長時間実行に耐えうるサーチエンジンシステムの構築

Development of Long-Term Executable Search Engine System

幸谷智紀*
Tomonori KOUYA*

Abstract: For 3 years, we have been continuing to develop a small-scale search engine which can automatically collect a large amount of Web data on the Internet. In this paper, we describe the 3 themes about our search engine which we have studied in this year:

- Confirmation of long-term executability
- Highly performed searching for collected Web datum by using "noun tables"
- Parallel performance evaluation of distributed Web robots.

1. 初めに

我々は3年間に渡り，自動的にWebデータを収集する小規模なサーチエンジンを作成し，その開発研究を進めてきた。本稿では本研究室で2008年度に行った次の3つのテーマについての報告を行う。

1. 長期運用能力の確認
2. 名詞テーブルを用いた検索処理速度の向上
3. Web Robotの分散処理能力の計測

現在では既にGoogleをはじめとする大規模な自動収集サーチエンジンが稼働し，商用的な成功も収めている。また，大規模サーチエンジンが収集したデータを部外者が，データ量の制限付きながらも自由に入手・加工できるようAPIを公開しているケースも見られる。このような状況下で，かつ，ネットワーク帯域もマシン性能もごく限定される大学の一研究室においてスクラッチからサーチエンジンシステムを構築する意義を疑問視する向きも多い。しかし我々はこのような状況下である現在であるからこそ，その意義があると考えている。本稿における3つの報告事項に絡めてその理由を以下に述べる。

日本をはじめとする先進諸国では既にThe Internetの普及率は飽和状態であるが，それ以外の地域ではまだまだ普及の途上にある。従って，The Internetにおける主要サービスの一つであるWebサーバは今後も増えていくと予想される。よって既に現在でも小規模なPC clusterではThe Internet上の全てのWebサーバ上のデータを捕捉することは不可能であり，将来においては更に不可能性が増すと予想される。従って，コンピュータ資源に制限のある環境下で，ほぼ無限大と等しいデータの海の中で溺れずに，しかもユーザが求めるデータを効率的に収集・管理・検索する技術がこれからは重要になる。我々は昨年度まで，MySQLを基盤としてその上に構築してきたWeb Robot(Webデータ自動収集スクリプト)と検索インターフェースを用い，この目的に少しでも近づけるための研究開発を本年度行った。

まず，無限大のWebデータの海の中でどのように既存のWeb Robotが動作するか，そして長期的にどの程度耐えられるかを検証した。その結果を第2節で述べる。次に，昨年度作成した名詞を対象として検索を行うインターフェースは，検索速度が遅く，また分散処理にも対応できるものではなかったので，本年はテーブル構造を検索用に変更し，高速化がどの程度図れるかをベンチマークテストで確認した。その概要とベンチマークテストの結果を第3節で述べる。最後に，大量のWebデータを収集するための高性能化を図るため，Web Robotの並列分散化がどの程度有効なのかを調査した結果を第4節で述べる。

なお，今回行った実験のうち，Web Robotを稼働させたコンピュータ環境は下記の通りである。

CPU Intel Core2Quad 6600 (4 Cores)
RAM 4GB DIMM
OS CentOS 5.2
RDBMS MySQL 5.0.45
Language Perl 5.8.8, PHP 5.1.6
ISP OCN + NTT West B Flets(Max. Bandwidth: 100Mbps(best effort))

2. 長期安定動作の確認

Web Robotの基本動作はごく単純である。既知のURLにアクセスし，本文データのリンク部分から新たなURLを収集する，というループを繰り返すだけである。しかし実際に動作させてみると高性能かつ頑健なWeb Robotを書くためには細かいノウハウが必要となる。本研究室の卒業研究において，Perl+RDBMSを基盤とするWeb Robotはいくつか実装してきたが，いずれも高性能性や頑健性において問題があった。そこで本年度はすべてのWebデータを竹口によるWeb Robot実装[1]を用いて収集することにした。まずこのWeb Robotのデータ収集アルゴリズムを簡単に述べる。

データベース上に，MainテーブルとReserveテーブルを用意する。Mainテーブルはユーザの検索要求がなされたときに，検索対象となるデータを保存しておく。具体的には

