ずつ取り出し並べ替えるもので、個々の文字コードの値が、並べ替え後の値に与える影響を小さくする効果が得られる。元の文字コードを 21bit の数値とすると、内部文字コードの bit 長は、1-gram で 21bit、2-gram で 42bit としている。同様にすることにより、1～3-gram までを 64bit の数値の範囲で識別することが可能である。

4.2 演算処理の高速化

図 1の例では、1bit単位でbit転置と交互配置を適用した。1bit単位で、この算出処理を行うと、内部文字コードの分散の度合いを高めることが可能であるが、一方で内部文字コードの算出にかかる計算量が増加する。そこで、内部文字コードの分散度合いを最低限維持しつつ、処理単位のbit数を増やす方式が考えられる。さらに、入れ替える単位となるbit数を固定ではなく、bit位置によって変化させることにより、計算量を削減させつつ、分散の度合いを高める方式が考えられる。

実際に、複数パターンの bit 転置と交互配置の組み合わせについて比較・検討し、内部文字コードの算出方式を決定した。今回の索引方式では、索引ブロックの数を約 256K 個（=18bit）としている。即ち、内部文字コードの先頭から 18bit の値によって、どの索引ブロックに格納されるかが決定される。よって、2-gram の場合は、その 18bit の中に元の文字の下位の 9bit が割りつくようにし、内部文字コードの分散を図った。それ以外の領域では、複数 bit 単位で交互配置することにより計算量の削減を図った。

5. 評価

5.1 評価方針

本稿で提案した全文検索エンジンの索引方式の有効性を確認するため、検索速度性能の評価を実施した。今回は、本全文検索エンジンを、文献[3]のメールアーカイブシステムに適用し、同様の評価を実施した。このメールアーカイブシステムは、日本語文字コードを検索の対象としたシステムである。これと同等の検索速度性能（メール 1TBから 1 秒）であることを確認することにより、提案の索引方式の有効性を確認する。

評価システムの構成を表 1に示す。また、索引は 1-gram と 2-gramを併用した索引とした（表 2）。測定は、10 種類の検索キーワードに対して、その検索時間を測定した。

表 1　評価システム構成

OS	Windows 2003 Server x64 Enterprise Edition
CPU	Intel Xeon E5345 2.33GHz（Core2×4）
Memory	24.0 GB
HDD	台数：8 台(RAID5), 回転速度：15,000rpm, キャッシュ：8MB, 接続 I/F：Ultra SCSI 320

表 2　メールと索引

メールの件数	20,802,390
メールの合計サイズ	1,028.534 [GB]
文字数	103,582,078,684
索引サイズ	1,364.980[GB]

5.2 索引の検索速度性能

検索速度性能の測定結果（図 2）に示した通り、1TBの大規模メールアーカイブに対して 1 秒で検索可能であることを確認した。

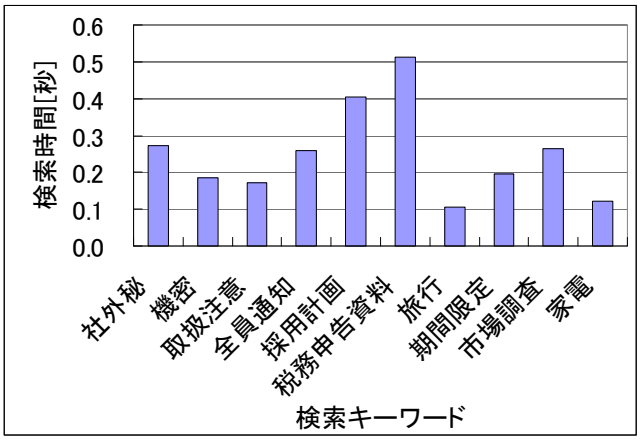

図2　検索速度性能（単位：秒）

6. おわりに

N-gram 索引方式に基づく全文検索エンジンにおいて、多言語検索に対応した場合、検索性能が低下する課題があることを示した。課題に対して、各 N-gram の位置情報と索引ブロックの対応を適切に設定することにより、文字の出現頻度の偏りによる、データ I/O 効率の低下を抑止した。また、提案方式を実装し、その効果を確認した。

参考文献

[1]郡 光則, 山岸 義徳, 清水 英弘, 金子 洋介, “検索機能を備えたストレージシステムによる大規模並列全文検索”, 信学技報, Vol.102, No.276, 41-46, (2002).

[2]清水 英弘, 山岸 義徳, 郡 光則, “n-gram 索引による大規模・並列全文検索方式 － (2) 索引の最適化”, 信学会ソサエティ大会, D-4-4, pp-21 (2001).

[3]竹内 丈志, 加藤 守, 山岸 義徳, 中村 隆顕, 郡 光則, “大規模電子メールアーカイブシステム向け高速蓄積・検索方式”, FIT2007, D-018, (2007).