2005－SE－147（3）
2005／3／17

ヒューリスティックを用いた 変更履歴に対応する討議スレッドの抽出法

山内 崇† **藤枝 和宏**† **落水 浩一郎**†‡

†北陸先端科学技術大学院大学 情報科学研究科

〒 923-1292 石川県能美市旭台 1-1

e-mail: {yamataka, fujieda, ochimizu}@jaist.ac.jp

‡国立情報学研究所

〒 101-8430 東京都千代田区一ツ橋 2-1-2

概要

本稿では、バージョン管理システム CVS と電子メールによるメーリングリストを用いたソフトウェア開発モデルを対象に、開発者の設計根拠を自動抽出する手法を提案する。検索手法としてベクトル空間モデルを用い、ヒューリスティックで検索対象を絞り込むことで精度の向上を図る。FreeBSD CURRENT の開発プロジェクトを対象に実験した結果、検索精度の向上を確認できた。

和文キーワード
オープンソース・ソフトウェア開発, CVS, 電子メール, ベクトル空間モデル, ヒューリスティック

Extraction of deliberation threads corresponding to a particular change of artifacts with heuristics on a commit log

Takashi YAMAUCHI† **Kazuhiro FUJIEDA**† **Koichiro OCHIMIZU**†‡

†School of Information Science, JAIST
Asahidai 1-1, Nomi-shi, Ishikawa 923-1292

‡National Institute of Informatics
Hitotsubashi 2-1-2, Chiyoda-ku, Tokyo 101-8430

Abstract

In this paper, we propose the technique of extracting design rationale for the development model which use CVS (Concurrent Versions System) and a mailing list. This system uses a vector space model to connect the CVS commit log to deliberation thread in a mailing list. We also use heuristics to improve the precision. We applied our method to the development project of FreeBSD CURRENT, and the effectiveness of our approach is proved experimentally.

英文 key words
open source software development, CVS, E-mail, vector space model, heuristics

1　はじめに

オープンソース・ソフトウェア開発では、成果物の管理にバージョン管理システム CVS が用いられる。また、開発者間のコミュニケーションにメーリングリストが用いられる。CVS リポジトリには成果物と共に、変更した理由や日付、変更した人がコミットログとして記録される。メーリによって行われる変更に関する討議はメーリングリスト・アーカイブに残る。メールの返信情報をもとに生成したメールスレッドから討議スレッドが取得できる。開発者は、CVS の変更履歴やメーリングリスト・アーカイブを閲覧することにより、過去の変更理由やその変更に至るまでの経緯を理解できる。

しかし、CVS リポジトリやメーリングリスト・アーカイブには大量の情報が蓄積されるため、開発者が必要な情報を探すのが困難になる。例えば、FreeBSD の開発者向けメーリングリストでは、一ヶ月に約 2500 通のメールが投稿される。そのため、開発者がすべてを読むのは容易ではなく、必要なメールを発見するのも困難である。

本研究では、開発者が変更履歴に対応する討議スレッドの検索を容易にするために、CVS リポジトリとメーリングリスト・アーカイブを用い、CVS リポジトリ中のコミットログからメーリングリスト・アーカイブの中の対応するメールスレッドを検索する手法を提案する。

佐々木ら [1] は。ソースコードに対応するメールを提示するために、ソースコードとメールに含まれるキーワードを用いている。

本稿は、以下のよう構成される。まず、ベクトル空間モデルとヒューリスティックの概略について 2 章で述べる。次に、3 章でベクトル空間モデルを用いた実験の評価をする。4 章では採用したヒューリスティックの説明と、それを用いた実験の評価をする。最後に、5 章でまとめと今後の課題について述べる。

2　提案手法

本研究で実現するシステムの概略を図 1 に示す。利用者がコミットログを指定すると、コミットログに対応するメールスレッド群を利用者に提示する。コミットログに対応するメールスレッドの検索手法

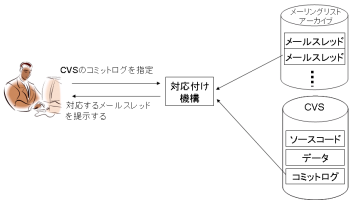

図 1: システムの概略

を図 2 に示す。まず、コミットログを入力しメール

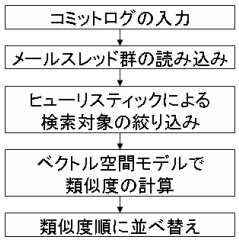

図 2: 検索手法

スレッド群を読み込む。次にヒューリスティックを用いて検索対象を絞り込み、コミットログとメールスレッド群の類似度を求める。最後に、類似度の降順にメールスレッド群を並び替える。

