Solr の検索性能

2007 年 9 月 29 日
株式会社 ロンウイット　　関口宏司

はじめに

本書は「Apache Solr（以下 Solr）」の検索性能の検証結果および考察を述べたものである。

Solr について

Solr は高性能の全文検索ライブラリ Apache Lucene(*1)を使って構築された、「検索エンジンサーバ」である。もとは CNET のために構築された検索エンジンフレームワークであるが、CNET の許諾により Apache Software Foundation にコードが寄贈され、現在は Lucene のサブプロジェクトとしてオープンソースで開発・管理がなされている。

(*1) Apache Lucene
http://lucene.apache.org/java/docs/index.html

Apache Solr
http://lucene.apache.org/solr/

Solr は全文検索機能を API で提供する Lucene よりも使い方が簡単なため、公開されると同時に利用事例が急速に拡大している。

Powered by Solr
http://wiki.apache.org/solr/PublicServers

弊社も顧客向けに Solr の導入コンサルティングを多数行っており、事例は国内だけでなく海外にも広がりつつある（諸事情により、弊社の顧客事例は公開できない）。

Solr の特徴を以下に列挙する。

- 商用利用可能な無料のオープンソースの検索エンジンサーバ。
- HTTP/XML による文書の索引付けと検索。検索結果は XML で返すため、フロントエンドアプリケーションは必ずしも Java で開発する必要がない。
- Lucene フィールドに「データ型」を導入。
- 検索を高速化するための各種キャッシュ機構の装備。
- キャッシュを最大限に利用した高速絞り込み検索。
- スケーラビリティと運用性を考慮したインデックスレプリケーション。
- 「類義語検索」や「ハイライター」などの各種ツール類の装備。
- 管理画面。
- 各種プラグイン開発インタフェース提供による高い拡張性と柔軟性。

検証に用いたデータ

性能検証には、「Yahoo!オークション」から約 201 万件の商品データをクロールしてインデックスに登録したものを使用した。商品データの収集は、「Yahoo!デベロッパーネットワーク」が提供している Web サービスを使って行った。

Yahoo!デベロッパーネットワーク - Yahoo!オークション
http://developer.yahoo.co.jp/auctions/

インデックスのファイルサイズは合計で約 14GB となった。

インデックスのスキーマは、下表のとおりである。

フィールド名	型	概要
auction_id	string	オークション ID

category_id	string	カテゴリ ID
category_path	text_ja	カテゴリパス
title	text_ja	商品タイトル
url	string	商品 URL
seller_id	string	出品者 ID
image1_url	string	画像 1 の URL
image2_url	string	画像 2 の URL
image3_url	string	画像 3 の URL
init_price	sint	開始価格
price	sint	現在価格
start_time	date	オークション開始日時
end_time	date	オークション終了日時
description	text_ja	商品説明
text	text_ja	(*2)
timestamp	date	インデックス登録日時

(*2) auction_id, category_id, category_path, title, seller_id および description のフィールドをコピーして合成したもの。

上記フィールドのうち auction_id をユニークキーとし、text フィールドをデフォルトの検索フィールドに指定した。

一部のフィールドの型について若干補足する。text_ja はインデックスに索引付けする日本語のテキストを示し、形態素解析により単語分割をしている。形態素解析器には Sen を使用した。

Sen Project
http://ultimania.org/sen/

sint 型は Sortable Integer であり、当該フィールドで検索結果一覧のソートを行ったり、範囲検索を行ったりする場合に使用する Int 型である。

検証環境

検証環境を下表に示す。

項目	説明	
ハードウ	機種名	DELL Inspiron 1501
	CPU	AMD Athlon(TM) 64 X2 デュアルコア・プロセッサ TK-53
	メモリ	1GB(512MBx2) デュアルチャネル DDR2-SDRAM メモリ
	ハードディスク	120GB SATA HDD (5400 回転)
	ネットワーク	100MB Ethernet
OS		Fedora Core 6 Kernel Rel. 2.6.20-1.2962.fc6
Java		build 1.5.0_12-b04
Java ヒープ設定		-Xms256m -Xmx256m
Servlet コンテナ		Tomcat 5.5.23
Solr		rev575809
負荷試験ツール		Apache JMeter 2.2

Solr を実行するサーバマシンには DELL 製のノート PC を使用した (本体価格：57,124 円)。ハードディスクは低速なため、次のようにページキャッシュとハードディスクの速度に大きな開きがある。

```
# hdparm -ftT /dev/sda

/dev/sda:
 Timing cached reads:   3208 MB in  2.00 seconds = 1604.93 MB/sec
 Timing buffered disk reads:  122 MB in  3.06 seconds =  39.93 MB/sec
```

そこで今回の検証では、OS のページキャッシュが十分に効いた状態で性能を測定することとした。

サーバに負荷をかけるクライアントマシンであるが、事前調査の段階でほとんどの場合においてネットワークがボトルネックとなることが判明したため、ノート PC 1 台で行った。クライアントからは JMeter で 200 同時ユーザ (スレッド) を起動して負荷をかけた。Ramp Up は 400 秒に設定した。

