全文検索システム Akao の検索手法と性能評価

データ変換研究所　システム事業部
森本　哲也

発表の流れ

1. 検索システム市場の動向
2. 全文検索システムAkao(アカオ)
3. 検索アルゴリズムの紹介
4. 検索システムの検証　形態素解析 or N-gram
5. 測定結果の評価・考察　Namazu vs Akao
6. まとめと今後の展望

検索システム市場の動向

全文検索システムの利用状況とコスト

全文検索システムの利用状況

- 全文検索システム協議会　活動報告 (2002/10)
  [http://www.ftsanet.com/index.html]
  インターネット視聴率の上位サイトや「日本の定番サイト100」などのサイトを対象に全文検索システムの導入率を調査
  - ポータルプロバイダ　100%
  - 音楽　80%
  - メディア・ビジネス　67%
  - 学術教育・行政　60%
  - コンピュータ　55%
  - 旅行・タウン情報　55%
  - 生活ショッピング　40%
  - エンターテイメント　38%
  - 趣味スポーツ　23%

調査対象サイト　：96
全文検索導入率：avg. 56%

⇩

全文検索システムの需要が高まりつつある

イントラネット上の検索システムの必要性

- Nielsen Norman Group レポート(2002/11/7)
  [http://www.nngroup.com/]
  - ユーザビリティを専門とするコンサルティング会社
  - 使い勝手の良いイントラネットとそうでないものに対して生産性コストを調査

良いイントラネット	(年間)作業時間：27時間
悪いイントラネット	196時間

  - 1万人規模の企業で年間約15億円の生産性損失
    - このうち検索(システム)の占める割合　：　43%
  - 100人規模の企業の検索システムに占める生産性損失を算出すると、15億 x 0.43 ÷ 100 = 約645万円

全文検索システム Akaoの概要

全文検索システムAkao(アカオ)

- Windows / Solaris / Linux などのマルチOS環境下のファイルサーバを一括管理できる全文検索システム

Akaoの主要機能

1. マルチOS分散ファイルサーバの統合
2. 多くのファイルフォーマットに対応
3. 高速なインデックス作成と検索処理
4. 充実した管理ツール(閲覧管理機能)

Akaoの機能 (1)

1. **マルチOS分散ファイルサーバの統合**
   Windows / Solaris / Linux など各種OSで構築されたファイルサーバに接続することができる

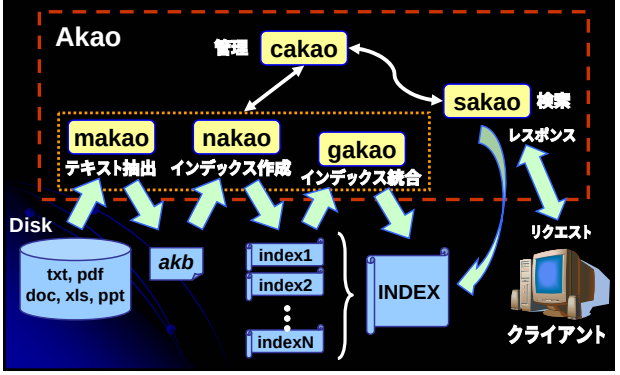
2. **多くのファイルフォーマットに対応**
   MS-Office文書、PDF、一太郎などの文書ファイルに加えて、AutoCADをはじめとした複数のCAD系ファイルにも対応

Akaoの機能 (2)

3. **高速なインデックス作成と検索処理**
   今回、評価を行った　後ほど詳細説明

4. **充実した管理ツール(閲覧管理機能)**
   全ての設定がWebブラウザ経由でGUI上で可能
   複数のファイルサーバやそのディレクトリに対して、ユーザ・グループ単位でアクセス制御が可能

Akaoのモジュール構成図



検索アルゴリズム

形態素解析とN-gram方式

検索アルゴリズム

Akao 開発時に検討した検索アルゴリズムは2つ、当時シェアをもっていた Namazu は形態素解析を採用

- 形態素解析　分かち書き：日本語を意味の通る単語に区切ること
  - あらかじめ単語登録してある辞書に基づいて文章の中から単語を抽出する手法
- N-gram
  - 単語を「N個の文字の組み合わせ」と捉えて、文字列をN文字ずつ区切っていく手法

形態素解析

Namazu の検索方式

- あらかじめ単語登録してある辞書に基づいて文章の中から単語を抽出する手法
  - メリット
    検索キーワードがヒットした場合、一定水準以上の精度の高い検索が行なえる
  - デメリット
    検索精度を維持するためには辞書のメンテナンスが必要
    例）「全文検索システム」のような複合語
    「インタフェース」「インタフェイス」の曖昧な外来語
    固有名詞や新語には対応できない

