P2P システムのためのスケーラブルな木構造ベースの整合性維持手法

広島大学大学院工学研究科情報工学専攻　分散システム学研究室　中島 大志

概要

P2P ファイル共有システムのための新しい整合性維持手法を提案する．提案手法の基本的な考え方は，共有ファイルごとに静的な木を構築することによって，更新情報を全レプリカノードに効率的に伝播させるというものである．木の根へのリンクは，共有ファイルから木の根へのマッピングを保存した Chord リングを参照することで得られる．提案手法の性能はシミュレーションによって評価される．シミュレーションの結果，メッセージ数・更新伝播遅延・離脱耐性において従来手法に対して優位性があることが示された．

1　はじめに

動画や音楽といったコンテンツを P2P ネットワーク上で共有する P2P ファイル共有システムは，世界中で活用されている．Gnutella, WinMX といった従来の P2P ファイル共有システムは，ユーザが共有ファイルを更新することができなかった．しかし，近年では，オンライン・ストレージや共同編集システムなど，ユーザが共有ファイルを更新できる P2P システムが高い注目を集めている．

一方，P2P ファイル共有システムでは，共有ファイルのレプリカ（複製）を分散配置することが一般的である．レプリカによって，負荷分散，離脱耐性，クエリ検索速度の向上といった性能面での利点が得られる．

もしも P2P システムがレプリカを配置しながら，そのユーザが共有ファイルを更新できる機能を提供する場合，共有ファイルの更新時にその全レプリカに更新情報を伝達してレプリカも正しく更新を行う必要がある．それを行う機構を**整合性維持機構**といい，これまでに多くの整合性維持手法が考えられてきた．中でも，Li らの手法 [4] は，レプリカノードのみで構成される Chord リングを用いて更新情報を伝えることで，効率的に更新情報の伝播を行うことに成功している．

本研究では，Li らの手法の問題点に着目して，新しい木構造ベースの整合性維持手法を提案する．提案手法では，静的にレプリカノードのみから木構造を構成して，共有ファイルが更新されたときにはその木の根から葉へ向けて更新情報を伝播する．更新時に木構造の根ノードを知るため，システム全体で 1 つの Chord リングを構成しておき，共有ファイルから木の根ノードへのマッピングを提供する．ノードが離脱した場合には木構造が崩れるため，先祖ノードも記憶しておくといった離脱耐性手法をとっている．

本研究の性能は P2P シミュレータである OverSim[1] によるシミュレーションで評価し，提案手法は Li らの手法よりメッセージ数・更新伝播遅延・離脱率においてまさっていることが示された．

2　関連研究

既存の整合性維持手法としては，push/pull[3]，IRM[5]，SCOPE[2] などがあるが，中でも効率がよいことが実験的に示されている手法が Li らの手法 [4] である．Li らの手法では，共有ファイルごとにレプリカノードのみからなる Chord リング [6] を構成する．そして更新時には，Chord リングから木構造を動的に抽出して，その根から葉へ向けて更新情報を伝播していく．ノード x が共有ファイル f の更新を行ったとして，f の Chord リングから木構造を抽出する方法は次の通りである．

- まず，ノード x は木構造の根ノードになる．
- x を除いた f の Chord リング空間を考え，この空間を d 個の部分空間に分割する．
- d 個の部分空間 S_i のそれぞれ最初のノード i_1 を，x の子ノードとする．また，i_1 の担当空間を S_i とする．
- 同様に，各子ノードは自分の担当空間を d 個の部分空間に分割して，それぞれの最初のノードを自分の子ノードとする．
- 以上を繰り返すことで，x を根として Chord リング上のノードだけからなる木構造が構成される．

Li らの手法では，共有ファイルごとに Chord リングを構成するため，共有ファイルの数が増加するにつれて Chord リングのメンテナンス・コストが増加していくという欠点がある．

3 提案手法

3.1 概要

提案手法では，共有ファイルごとに静的にレプリカノードからなる木構造を構成する．そして，共有ファイルの更新時には，(1) 木構造の根ノードを発見して，(2) 根ノードから順に子ノードに対して更新情報を伝達していくことで，更新情報の伝播を行う．Li らの手法と異なり，共有ファイルごとに Chord リングを構成する必要がないため，メンテナンス・コストは Li らの手法より少ないと考えられる．実際，Li らの手法では，共有ファイルごとに各ノードは $\Theta(\log X)$ 個の隣接ノードを管理する必要がある一方で，提案手法では，共有ファイルごとに各ノードは自身の子ノードと定数個の先祖ノードのみを管理すればよい．

木構造を設計する上で考慮した課題は次の通りである：

- 共有ファイル f の更新を行うためには，まず f に対応する木の根ノードを発見する必要がある．提案手法では，Chord リングを用いて共有ファイルから根ノードへのマッピングを用意することで，高速な根ノードの発見を可能にしている．
- 木のリンクに沿って更新情報を伝播する場合，木構造はできるだけ平衡化されていることが望ましい．また，親ノードの負荷を抑えるため，子ノード数を制限する必要がある．提案手法では，レプリカノードの追加時に，子ノード数を一定に制限する一方で，子孫ノード数を利用して木構造をできるだけ平衡化する方法を取り入れている．
- 葉以外のノードが木から離脱すると，木構造が分断されてしまい，更新情報が伝播されなくなってしまう．提案手法では，ノード離脱時に木構造を修復する方法を提供している．

