L-004

データ間の関連性を基にしたトラフィックスパイクの波及に対する予防手法の提案
A Data Relationship-based Approach to Preventing the Effect of Traffic Spikes

尾上 裕太郎† 王 家宏† 児玉 英一郎† 高田 豊雄†
Yutaro Onoue† Jiahong Wang† Eiichiro Kodama† Toyoo Takata†

1. はじめに

近年，Amazon Dynamo[1]や IIJ GIO[2]など，システムにサーバの追加，削除を行うだけで必要に応じた性能の拡張が可能な分散ストレージサービスが登場してきた．しかし，サーバの増減による性能の拡張だけでは解決できない問題が指摘されている．それは，特定のデータにユーザからのアクセスが集中するようなワークロード下において，アクセスの集中が発生したデータを保持するサーバの応答性が低下してしまうという問題[3]である．

この問題を解決する研究として，Intelligent Workload Factoring 機構[4]や，Adaptive Replication Degree 機構[5]が提案されている．前者は，アクセス集中によるシステム負荷が一定以上になると，アクセスの集中したデータを自動的にパブリッククラウドに移動させることで，後者は，ユーザからのアクセスをリアルタイムで監視し，ユーザからの急激なアクセス集中が発生した際に，アクセス集中したデータの複製を別のサーバにも配置することでこの問題を解決している．

しかし，これらの研究ではアクセス集中（トラフィックスパイク）が発生した後に，それを検知し対処を行うため，対処より先にシステムの応答性能の低下が発生してしまう可能性がある．また，スパイクの発生したデータと関連する別のデータにまでアクセスの集中が波及する可能性があるが，既存の研究では，この種の発生要因を持つトラフィックスパイクを検知することができない．そこで，本稿では，データマイニングを用い，予めアクセス集中が予測されるデータを探知しておき，探知したデータの複製（レプリカ）を別のサーバにも配置（レプリケーション）することで，トラフィックスパイク発生時におけるサーバの応答性能低下の問題を解決する手法を提案する．

2. 関連性によるアクセス集中データの予測

本提案手法は IIJ GIO や Amazon Dynamo 等 Web サーバとして運用するための大量のローカルデータを保有する分散ストレージサービスでの利用を想定している．対象サービスに対し，本手法を適用することで，アクセス負荷の分散を行い，サーバの応答性能低下の問題を解決する．

本提案手法ではトラフィックスパイクが発生したデータと関連性の高いデータにもトラフィックスパイクが発生しやすいと考え，対象サービスが保有するストレージプール内のローカルデータ間の関連性を分析する事により，アクセスが集中するデータの予測を行う．本稿ではより詳細に分析するため，関連性を因果関連性，共起関連性，時間的関連性，構造的関連性の４つに分類して考える．以下，それぞれの関連性について事例を交えて説明する．

2.1 因果関連性

因果関連性とは，「ストレージプール内の特定のデータのアクセスが上昇したら，それが原因となり，関連する他のデータにもアクセスが集中する」という考えに基づく関連性である．以下に，この関連性によるトラフィックスパイクが発生した事例を紹介する．

事例１：あるユーザがブログで企業の不正の告発を行う．それが原因となり，告発された企業のページに対してアクセスが集中した．

事例２：ある有名アーティストの死去がWebニュースで報道される．それが原因となり，そのアーティストの作品を販売しているＥＣサイトにアクセスが集中した．

2.2 共起関連性

共起関連性とは，「ストレージプール内の特定のデータへアクセスが集中した際，それと共起し，関連する他のデータへもアクセスが集中する」という考えに基づく関連性である．以下に，この関連性によるトラフィックスパイクが発生した事例を紹介する．

事例１：有名音楽ユニットが解散し，ユニットを構成するメンバー１とメンバー２のブログに対しアクセスが集中した．

事例２：原発問題が社会的ニュースになり，原発について述べている複数のブログ，ニュースサイトに対しアクセスが集中した．

2.3 時間的関連性

時間的関連性とは「特定のデータにアクセス集中が発生した一定時間後に，関連する他のデータにもアクセスが集中する」という考えに基づく関連性である．以下に，この関連性によるトラフィックスパイクが発生した事例を紹介する．