- URL
- タイトル
- 更新日付
- 本文 (HTML, XMLタグは排除したプレーンテキスト)
- リンクURL
- WebRank

2009年3月5日受理
*総合情報学部コンピュータシステム学科

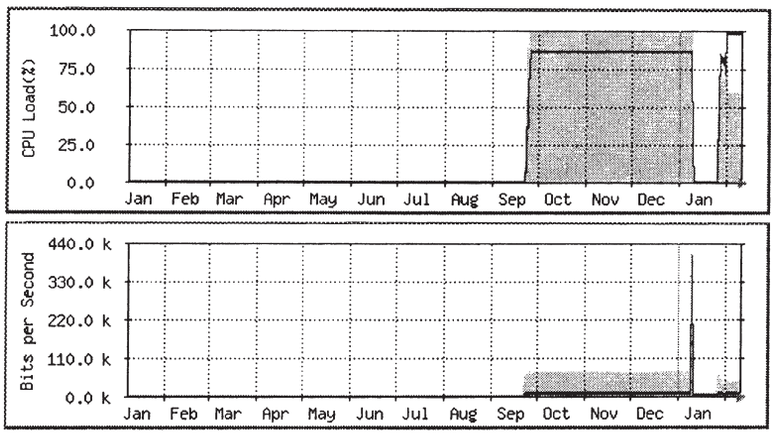

Fig. 1: CPU 負荷履歴 (上) と帯域使用履歴 (下)

が最低限必要となる。当然，接続可能かつ検索に意味のある URL の Web データだけが格納されていなければならない。しかし，Web データから新たな URL を抽出すると，大量の新規 URL が取得できることもあるが，そのうちアクセス可能なものはそれほど多くなく，データ収集が進むと既知の URL もたくさん見つかる。従って，新規の URL は一旦 Reserve テーブルに URL, リンク元 URL と，アクセスに必要な最低限度のデータのみ格納しておき，アクセス可能なことが判明したもののみ Main テーブルに追加するようにすると，データ収集能力の向上と，収集したデータの信頼性の両方を高めることができる。

このようなアルゴリズムによる Web Robot を 2008 年 9 月から 24 時間ノンストップで動作させてみた。最初の収集 URL は Yahoo! Japan のトップページ (`http://www.yahoo.co.jp/`) である。その過程を MRTG で記録した結果を Fig.1 に示す。上が CPU 負荷履歴，下が Ethernet の帯域使用履歴である。残念ながら 2009 年 1 月上旬に不慮の事故による停電が短時間に 2 回発生し，そこで Web Robot は一度停止し，その際に Main テーブルが破損した。従って，連続動作としては約 4 か月達成されたことになり，停電がなければもっと動作記録は伸ばせたものと思われる。この時，Main テーブルには約 60 万 URL, Reserve テーブルには約 2000 万 URL が記録されていた。

収集した URL のリンク関係を，収集回数ごとにドメイン単位でまとめた図 [3] を Fig.2 に示す。URL 数は約 54000, ドメイン数は約 2000 である。図の一番上が Yahoo! Japan であり，4 段階目で一気に URL 数が増えていることが分かる。

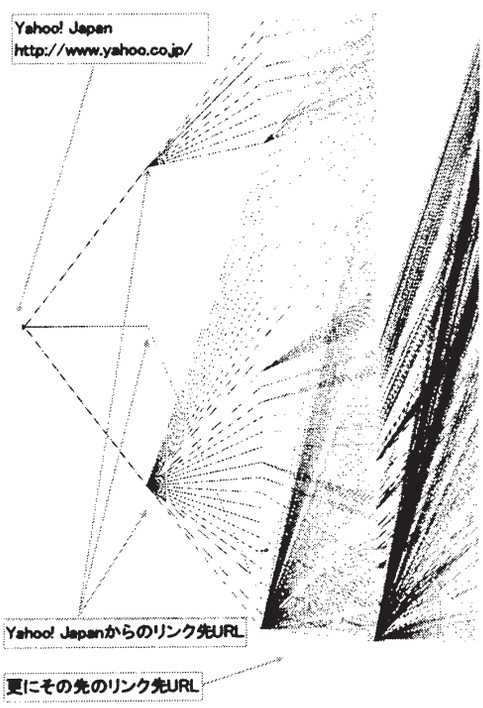

Fig. 2: リンク関係図

3. 名詞テーブルを用いた検索処理速度の向上

昨年度作成した検索インターフェイス [2] は，ユーザから入力されたキーワードを日本語形態素解析ソフトウェアを用いて分解し，その中の名詞のみを対象として検索を行うものである。本年度新たに作成したそのトップページを Fig.3 に示す。