また、サーバ/クライアントとも、CPU を使い切ることはなかった。

Solr 検索性能サマリー

ユーザから見た Solr の検索性能（応答時間とスループット（QPS；1 秒あたりのクエリ数））のサマリーを以下に記す。

検索単語数	同時ユーザ数	平均応答時間（ms）	QPS
1	1	7	123.9
	10	46	187.3
	50	143	203.3
	100	262	210.2
	200	523	215.8
2	200	975	79.4

「検索単語数」が 1 のものについては、絞込みが十分に行われない場面を想定し、検索結果文書数が 1,001～10,000 件になる検索キーワードをインデックスから 1,024 件抽出して負荷をかけた。また、「検索単語数」が 2 のものについては、AND 検索によってある程度絞込みが行われた場面を想定し、検索結果文書数が 101～1,000 件になる検索キーワード 2 単語 1 組をインデックスから 1,024 組抽出して負荷をかけた。

単語数 1 では同時 200 ユーザで応答時間 0.5 秒、200QPS を達成した。単語数 2 では同時 200 ユーザで応答時間 1 秒、80QPS となった。

以上の結果を踏まえ、次節以降は、「単語数 1・同時 200 ユーザ」の基本モデルにさまざまな検索条件を付加して検証を行った結果を述べる（ただし、一部の検証では「同時 50 ユーザ」で行っており、その旨明記してある）。

ソートフィールドの数

検索結果一覧の文書のランキング（文書の表示順）は、通常、検索式と文書の適合度合いを数値化した「スコア」により決められる。しかし、アプリケーションによっては「スコア」よりも文書のフィールド値の大小によるソート機能が求められることがあり、Solr もこれをサポートしている。

ソートに使用するフィールドは 1 つ以上指定可能である。これには「スコア」を含めることができる。

ソートフィールドには以下を使用した。

項目名	ソートフィールド数	ソート対象フィールド
sort-0	1	score desc（デフォルト）(*3)
sort-1	1	category_id asc
sort-2	2	category_id asc, end_time asc
sort-3	3	category_id asc, end_time asc, score desc

(*3) asc は昇順、desc は降順を示す。

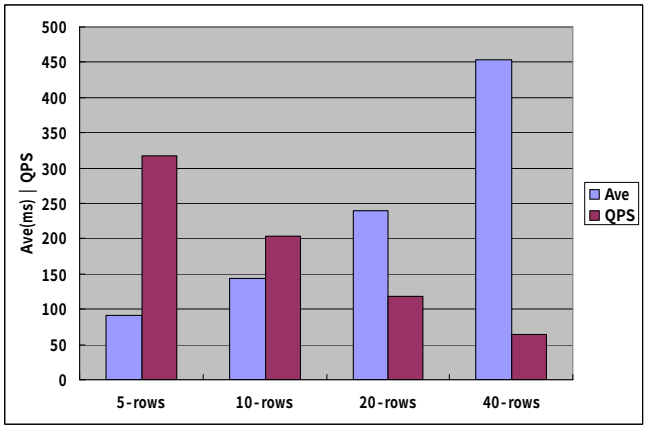
検証結果では、ソートフィールドの数の上昇による応答時間や QPS の性能劣化は見られなかった。ただし、サーバの CPU 平均使用率は 27%（sort-0）から 35%（sort-3）に上昇したため、CPU に余裕のない場合はソートフィールドの指定による性能劣化は起こりうると考えられる。

1 レスポンスあたりのデータ量

Solr では検索結果一覧を XML で返すが、XML のサイズが大きくなれば性能にも影響する。XML のサイズが変わる要因にはいくつか考えられるが、ここでは Solr の代表的な HTTP パラメータである「rows」、「fl」、「hl」を変更することで XML のサイズを変更することとした。

rows パラメータは、1 件の XML に含まれる検索結果の文書件数を指定する。デフォルトは 10 である。このパラメータを 5～40 で指定した結果を以下に示す。

rows	平均応答時間 (ms)	QPS	データ量 (KB/s)
5	91	317.9	722.68
10	143	203.3	834.98
20	240	118.5	922.82
40	453	64.3	977.18



rows が大きくなるにつれ、平均応答時間が延び、QPS が下がった。このことから、アプリケーションの要件によって適切な rows を設定することが重要であることがわかる（たとえば、携帯電話向けのアプリケーションでは rows を小さくするなど）。

なお、このテストでは rows=10 で早々にネットワークがボトルネックとなってしまうため、同時 50 ユーザで行った。

fl パラメータは、XML に含める文書のフィールド名を指定する。デフォルトは fl=* であり、「文書を構成するすべてのフィールド」を XML で返す意味となる。

fl パラメータは、以下のバリエーションでテストを行った。

項目名	fl
fl-1	auction_id, title
fl-2 (*4)	auction_id, bids, category_id, end_time, price, title, url
fl-3	auction_id, bids, category_id, end_time, image1_url, init_price, price, seller_id, start_time, title, url
fl-4	*, score (*5)