必ず**検索漏れ**が生じる

N-gram

Akao の検索方式

- 単語を「N個の文字の組み合わせ」と捉えて、文字列をN文字ずつ区切っていく手法
  - メリット
    辞書に依存せず、検索漏れは理論上ありえない
  - デメリット
    検索語によっては全く関係ない単語をヒットさせてしまう
    例）「検索システム」を2文字ずつ区切る
    → 「検索」「索シ」「シス」「ステ」「テム」
    検索語 “オア**シス**”、“アイ**テム**” でヒット

Akaoの実装

N-gramでは何文字で区切るかで3通りの方法がある

- 1-gram(uni-gram) ・・・ 1文字ずつ区切る
  例） 「検索システム」→「検」「索」「シ」「ス」「テ」「ム」
  何でもかんでもヒットしてしまう、ノイズ多
- 2-gram(bi-gram) ・・・ 2文字ずつ区切る
  → 「検索」「索シ」「シス」「ステ」「テム」
- 3-gram(tri-gram) ・・・ 3文字ずつ区切る
  → 「検索シ」「索シス」「システ」「ステム」
  実装が難しい、運用上も有効であるか不明

実装は工夫次第
検索精度良

検索アルゴリズムの評価

	形態素解析	N-gram
インデックス容量	小	大
インデックス作成速度	遅い	速い
検索速度	やや速い	やや速い
検索ヒット数	少	多
検索精度	漏れる	漏れない

本当にこのような結果になるのかを実験より定量的に評価する

インデックス作成と検索処理の評価

NamazuとAkao

検証環境

- マシン環境
  - CPU：Pentium II 300MHz
  - MEM : SDRAM 512MB
  - HD : Ultra-ATA / 33 (20GB)
  - OS : Redhat 8.0 (kernel 2.4.18)
- ソフトウェア環境
  - N-gram型の全文検索システム ： Akao 1.15
  - 形態素解析型の全文検索システム ： Namazu 2.0.12
  - Web サーバ : apache 2.0.40

インデックス作成の比較

- Akao, Namazu 双方で、大量のテキストファイル、PDFファイルをインデックス化し、インデックス容量と処理時間を比較する

（注） Namazuの環境設定

デフォルト状態より高速化

- 分かち書きツールに KAKASI を使用
- Text :: KAKASI の Perl モジュール追加
- ON_MEMORY_MAX → 50MB
- pdfフィルタとして xpdf の pdftotext を使用

インデックス作成の測定結果

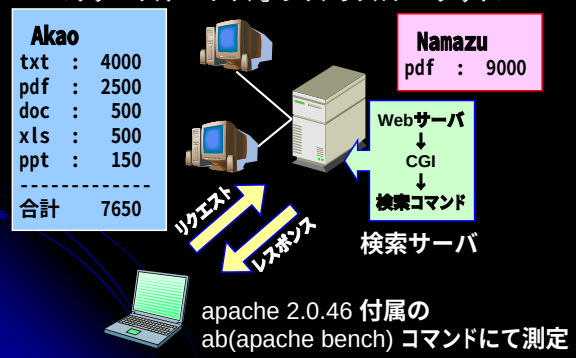
システム	ファイル数 / 形式	ファイル容量	インデックス容量	処理時間
Akao	1000 / txt	51 MB	21 MB	19分4秒
Namazu			18 MB	57分13秒
Akao	10000 / txt	505 MB	197 MB	55分22秒
Namazu			145 MB	11時間8分23秒
Akao	1000 / pdf	290 MB	9.4 MB	16分33秒
Namazu			8.9 MB	50分49秒
Akao	10000 / pdf	2.9 GB	71 MB	2時間1分55秒
Namazu			56 MB	11時間58分48秒

インデックス容量 ・・・ Namazuの方がやや小さい
インデックス作成速度 ・・・ Akaoの方が断トツに速い

インデックス作成の評価

- インデックス容量
  - Akao ： 全て2文字ずつ単語に区切る （大）
  - Namazu ： 2文字以上の単語や ストップワードが存在 （やや小さい）
- インデックス作成速度
  - Akao ： 単純処理 （かなり遅い）
  - Namazu ： 辞書検索、ソートなどのオーバヘッドが大 （遅い）