検索対象となる Main テーブルのフィールドは，URL，タイトル，本文である。しかし，実際に検索を行ってみると，検索結果が表示されるまで分単位の時間を要することがしばしば見られた。我々の検索インターフェースは，ドメイン検索やタイトルのみの検索をオプションとして指定しない限り，本文データを検索対象とするが，大量の URL 件数になればなるほど，本文データを逐一取り出して複数の名詞を検索するのはかなりの時間を要する作業となりうる。

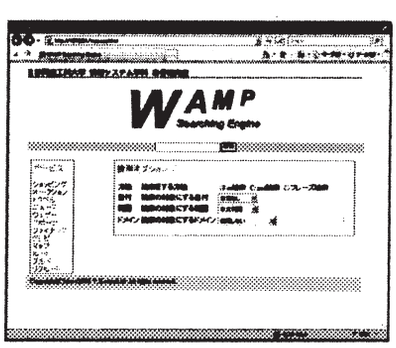

Fig. 3: 検索インターフェース

検索時間を短くする工夫としては，データベースのファイル形式を変更したり，ファイルを格納するディスクを高速にしたり，キャッシュを利かせるようにする等，I/O 性能を上げることが一番手っ取り早い方法である。しかし我々が目指すサーチエンジンシステムでは扱うデータ量が膨大になるので，I/O 性能を上げる工夫は重要ではあるが，それ以上に性能が向上できる仕組みを，もっと上位階層で行うことを考えなければならない。そこで，データベースを名詞単位で細かく分割し（これを名詞テーブルと呼ぶことにする），Main テーブルの情報を検索用に小分けにすることを考えた。つまり名詞ごとのインデックスをあらかじめ作成しておこうということである。このシステム概要図を Fig.4 に示す。。

Main テーブルそのものは改変せず，あくまで検索用に別データベースを作成し，そこに PHP スクリプトと日本語形態素解析ソフトウェアを使って名詞テーブルを作成する。これによって，検索要求があるごとに本文データを見る必要はなくなり，理論的には OR 検索，AND 検索は名詞テーブルごとの集合演算 (SQL の JOIN 命令) で実行できるはずである*。

これによる速度向上率を昨年度の検索スクリプトと同じヒット率になるキーワードで比較したところ，約 50 %の性能向上率が得られた。検索時間は半分程度になったものの，それ以上の向上を期待していた我々としては満足のいくレベルではない。この理由としては，膨大な数になる名詞テーブルそのものの検索に時間がかかっているせいではないかと推察される。よって，更なる性能向上を目指すためには，名詞テーブルそのものを分散・階層配置する必要があると思われる。

4. Web Robot の分散処理の実験

無限大の Web データを大量に取得するためには，なるべく IX(Internet Exchange) に近い幹線のような高速なネットワーク環境の下で，Web Robot を大量に分散配置し，並列に Web データを取得する仕組みが不可欠である。しかし，単純に Web Robot を並列に配置しても，取得した URL は重複するであろうし，同時に同じ Web サーバに複数アクセスを行ってネットワーク攻撃とみなされることも考えられる。従って，ある一定の秩序を保ち，どこかで集中的に分散配置した Web Robot をコントロールする機構が不可欠である。

そこで我々は，もっとも簡単な Web Robot コントロール技法として，Main/Reserve テーブルを配置する MySQL データベースサーバを一台だけとし，テーブルを共有させてドメイン単位に担当する Web Robot を配置するようなコントロール技法を使用し，どの程度並列動作の効用が得られるかどうかを検証してみた。その結果を Fig.5 に示す。

