7J-01

タブレットDBとハッシュチェインを用いたイメージデータ収集システム

童　磊†　　小林　洋†
東海大学†

1．はじめに

本研究では，タブレットやスマートフォン（以降，タブレット等）を活用して画像等のイメージデータを収集し，一旦，そのリレーショナルデータベース（RDB）に保存し，回線やサーバの負荷の軽い時にタブレット等のRDBからサーバのRDBにデータを送るという形態のイメージデータ収集システムの開発を行っている．このシステムでは，データ間でハッシュチェイン(hash chain)を構成することにより，クライアントまたはサーバの一方のデータの異常が検出された際には，他方のデータを用いて回復が可能となる．本研究では，RDB間の転送方式およびハッシュチェインを用いた回復方式を提案すると共に，更には，プロトタイプシステムの開発を行っている．

2．背景

近年，センサ技術の発達により，Android や iOS のタブレット等で，画像等のイメージデータが簡単に収集できるようになって来ており，文書や音声等の記録をタブレット等で手軽にイメージデータとして記録するような状況も増えて来ている．また，タブレット等では，RDBがライブラリとして簡単に用いる事が出来るようになっている．タブレット等とサーバ間のデータのやり取りは，RDB同士であれば整合性の維持が容易なのではないかと考えられる．そこで，本研究では，タブレット等で手軽に収集した画像等のイメージデータを一旦その RDB に保存し，後でまとめてサーバのRDBに送るというイメージデータ収集システムの開発を試みることにした．このシステムの特徴は，タブレット等とサーバのRDB間のデータの整合性を保つために，データ間でハッシュチェインを構成していることである．ハッシュチェインにより，一方のデータの一部が失われた場合には，他方のデータとハッシュチェインにより，データを回復(recovery)することが可能となる．イメージデータをRDBに保存するには，blob(binary large object）形式（以降，blob 方式）で表に保存可能であるが，容量の問題からイメージデータへのリンクのみを表に保存しイメージデータ自体は RDB の外部に保存する方式（以降，link 方式）も良く行われているので，本研究では，両者を取り扱うことにする．ハッシュチェインの原理は Lamport により提案され[1]，最近では複製（replica）のある分散データベースの高信頼化のための practical Byzantine fault（PBFT）プロトコル[2]の一種である Zyzzyva[3-4]等で用いられている他，ビットコインシステム[5]のブロックチェインで P2P 向けに複雑な構成にしたものが用いられているが，本研究でのハッシュチェインは基本的なものを用いる．

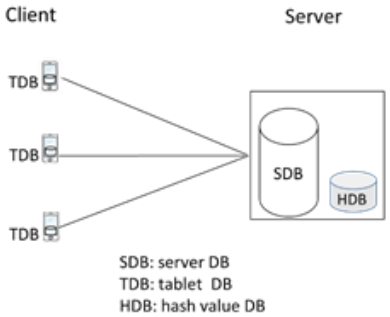

図1　システムの構成の概要

3．システムの構成と処理の概要

本システムの構成を図1に示す．本システムは，イメージデータを取得するための複数台のタブレット等からなるクライアントと，これらからイメージデータを収集するサーバで構成する．サーバ側にはハッシュチェインでのハッシュ値を保存するためのハッシュ値 DB（HDB）を設ける．HDB は，場合によっては，別ディスクとするなど，特に信頼性の

Data Collection System Using Tablet DBs and a Hash Chain Protocol

†Lei Tong, Hiromi Kobayashi・Tokai University

高いアーキテクチャとすることも考えられる．システムでの処理手順は，アプリケーションにより次の手順のように行う．

① タブレット側で写真を撮影後，撮影用フォルダからタブレット DB にイメージデータを保存する（但し，link 方式ではイメージデータは RDB の外部に保存し，そのリンクのみ RDB に格納する）．

