2N5-OS-21b-3

頻出アイテム集合の即時圧縮を行う準オンライン型ストリームマイニング

Semi-Online Stream Mining Which Compresses Frequent Item Set Instantly

福田　翔士[*1]　岩沼　宏治[*2]　山本　泰生[*2]
Shoshi Fukuda　Koji Iwanuma　Yoshitaka Yamamoto

[*1]山梨大学大学院医学工学総合教育部コンピュータ・メディア工学専攻
Department of Computer Science and Media Engineering,Interdisciplinary Graduate School of Medicine and Engineering, University of Yamanashi

[*2]山梨大学大学院コンピュータ・メディア工学専攻
Department of Computer Science and Media Engineering, University of Yamanashi

Frequent item sets mining over stream data is difficult to be an on-line form for dealing with stream data because it should extract a huge number of item sets. In our study, we propose a novel semi-online algorithm that can extract frequent item sets from stream data, where it compresses frequent item sets incrementaly. Using the lossy compression proposed by Dong Xin, et al., we actualize a significant semi-online extraction and compression of frequent item sets.

1. 研究背景

データマイニングとは, 巨大なデータベースから有用な情報を高速で自動抽出する技術である. 近年では, 通信技術の向上やストレージ容量の増加により, 大量のデータが高速で流れるようになった. 例えばクレジットカードの利用履歴や通信のトラフィックなど様々なものがある. このようなデータをストリームデータという. ストリームデータをマイニングする際, リアルタイムでデータマイニングを行いたいという要求が生じる. そこで, 近年ではデータベースを一度しか読まずに, 高速に処理を行うオンライン型アルゴリズムの研究が盛んに行われている. オンライン型アルゴリズムでは無限長のデータを扱うことを仮定するため, 正確な計算を行うことは原理的に不可能であり, いかにメモリ空間消費を抑えられるかが重要課題となる. 厳密解を求めていてはこの問題の解決が困難なため, 近似解を求めて効率の良い計算を実現する戦略が一般的である.

本論文では, データマイニングの主要な課題である頻出アイテム集合マイニングをオンライン型に拡張することを考える. 頻出アイテム集合マイニングでは, 抽出される頻出アイテム集合の候補数が非常に膨大になり, 大量のメモリを要するため, オンライン型への拡張は難しいことが知られている.

本研究では, メモリ使用量の抑制を目的として, 現時点までのデータから抽出される頻出アイテム集合を随時圧縮し, その結果だけを保持してメモリ節約を行う準オンライン型アルゴリズムを提案する. 圧縮法は Xin らの非可逆圧縮手法 [1] を採用し, 大幅な圧縮効果を目指す.

2. 準備

本稿で用いる表記法と用語の定義を行い, その後, Xin らの非可逆圧縮法 [1] について述べる.

定義 1 全てのアイテムの集合を $I = \{i_1, i_2, \cdots, i_n\}$ としたとき, I の部分集合を**アイテム集合**とよぶ. また, アイテム集合を適宜, トランザクションとも呼ぶ. トランザクションデータベース *(*以下, *TDD* と略す*)* とはトランザクションの集合である. トランザクションの無限列 $t_1, t_2, t_3, \cdots$ をストリームデータと呼ぶ.

連絡先: 福田翔士, 山梨大学大学院医学工学総合教育部修士課程コンピュータ・メディア工学専攻, g13mk021@yamanashi.ac.jp

例 1 *TDD* の例を示す. 図 *1* の例では, アイテムは A, B, C, D, E の 5 種類であり, トランザクションの集合は *6* 個である.

TID	Items
t_1	{A, B, C, D}
t_2	{A, B, C}
t_3	{A, B, C, E}
t_4	{A, B, C, D, E}
t_5	{A, B, E}
t_6	{B, C, D}

4つ以上に含まれるアイテム集合

{A} {B} {C}
{A, B} {A, C} {B, C}
{A, B, C}

図 1: TDD と頻出アイテム集合の例

定義 2 T を *TDD*, P をアイテム集合とするとき, $T(P)$ を P が出現する T 中のトランザクションの集合とする. このとき, P の頻度 $sup(P)$ は $sup(P) = |T(P)|$ と定める. **最小頻度**とよばれる閾値 θ 以上の頻度をもつアイテム集合を**頻出アイテム集合**と定義する.

図 *1* の右側に示したアイテム集合はどれも少なくとも *4* つ以上のトランザクションに含まれるため, 最小頻度 *4* に対する頻出アイテム集合である.