ヒューリスティックとベクトル空間モデルの役割を以下に示す。

- **ヒューリスティックによる検索対象の絞り込み**
  検索精度を向上するために、開発者が手作業で探し出すときの手がかりをヒューリスティックとして定義し、これを用いて検索対象を絞り込む。

- **ベクトル空間モデルによる類似検索**
  ベクトル空間モデル [2][3] は、検索に用いる文書をベクトルであらわすことにより文書間の類似検索を実現する情報検索技術のひとつである。本研究では、検索時の入力 (検索質問) にコミットログ、検索対象にメールスレッド群を用いる。

3　ベクトル空間モデル

ベクトル空間モデルは、文書をベクトルであらわすことで文書間の類似検索を実現する。ベクトル

空間モデルを用いた検索例を図 3 に示す。文書を

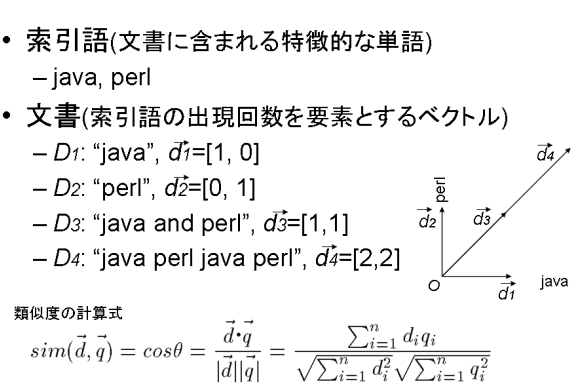

図 3: 提案手法

D_1, D_2, D_3, D_4 とし、これらの文書に含まれる単語から “java” と “perl” のふたつの単語を索引語 (3.2 節にて後述) として抜き出す。このとき、文書 D_i を索引語の出現回数を要素とするベクトル $\vec{d_i}$ で表すと、$\vec{d_1}, \vec{d_2}, \vec{d_3}, \vec{d_4}$ は右のベクトルとして図示できる。たとえば、D_1 と D_3 の文書間の類似度を求めるには、$\vec{d_1}$ と $\vec{d_3}$ のなす角 θ の $cos\theta$ を求めればよいので $\frac{1}{\sqrt{2}}$ となる。

3.1 検索処理の流れ

ベクトル空間モデルを用いた検索処理の流れを図 4 に示す。

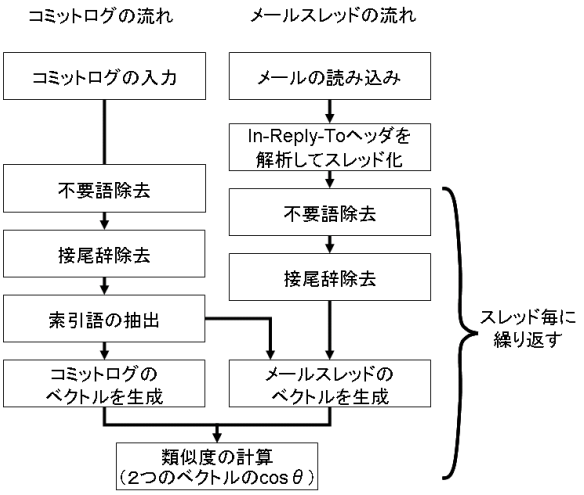

図 4: 検索処理の流れ

図 4 において、コミットログの流れの処理を行う。利用者によって入力されたコミットログから不要語と接尾辞を除去する。ここで残った単語を索引語にする。

メールスレッドの処理は以下のように行う。開発者向けメーリングリストのメールを読み込み、返信情報 (In-Reply-To ヘッダ) を解析してメールスレッドにする。各メールスレッドの本文から不要語と接尾辞を除去する。

コミットログから抽出した索引語を用いて、コミットログの索引語ベクトルと各メールスレッドの索引語ベクトルを求める。これらのベクトルに対し、類似度の計算式を適用するとコミットログと各メールスレッドの類似度が求まる。この類似度の降順に各メールスレッドを並べ替えることにより、コミットログに対応するメールスレッドの候補のリストが得られる。

