日本語形態素解析への複数ハッシュふるい分けの応用

6M－5

中本賢一　山本毅雄
図書館情報大学

1. はじめに

最近、大きなメモリが利用できるようになり、並列処理が普及しつつある。自然言語処理のアルゴリズムはこの変化を反映する必要がある。しかしメモリが大きくなっても、長い語長のキーをきわめて多数持つ辞書を、通常のハッシュ表にすることは、必ずしも常に可能ではない。

本研究では「複数ハッシュふるい分け」[1][2][3]の形態素解析への応用について述べる。「ふるい分け」とは、与えられたキーがあるデータ集合中に存在するかどうかを調べることである。複数ハッシュふるい分けでは、複数のハッシュアドレス生成に互いに独立なn個の異なるハッシュ関数を使用し、ハッシュ表の該当アドレスにはデータの存在を示す1ビットのフラグを置く。探索時には同じハッシュ関数のセットを用いて探索キーをハッシュし、すべてのアドレスの内容が1であれば、辞書中のキーと一致したとみなす。この方法ではハッシュ表が比較的小さくなり、また並列処理の適用が容易である。

また、本研究では長い文字列を「畳み込む」ためのアルゴリズムについても検討する。

2. 日本語文字列の畳み込み

長い文字列Zから、このハッシュアドレスを求める過程は、普通二つの段階に分けられる[2][4][5]:Zから比較的大きなビット長をもつハッシュキーKを求める操作T(Z)、Kからハッシュアドレスを求める操作h(K)である。T(Z)は通常「畳み込み」と呼ばれる。h(K)については多くの研究があるが、T(Z)については調べられることが少ない。しかし英語と異なり、日本語においては長大な文字列の処理が頻出するので、畳み込みT(Z)の品質について検討した。

畳み込みには文字列を適当な長さに切り分けて部分列を作り、これらの、あるいはこれらに適当な演算を施したもの、の和あるいは排他的論理和をとる。ここではEUCコードを使用した日本語の文字列Zから4byteのキーを生成する畳み込みについて、次の各方法を比較した。

ここで $\bigoplus_{i=0}^{n/2}$ はi＝0,…,⌈n/2⌉までの各項の排他的論理和を表わし、m_iはZの2文字をとった部分文字列(z_{2i}, z_{2i+1})(ビット列あるいは2進数)である。nが奇数のとき$m_{n/2}$は$(0, z_n)$とする。cy^iは左へ2進i桁の巡回シフト、$prime^i$は3から数えてi番目の素数である。

(A)　単純排他的論理和

$$T(Z)=\bigoplus_{i=0}^{n/2} m_i$$

(B)　巡回シフト後排他的論理和

$$T(Z)=\bigoplus_{i=0}^{n/2} cy^i(m_i)$$

(C)　異なる素数を乗じて排他的論理和

$$T(Z)=\bigoplus_{i=0}^{n/2} prime^i \times m_i$$

(D)　ビットシフト後総和[6]

$$T(Z)=\sum_{i=0}^{14}(2^i\times z_i)+\sum_{i=15}^{n} Z_i$$

上の(A)-(D)では日本語文字列を畳み込むと表1のように多数の衝突を起こす。本研究ではより衝突の少ない次の方法を用いた。

(E)　擬乱数化後排他的論理和

$$T(Z)=\bigoplus_{i=0}^{n} f^i(z_i)$$

ここで$f(z)=f^0(z)$はzを種とする擬乱数生成系,$f^1(z)=f(f(z))$,$f^2(z)=f(f(f(z)))$,……である。これをEUCコードから第0、第4の2ビットを除いた14ビットを第1インデックスとした(2^{14},

---

Screening by multiple open hashing for Japanese token analysis
Ken'iti NAKAMOTO and Takeo YAMAMOTO
University of Library and Information Science

