

LDAP ディレクトリ・サーバにおけるキャッシング機構の最適化

菊地 聡[†], 平島 陽子[†], 松岡 祐介[‡], 小瀧 伯泰[¶]

(株)日立製作所

[†]システム開発研究所, [‡]半導体事業本部, [¶]ソフトウェア事業部

分散システムの各種資源を一元的に管理する基盤技術として注目されるディレクトリ・サービスは, 更なる高性能化が要求されている. 本研究では, LDAP ディレクトリ・サーバにおける高性能キャッシング技術として, 検索種別毎の LRU 管理により安定した応答性能を確保する Partitioned LRU 方式, アクセス属性に限定したキャッシュ保持により応答性能を向上する Fractional Cache 方式を開発・評価し, 目標性能を達成した.

High Performance Caching Techniques for LDAP Directory Servers

Satoshi Kikuchi[†], Youko Hirashima[†], Yuusuke Matsuoka[‡], Michiyasu Odaki[¶]

[†]Systems Development Laboratory, [‡]Semiconductor & Integrated Circuits Group, [¶]Software Division
Hitachi, Ltd.

With the spread of large scale distributed environments, high-performance becomes very important feature for a directory service which manages the enormous number of resources in various systems as a unified repository. We propose a Partitioned LRU and a Fractional Cache as high-performance caching techniques for LDAP directory servers. The Partitioned LRU can steady the performance of response by dividing a LRU into search categories, and the Fractional Cache can enhance the performance by keeping only parts of attributes to be requested by clients in the cache. By these techniques, we could achieve our performance goal.

1. はじめに

インターネットの浸透, ネットワーク・セキュリティ技術の進歩に伴い, 各企業内の情報システムにインターネット技術を活用したイントラネットが定着しつつある. 更に今日では, 市場動向, 環境変化に即応可能な仮想企業の構築に向け, 各企業のイントラネット同士を接続し, 企業間の情報共有, EC (Electronic Commerce) を実現するエクストラネットも脚光を浴び始めている. このようにインターネット技術は, 組織内の情報流通やコミュニケーションの活性化に留まらず, 組織の利益に直結するミッション・クリティカルな基幹システムへの適用が著しい.

このような情報システムの大規模分散化に伴い, システムの各種資源を一元管理する基

盤技術として, CCITT (International Telegraph and Telephone Consultative Committee) の X.500^{1),2)}に代表されるディレクトリ・サービスが注目されている. X.500 の特徴は, 各種資源の階層管理, 即ち階層型ネーミング・モデルであり, 大規模分散システムにおける各種資源をユニークに識別可能である. ところがユーザは, OSI (Open Systems Interconnection) の7レイヤ構造による処理負荷の重さから X.500 を敬遠しがちであった. この課題を解決する為, US ミシガン大学はクライアントから X.500 準拠のディレクトリ・サーバを簡易にアクセス可能なプロトコル LDAP (Lightweight Directory Access Protocol) を開発した³⁾. LDAP は IETF (Internet Engineering Task Force) が規定したインターネット標準のディレクトリ・アクセス・プロトコル⁴⁾であり, 多数のベンダ

やユーザの支持を受け、多種多様なシステムへの適用が試みられている。また最近では、X.500を必要としないスタンドアロンのLDAPサーバ⁶⁾も市場に出荷され始めている。

筆者らは、LDAP ディレクトリ・システムに関する各種実装技術、運用管理手法等の研究開発^{6),7)}を推進中である。本稿では、始めにLDAPサーバにおける従来のキャッシング方式を説明し、次に従来方式に潜在する問題、それを解決する本研究のキャッシング方式及び性能評価について報告する。

2. LRU アルゴリズム

高性能なディレクトリ検索を提供する為、US ミシガン大学が開発したLDAPサーバはインデックス及びキャッシュ機構を備える。

LDAP 検索要求の受信時、サーバは、先にインデックスを用いて検索条件に合致するエントリを選別し、その後、各エントリの内容をDBから逐次読み出す。この際、抽出された全情報はキャッシュとしてメモリ上に格納される。同一エントリへの再アクセス時は、キャッシュに保持した情報を用いて応答する事により、DBアクセス回数(ファイルI/O発行回数)を削減し、応答性能の向上を図っている。

一般にキャッシングは、過去にアクセスされたデータを既定量のメモリ領域に記憶する。新たなデータの追加によりキャッシュ溢れが生じる場合は、キャッシュ内のあるデータを新規データに置き換える。キャッシュ置