3.2 索引語の抽出

ベクトル空間モデルを用いた検索では、検索質問から目的の文書を探し出すために、検索質問と検索対象の文書群に含まれる単語による比較を行う。しかし、文書に含まれるすべての単語を対象に比較を行うと、文書に含まれる “文書を特徴づけない単語” も比較対象になり、検索の精度が低下する問題が起こる。この問題を解決するために、“文書を特徴づけない単語” をあらかじめ除去する。この “文書を特徴づけない単語” を不要語という。たとえば、日本語の場合は助詞や助動詞など、英語の場合は冠詞や前置詞などは不要語である。不要語を除去して残った単語を索引語という。

索引語の抽出では、一般的に事前に決めた不要語のリストをもとに機械的に除去する方法が用いられる。理由は、手作業で索引語を抽出すると作業コストが高く、抽出される単語に偏りが生じる可能性があるからだ。本研究では、英語で書かれたメールを検索対象とし、不要語の除去に文献検索システム SMART で採用された 571 語の不要語 [4] を用いた。

また、英語は単語の語形変化により数、時制などをあらわす。そのため、単語の語形変化を考慮しないと、本来は同じ意味の単語がそれぞれ別の単語として扱われてしまう問題がある。この問題は、接尾辞と呼ばれる語形変化の部分を除去することで対処

できる。本研究では、ポーター・アルゴリズム [5] を用いて接尾辞の除去を行う。

不要語除去と接尾辞除去の例を図 5 に示す。元の

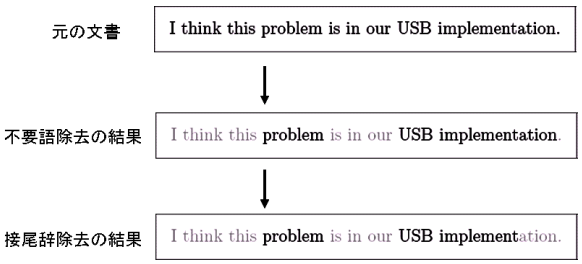

図 5: 不要語除去と接尾辞除去の例

文書から、不要語除去を行うと “I”, “think”, “this”, “is”, “in”, “our” といった単語が除去される。さらに、接尾辞除去を行うと “implementation” の語形変化部分が除去され “implement” になる。

3.3 ベクトル空間モデルを用いた実験の評価

図 4 で示した検索手法を、実際の開発プロジェクトに用いて実験し評価した。

対象は、FreeBSD CURRENT の 2003 年 12 月から 2004 年 11 月まで 1 年間の、CVS リポジトリのコミットログ (コミット数: 21535) とメーリングリストへ投稿されたメール (メール総数 28273 通、メールスレッド数は 8480 スレッド) である。

実験サンプルは以下の手順で選び出された。

1. 実験対象から無作為にコミットログを選ぶ
2. 対応するコミットが明らかなメールスレッドを手作業で探し出す
3. 対象のメールスレッドが見つからない場合は 1. に戻り見つかるまで繰り返す

この手順で 12 組の実験サンプルを選び出し、これを**正解の組み合わせ**として実験する。

実験結果の評価は以下の手順で行った。

1. コミットログと各メールスレッドの類似度を計算する
2. メールスレッドを類似度順に並べ替える
3. 正解のメールスレッドの順位と類似度の関係を調べる

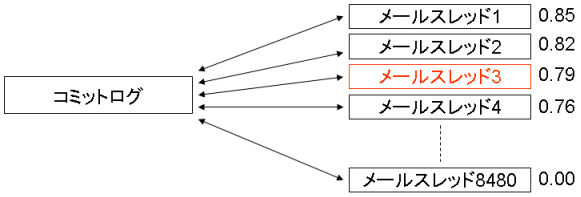

図 6: コミットログと各メールスレッドの類似度計算の例

図 6 は、コミットログと各メールスレッドの類似度を計算して並べ替えたときの例である。たとえば、メールスレッド 3 の順位は 3 位となる。

実験の結果、順位は表 1 のようになった。実験サンプルの 12 組は、**正解の組み合わせの ID** C1, C2 ... C12 であらわす。**索引語数**は、各コミットログに含まれる索引語数である。**順位**は、正解の組み合わせの順位をあらわす。**類似度**はコミットログと各メールスレッドの類似度である。

