C-038

ピュア P2P 型分散ファイルシステムにおける均一なデータ配置の実装と提案
Study on Uniformity Data Arrangement in the Pure P2P Distributed File System

平松 謙隆‡　赤坂 翔太‡　小林 孝史‡
Yoshitaka Hiramatsu　Shota Akasaka　Takashi Kobayashi

1. はじめに

近年，情報通信技術の発展と共に情報処理で扱われるデータは，構造化された定型データに加え，動画や音声などの非定型データなどが急激に増大している．このような大容量データは今後も増加する一方と考えられる．そのため，システムの拡張への対応などスケーラビリティが求められ，同時にデータの消失を防ぐ安定的な運用が求められる．

これらを実現する手段として，分散ファイルシステムが存在する．これは，ファイルシステムを持った複数のマシンをネットワークで繋ぐことで，一つの巨大なストレージを構成するシステムである．現在，リリースされている分散ファイルシステムの例として GFS や pNFS などが挙げられる．これらは，複数のマシンにまたがって格納されるファイルの所在地を統括して管理する管理ノードと呼ばれるマシンを用いる．しかし，管理ノードが停止した場合はファイルの所在地を参照することが出来ないため，最悪の場合にシステム全体が停止してしまう恐れがあり，単一障害点を持つことになる．

田中の研究 [1] では分散ファイルシステムにおける単一障害点の問題を解決するため，P2P 技術を応用したファイルシステムの開発を行った．P2P 技術を用いることにより，ネットワークに参加するノードが相互に経路状態を管理することで，ファイルの所在位置を管理する．しかし，田中の開発したシステムの課題として，ノードごとのデータ配置に偏りが生じることが挙げられる．これは，ファイル管理方法において，ファイルはハッシュ値により格納先を一意に結び付けられる．そのため，ファイルサイズにばらつきが存在する場合に，特定のノードのディスク使用率や負荷の集中がファイル処理のボトルネックになる．

そこで本研究では，各ノードのディスク使用量を均一化し，システムのリソースを効率利用するために，ピュア P2P 型ネットワークで構成された各ノードに対して，一つのファイルを分割して配置する分散ファイルシステムを実装した．そして，各ノードに対するファイルサイズの分散値を求め評価を行った．

2. 先行研究

P2P 技術を応用した分散ファイルシステムの例として，田中の研究が挙げられる．田中の研究では，Chord アルゴリズム [2] を用いて複数のノード間で分散ハッシュテーブル（以下，DHT と呼ぶ）を構成する．これにより，管理ノードを持たないピュア P2P 型ネットワークを構築した．

Chord アルゴリズムを組み込んだ P2P ネットワークにおいて，論理空間上に属するノードは，ノード名をもとにハッシュ値を生成し，その大小関係から時計回りのリング状にマッピングされる．また，各ノードは経路表を保持し，時計回りで一つ先に位置するノードを Successor と呼び，一つ後ろに位置するノードを Predecessor と呼ぶ．さらに，ノード探索を効率化するための経路として FingerTable を持つ．これらの経路を常に更新することで，ネットワーク構造を維持しながら担当ノードの検索を実現する．

田中の開発した分散ファイルシステムでは，ネットワークから探索された担当ノードに対して，ファイルが散らばって配置される．そのため，分散ファイルシステム全体で統一されたファイルの階層構造を維持する必要がある．そこで，ディレクトリエントリを格納したファイルを用いて，これを実現した．

3. 提案手法

本研究では，P2P 型分散ファイルシステムにおけるデータ配置について，各ノードに配置されるデータサイズを均一化する手法を提案する．本システムの概要を図 1 に示す．

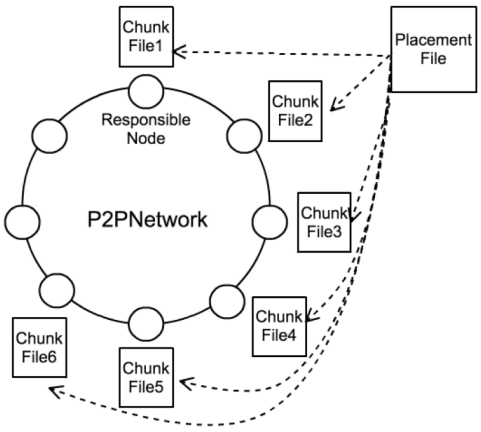

図 1　提案手法

田中のシステムでは，コンシステント・ハッシュ法に基づいてファイル配置が行われることから，ハッシュ値を持つデータに対して担当ノードは一対一の関係にある．そこで本手法は，データサイズの大きいファイルに限定して分割配置を行うことで，特定のノードに対するデータの偏りを軽減する．

本システムでは，ファイルを分割する際に基準となるチャンクサイズを設ける．また，分割されたファイルをチャンクファイルと呼び，分割順に要素番号をファイル名の末尾に加える．チャンクファイルの配置先は，各ノードがシステムを構成するノードをノードリストとして保持し，これを用いて決定する．そして，元ファイルの担当ノードを始点として各ノードへ順に配置する．また，ディレクトリを管理するためのディレクトリファイルには，それ以下のディレクトリエントリだけでなく，ファイル情報や分割数も含めて保存する．これにより，ディレクトリファイルを参照することで，一つのファイルとしてチャンクファイ全体の情報を取得できる．

† 関西大学大学院 総合情報学研究科 知識情報学専攻
‡ 関西大学 総合情報学部

ディスクアクセスを伴うファイル処理として，書き込み時は，チャンクサイズを基準に一つのファイルを分割し，ノードリストを辿ることで，各ノードに対する書き込みを行う．これにより分割処理は，ファイルシステムにおけるファイル処理の段階で行われることから，再配置のスケジューリングを行わないため，データ競合の状態が発生しないことが期待できる．