換ポリシーは、再アクセスされる可能性の低さを基準に置換すべきデータを決定する手法であり、最も過去にアクセスされたデータを置換対象とするLRU (Least Recently Used) アルゴリズムは、ファイル・システム、Webプロキシ等、様々なシステムに採用されている。

図1に、LRU アルゴリズムによるLDAPサーバのキャッシング方式を示す。

LRU リストは、キャッシュに保持しているエントリのアクセス順序を表現している。LRU リストの先頭には最近アクセスされたエントリ、最後尾には最も過去にアクセスされたエントリが配置される。キャッシュに格納されたエントリ数が閾値(キャッシュ・サイズ)を超える場合、LRU リストの最後尾に位置するエントリをキャッシュから破棄した上で、新たなエントリを登録する。

ところが机上検討及び実験の結果、LRU アルゴリズムを用いた従来のキャッシング方式には、次章に述べるディレクトリ・サーバ特有の問題がある事が判明した。

3. キャッシング機構の最適化

3.1 Partitioned LRU 方式

ディレクトリ・サービスの主たる用途は情報検索であり、ユーザは用途に応じて表1に示す3種の検索要求を使い分ける。

リードは特定の1エントリを対象とした検索、リストはあるエントリの直下に位置するエントリ群を対象とする検索、サーチはあるエントリの下位に位置する全エントリ群を対象とする検索である。リストまたはサーチによる検索結果数は検索条件に左右される。例えば、「全社の中から、姓に"a"が含まれたユー

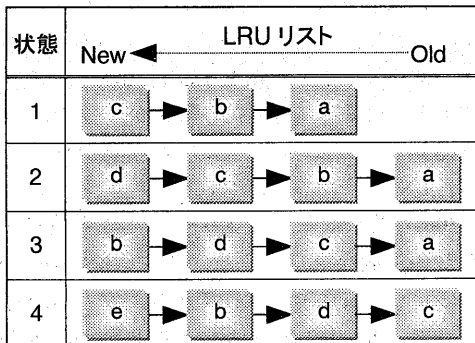


図1 LRU アルゴリズム

表1 ディレクトリ検索機能

| 種別 | 結果 | 比率[%] | 反復性 |
|-----|----|-------|-----|
| リード | 1 | 70 | 高 |
| リスト | 少 | 10 | 中 |
| サーチ | 多 | 20 | 低 |

ザを検索せよ」といった絞り込みが不十分なサーチ検索が要求されると、多数のエントリが条件に合致してしまう。

一般にリスト及びサーチ検索は、電子メール・システムにおける宛先検索等、候補の絞り込みに利用され、クライアントは人間であるケースが多い。しかし今後のディレクトリ適用分野として注目されている分散システムのリポジトリにおいては、プログラムがクライアントであるケースが多く、ノイズを除去する必要のあるリスト、サーチ検索は通常利用されず、リード検索の使用頻度が高い。本研究では、今後のディレクトリ・サービス適用分野におけるリード、リスト、サーチ検索要求の利用比率を7:1:2と推測した。

またキャッシュ・ヒット率を向上するには、繰り返しアクセスされる可能性（反復性）が高いエントリをキャッシュに残す事が望ましい。一般にサーチ検索により得られたエントリは検索条件に偶然合致したオブジェクトであり、反復性は低い。一方リード検索されたエントリはクライアントにより明示的に指定されたオブジェクトであり、反復性が高く、キャッシュの効果が得られ易い。

ところが従来のキャッシング方式によると、条件に合致するエントリが多数となるような曖昧なリストまたはサーチ検索を受けると、これらのエントリをキャッシュに登録すべく、キャッシュから溢れる多くのエントリを廃棄してしまい、本来、反復性が高くキャッシュの効果を期待できるリード検索の応答性能に悪影響を及ぼす。

実験によると、3種の検索要求が混在する場合、大半のリード検索の応答時間は0.2から0.3sに集中するが、約11%のリード検索が0.3s以上に劣化する事が判明した。