正解の組合わせの ID	索引語数	順位	類似度
C1	69	71	0.59
C2	35	845	0.52
C3	26	83	0.57
C4	40	1	0.84
C5	56	105	0.83
C6	20	2400	0.29
C7	126	252	0.59
C8	51	14	0.57
C9	31	1	0.86
C10	62	1916	0.29
C11	40	389	0.73
C12	67	1285	0.54

表 1: 正解の組み合わせと同じメールスレッドの順位

表 1 より、1 位から 2400 位まで順位の差が大きいことがわかる。順位が高いものに注目すると、C4 の順位は 1 位で類似度が 0.84、C9 の順位も 1 位で類似度が 0.86 である。順位が低いものでは、C6 の順位は 2400 位で類似度が 0.29、C10 の順位は 1916

位で類似度が 0.29 である。このように、順位と類似度には相関関係があることがわかる。しかし、C8 や C12 のように相関関係がないものもある。C8 は 0.57 で順位は 14 位であるが、C12 は類似度が 0.54 で順位は 1285 位である。よって、順位と類似度には必ずしも相関があるとはいえない。

4　ヒューリスティック

すべてのメールスレッドを検索対象として類似度を計算すると、本来対応するメールスレッドに関係のないメールスレッドに埋もれてしまい、順位が下がる問題がある。原因は、コミットログに関係がない同じような文脈のメールスレッドが存在するからである。

手作業で探し出すときは、経験に基づくの手がかりを用いる。例えば、あるコミットログに対応する討議が行われたメールスレッドは、コミットの日付に近いはずだと考え、たとえばコミットの日付と同じ月に範囲を絞って探す。もし、見つからなければさらに範囲を広げる。この手作業で探し出すときの手がかりを検索処理に利用することを考えた。

ベクトル空間モデルを用いた実験では、手作業によって 12 組の実験サンプルを選び出した。このサンプルを選び出す過程で用いた手がかりを以下に示す。

- コミットの日付に近いメール
- コミットした人の名前
- 内容を特徴づける単語

本研究では、手作業で探し出すときの手がかりをモデル化し、ヒューリスティックと定義する。ヒューリスティックを用いて検索対象を絞り込むことにより、コミットログに関係がない同じような文脈のメールスレッドを省き、正解を発見する可能性を向上させる。

4.1　採用したヒューリスティック

採用したヒューリスティックを以下に示す。

1. コミットした人で絞り込む
    - コミットした人が投稿しているメールスレッドのみを残す
2. コミットの日付で絞り込む
    - コミットの日付から前後 n 日以内に投稿されたメールスレッドのみを残す

1. は、コミットログに対応するメールスレッドには、コミットした人が参加していると考え、2. は、コミットログに対応するメールスレッドへの最後の投稿と、コミットの日付は近いと考え採用した。

ここで、2. のヒューリスティックにおいて、C1 から C12 までの 12 組のサンプルについて “前後 n 日以内” の n を決定するために調べた結果を図 7 に示す。　この図は、コミットの日付とメールスレッド

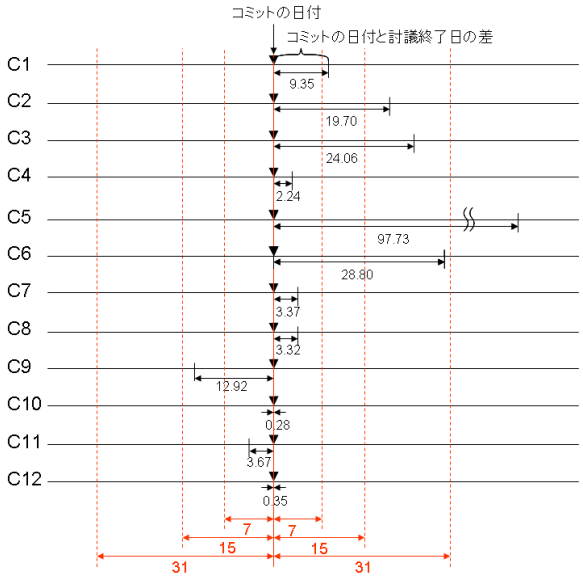

図 7: コミットログと各メールスレッドの類似度計算の例

の終わりの相対関係をあらわしている。横軸は時間軸で、時間の進行方向は右である。コミットの日付は、▼ (黒い下向きの三角) であらわす。|⟷|(2 本の縦線の間に双方向の矢印) は、コミットの日付とメールスレッドの最後の投稿の日付との差をあらわす。矢印の下の数値は差の長さ（単位は日）をあらわし、C1 の例の 9.35 は “9 日と 8 時間 24 分” を意味する。また、C1 はコミットの日付の右に矢印が伸びているが、これはコミットした後でそのコミットに対する議論が行われたことを意味し、C9 はコミットの日付の左に矢印が伸びているが、これは議論が行われた後でコミットしたことを意味する。

