M-19 協調サーチエンジンにおける永続的キャッシュの実装

Implementation of Persistent Cache in Cooperative Search Engine

佐藤 永欣† 宇田川 稔† 上原 稔† 酒井 義文† 森 秀樹†
Nobuyoshi Sato Minoru Udagawa Minoru Uehara Yoshifumi Sakai Hideki Mori

1 はじめに

インターネットの情報検索では集中型サーチエンジンの使用が一般的であるが、更新間隔の極端な短縮は困難である。そこで、更新間隔を短縮するため、分散型アーキテクチャによる協調サーチエンジン (Cooperative Search Engine, CSE) [1] を開発した。

CSE は各 Web サイトの局所的サーチエンジンが索引付けを行う。また、各局所的サーチエンジンを統合し、一つの大域的サーチエンジンを構成する。このため CSE では検索時に通信による遅延が生じ、100 台以上の規模のネットワークに適用することが困難であった。CSE ではキャッシュを用い、様々な工夫をすることにより検索時の遅延を隠蔽している。しかし、CSE では更新時間が短いため、すぐにキャッシュが無効になってしまう。そこで、更新後にも有効なデータを保持する永続的キャッシュが必要とされる。本文では、CSE における永続的キャッシュの原理と実現について述べる。

2 協調サーチエンジン

CSE は以下のコンポーネントからなる（Fig.1 参照）。

- Location Server (LS) は各 Web サイトに含まれるキーワードの表を管理する。LS は Site selection Cache (SC) を持つ。
- Cache Server (CS) は検索結果をキャッシュし、先読みを行う。CS は Retrieval Cache (RC) と SC (LS の SC のコピー) を持つ。
- Local Meta Search Engine (LMSE) はユーザと対話し、LSE の違いを隠蔽する。
- Local Search Engine (LSE) は検索、インデックス作成を行う。

CSE は以下のようにインデックスを更新する。

1. LSE の Gatherer が対象サイトの文書を収集する。
2. LSE の Indexer は Gatherer が収集した文書のインデックスを作成する。この時、並列処理を行う。
   (a) $LMSE_i$ の Engine I/F は LSE_i からキーワード、スコア情報を抽出して LS に送信する。
   (b) LS は送信された情報を検索用に記録する。

以下では検索がどのように行われるかを述べる。

†東洋大学情報工学科, Department of Information and Computer Sciecces, Toyo University

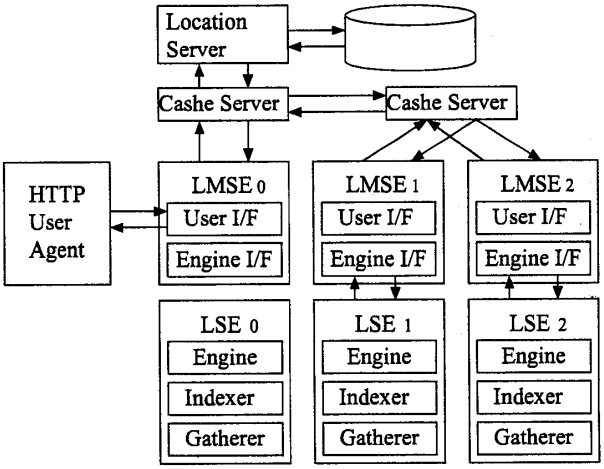

Fig. 1. CSE の構成と概要

1. $LMSE_0$ はユーザーからクエリーを受け取り、CS に検索を依頼する。
2. CS は LS にクエリーを送り、どの LMSE がクエリーに適合する文書を持っているか尋ねる。
3. CS は LS がクエリーに適合する文書を持つと回答した LMSE にクエリーを送る。
4. 各 LMSE は LSE を用いてクエリーに適合する文書を検索し、CS に返答する
5. CS は検索結果をまとめて $LMSE_0$ に返す。
6. $LMSE_0$ は検索結果を整形してユーザーのブラウザに表示する。

CSE の検索時は多くの通信を伴うため、通信量を削減することが検索の高速化につながる。CSE では以下のような方法を用いて通信量を削減している。

Query based Site Selection (QbSS) CSE の論理型検索では AND、OR、2 項 NOT（差分）をサポートしている。これを利用して検索対象サイトを絞りこむ。

継続検索における先読みキャッシュ 検索時の遅延を最小化するため、CS は「次の 10 件」の検索結果の先読みを行う。

Score based Site Selection (SbSS) 継続検索では次回の検索で得られる各サイトの文書のスコアを前回の検索時に CS が知ることができる。このため、順位の低い文書しか持たない LMSE への検索要求を抑制できる。