② タブレット DB の保存データを，定期的にサーバ DB に転送する．このとき，4.で示すようなハッシュ計算の漸化式を用いてハッシュ計算を行う．

③ サーバ側で，受信したデータをサーバ DB に保存する．ここでも，漸化式のハッシュ計算を行う．

4．ハッシュチェインを用いたプロトコル

ハッシュチェインは，基本的には $h_{j,i}=H(h_{j,i-1}, d_{j,i})$ という漸化式でデータを繋いだものである．ここで，H:ハッシュ関数，$h_{j,i}$:ノード j のステップ i でのハッシュ値，$d_{j,i}$:データの要約値で，$h_{j,i}$ の系列はデータの履歴の要約値を表している．図 2 にハッシュチェインを用いた転送方式のシーケンス図を示す．$D_{j,i}$ はステップ i でクライアント j からサーバ 0 に送られる画像とその属性値からなるデータである．サーバ側では，m 個のクライアントから送られるデータの和 $D_{0,i} = \Sigma D_{j,i}$ がステップ i での増分データとなる．

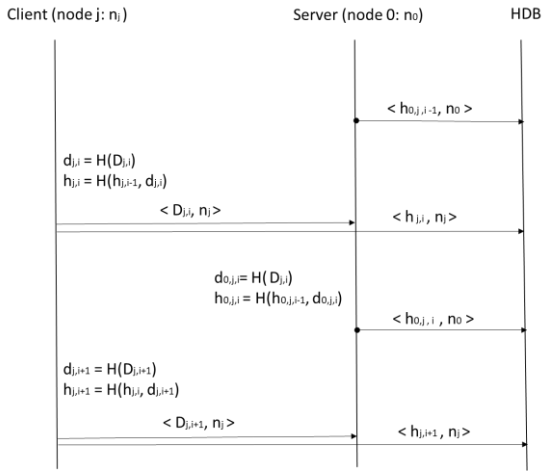

図２　データ転送方式のシーケンス図

5．回復方法

サーバで，定期的に以下のような処理を行い，異常がある場合にはクライアントから該当するデータを受け取る事により回復を行う．

① 区間[p, q]におけるクライアント j から受け取ったデータ $D_{j,p}$ から $D_{j,q}$ までのチェックを行うために，HDB から $h_{0,j,p-1}$ を受け取る．

② ハッシュ計算の漸化式を用いて p から q までハッシュ値 $h'_{0,j,q}$ の再計算を行う．

③ HDB から $h_{0,j,q}$ を受け取りハミング距離を用いてハッシュ値のマッチングを行う．値が一致した時はこの区間のチェックを終了するが，値が一致しなかった時には異常データの箇所を特定するために④へ．

④ 区間[p, q]で二分探索法を基にした方法で，最初の異常データの箇所 r を特定する．

⑤ サーバは，クライアント j から異常データの箇所 r に該当するデータを受け取り回復させる．

⑥ r の次を始点として残りの区間[r+1,q]のチェックを行うため①へ．

6．おわりに

現在，先ずは要素技術ごとの実装による処理の確認を行っている．サーバ側は MySQL と PHP，タブレット側は SQLite と Java を用い，データ転送には OKhttp3，画像データのハッシュ化には Perceptual Hash 法[6]を用いている．本研究では，次にシミュレーションプログラムの作成による転送と回復方式の検討，更には，全体の実装を行う予定である．

参考文献

[1] L. Lamport, Password Authentication with Insecure Communication, CACM 24, 11, pp. 770-772, 1981.

[2] M. Castro, and B.Liskov, Practical byzantine fault tolerance and proactive recovery, ACM Trans. on Computer Systems,20, 4, pp.398-461,2002.

[3] R. Kotla, L. Alvisi, M. Dahlin, A. Clement, and E. Wong, Zyzzyva: Speculative Byzantine Fault Tolerance, ACM Trans. on Computer Systems, 27, 4, pp. 1-39, 2009.

[4] Y. Matsumoto and H. Kobayashi, A speculative Byzantine algorithm for P2P system, Proc. IEEE 2010 PRDC, pp. 231-232, 2010.

[5] S. Nakamoto, Bitcoin: A peer-to-peer electronic cash system, https://bitcoin.org/bitcoin.pdf.

[6]N.Krawetz ,http://www.hackerfactor.com/blog/index.php?/archives/432-Looks-Like-It.html