事例１：スマートフォンの新製品が発表され，その一週間後に、各キャリアの料金プランを表示するサイトにアクセスが集中した．

事例２：有名政治家が死去し，その一定時間後に，暗殺説が流布し，その話題を紹介している Web サイトにアクセスが集中した．

2.4 構造的関連性

構造的関連性は，上記３つの関連性と異なり，データの構造に基づく関連性である．以下に，この関連性によるトラフィックスパイクが発生した事例を紹介する．

事例１：スパイクが発生したデータのハイパーリンク先に対して，アクセスが集中した．

事例２：ある企業が掲載した謝罪文にアクセスが集中，その企業のフロントページにもアクセスが集中した．

3. システムモデル

図１に提案手法のシステムモデル図を示す．本システムは，クライアントからのアクセスを解析するロードバランサと，スケーラブルなサーバクラスタによるストレージプールで構成されており，ストレージプール内には対象サービスが運用する大量のデータが格納されている．

ロードバランサは，トラフィックスパイク検出を行うトラフィックスパイク検出器①と，どのレプリカを何個配置するか，レプリカをどのサーバに配置するかを決定するレプリカ配置装置②，クライアントからのアクセスログを保存しているアクセス履歴ＤＢ③，データ間の関連性の計算を行う関連性計算機④，計算された関連性を格納する関連性ＤＢ⑤の５つの要素により構成される．

†岩手県立大学 Iwate Prefectural University

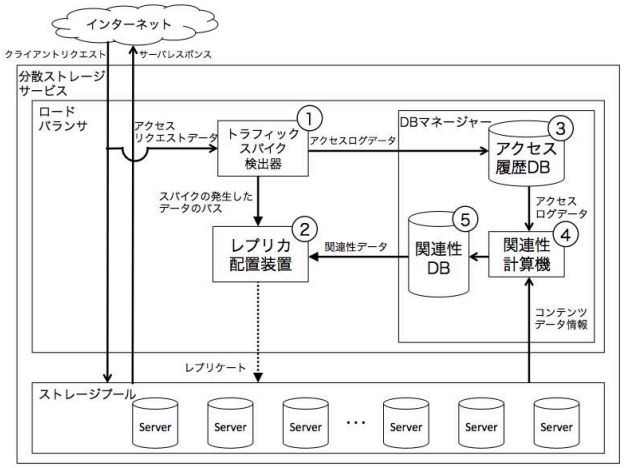

図 1：システムモデル図

以下に，本システムの動作について述べる．まず始めに，関連性によるレプリケーションを行うために必要となる関連性ＤＢの構築を行う．関連性ＤＢにはストレージプール内に保管されているデータ同士の関連性の情報が格納されている．関連性は，アクセス履歴ＤＢに保存されているアクセスのログデータと，ストレージプール内に保管されている対象サービスのコンテンツデータを用い，関連性計算機で計算することにより算出される．算出されたデータ間の関連性データを関連性ＤＢに格納することにより，関連性ＤＢを構築する．

次に，トラフィックスパイク発生時のシステムの動作について述べる．まず，ユーザからのアクセスリクエストデータをトラフィックスパイク検知器で監視する．そこで，トラフィックスパイクが観測された際，トラフィックスパイクが発生したデータのパスをレプリカ配置装置に送る．レプリカ配置装置は関連性ＤＢを参照し，トラフィックスパイクが発生したデータと関連性のあるデータを検索する．その後，各データに割り当てるレプリカ数の計算を行う．計算が終了した後，各データのレプリカをストレージプール内のどのサーバに配置するか決定し，トラフィックスパイクが発生したデータと，それと関連性のあるデータに対しレプリケーションを行う．

以上の動作により，トラフィックスパイクの発生したデータとそれに関連するデータのレプリケーションを行い，トラフィックスパイクの発生する危険性の高いデータに対してトラフィックスパイクが発生する前にレプリカを配置することが可能となる．

4. 因果関連性の抽出手法

２節では４つの関連性を挙げたが，本稿では，スペースの都合上，因果関連性の抽出手法についてのみ述べる．

アクセス履歴ＤＢに保存してある過去のユーザからのアクセスを解析することで相関ルールを発見し，因果関連性の抽出を行う．なお，アクセス履歴ＤＢにはクライアントからシステムへのアクセスログがそのまま格納されている．今回は，相関ルールの抽出にAprioriアルゴリズムを用いた．抽出手法を以下に示す．

