1997年度日本オペレーションズ・リサーチ学会
秋季研究発表会

分散キャッシュサーバのための問い合わせ関係構成

01011240 NTT 朝香 卓也 ASAKA Takuya
01605520 NTT 巳波 弘佳 MIWA Hiroyoshi

1 はじめに

通信ネットワークにおける分散キャッシュサーバ(以下，CS)は，クライアントのデータ高速取得を可能にし，かつネットワークやオリジナルサーバへの負荷の軽減を可能とする．大規模な分散CSを可能にする「問い合わせキャッシング方式」[1]が提案されている．本稿では，本方式を効率的に利用するための問い合わせ関係構成アルゴリズムと得られた問い合わせ関係について述べる．

2 背景

現在の通信ネットワーク，特にインターネットではDNS (Domain Name Server)[2]やWWW (World Wide Web) 代理サーバ[3]等のキャッシュ機能を持つ各種サーバが数多く利用されている．これらキャッシュ機能を持つサーバの多くは現在単独で利用されているため，近くの他のCSがオブジェクト(DNSの場合はIPアドレス，WWWの場合はテキストや画像データ)をキャッシュしていても，直接オリジナルサーバへアクセスしてしまう．これらCSが協調すれば互いのキャッシュデータを有効利用できる．

今までに提案されていた分散CS方式[4][5]では，クライアントからのアクセス時に他CSにキャッシュデータの存在を問い合わせる．よって，大規模な分散CSではCS間の問い合わせ数が増大するという問題があった．問い合わせキャッシング方式[1]では，分散CS内の限られた数のCSに問い合わせるだけで目的とするオブジェトを発見することができるので，多数のCSで分散CSを構成できる．

3 問い合わせキャッシング方式

本方式では，クライアントからアクセスされたCSは目的とするオブジェクトを発見し，そのオブジェクトをクライアントへダウンロードすると同時に自分自身もキャッシュする．さらに各CSはキャッシュデータとして,「オブジェクト＋オブジェクト名」に加えて,「他CSから過去に問い合わされたオブジェクト名+問い合わせ元CS名」を持つ．あるCSが他のCSに対して過去に問い合わせたということは，

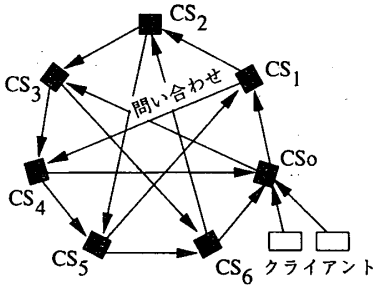

図1: 問い合わせキャッシング方式の動作

その問い合わせ元CSはいずれかのCSあるいはオリジナルサーバからオブジェクトを取得しキャッシュしていることになる．また，図1のように各CSの問い合わせ先CSはあらかじめ決めておく．

次に，図1を例にCSの動作について述べる．クライアントがCS_0にオブジェクトObjを要求した時，CS_0がObjをキャッシュしているなら，CS_0はクライアントへObjをダウンロードし終了する．CS_0がObjをキャッシュしておらず，かつ過去に他のCS(CS_4とCS_6)から問い合わされた経験があるなら，その問い合わせ元CSからObjをダウンロードし終了する．それ以外の場合には，CS_0は他CS(CS_1とCS_3)に問い合わせ，CS_1とCS_3は，Objの有無と過去に他CSから問い合わされた経験の有無も返答する．CS_0は得られた返答をもとに，Objをキャッシュする CSか，またはオリジナルサーバからObjをダウンロードし終了する．

これにより，問い合わせを行うCSを少なく抑え，大規模な分散CSを構築できる．この時，特定のCSへの問い合わせの集中を避け，かつネットワーク内の問い合わせためのトラヒックを少なくするための適切な問い合わせ関係を構成する必要がある．

4 問い合わせ関係構成

CS間の問い合わせ関係は，以下のような有向グラフから構成できる．

定義 (問い合わせ関係グラフ) 節点数n，自然数k，Lが与えられた時，任意の2節点に対して，[条件1]節点間に有向枝が存在する [条件2]それぞれ同一節点を終点とする有向枝の始点になっている，のいずれかを満たし，かつ全ての節点

における入次数はk以下で，かつ枝数がL以下である有向グラフの集合をパラメータ(n,k,L)の問い合わせ関係グラフの集合といい，その各要素の有向グラフを問い合わせ関係グラフといい，$G(n,k,L)$と表す．□

