協調サーチエンジンにおける近接検索

4X－01

上原　稔
東洋大学情報工学科

1. はじめに

インターネットにおけるWebページの増加に伴いAltaVista、Googleなどの集中型サーチエンジンでは新鮮な情報を検索するのが困難となってきた。そこで、我々は分散型アーキテクチャに基づく協調サーチエンジン(Cooperative Search Engine, CSE)[1]を開発した。CSEでは、各サーバに配置された局所的サーチエンジンをメタサーチエンジンで統合し、大域的サーチエンジンを構成する。

膨大なWebページから適切な文書を検索するには、ブーリアン検索以上に詳細な条件指定が必要である。Lycos[3]などでは豊富な近接演算をサポートすることで高度な検索要求に応えている。近接演算(proximity operation)とは語順や語間の距離を考慮した演算である。近接演算は絞り込みに有効であるが、インデックスのサイズが増大するため、小規模なサーチエンジンではサポートしていないことが多い。我々のCSEでは局所的サーチエンジンとしてnamazu[2]を利用可能であるが、namazuも近接演算はサポートしていない。そこで、我々はCSEのために近接演算機能を持つサーチエンジンを開発した。

本章の構成は以下の通りである。2章では近接演算について説明し、3章では我々の仕様を述べ、4章では評価について述べる。最後に結論を述べる。

Proximity Search in Cooperative Search Engine
Minoru Uehara
Dept. of Info. and Com. Sciences, Toyo Univ.

2. 近接演算

Lycos[3]には豊富な近接演算子が用意されている。

A adj B	AとBが1語以内
A near B	AとBが25語以内
A far B	AとBが25語以上

これら演算子の頭に“o”が付くと語順を考慮する。また、語間距離と無関係なものもある。

A before B	語順のAかつB

STARTS[4]では2語A、Bの関係として以下の近接演算子を提案している。

A prox[n,X] B

ここで、nはAB間の距離を表す自然数で、AとBの間の距離がn以下であることを示す。たま、Xは語順を考慮するか否かを示すブール値である。STARTSの演算子を用いるとfar以外のLycosの近接演算子を表現できる。

A adj B = A prox[1,F] B
A near B = A prox[25,F] B
A before B = A prox[∞,T] B

なお、STARTSに無限大の記法はないが、ここでは理解を容易にするため採用した。

STARTSでは、近接演算の被演算子をキーワードに限定している。そのため、A oadj (B_1 or B_2) oadj Cのようなパターンを表現することはできない。

3. CSEの近接演算

CSEでは、Lycosの近接演算をすべて表現するため以下のような演算子を採用した。

A prox[m,n,X] B

これは、ある2語AとBが最低m語、最大n語離れることを表し（m、nは非負数、m<n）、Xは語順の有無を表す。これを用いて他の演算子は以下のようにあらわせる。

A prox[n,X] B = A prox[1,n,X] B

A far B = A prox[25,∞,F] B

通常の検索におけるキーワード検索の結果は、文書IDとスコアの対の集合となるが、近接演算におけるキーワード検索では、文書IDに対応するのは単一のスコアではなく、語の出現位置とスコアの集合である。

したがって、A prox[m,n,X] Bは該当する各文書に対して以下の値を返す。

$\{(\min(P_a,P_b),f(S_a,S_b)) \mid P_b \text{ in } Z\}$

$Z=[P_a+m,P_a+n]$ if X=T,

$Z=[P_a-n,P_a-m] \cup [P_a+m,P_a+n]$ if X=F.

ここで、P_a, P_bはA, Bの位置、S_a, S_bはA, Bのスコア、fはスコア関数である。なお、m=∞のときは、A not Bを意味する。

A and B = A prox[1,∞,F] B

A not B = A prox[∞,∞,F] B

また、近接演算子のorと組み合わせてA oadj (B_1 or B_2) oadj Cのようなパターンも表現できる。今後、繰り返しが表現できれば正規表現として使うことができる。

4. 評価

検索時間は、文書数nと語数mに対してO(nm)であった（表1参照）。また、インデックスサイズを図1に示す。いずれも1000程度の文書数では問題とならない。CSEではインデックスを分割し、並列・分散検索できるため、それほど性能を落とさず、精度を向上させることができる。

表1. 検索時間

文書数*1	100	1000	10000
検索時間	0.17s	1.6s	17s
文中語数*2	10	100	1000
検索時間	0.26s	1.8s	16s

*1 文中語数は1000、*2 文書数は10000で固定

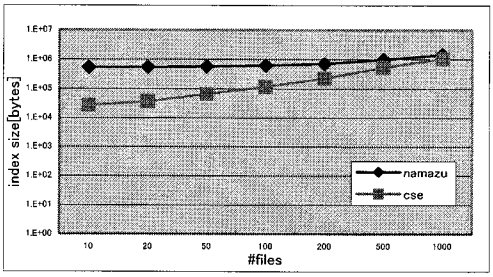

図1 インデックスサイズ

5. まとめ

既存近接演算のほとんどを表現可能な近接演算子を提案し、論理演算と自由に組み合わせることのできる近接演算を実現した。分散型サーチエンジンでは近接演算の問題点を回避できるため実用の可能性がある。

参考文献

[1] N. Sato, M. Uehara, Y. Sakai, H. Mori, "Fresh Information Retrieval in Cooperative Search Engine", In Proceedings of SNPD'01, pp.104-111, (2001.8.20)

[2] Namazu, http://www.namazu.org/

[3] Lycos, http://www.lycos.co.jp/search/help.html

[4] Luis Gravano, et al.: STARTS – Stanford Protocol Proposal for Internet Retrieval and Search –, Stanford University, 1997