正解の組み合わせの ID	ヒューリスティック不適用		コミットした人の名前		コミットの日付					
					前後 31 日		前後 15 日		前後 7 日	
	順位	検索対象のスレッド数	順位	検索対象のスレッド数	順位	検索対象のスレッド数	順位	検索対象のスレッド数	順位	検索対象のスレッド数
C1	71	8480	0	429	15	1486	10	807	0	209
C2	845	8480	0	141	190	1543	0	770	0	388
C3	83	8480	0	88	16	1520	0	739	0	353
C4	1	8480	0	29	1	1052	1	525	1	227
C5	105	8480	0	379	0	1405	0	585	0	326
C6	2400	8480	0	141	440	1343	0	638	0	298
C7	252	8480	49	365	37	1406	19	760	11	354
C8	14	8480	4	53	2	1505	1	713	1	310
C9	1	8480	1	18	1	1596	1	706	0	321
C10	1916	8480	113	168	375	1731	204	882	93	419
C11	389	8480	0	189	73	1785	41	905	21	429
C12	1285	8480	11	53	200	1743	87	878	36	416

表 2: ヒューリスティック不適用と適用の場合の、順位と検索対象のスレッド数

以上を踏まえて図 7 を見ると、コミットの日付とメールスレッドの最後の投稿との差 (|⟷| の下の数値) が C5 以外はすべて前後 31 日以内であることがわかる。さらに、前後 15 日以内に絞り込むと 4 箇所が範囲から外れ、前後 7 日以内に絞り込むと半数が範囲から外れる。

このように、コミットした人の名前やコミットの日付を用いて検索対象を絞り込むことにより、正解を発見する可能性が向上する。

4.2 ヒューリスティックを用いた実験の評価

実験対象はベクトル空間モデルの実験と同じである。実験の評価方法は、ヒューリスティックを用いて検索対象のメールスレッド群を絞り込んだ後、ベクトル空間モデルによる類似検索を行い順位を調べる。

表 2 はヒューリスティックを用いた実験の結果である。実験サンプルの 12 組は、**正解の組み合わせの ID** C1, C2 ... C12 であらわす。ヒューリスティック不適用の列は、一年間すべてのメールスレッドを対象にしたときの順位と、そのときの検索対象のスレッド数をあらわす。**コミットした人の名前**の列は、コミットした人が討議に参加しているメールスレッドのみを残すというヒューリスティックを適用したときの順位と、そのときの検索対象のスレッド数をあらわす。**コミットの日付**の列は、コミットの日付から前後 n 日以内 (n は 31, 15, 7 に変えて試行) に投稿されたメールスレッドを残すというヒューリスティックを適用したときの順位と、そのときの検索対象のスレッド数をあらわす。

ヒューリスティックを用いることで検索対象が絞り込まれているか確認する。ヒューリスティック不適用の検索対象のスレッド数を見ると、一年間に投稿されたメールスレッドすべて (8480 スレッド) が検索対象になることがわかる。これに対し、コミットした人の名前による絞り込みを用いた場合は、検索対象の数が減っている。たとえば、コミットの日付による絞り込みを用いた場合も、前後 n 日以内の n が 31, 15, 7 と小さくなるに伴い検索対象のスレッド数が減っている。以上から、これらのヒューリスティックを用いることにより検索対象が絞り込まれていることがわかる。

次に、検索対象を絞り込むことで、類似度検索による順位の向上を確認する。コミットした人の名前で検索対象を絞り込んだ場合の順位を見ると、12 組のうち 5 組の順位が上がった。一方、7 組は 0 となっている。これは、コミットした名前で絞り込んだ結

果、正解の組み合わせのメールスレッドが検索対象から外れてしまったことを意味する。ベクトル空間モデルを用いた実験で類似度と順位の相関がなかった C8 と C12 の例では、C8 は 14 位から 4 位に上がり、C12 は 1285 位から 11 位に上がった。