(*4) 基本モデルでは fl-2 の fl を使用している。

(*5) score は検索結果 XML にスコア値を含めることを指定する。

結果は次のとおりで、fl に指定するフィールドが増えると XML のサイズが大きくなるため、rows パラメータを変更したときと同じ傾向となる。

項目名	平均応答時間 (ms)	QPS	データ量 (KB/s)
fl-1	258	393.8	697.25
fl-2	523	215.8	886.26
fl-3	720	142.4	953.00
fl-4	1595	63.1	1018.92

fl=*,score を指定している fl-4 では平均応答時間が 1.6 秒にもなったが、これはオークション商品の詳細説明が記述された description フィールドが大きいためである（最大 8KB）。通常、検索結果一覧には商品の細かい説明は不要であり、この結果からもアプリケーションの要件によって適切な fl パラメータを指定することが重要であることがわかる。

hl パラメータには、ハイライト（検索語の強調表示と要約文の抽出）の使用有無を指定する。hl=true でハイライト機能を ON にし、hl.fl でハイライトするフィールド名を 1 つ以上指定する。今回のスキーマではハイライトに適したフィールドが title と description 以外にないので、hl.fl にはこの 2 つのフィールドを指定することとし、hl.snippets で要約文の断片数を 1,2,3 と変えて比較した。

結果は次のとおりで、断片数の数による性能の変化は見られない。

断片数	平均応答時間 (ms)	QPS	データ量 (KB/s)
1	1495	37.7	212.90
2	1567	37.9	219.43
3	1559	37.6	220.87

平均応答時間が 1.5 秒となったが、中央値は 100 ミリ秒であったことを注記しておく。また、CPU の使用率は hl=true は hl=false の 2 倍強となった。さらに GC 時間の占める割合を比べると、hl=true は hl=false の 8 倍にも達した。ハイライトでは Full GC は発生していないことから、短寿命のオブジェクトが多数生成されていることが考えられる。

Solr キャッシュ効果

Solr には検索結果リストをヒープに保存する「検索結果キャッシュ」、ロードした文書データをヒープに保存する「文書キャッシュ」、フィルタをヒープに保存する「フィルタキャッシュ」がある。

「検索結果キャッシュ」については、キャッシュの大きさを変えることでヒット率 52%と 98%、およびキャッシュを使用しない場合（0%）の性能を測定した。結果は次のとおりとなった。

ヒット率	平均応答時間（ms）	QPS	CPU 使用率（%）
0%	486	142.9	88.96
52%	484	206.5	78.11
98%	486	212.6	22.05

ヒット率が 0%から 52%に上がると、QPS がやや改善したが、52%から 98%ではほとんど改善しない。ただし、ヒット率が上がると CPU 使用率の低減に大きく貢献することがわかる。

「フィルタキャッシュ」についても、キャッシュの大きさを変えることでヒット率 54%と 99%、およびキャッシュを使用しない場合（0%）の性能を測定した。結果は次のとおりとなった。

ヒット率	平均応答時間（ms）	QPS	CPU 使用率（%）
0%	558	137.4	81.25
54%	637	156.0	72.88
99%	564	179.3	48.91

ヒット率 0～99%の間で QPS 性能は改善している。

「文書キャッシュ」についてはヒット率の違いで検索性能の変化は見られなかった。これは OS のページキャッシュが効いている状況下では「文書キャッシュ」の威力が発揮されないためである。Solr に装備されているフィールドの遅延ロード機能もこの場合性能改善には役立たない。ただし、高速ハードディスク・少メモリのマシンでは「文書キャッシュ」を使って Solr の検索性能を大きく改善できる可能性がある。

まとめ

単純な検索ながら、文書数 200 万件・ヒット件数 1,000 件超・同時 200 ユーザ接続という条件で 200QPS/0.5 秒のレスポンスを 6 万円のノート PC で達成できた。高速なディスクのサーバマシンの利用や、複数サーバを並べてスケールアップを図ることでさらなる QPS 向上が期待できる。

性能検証の残項目

今回は時間的・リソース的制約から、以下に列挙する項目については検証しなかった。

- N-gram による検索性能。
- リクエストハンドラ種別による検索性能の違い。
- JDK バージョンによる違い。
- 1,000 万件程度の文書量に対する検索性能（クロール時間とディスク容量の制約により今回は見送った）。
- インデックスへの文書の登録性能。

（株）ロンウイットについて

ロンウイットはオープンソースの全文検索エンジン Lucene/Solr を企業システムに導入する支援サービス事業を展開している。

お問い合わせ先

〒100-0005
東京都千代田区丸の内 1-1-3 AIG ビル B1F
電話：03-5288-5927　　FAX：03-5288-5353
メール：sales@rondhuit.com
ホームページ：http://www.rondhuit.com/