定義 3 T を *TDD* とするとき, $\|T\|$ を T 中のトランザクションの総数とする. このときアイテム集合 P の**相対頻度** *R-sup(P)* を以下のように定める.

$$R - sup\,(P) = \frac{sup\,(P)}{|T|}$$

Xin らの PRglobal あるいは RPlocal 法 [1] は, 頻出アイテム集合マイニングにおける非可逆圧縮技術であり, TDD から代表パターンのみを抽出する.

Xin らの手法では，アイテム集合間の距離尺度による誤差保証を行いながら，代表パターンの集合を抽出する

定義 4 *TDD* に存在する，アイテム集合 P_1 および P_2 の距離を以下の式で定義する.

$$D(P_1, P_2) = 1 - \frac{|T(P_1) \cap T(P_2)|}{|T(P_1) \cup T(P_2)|}$$

この距離尺度は，パターン間の距離を出現トランザクション集合の間の *Jaccard* 距離である.

例 2 図 *1* の *TDD* を考え，$P_1 = \{A, B\}, P_2 = \{B, C\}$ であるとする．このとき，$T(P_1) = \{t_1, t_2, t_3, t_4, t_5\}$，$T(P_2) = \{t_1, t_2, t_3, t_4, t_6\}$ であるので，P_1 および P_2 の距離は，$D(P_1, P_2) = 1 - \frac{4}{6} = \frac{1}{3}$ である.

定義 5 アイテム集合 P が別のアイテム集合 P' を δ カバーするとは, $P \supseteq P'$, $D(P, P') \leq \delta$ という条件が成り立つことである. δ *(0 ≤ δ ≤ 1)* はクラスタの堅固さの指標である．アイテム集合 P がアイテム集合の集合 S の代表パターンであるとは, P が S の各元 Q_i を δ カバーする場合をいう．S が δ **クラスタ**を成すとは，S の代表パターン P_r が S に存在することである.

Xin のアルゴリズムは, 頻出（飽和）アイテム集合の集合 S が与えられたとき, $S = s_1 \cup s_2 \cdots s_m$ となる δ クラスタとそれぞれの代表パターン $P_{r1}, P_{r2} \cdots P_{rm}$ を近似計算するものである．代表パターン $P_{r1} \cdots P_{rm}$ の算出問題は NP-hard であり大変難しい．Xin らの RPglobal 及び RPlocal は一定の誤差保証の下で近似計算する.

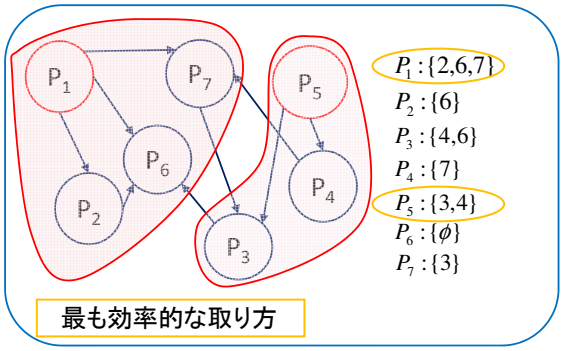

図 2: 代表パターンの抽出例

例 3 図 *2* に代表パターンの抽出例を示す．P_1～P_7 は頻出飽和パターンである．各パターンのカバー関係を矢印の向きで表している．例えば，P_1 は P_2，P_6，P_7 を δ カバーしている．このカバー情報はグラフの右側に示している．代表パターンとして，様々な抽出例が考えられるが，この例では P_1 と P_5 を代表パターンとして抽出することで，代表パターンの数が最小になる.

3. 提案手法

本研究は, Xin らの非可逆圧縮法を拡張し, ストリームデータに対して, 受け取ったトランザクションデータを即座に圧縮を行うオンライン型アルゴリズムを目指す．しかし, オフライン型である Xin らの手法を完全なオンラインに単純に拡張することは困難である．そこで, 図 3 のような準オンライン型のアルゴリズムを提案する.

まず, データストリームをメモリに読み込む．一定量のデータストリーム（以下, ブロックと呼ぶ）をメモリに保存した後, このブロックに対して, オフライン処理でデータ圧縮を行い, その後, データストリームの読み込みを続ける．圧縮には Xin らの非可逆圧縮を行い, メモリに保存されているデータを代表パターンの集合に圧縮し, その代表パターンの集合を過去の情報として保持していく．このような処理を繰り返すことで, メモリ消費を一定量に抑制し, データストリームから頻出アイテム集合の代表パターンを随時抽出する.