コミットの日付で検索対象を絞り込んだ場合の順位を見ると、前後 31 日では 12 組のうち 11 組、前後 15 日では 7 組、前後 7 日では 6 組の順位が上がった。一方、絞り込みの日数 n が小さくなるに伴い、順位が 0 になる割合が増えて、正解の組み合わせのメールスレッドが検索対象から外れる傾向がある。C8 と C12 の例に注目すると、C8 は 14 位から前後 31 日では 14 位から 2 位に、前後 15 日では 1 位に、前後 7 日では 1 位に上がっている。C12 は 1285 位から前後 31 日では 200 位に、前後 15 日では 87 位に、前後 7 日では 36 位に上がった。

C8 や C12 の順位が向上したことをグラフで確認する。C8 の順位と類似度の関係を図 8 に示す。ヒューリスティックを用いない場合のグラフと比べ

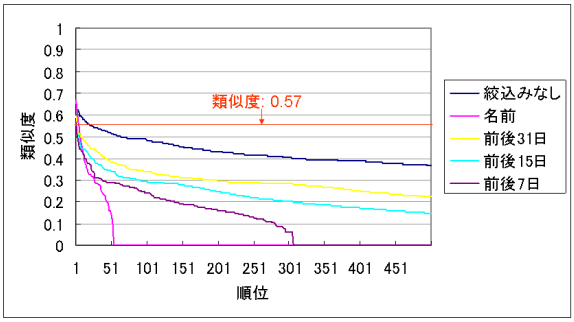

図 8: C8(類似度: 0.57, 14 位) の例

て、コミットした人の名前やコミットの日付で絞り込んだ場合のグラフの傾きが急になっている。類似度 0.57 のときのグラフとの交点を見ると、これらのヒューリスティックを用いることにより順位が上がっていることがわかる。

C12 の順位と類似度の関係を図 9 に示す。C8 と同様に、ヒューリスティックを用いない場合のグラフと比べて、コミットした人の名前やコミットの日付で絞り込んだ場合のグラフの傾きが急になっており、類似度が 0.54 のときのグラフとの交点を見ると、順位が上がっていることがわかる。

以上の結果を考察すると、コミットした人の名前では、直接討議に参加していない人がコミットしている場合、正解を取りこぼす傾向がある。コミット

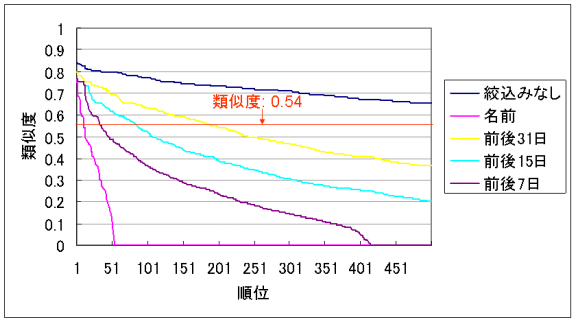

図 9: C12(類似度: 0.54, 1285 位) の例

の日付では、似たような語群が出現するメールスレッドが多い場合や、似たような討議が近い日付に複数ある場合は、正解を発見できる可能性が下がる傾向がある。この実験から、ヒューリスティックの有効性が認められた。

5 まとめ

本研究ではヒューリスティックとベクトル空間モデルによる検索手法を提案した。コミットした人の名前と、コミットの日付のヒューリスティックを用いて実験した結果、検索対象に入らない例もあるが、順位が上がることを確認できた。しかし、4.2 節で述べたようにヒューリスティックの使用には注意が必要である。

今後の課題は、今回提案したコミットした人の名前やコミットの日付以外に、検索対象を絞り込むことができるヒューリスティックを検討することがあげられる。

参考文献

[1] 佐々木 啓, 松下 誠, 井上 克郎. リビジョン情報と電子メールを用いたオープンソース開発向き情報検索システム, 電子情報通信学会技術研究報告 Vol.103, 2003.

[2] 長尾 真, 黒橋 禎夫, 佐藤 理史, 池原 悟, 中野 洋. 言語情報処理, 岩波書店, 1998.

[3] 北 研二, 津田 和彦, 獅々堀 正幹. 情報検索アルゴリズム, 共立出版, 2002.

[4] Salton, G. SMART system: stop word list.
ftp://ftp.cs.cornell.edu/pub/smart/english.stop

[5] Fotis Lazarinis. Porter stemming algorithm.
http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/