4. 実装

本システムは UNIX 系 OS 環境におけるファイルシステムとして動作する．またユーザ空間でファイル処理を制御するため，ファイルシステムの実装には FUSE を用いる．提案手法である分割配置処理は，従来の P2P 型分散ファイルシステムの構成に加え，拡張機能として以下の三つの機能を組み込む．

- チャンクファイル管理
- 配置先ノードリスト
- ファイル分割配置

チャンクサイズは，設定ファイルに記述することで任意のサイズに設定する．また，ファイルの作成時はハッシュ値から生成されるファイル名の末尾に分割順に要素番号を加える．これにより，参照時に競合することを防ぐ．さらに，チャンクファイルを効率的に管理するため，ディレクトリファイルには全てのチャンクファイルを合わせた元ファイルの情報を加える．これにはファイルの総サイズや分割数を加えて記載することで，ディレクトリファイルを参照して，全てのファイル操作を可能にする．

配置先ノードリストは，チャンクファイルの配置先を示し，P2P 経路と同様に各ノードにより更新される．配置先は，本システムが稼働しているノードをネットワーク上から検索することで，実体が存在するノードのみを追加する．これにより，P2P ネットワーク上の経路とは異なる経路として配置先を管理する．

ファイル分割処理は，ファイル操作に対応する処理内部へ組み込まれる．チャンクファイルの書き込み時は，ディレクトリファイルから現状のファイルの分割数を読み込み，ファイル名の末尾に加える．そして，対象となるチャンクファイルのサイズを調べることで書き込み可能なサイズを求め，そのサイズが 0 になるまで書き込みを続ける．一方で書き込み可能サイズを超えた場合，ノードリストから次の配置先の参照を行う．最後に，残りのデータについて，ディレクトリファイルに保存された分割数をインクリメントし更新した上で，新たなチャンクファイルの作成を行う．

5. 評価

提案手法の有効時と無効時の状態について，配置ファイルの個数を変化させた時の各ノードにおける分散の値を比較し検証を行った．検証については，配置するファイル数を 20～100 個まで 20 個ずつ変化させ，その時の各ノードへの配置数を計測した．また，各ファイルサイズについては，N個のファイル数に対し，1~NKB まで 1KB ずつ増加させたものを使用することで，偏りのあるファイル配置を設定した．分割サイズの基準となるチャンクサイズについては 10KB とした．これらの結果を図 2 及び図 3 に示す．

検証の結果，提案手法の有効時と無効時を比較すると，有効時では分散の値が無効時と比べておよそ 10 分の 1 程度まで減少した．

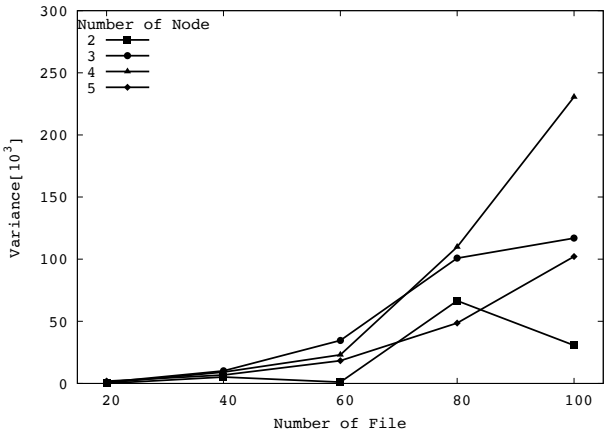

図 2　配置データの分散の値(既存手法)

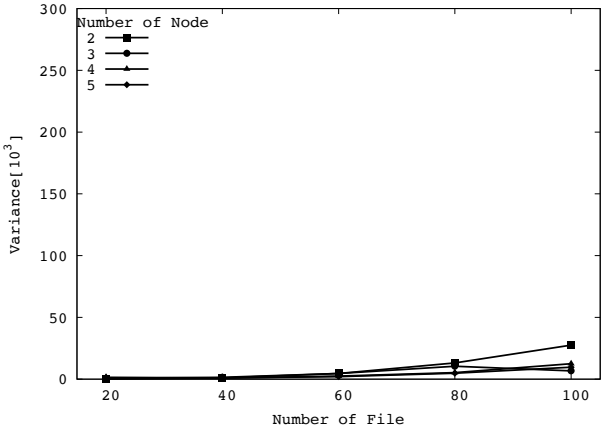

図 3　配置データの分散の値(提案手法)

このことから，本手法は，サイズの大きなファイルを分割配置することで，各ノードのディスク使用率を均一化でき，ファイルサイズに偏りがある場合の配置に対しても有効であるといえる．

6. おわりに

本研究では，サイズの大きいファイルをチャンクサイズごとに分割し，ネットワークを構成するノードへ分散配置することで，データ配置の偏りを減少させることが可能となった． 今後の展望として，チャンクサイズの値を固定から可変に変えて適切な分割基準を定めることや，分割処理によるオーバーヘッドを減らすためにチャンクファイルへの並列操作などについて検討する．

謝辞

本研究の一部は，平成 24 年度関西大学在外研究の成果である．

参考文献

[1] 田中英昭, “P2P 技術を利用した分散ファイルシステムの実装”, 電子情報通信学会技術研究報告. NS, ネットワークシステム, Vol.111, No.232, pp.99-104, (2011).

[2] STOICA Ion, “Chord: A Scalable Peer-to-peer Lookup Protocolfor Internet Apprications”, Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications. pp.149-160, (2001).