今後のディレクトリ・サービスにおいては利用頻度が高いリード検索の高応答性が重視され、上記のような突発的な性能劣化を防止しなければならない。従来方式の問題は、検

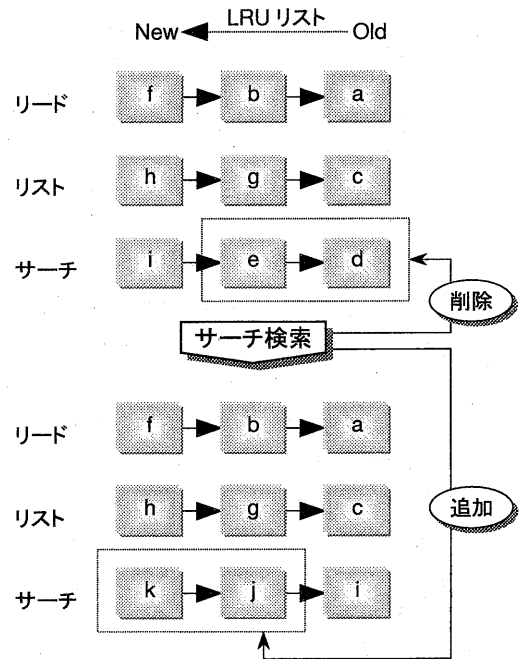


図2 Partitioned LRU

索結果数やアクセス頻度、反復性等の特性が異なる各種検索を単一のLRUで管理していた事に起因する。

そこで、LDAPの検索要求種別毎にキャッシュを分割管理するPartitioned LRU方式(図2)を開発した。本方式の目標は安定した応答性能の確保であり、具体的には全リード検索の95%を0.3s以内に応答する事である。

Partitioned LRU方式は、検索種別毎に分割されたLRUリストでキャッシュを管理する。つまり、リード、リスト、サーチ検索に対応する3本のLRUリストを備える。サーバは、検索要求を受信した際、検索種別に相当するLRUリストを対象にLRUアルゴリズムを実行する。このため、他のLRUリストに連結されたエントリは影響を受けない。

図2の例では、サーチLRUリスト内のエントリd, eが、サーチ検索の結果であるエントリj, kに置換される。リードまたはリスト検索によりキャッシュに登録された他のエントリが削除される事は無い。

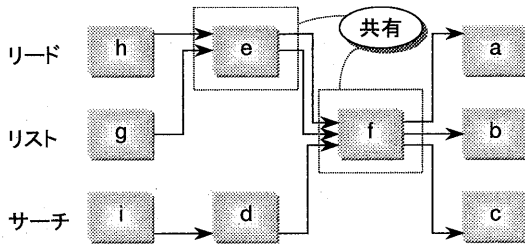


図3 冗長キャッシュの除去

次に、開発した Partitioned LRU 方式における冗長キャッシュの除去について説明する。

LDAP サーバに格納された各エントリーは、3種の検索要求の何れからでもアクセス可能である。Partitioned LRU は検索種別毎にキャッシュを分割管理する方式であり、異なる検索種別により同一のエントリーがアクセスされると、各々がキャッシュに重複登録され、余計なメモリを消費してしまう。

そこで、図3に示すように、重複するエントリーの共有により、冗長性を排除し、有限なキャッシュの利用効率低下を防止した。

重複キャッシュを共有する為、キャッシュ上の個々のエントリーに、各LRUリストに連結する2個のポインタを追加した。サーバは、キャッシュに新たなエントリーを登録する際、そのエントリーが既に他のLRUリストに存在し

ないか確認する。存在する場合は、エントリーを新たに追加せず、登録済みのエントリーを処理対象のLRUリストにも連結する。

3.2 Fractional Cache 方式

次に、本研究の第二の特徴である Fractional Cache 方式を説明する。

キャッシュ・ヒット率の向上を容易に実現するには、メモリを増設しキャッシュ数を増やせば良いが、反面、ユーザにとってコスト増となる。そこで本研究では、従来同等のメモリ所要量でも保持する最大キャッシュ数を増加可能な方式を検討した。

各ディレクトリ・エントリーは、そのオブジェクトに関連する様々な属性を保持可能である。ユーザまたはクライアント AP は、LDAP 検索要求を発行する際、そのエントリーが保持する属性群の中から所望の属性を選択する。

ところが従来のキャッシング方式は、抽出属性に関わらず、アクセスされた各々のエントリーが持つ全属性をメモリ上にキャッシュとして保持する為、キャッシュの所要メモリは各エントリーに格納された属性数及び属性値長に比例する。このためアクセス頻度の低い属性(例えば各ユーザの JPEG 画像等)により大量にメモリを消費する場合がある。