前述した Quad-core マシン上に Web Robot を 1 本もしくは 4 本同時に 24 時間連続動作させ，Yahoo! Japan を収集の起点として，どの程度収集 URL が増加するかを調査した。Fig.5 の左が Main テーブルの URL 増加数，右が Reserve テーブルの URL 増加数である。どちらも分散処理の効用が現れているとは言い難く，特に収集データ量の多い Main テーブルでは，4 本並列動作の方が，実行時間が長くなり URL 数が増加するにつれて増加量は鈍化していることが分かる。それに対して Reserve テーブルの方は，並列動作の効用はないとはいえ，増加量の劣化は起こっていない。

これは MySQL サーバが一台しかなく，そこに集中的に SQL query が寄せられることによって I/O 性能のボトルネックが発生しているためと思われる。従って，高性能な分散並列 Web Robot の開発には，分散データベースとの併用が不可欠という結論が導かれる。

5. 結論と今後の課題

以上述べてきたように，我々の開発してきた自動収集サーチエンジンシステムは長期的に実行可能であることが確認できたが，次のような問題を抱えていることも同時に明らかとなった。

1. Web Robot 実行中のデータベースのバックアップ体制がないため，突然の停電などのファイル破損に対して無防備である。
2. 検索インターフェースは昨年度に比較して半分程度の検索時間になったものの，実用的にはさらに短縮する必要がある。また，名詞テーブル作成時間が膨大になることも問題である。
3. Web Robot を分散させて並列実行させても，データベースサーバが単独ではそこがボトルネックになって収集性能は並列度を増やしても上がらない。

これらの問題の解決策を考案するのが今後の課題である。現時点としては次のようなことが考えられる。

5.1 収集 URL データのバックアップ体制

データベースのバックアップは，Main テーブル，Reserve テーブルそれぞれで行い，なるべくデータ量は小分けにして行い，データベースのバックアップと復旧が迅速に行えるようにする。特にデータ量の多い Main テーブルをどう小分けにするか，どの程度の頻度でバックアップ作業を行うかをよく考える必要がある。もちろん，バックアップ用のサーバを一台用意し，互いにバックアップファイルを持ち合うようにすることも考えねばならない。

5.2 検索インターフェースに関する問題の解決策

まず，現状の名詞テーブル配置のまま，I/O 性能を最大限発揮できるよう，既知のノウハウを習得してできうる限りの高速化を行う。その上で，収集した Web データ中に含まれる名詞（単語）の統計分布を調査し，それに基づいて適切な分散配置を実行できるようなアルゴリズムを考案する。将来的には，異なる性能を持つ PC 群に対して，個々の PC の性能に見合った名詞テーブルの配置が可能なような機能も，実用的には不可欠であろう。

5.3 分散並列 Web Robot の高性能化

名詞テーブルに関する問題解決同様，データベースサーバ単体での I/O 性能をできうる限り引き出すことをまず考えねばならない。その上で，既存の RDBMS ソフトウェアを，ライセンスの許可する範囲内で独自にクラスタ化し，その上で分散データベースを構築し，そこで最大限の能力が発揮できるような Web Robot の配置，並列動作方法，データの同期方法を考えねばならない。

この方法としては，全ての分散データベースの機能としてミドルウェア的に解決する方法と，個々の Web Robot がエージェントとして独自に考えつつ、あまり同期の手間を増やさないようにする方法が考えられる。どちらの方法にしても一長一短あるので，まずは Web Robot としてふさわしい方法を選択することにしたい。