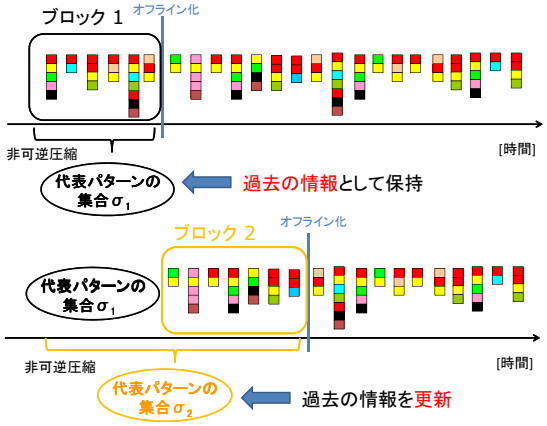

図 3: 提案手法の動作

図 3 は, 提案手法の動作を表したものである．図中のブロックとは, 前回の圧縮後から今まで読み込んだデータのことを指す．図中の上部は, データストリーム上で, 初めて非可逆圧縮により代表パターンの集合を求める様子を示している．下部は, 前回のブロックで抽出した代表パターンと現在のブロックから新たな代表パターンの集合を求める様子を示している.

提案手法は, 部分的なオフライン処理で圧縮を行う．全体の処理を通してみればオンライン型アルゴリズムであるが, 完全なオンライン処理ではないため, **準オンライン型**と呼んでいる．以下に, 飽和アイテム集合についての説明を行い, 次に, 見積もり頻度の定義を示す．その後, 提案手法のアルゴリズムについて述べる.

3.1 飽和アイテム集合

飽和アイテム集合とは, あるアイテム集合 P において, $P \subset P'$ かつ $sup(P) = sup(P')$ となる集合 P' が存在しないとき, P は飽和アイテム集合と呼ぶ．飽和アイテム集合は可逆圧縮法の 1 つである．本研究において, 飽和アイテム集合とは, あるアイテム集合を $\delta = 0$ で δ カバーする代表パターンのこととなる.

一度候補アイテム集合を飽和アイテム集合に可逆圧縮を行った後, さらに Xin らの非可逆圧縮を行い代表パターンを抽出する 2 段階の圧縮を検討する．これにより, 代表パターンの候補集合の数を削減し, 提案手法の処理速度を速めることを目指す.

3.2 見積もり頻度

Xin らの非可逆圧縮を行う上で, 各アイテム集合の δ カバー関係は頻度情報から求める. このとき, 必要な頻度とは, データの読み込みを始めてから現時刻までの頻度である. 過去の情報は, 圧縮結果である代表パターンしか保持していない. したがって, アイテム集合の過去の完全な頻度情報を知ることはできない.

しかし, 代表パターンから過去の出現で考えられる最大の頻度を求めることはできる. この過去の最大頻度と現 Block の実頻度を加算した頻度を**見積もり頻度**とし, 見積もり頻度から δ カバー関係を調べる. 見積もり頻度は, 過去の最大頻度を考えているため, 過去から現在までの真の頻度より必ず大きくなる. そのため, 実際には非頻出であったアイテム集合が, 見積もり頻度では頻出となり, 誤って検出される場合がある. しかし, ここで重要なことは, 元々頻出であるアイテム集合をすべて検出する完全性である. 見積もり頻度を用いれば, 完全性を保証できる. 以上のことから, 以下では見積もり頻度で処理を行っていく.