グラフの節点をCSに対応させ，有向枝をその枝の始点に対応するCSから終点に対応するCSに対する問い合わせ関係に対応させる．この時，$G(n,k,L)$は，条件1と2のいずれかを満足することより，互いのキャッシュ情報(注目するObjの有無とそのObjについて問い合わせてきたCS名の有無)を得ることができる．さらに，入次数がk以下ということから特定のCSに対して問い合わせが集中せず，枝数がL以下ということから問い合わせのためのトラヒック量が制限される．従って，CS数nと適当に固定したkとLが与えられた時の$G(n,k,L)$から，CS間の問い合わせ関係を構成できる．

問い合わせ関係グラフの存在性について，次の定理1は問い合わせ関係グラフが存在するための必要条件を与えている．

定理1 問い合わせ関係グラフ$G(n,k,L)$が存在するならば，$n \leq k^2+k+1$が成り立つ．[証明 略]

しかし，任意の(n,k,L)の組に対して$G(n,k,L)$が存在するわけではない．また，存在しても実際にグラフを構成することは容易でない．そこで，入次数ができるだけ小さく，かつ枝数ができるだけ少ない$G(n,k,L)$を構成するアルゴリズムを挙げる．

アルゴリズム

SETP1：$k := \lceil (n-3/4)^{1/2} - 1/2 \rceil$とする．ただし，$\lceil \cdot \rceil$は·を下まわらない最小の整数を表わす．

SETP2：集合$H = \{p_1, p_2, ..., p_k | p_i \in \mathcal{Z}, 1 \leq p_1 < p_2 < ... < p_k \leq n-1\}$を生成する．

SETP3：有向グラフ$\tilde{G}$を生成する．ただし，$\tilde{G}$の節点集合を$V = \{v_0, v_1, ..., v_{n-1}\}$，有向枝集合を$E = \cup_{i=1}^{k} \cup_{j=0}^{n-1} \{(v_j, v_{q_{ij}}) \mid q_{ij} = p_i + j \pmod n\}$とする．

SETP4：$\tilde{G}$が問い合わせ関係グラフの定義1)と2)を満たさなければSETP7へ行く．

SETP5：$\tilde{G}$の全ての枝に対して，その枝を除去しても$\tilde{G}$が問い合わせ関係グラフ定義の条件2を満たしているかチェックし，満たすならその枝を除去する．

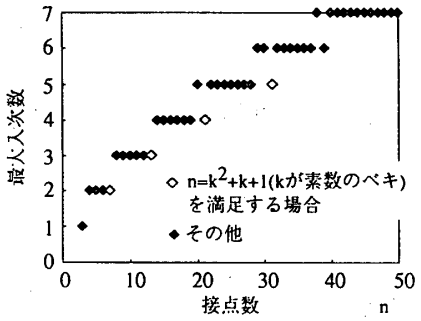

図2: 問い合わせ関係グラフの最大入次数

SETP6：$G(n,k,|E|)$がまだ生成されていないか，あるいは$\tilde{G}$の枝数が$G(n,k,|E|)$の枝数を下回れば，$G(n,k,|E|) \equiv \tilde{G}$とする．

SETP7：新たにHを生成してSTEP3へ行く．新たにHが生成できず，かつ既に$G(n,k,|E|)$が生成されていれば終了する．さもなければ$k := k+1$としてSTEP2へ行く．

このアルゴリズムでは，定理1の必要条件から得られる最小の入次数kから開始し，その入次数の下で問い合わせ関係グラフが構成できるか否か確認し，構成できるまで入次数を1ずつインクリメントすることを繰り返し帰納的に有向グラフを構成する．

節点数nが$n = k^2+k+1$(ただし，kが素数のベキ)を満足する場合には，このアルゴリズムで最小の入次数でかつ最小の枝数の問い合わせ関係グラフを構成できる(図1は$k=2$の場合)．また，このアルゴリズムを用いて得られた問い合わせ関係グラフの最大の入次数を図2に示す．

5 おわりに

今後の課題として，新規CS導入時の問い合わせ関係の再構成方法の検討があげられる．

参考文献

[1] 朝香, "WWW協調型キャッシュサーバネットワークのための問い合わせキャシング方式," 信学会1997年総合大会, pp.271, B-7-142, Mar., 1997.

[2] P. Mockapetris, *"Domain Names - Implementation and Specification,"* STD 13, RFC 1035, USC/Information Sciences Institute, Nov., 1987.

[3] World Wide Web Consortium, http://www.w3.org/pub/WWW/.

[4] D. Wessels, *"Squid Cache Server,"* http://www.nlanr.net/Squid/.

[5] L. Zhang, and S. Floyd and V. Jacobson, *"Adaptive Web Caching,"* Cache Workshop '97, 1997.