3 永続的キャッシュ

通常、更新終了後にはキャッシュと実際の検索結果の一貫性が失われるため、キャッシュをクリアする必要がある。永続的キャッシュでは更新時に予備的な検索を行いキャッシュしてキャッシュ無効化を回避する。永続的キャッシュの実現にはユーザーが使用した検索式を記録し、更新時に再検索する必要があるが、各サイトでの最高スコアがわかるため常にSbSSが使用可能である。

以下に永続的キャッシュの更新時の動作を述べる。

1. LMSEは通常の更新作業を行う。
2. LMSEはLSからクエリーリスト等を受け取り、予備的な検索を行い、最高スコアをLSに送信する。LSは受け取ったスコアでSCを更新する。
3. LSはCSにキャッシュの無効を通知する。
4. 各CSはSCとRCをクリアする。

このように予備的検索によるネットワークへの負荷は小さい。また、遅くとも更新終了後にはサイト選択の結果を知ることが可能である。よって、既に使用されたクエリの検索時のスケーラビリティが大幅に向上する。一方、永続的キャッシュでは更新時にLSから受け取ったクエリーを検索する必要があり、検索時間が長いと更新間隔を短縮できない。そこで、一度に全てのクエリーを一括して検索する。

4 永続的キャッシュの評価

更新前と更新後の検索時間を比較するため、通常のキャッシュと永続的キャッシュを用いた場合の検索時間を比較した。QbSS、SbSSでのサイト絞り込はできない状況である。Table.1に結果を示す。更新前にはSCはヒットしないが、更新終了後には必ずヒットするため永続的キャッシュでは検索が早くなる。ただし、1ページ目の検索のRCはヒットしない。

Fig.2に通常キャッシュ、永続的キャッシュの検索時間を示す。サイト数が50に増えると、通常キャッシュでは検索に50秒以上かかる。永続的キャッシュではサイト数が20、50でも10サイトの時と同じ時間で、ほぼ定数時間とみなしてよいため検索時のスケーラビリティは大幅に向上する。

我々は論文[1]でCSEの更新時間は通常数分以内であることを示した。よって、短い更新時間を維持するためには短時間で予備的検索を終る必要がある/ Table.2に更新終了後の予備的検索の実行時間を示す。この結果、10000種類のクエリーの予備的検索は一括して検索する場合、42秒程度で終了し、CSEの更新時間と較べても十分短い。予備的検索にはPentium III 700MHz、メモリ192MB、FreeBSD 4.5-RELEASEのPCを用いた。この結果から、予備的検索はCSEの短い更新間隔に大きな悪影響を及ぼさないと考えられる。

Table 1. 通常キャッシュと永続的キャッシュの検索時間

	更新前	更新後
通常キャッシュ	2.32 [sec]	2.17 [sec]
永続的キャッシュ	2.23 [sec]	1.23 [sec]

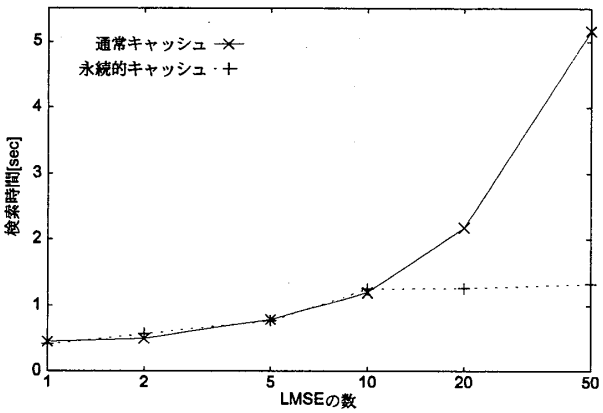

Fig. 2. 永続的キャッシュのスケーラビリティ

Table 2. 永続的キャッシュの予備的検索の所要時間

クエリー数	一括検索	1クエリーずつ検索
10	0:00.23	0:01.99
100	0:00.61	0:19:97
1000	0:04.45	3:19.40
10000	0:42.20	33:11.39

東洋大学のWWWプロキシサーバの2000年4月から11月までのログを調査したところ、12681種類のクエリーが使われていた。この調査結果は直ちに一般化できないが、クエリーが約8000個の時、新規クエリーの割合は25%程度であった。また、約160000個の時には8%程度であった。したがって、クエリーの種類は10000種類程度で飽和すると考えられる。

5 まとめ

本論文ではCSEのための永続的キャッシュの実装について述べた。永続的キャッシュは一度検索された検索結果を半永久的に保持すし、検索時の性能とスケーラビリティが向上する。特に、スケーラビリティは検索所要時間がほぼ定数時間になるなど大幅に向上する。

謝辞

また、本研究の一部は文部科学省科学研究費（課題番号14780242）の支援を受けて行われた。

参考文献

[1] 佐藤永欣, 上原稔, 酒井義文, 森秀樹. 最新情報の検索のための分散型サーチエンジン. 情報処理学会論文誌, Vol. 43, No. 2, pp. 321–331, 2002.