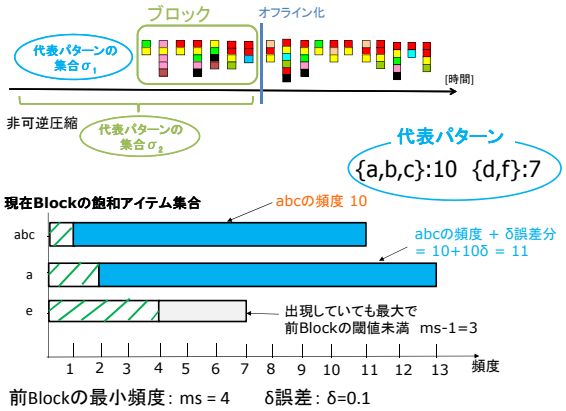

図 4: 見積もり頻度の導出

図 4 は, 見積もり頻度の導出の一例である. 現ブロックで新しく 3 つの飽和アイテム集合 $\{a\}$, $\{e\}$, $\{a,b,c\}$ が, それぞれ頻度 $2,4,1$ で保存されたとする. 過去の情報として, 代表パターンとして $\{a,b,c\}$ が頻度 10, $\{d,f\}$ が頻度 7 として保存されていたときの, $\{a\}\{e\}\{a,b,c\}$ の見積もり頻度の導出を示している. 斜線部分が飽和アイテム集合の現 ブロックでの実頻度である. $\{a,b,c\}$ はそれ自身が代表パターンとして存在しているため, 過去の情報が正確に残っている. したがって, 代表パターン $\{a,b,c\}$ の頻度をそのまま加算し, 見積もり頻度とする. $\{a\}$ と代表パターンを比較すると, 代表パターン $\{a,b,c\}$ の部分集合となっており, また, その他に $\{a\}$ を含む代表パターンは無い. したがって, $\{a\}$ は存在していたが, $\{a,b,c\}$ に圧縮されたと考えられる. したがって, $\{a\}$ の過去の最大頻度は代表パターン $\{a,b,c\}$ の頻度に δ 誤差を加算したものである. $\{e\}$ は代表パターンにカバーされているものがない. このことから, $\{e\}$ は, 前ブロックでの最小頻度を満たさない頻度しかなく, 削除したと考えられる. したがって, $\{e\}$ の見積もり頻度は現ブロックでの実頻度に前ブロックの最小頻度から 1 引いた値を加算したものである. 以上のような, 見積もり頻度の導出を考える.

3.3 提案手法のアルゴリズム

データストリームに対して, 代表パターンの集合を抽出する提案アルゴリズムを以下に示す. 飽和アイテム集合の抽出には, 宇野らの LCM[2] を採用する.

入力：データストリーム, 相対最小頻度 σ, 誤差 δ,
ブロックデータの大きさ T
出力：代表パターンの集合

変数:
アイテム集合の頻度, δ カバーの情報を保存する配列（以下, 頻度表とする） ST
前ブロックで抽出した代表パターンの集合を保存する配列 RP
一時的にアイテム集合を保存する配列 $temp$
アイテム集合 P
読み込んだトランザクションの総数 n
現ブロックでの最小頻度 ms
前ブロックでの最小頻度 $prems$

変数の初期化:
$ST = \emptyset$
RP に前ブロックでの代表パターンを登録
$temp = \emptyset$
$P = 0$

手続き:
ユーザの出力要求があるまで, 以下の処理を繰り返す.
データストリームをメモリ上に保存する.
読み込んだトランザクション数が T を超えた場合, ブロックデータに対して, 以下の圧縮処理を行う.

1. temp = 現ブロック中の飽和アイテム集合の集合
2. データストリームの読み込み開始から現ブロックまでの飽和アイテム集合を求める
   (a) ST に RP を登録
   (b) $ms = n \cdot \sigma$
   (c) $prems = (n - T)\sigma$
   (d) RP を頻度に対して, 降順にソートする
   (e) for temp $\neq \emptyset$ //以下の処理より, 現ブロックまでの飽和アイテム集合を求める
      i. temp からアイテム集合を 1 つ取り出し, P に保存
      ii. for $i = 0; i < n(RP); i++$ //$n(RP)$ とは配列 RP の要素数
         A. if RP[i] $\subset$ P かつ ST 中の $sup(RP[i])$ が $sup(RP[i]) \leq sup(RP[i]) + sup(P)$
            ST 中で $sup(RP[i])$ += $sup(P)$ //$sup()$ とは, 括弧内に入る集合の頻度を表す
         B. if $RP[i] == P$
            ST 中で $sup(RP[i])+ = sup(P)$ //見積もり頻度を求める
            break