Algorithm: 因果関連性の抽出アルゴリズム
Input: 閾値S，時間T，アクセス履歴ＤＢ
Output: 因果関連性を関連性ＤＢへ格納

1. アクセス履歴ＤＢを参照し，ストレージプールに保存されている全てのhtmlファイルへのアクセスが何回行われたかをカウントする．
2. 閾値S以上のアクセスが行われたhtmlファイルを抽出する．
3. 抽出したhtmlファイル毎に，該当htmlファイルに対して行われた全てのアクセスから，T時間以内に行われた別のhtmlファイルへのアクセスを抽出する．
4. 抽出したデータを用い，htmlのファイル名と，一定時間内に発生したhtmlファイルへのアクセスを一つのトランザクションとしてまとめ、Aprioriへの入力データを生成する．
5. 生成した入力データを用い，Aprioriアルゴリズムで相関ルールの抽出を行う．
6. 抽出した相関ルールをデータ間の因果関連性と定義して関連性ＤＢへ格納する．

5. システム構築について

上記では関連性によるレプリカ配置を提案したが，システムを構築するためには，これ以外にも，レプリカの配置場所の選択や関連性データベースの構築などを行う必要がある．

レプリカの配置は，html ファイルとその中で参照されているコンテンツデータ（画像，動画等）を一つのレプリカの単位とし，このレプリカをコネクション数が最も少ないサーバに割り振る事でアクセス分散を行う．割り振るレプリカの数については，発生したトラフィックスパイクの規模に応じ，段階的に分けることにより配置するレプリカ数を決定する．レプリカのリリースに関しては，リリース自体にもコストがかかる事と，今後も時間をおいてスパイクが発生することを考慮し，能動的に行うことはしない．ただし，後に，別のトラフィックスパイクが発生した際は，過去のレプリカデータと現在のレプリカデータを入れ替えることで受動的に削除を行う．

関連性データベースは，“レコードＩＤ”，“関連性の種類”，“データ１のパス”，“データ２のパス”を１つのレコードとして持ち，１つのレコードがデータ１とデータ２との関連性を表すものとする．なお，ストレージプール内でデータの更新がある度に，関連性データベースも更新する．

6. おわりに

本稿では，データ間の関連性を利用し，トラフィックスパイクの波及に対する予防手法の提案を行った．提案手法では，トラフィックスパイクが発生したデータと関連性の高いデータにもスパイクが発生すると仮定し，関連性の高いデータの複製を事前に別のサーバに配置するという手法をとっている．これにより，トラフィックスパイクが別のデータに波及した際にも対応でき，また，大規模なトラフィックスパイク発生時における、サーバの応答性能の低下の問題も，既存研究に比べ改善できると考えられる．しかし，トラッフィックスパイクの発生しないデータまで複製してしまう可能性や，データマイニングを利用しているために，処理にかかるコストが大きくなることも懸念される．今後は，実験により本提案手法の有効性を検証していく予定である．

参考文献

[1] G. D. Candia, D. Hastorun, M. Jampani, G. Kakulapati, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels, “Dynamo: Amazons Highly Available Key-Value Store”, In Proc. ACM SIGOPS Symposium on Operating Systems Principles, pp.205-220 (2007).
[2] IIJ GIO　http://www.iij.ad.jp/GIO/
[3] P. Bodik, A. Fox, M. J. Flanklin, M. I. Jordan and D. A. Patterson, “Characterizing, Modeling, and Generating Workload Spikes for Statefull Services”, In Proc. ACM Symposium on Cloud Computing, pp.241–252 (2010).
[4] H. Zhang, G. Jiang, K. Yoshihira, H. Chen, and A. Saxena, “Intelligent Workload Factoring for a Hybrid Cloud Computing Model”, In Proc. World Conference on Services, pp.701–708 (2009).
[5] 加藤 純, 前田 宗則, 小沢 年弘, “動的なレプリカ数調節によるデータスパイク平準化”, 情報処理学会報告, No.8, pp.1–10 (2012).