---

*今回作成したものは PHP スクリプト内部で独自に OR，AND 処理を行っている。

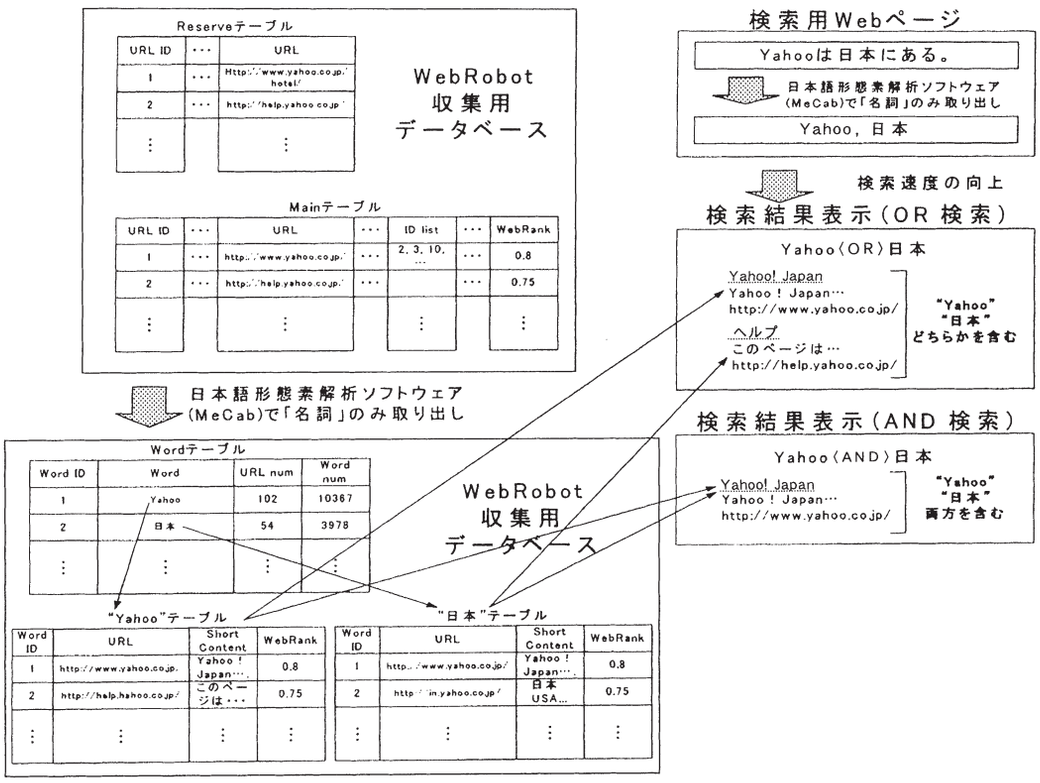

Fig. 4: 名詞テーブルと検索処理の概要

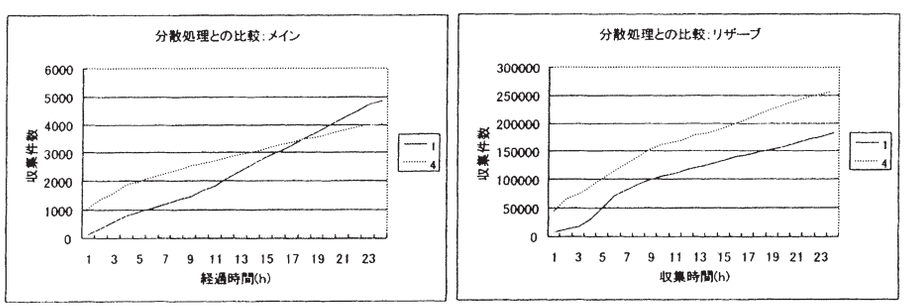

Fig. 5: 分散処理による効果の検証

謝辞

本研究は，平成20年度卒業研究として，仁藤昌彦，吉田嗣，黒田博雅によってなされたものである。教員の独断によって設定された研究テーマに一年間お付き合い頂いた彼らの労に対して感謝申し上げる。また本研究に使用したPC機材は静岡理工科大学学内研究費の補助で購入したものである。関係者各位に感謝する。

参考文献

1) 竹口友大・幸谷智紀, ランク機能付きサーチエンジンの開発およびI/Oボトルネック対策, 第70回情報処理学会全国大会講演集, 2008.
2) 山本達文, "PHP+MySQLと茶筌(Chasen)を用いたサーチエンジンの作成", 2007年度静岡理工科大学情報システム学科卒業研究.
3) 吉田嗣, "Webデータ収集構造を視覚化するためのプログラムの作成", 2008年度静岡理工科大学情報システム学科卒業研究.
4) 仁藤昌彦, "サーチエンジンの検索速度向上に関する研究", 2008年度静岡理工科大学情報システム学科卒業研究.
5) 黒田博雅, "並列分散型Webデータ収集システムの改良", 2008年度静岡理工科大学情報システム学科卒業研究.