C. else if $RP[i] \supset P$ かつ ST 中に P が存在しない
ST に P を登録
ST 中で $sup(P)$ += $sup(RP[i])$*(1+δ)
//見積もり頻度を求める

D. else if $RP[i]$ と P に共通部分 P' が存在する かつ ST 中に P' が存在しない
S に P' を登録
ST 中で $sup(P')$ += $sup(RP[i])$*(1+δ)
//見積もり頻度を求める

iii. if P が ST に登録されなかった または ST 中に既に登録されているものよりも P の見積もり頻度のほうが大きい

A. ST に P を登録

B. ST 中の sup(P) += $prems$ //見積もり頻度を求める

(f) 頻出飽和アイテム集合を求める

i. 見積もり頻度が, ms 未満のアイテム集合を ST 中から削除

3. ST 中のアイテム集合間の δ カバー関係を調べる

4. 代表パターンを貪欲法で近似計算する

(a) 最も多くの集合を δ カバーする集合を代表パターンとして抽出する

(b) 代表パターンが δ カバーするアイテム集合の情報を, ST 中の集合が δ カバーする集合の情報から削除する

(c) 代表パターンと δ カバーする集合が存在しない集合を, ST 中から削除する

(d) 候補集合が空になるまで, 上記 3 つを繰り返す

圧縮処理が終了したら, ブロックデータを破棄する.

提案手法のアルゴリズム

4. 予備実験とその考察

アイテムの種類数 12 個, トランザクション数 10,000, 1 トランザクションあたりアイテムは最大でも 10 個, という条件の下, データセットをランダムで作成した. そのデータセットに対して, 頻出アイテム集合, 頻出飽和アイテム集合, Xin らの非可逆圧縮により抽出した代表パターンの集合, 提案手法により抽出した代表パターンの集合を求め, それらの比較を行う予備実験を行った.

トランザクション数 10,000, アイテム総数 55,350, 1 トランザクションあたり平均 5.5 個のアイテムを含むデータセットに対して, 相対最小頻度 $\sigma = 0.05$, δ 誤差 $\delta = 0.4$ の設定の下, 予備実験を行った.

予備実験の結果を以下の表に示す. このとき, ブロック長とは, 提案手法における, 一度に読み込むトランザクションの数である.

表 1 より, データセットから 1585 個の頻出アイテム集合が抽出されていることが見て取れる. 次に, このデータセットに対して, 頻出飽和アイテム集合の抽出を行うと, 1585 個のアイテム集合が抽出される. 頻出アイテム集合の数と比較すると,

表 1: 実験結果

手法 \ ブロック長	500	1000	5000	10000
頻出アイテム集合				1585
頻出飽和アイテム集合				1585
Xin らの研究				530
提案手法	786	878	897	896

そのアイテム数は変化していないことから, このデータセットでは, 飽和アイテム性による可逆圧縮の効果はないことが見て取れる. Xin らの非可逆圧縮を行うと, 抽出される代表パターンの数は 530 個であった. 頻出飽和アイテム集合のみの抽出ではアイテム集合の数を削減することができなかったことに対し, その数を 3 分の 1 に削減している. 最後に提案手法の抽出結果では, どのブロック長でも頻出アイテム集合の数を下回る数の代表パターンを抽出していることが見受けられる.

4.1 考察

頻出アイテム集合を準オンライン型処理で非可逆圧縮することで, その数を大幅に削減することができることを示した. 提案手法の実験結果は, Xin らのオフライン型非可逆圧縮により抽出した代表パターンの数を上回る代表パターンを抽出しているが, 頻出アイテム集合の数を削減することはできている.

5. まとめと今後の課題

オンライン型頻出アイテム集合マイニングでは, 抽出される膨大な数の頻出アイテム集合を, どのような手法を用いて縮約していくかが, 重要な課題である. 本研究では, 抽出される頻出アイテム集合の候補を, 非可逆圧縮することで縮約をするアプローチを提案した. 提案手法の素朴な実装は完了したが, 処理に多大な時間を要している. 今後の課題は, 本格的なシステムの実装を行い, 計算時間等に関する評価等を行うことである.

謝辞

本研究は一部，文科省科学研究費補助金（基盤 C：No.22500127 および No.25330256）の援助を受けている.

参考文献

[1] Xin, D., Han, J., Yan, X. and Cheng, H. : On compressing frequent patterns. Data & Knowledge Engineering 60, pp.5-29(2007).

[2] Uno, T., Kiyomi, M. and Arimura, H. : LCM ver.3: Collaboration of Array, Bitmap and Prefix Tree for Frequent Itemset Mining. OSDM'05, pp.77-86(2005).

[3] Illimine system package. http://illimine.cs.uiuc.edu/download/.

[4] Manku, G.S and Motwani, R. : Approximate frequency counts over data streams. Proc. VLDM'02, pp346-357(2002).

[5] 伊藤秀志，岩沼宏治，山本泰生：バースト出現へ対応を目的としたオンライン型系列マイニングへのメモリ制限の導入. 人工知能学会創立 25 周年記念合同研究会予稿集, pp.(2-42)-(2-49)(2011).