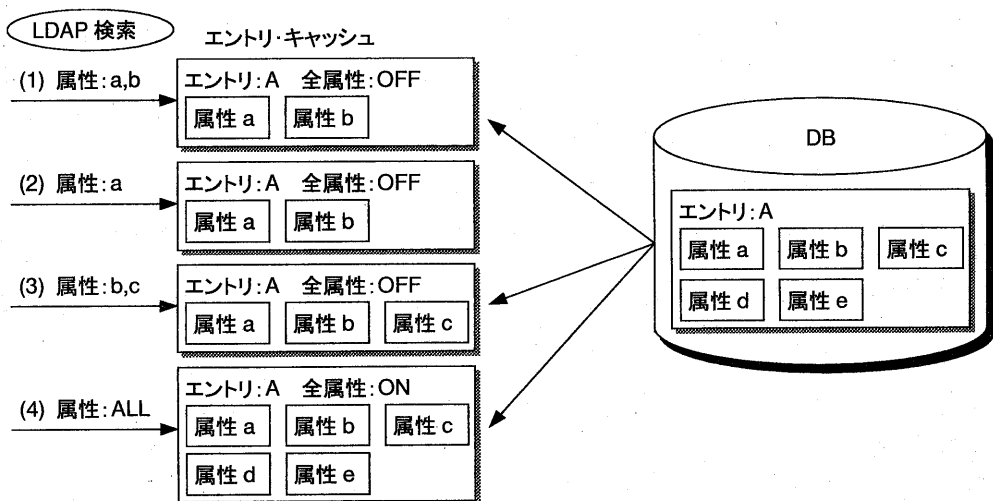


図4 Fractional Cache

そこで、メモリの利用効率を高める為、アクセスされた属性に限りキャッシュとして保持する Fractional Cache 方式 (図 4) を開発した。本方式の目標はリスト及びサーチ検索の応答性能を 20%向上する事である。

サーバは、(1)のように検索時に抽出を指定された属性に限りキャッシュに保持する。(3)のようにキャッシュに未保持の属性抽出を要求された場合はその属性値を追加する。

また LDAP 仕様によると、そのエントリが持つ全属性の抽出も指定可能である。このような要求に対してもキャッシュを有効活用する為、全属性を保持しているか否かを示すフラグをキャッシュ上の各エントリに設けた。例えば(4)のように全属性の抽出を要求された場合には、全属性値をキャッシュに登録すると共に、フラグを ON に設定する。再度、同一の要求を受けた場合は、キャッシュ上の情報を用いて検索結果を生成する。

4. 性能評価

本章では、Partitioned LRU 及び Fractional Cache 方式の性能測定結果を報告する。

4.1 測定環境

評価データは 10 万エントリであり、各エントリは、姓名、住所、電話番号等、25 個の属性を持つ。1 エントリ当たりのデータ量は平均 830 バイトである。上記のデータを LDAP サーバに登録し、4000 回 (リード:リスト:サーチ =7:1:2) の検索要求に対する応答時間を測定し

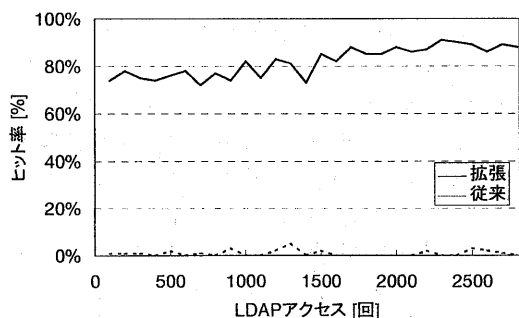


図5 キャッシュ・ヒット率 (リード検索)

表 2 平均応答性能

| 種別 | 従来[s] | 拡張[s] | 向上率[%] |
|-----|-------|-------|--------|
| リード | 0.23 | 0.21 | 9 |
| リスト | 2.88 | 2.15 | 25 |
| サーチ | 2.84 | 2.09 | 26 |

た、キャッシュ・サイズは、従来方式が 1000、本方式が 3000 である (メモリ所要量は同等)。

表 2 に、従来方式と本研究方式の平均応答時間を示す。

4.2 リード検索

図 5 は、従来方式と本研究方式 (拡張) のキャッシュ・ヒット率の比較である。値はリード検索 100 要求毎のヒット率である。

性能測定に用いたテスト・シーケンスは、実際のディレクトリ利用環境を想定し、10 個の検索要求の内、3 要求は複数のエントリが検索結果となり得るリストまたはサーチ検索である。従来のキャッシング方式では、これらの要求が外乱となり、多数のキャッシュが置換されてしまい、結果としてヒット率が平均 1% に劣化している。これに対し Partitioned LRU 方式は、リード検索により登録されたキャッシュがリストまたはサーチ検索による影響を一切受ける事が無い為、平均 82% のヒット率を達成できた。