3.2 根ノードの発見

共有ファイル f に対応する木を T_f，その木の根ノードを r_f と記すことにしよう．T_f は f のレプリカを持つ全ノードから構成される．f を更新した場合には，まず T_f の根ノードを発見して，更新情報を T_f の根ノード r_f へ送信する．更新情報を受け取った根ノードは自分の子ノードへ更新情報を伝達し，各子ノードも同様に更新情報を自分の子ノードへ伝えていく．結果として，f の全レプリカノードに更新情報が伝播される．ここで，根ノードの発見には Chord リングが用いられる．ファイル共有システムに参加している全ノードで 1 個の Chord リングを構成し，この Chord リングは共有ファイル f の集合から木の根ノード r_f の集合への写像を格納している．f の更新を行うノードは，Chord リングに対して木の根ノード r_f を問い合わせることで，更新情報を送信する木の根ノードを知ることができる．共有ファイルごとに Chord リングを構成する Li らの手法と異なり，提案手法はシステム全体で 1 個の Chord リングしか必要としないことに注意しよう．

根ノードの発見を高速に行うために，提案手法ではキャッシュ機構を用意する．あるノード x が 1 度 Chord リングの探索を行って，共有ファイル f から木の根ノード r_f へのキーバリューペアが得られたとしよう．x はこのキーバリューペアの情報をキャッシュしておき，再び f の更新を行った際には，Chord の探索を行わずにキャッシュされた r_f に直接更新情報を送る．もしも，キャッシュされた r_f が離脱しているか，すでに T_f の根ノードでなくなっていれば，x は Chord の探索を行って正しい根ノードを発見し直す．

3.3 レプリカノードの追加

T_f 中の各ノード u は，T_f における子ノード集合 $Child_f(u)$ と子孫ノード数 $D_f(u)$ を局所的に保持している．また，システムのパラメータとして最大子ノード数 d を用意する．ノード x が新たに f のレプリカノードになるとしよう．ノード x は T_f 中のノード y から f のレプリカを取得した後，自身の局所変数を $Child_f(z) := \emptyset, D_f(x) := 1$ という形で初期化して，ノード y に向けて参加要求 $\mathsf{Add}(x)$ を送る．$\mathsf{Add}(x)$ を受け取った y は，局所変数 $D_f(y)$ の値を 1 増やした後，

- もし $|Child_f(y)| < d$ ならば，$Child_f(y)$ に x を加えて x を自分の子とした上で，その旨を x に通知し，
- そうでなければ，$Child_f(y)$ の中でもっとも変数 D_f の値が小さな子ノードに向けて参加要求 $\mathsf{Add}(x)$ を転送する．

同様の手続きは，参加要求を受け取ったノード上でも，(x の親ノードが決まるまで) 繰り返し実行される．根から葉へのどんな経路でも，$|Child_f(y')| < d$ となるようノード y' は最低 1 個存在するため，この繰り返しは必ず終了することに注意しよう（たとえば葉ノードは必ずこの条件を満たす）．

3.4 離脱耐性

まずは，根ノード以外のノードが離脱した場合の対処法を示す．各ノードは定数世代分の祖先ノードを記録してお

くための変数 Anc_f を用意しておく．そして，隣接ノードの離脱が起きたかどうかは，隣接ノード間で定期的にメッセージ通信を行うことで検知できる．親ノードの離脱を検知したノード x は，Anc_f の中でシステムに参加しているもっとも根に近いノード y に向けて再参加要求 Add(x) を送る．Add(x) を受け取ったノード y の動きは，前節と同様である．

次に，根ノードが離脱した場合の対処法を示す．提案手法では，根ノードの子からランダムに $c(\geq 1)$ 個のノードを選択して根のコピーノードとする．そして，Chord リングには，キーバリューペアの値として，根ノードだけではなく，根ノードおよびそのコピーノードを登録する．根ノード r_f とそのコピーノードは，ファイル f のレプリカと変数 $Child_f(r), D_f(r)$ という 3 つの情報を共有する．ファイル f を更新するときや根ノードに対して Add 要求を送るときに根ノードが離脱していた場合，コピーノードをランダムに 1 つ選択して新しい根ノード r'_f として，残りのコピーノードは r'_f のコピーノードとなる．また，根ノード r_f は自らのコピーノードが c 個未満になった場合，まだコピーノードになっていない任意のノードを $Child_f(r_f)$ から選択して，コピーノードとする．

4 評価

4.1 設定

OverSim[1] を用いたシミュレーションにより，提案手法の性能を実験的に評価した．シミュレーションの設定は次の通りである．ノード数を 500，共有ファイル数を 500 に固定した P2P ファイル共有システムを考える．各ノードは $s = 1, N = 500$ の Zipf 分布に従った人気度によってファイルを共有している．各ファイルはレプリカノードによって秒単位で $\lambda = 0.05$ のポアソン分布に従って更新される．各ノードは秒単位で $\lambda = 10000, k = 1.0$ のワイブル分布にしたがって参加・離脱する．パラメータ d の値は $d = 16$ と定義している．