検索処理の比較
＜リクエストーレスポンスのスループット＞



apache 2.0.46 付属の
ab(apache bench) コマンドにて測定

検索処理<リクエストーレスポンス>の測定結果

システム	検索種別	コネクション	処理時間	Request / Sec
Akao	1文字検索	100	1131 秒	0.09
Namazu			10 秒	10.15
Akao	2文字検索	250	346 秒	0.72
Namazu			25 秒	9.79
Akao	3文字検索	250	339 秒	0.74
Namazu			28 秒	8.81
Akao	and 検索	250	364 秒	0.69
Namazu			26 秒	9.49
Akao	or 検索	250	598 秒	0.42
Namazu			31 秒	7.94

検索レスポンス ・・・ Namazuが圧倒的に速いね

検索処理<リクエストーレスポンス>の評価

- Akao
  - これはまずい

CGI → Perl

CGI実装の違いが結果に影響しているのでは？

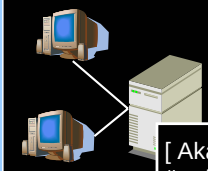
- Namazu
  - とても優れたパフォーマンス

CGI → C言語

検索コマンド単体での検証

Akao

txt	:	4000
pdf	:	2500
doc	:	500
xls	:	500
ppt	:	150
合計		7650



Namazu
pdf : 9000

[Akao]
 # sakao -d /ak/idx

[Namazu]
 # namazu -d /nmz/idx

ローカルでコマンドを100回実行し、
その処理時間を測定

検索コマンド単体の測定結果 (1)

コマンド	検索	ヒット数	実行回数	処理時間
sakao				1015 秒
sakao - n			0	899 秒
nama				6.9 秒
sakao	2文字検索		100	11.8 秒
sakao - no grep				9.9 秒
namazu				6.6 秒
sakao	3文字検索		100	7.9 秒
sakao - no grep				7.1 秒
namazu				7.1 秒

検索結果のヒット率の精度を
高めるための実装部分を削除
(検索そのものの処理とは無関係)

1文字検索を除けばコマンドレベルでは大差ない

検索コマンド単体の測定結果 (2)

コマンド	検索	(ヒット数)	実行回数	処理時間
sakao	and 検索	126	100	19 秒
sakao - no grep				14 秒
namazu		110		7 秒
sakao	or 検索	1641	100	183 秒
sakao - no grep				125 秒
namazu		1358		8.2 秒

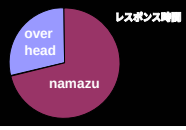
ヒット件数が増加すれば sakao のパフォーマンスは著しく低下

検索コマンド単体の評価

- レスポンス時間全体に対する検索コマンド単体の処理時間の割合を比較
  - Akao → 最大15倍以上
    - 日付指定検索
    - 検索対象サーバ選択
    - 閲覧管理機能

CGIの実装も含めたその他の処理の見直しが必要

  - Namazu → 最大 1.4倍程度



AkaoのC言語版CGIの測定結果

システム	検索種別	コネクション	Request / Sec	Rapid CGI Req / Sec
Akao	1文字検索	100	0.09	0.10
Namazu			10.15	
Akao	2文字検索	250	0.72	2.31
Namazu			9.79	
Akao	3文字検索	250	0.74	6.94
Namazu			8.81	
Akao	and 検索	250	0.69	2.11
Namazu			9.49	
Akao	or 検索	250	0.42	0.57
Namazu			7.94	

Perl → C : 大幅な性能向上が確認された

インデックス作成と検索処理の評価 (まとめ)

- インデックス作成
  - Akao が断トツに速いね

大多数ファイルのインデックス化には有効

- 検索レスポンス
  - Namazu の CGI は高速、コマンドレベルでもやや速い
  - Akao は 一部 Namazu に近い性能も出るが、
    検索ヒット数が増加すると性能低下

運用上のアプローチ
・検索ヒット数の上限を500程度に？

まとめと今後の展望

- まとめ
  - N-gram 型の Akao と形態素解析型の Namazu のインデックス作成と検索処理について定量的に評価した
  - Akaoの主要機能のうち、基本性能の部分を検証しました

Akaoの主要機能
1. マルチOS分散ファイルサーバの統合
2. 多くのファイルフォーマットに対応
3. 高速なインデックス作成と検索処理
4. 充実した管理ツール(閲覧管理機能)

- 今後の展望
  - CGI の高速化対応
  - Akao の有効性をユーザにアピール

    Namazu は優れた全文検索システム、

    だが、Akao には Akao のユニークな機能がある