図 6 は両方式のリード応答性能分布を 0.1s 刻みに分類したグラフであり、棒グラフ中の数値は各間隔毎の応答数を表す。

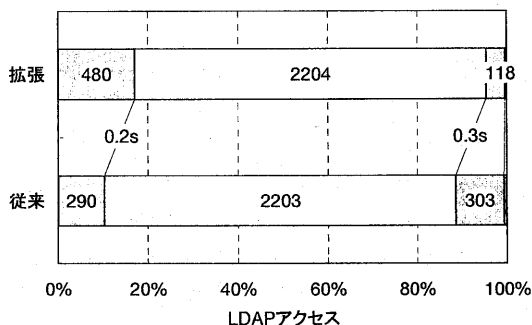


図6 リード検索・性能比較

従来方式は、リストまたはサーチ検索の混入により多くのキャッシュが一掃されてしまい、11%のリード検索の応答時間が0.3s以上に悪化する。これに対し Partitioned LRU では、0.3s 以上に応答性能が劣化するリード検索は5%であり、リード検索の安定した応答性能を確保した。

ところで、従来方式のキャッシュ・ヒット率が本方式に比べ極端に低いにも関わらず、両方式とも大部分のリード検索の応答時間が0.2から0.3s以内に集中している。これは、従来方式においてキャッシュにヒットしない場合にも、OS 提供のファイル・キャッシングが有効に働き、ファイル I/O のオーバーヘッドを削減できた為と考えられる。しかし実環境では、他サービスとのマシン共用やアクセスするファイル・ブロックの散乱等により、ファイル・キャッシュのヒット率が低下し、本測定結果よりも両方式の性能差が更に拡大するものと推測する。

4.3 リスト/サーチ検索

図7は両方式のサーチ応答性能分布を0.5s刻みに分類したグラフである。

測定によると、全体の75%以上の応答が、従来方式では2.5から3.0s以内に集中しているのに比べ、本方式は2.0から2.5s以内に集中している。また従来方式では21%の応答が3s以上に劣化するのに対し、本方式は99%の応答を3s以内に処理可能である。

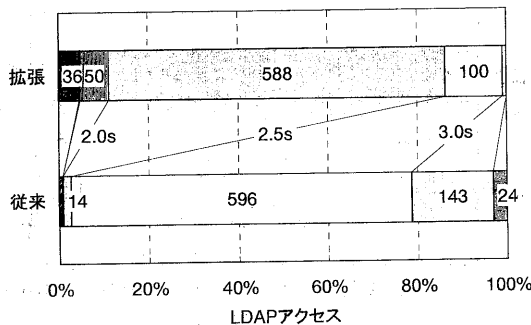


図7 サーチ検索・性能比較

上記の結果は、本研究の Fractional Cache 方式の効果である。本方式により従来と同等のメモリ所要量でも3倍のキャッシュを設ける事が可能となり、キャッシュ・ヒット率を向上できたものと考えられる。

表2の測定結果に示すように、本研究のキャッシング方式は、リスト及びサーチ検索とも、従来方式に比べ、平均応答時間を約0.7s、25%向上しており、性能目標とした20%を達成可能である事を確認した。

5. おわりに

LDAP サーバにおける高性能キャッシング技術として、検索種別毎のLRU管理により安定した応答性能を確保する Partitioned LRU 方式、アクセス属性に限定したキャッシュ保持により応答性能を向上する Fractional Cache 方式を開発・評価し、目標性能を達成した。

現在、本研究の製品適用を検討中である。

参考文献

- 1) ISO/IEC 9594-1~8 / CCITT X.500 シリーズ
- 2) 小花 貞夫他:リレーショナルアプローチによる OSI ディレクトリの DIB (ディレクトリ情報ベース) の実装と評価:情報処理学会論文誌, Vol.32, No.11, pp.1488-1497 (1988)
- 3) T.Howes 他: A Scalable, Deployable, Directory Service Framework for the Internet: INET'95 (1995)
- 4) M.Wahl 他: RFC2251 Lightweight Directory Access Protocol (v3): (1997)
- 5) 日立製作所:日立ディレクトリサーバ: <http://www.hitachi.co.jp/Prod/comp/soft1/directory/> (1998)
- 6) 菊地 聡他:基幹システムにおける高信頼ディレクトリ・サーバ:情報処理学会研究報告分散システム運用技術, 97-DSM-7 (平9)
- 7) 平島 陽子他:基幹システム向けディレクトリレプリケーション・サーバの開発:情報処理学会研究報告マルチメディア通信と分散処理, 98-DPS-87 (平10)