総メッセージ数・更新伝播遅延・離脱耐性という 3 つの項目に関して提案手法の性能を評価し，2 で述べた Li らの手法と比較した．

図 1 は総メッセージ数の結果で，図 2 は更新伝播遅延の結果である．提案手法は Li らの手法よりもよい結果を示し，またキャッシュの効果によりさらに総メッセージ数と更新伝播遅延の削減が行われていることが分かる．メッセージの内訳を分析したところ，Li らの手法は Chord リングのメンテナンスに大きな負荷がかかっていた．

全参加ノード中の D[%] が “同時に” 離脱した状況を考える．この離脱率 D に応じた更新伝播遅延を評価した．図 3 に結果を示す．提案手法は離脱率が上昇するとともに更新伝播遅延も増加しているが，それでも Li らの手法より低い離脱率に抑えられていることが分かる．

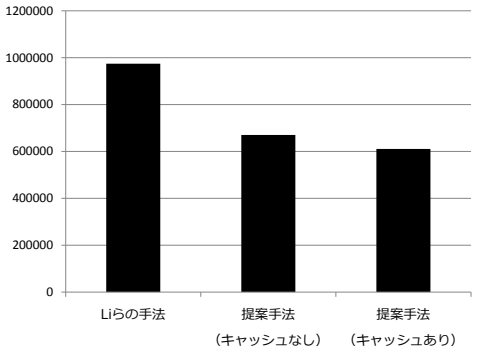

図 1　総メッセージ数.

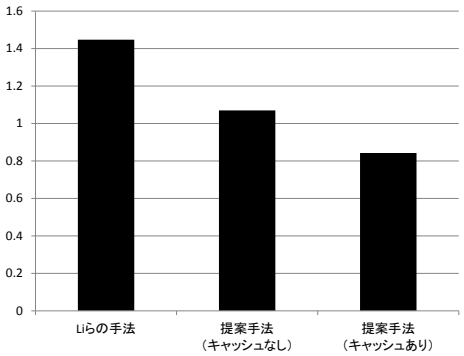

図 2　更新伝播遅延.

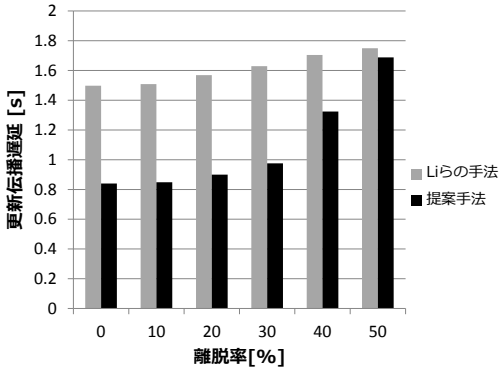

図 3　離脱率に応じた更新伝播遅延.

5　おわりに

本研究では，共有ファイルごとに静的な木を構築することによる，P2P ファイル共有システム向けの整合性維持手法を提案した．Li らの手法と異なり，提案手法は共有ファイルごとの Chord リングを必要とせず，根ノードの予備を用意して先祖ノードを記憶しておくことで離脱耐性を実現している．シミュレーションの結果，提案手法は Li らの手法よりメッセージ数・更新伝播遅延・離脱率においてまさっていることを示した．

今後の課題として，実際のアプリケーションを開発した上での実験評価などがある．また提案手法の向上方法として，木の子ノード数を適応的に決定していくことが考えられる．

参考文献

[1] Ingmar Baumgart, Bernhard Heep, and Stephan Krause. OverSim: A Flexible Overlay Network Simulation Framework. In *Proceedings of 10th IEEE Global Internet Symposium Symposium*, pages 79–84, 2007.

[2] Xin Chen, Shansi Ren, Haining Wang, and Xiaodong Zhang. SCOPE: Scalable Consistency Maintenance in Structured P2P Systems. In *IEEE INFOCOM*, volume 3, pages 1502–1513, 2005.

[3] Jiang Lan, Xiaotao Liu, Prashant Shenoy, and Krithi Ramamritham. Consistency Maintenance in Peer-to-Peer File Sharing Networks. In *Proceedings of the The Third IEEE Workshop on Internet Applications*, WIAPP '03, pages 90–94, 2003.

[4] Zhenyu Li, Gaogang Xie, and Zhongcheng Li. Efficient and Scalable Consistency Maintenance for Heterogeneous Peer-to-Peer Systems. *IEEE Transactions on Parallel and Distributed Systems*, 19(12):1695–1708, 2008.

[5] Haiying Shen. IRM: Integrated File Replication and Consistency Maintenance in P2P Systems. *IEEE Transactions on Parallel and Distributed Systems*, 21(1):100–113, 2010.

[6] Ion Stoica, Robert Morris, David Liben-Nowell, David R Karger, M Frans Kaashoek, Frank Dabek, and Hari Balakrishnan. Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications. *IEEE/ACM Transactions on Networking*, 11(1):17–32, 2003.