30)行列の文字コード表として保存する。これに必要な記憶容量は約1.2MBである。

実験で使用した日本語形態素辞書はICOTフリーソフトウェア「形態素辞書」をもとにしている。この辞書に含まれる動詞、形容詞の語幹および活用語尾を使用して、動詞、形容詞の活用形を作成し、辞書のエントリに加えた。その結果、作成した辞書は、表記が同一の形態素を一つとみなした場合、236541件の形態素を含む。形態素の長さの平均は約6.8byteである。f(z)としては $(z\times a) \bmod m, a = 16807, m = 2^{31}-1$ [7]を使用して、上の(A)-(B)の各方法で4byte整数としてT(Z)を求め、この衝突数を計算した。結果を表1に示す。ここで、クラスタ数は衝突によって生じた形態素のクラスタの数、衝突個数は上のクラスタに含まれる形態素の総数である。平均はこの二つの比である。

表1　畳み込みによる衝突

方法	クラスタ数	衝突個数	平均
A	11006	28813	2.62
B	4873	10837	2.22
C	3196	7154	2.24
D	46152	184872	4.01
E	13	26	2

3. 日本語形態素解析への応用

上の方法Eで作成した文字コード表と形態素辞書を使用し、文献[8]から得た8個のハッシュ関数を用いた複数ハッシュふるい分けによって、学術情報センターの学会報告データベースから得た日本語抄録データの形態素解析を試みた。ハッシュ表の大きさは約1MBである。ICOTの原辞書中にある語の接続情報は今のところ無視し、辞書にマッチした形態素のすべての可能な組合せを機械的に出している。結果の例を図1に示す。

現在のところ、専門語の多くが未知語になるが、スペルチェックなどの用途にはほぼ充分である。学術用語辞書を増強することにより、上のデータの実用的な解析が可能と思われる。

4. おわりに

電子出版の普及により、大量のテキストデータの処理が必要となっている。複数ハッシュふるい分けは、UNIXのスペルチェッカ[1]などにも使用されているが、形態素解析には適した方法といえる。また通常のハッシュ法と異なり、並列処理に適した性質を持つ。[3]

現在のところ形態素の可能な組合せを求めるアルゴリズムが並列化に適していないが、この点の工夫によって、高速の並列処理が可能になることが期待される。

参考文献

[1] McIlroy, M.D. Development of spelling list. *IEEE transactions on communications*, Vol.COM-30, No.1, p.91-99(1982)

[2] 渋谷政昭,山本毅雄. データ管理算法.東京,岩波書店, 1983(岩波情報科学11)

[3] Stanfill, C., Kahle, B. Parallel free-text search on the connection machine system. *Communications of ACM*, Vol.29, No.12, p.1229-1239(1986)

[4] Knuth,D.E .Sorting and searching. Addison-Wesley, 1973 (The Art of Computer Programming, Vol.3)

[5] 西原清一.ハッシングの技法と応用.情報処理, Vol.21, No.9, p.980-911(1980)

[6] 横山晶一,元吉文男,井佐原均.二次記憶上の大規模語彙を用いる自然言語処理システム.情報処理学会論文誌, Vol.29, No.6, p.570-580(1988)

[7] Stephen,K.P., Keith, W. M .Random number generators; Good ones are hard to find. *Communications of the ACM*, Vol.31, No.10, p.1192-1201(1988), 西村恕彦訳, *bit*, Vol.25, No.4, p.19-27, Vol.25,No.5, p.12-20(1993)

[8] L'Ecuyer, P. Efficient and portable combined random number generators. *Communications of the ACM*, Vol.31, No.6, p.742-749(1988)

(語族 語族) が (大きく 大き く 大きく 大き く)(異なる 異なる 異なる 異なる)(日本語 日本語 日本語)(から から)(英語 英語) へ の (翻訳 翻訳) は

原 (言語 言語) の 各 (単語 単語) の (訳語 訳語) を (並べ換える (並べ 並べ)(換える 換える))(だけ だけ)(では では)

図1　形態素解析の結果