1L-04

ZDDによるパターン頻度表を用いた頻出アイテム集合マイニングの追加データ処理手法

周文爍† 川原純† 湊真一†

京都大学大学院情報学研究科†

１．はじめに

LCM法 [1]は，頻出アイテム集合マイニングアルゴリズムの一種であり，頻出パターン数に対して線形時間で実行できることが知られている．さらに，ZDD（ゼロサプレス型二分決定グラフ）[2]と呼ばれるデータ構造を組み合わせると，LCM-ZDD[3]というさらに高速なアルゴリズムが得られる．LCM法では，入力データベースが更新されて少量のデータが追加された時に，差分だけを高速に処理する良い方法はこれまでになく，元データ件数が大きく追加データ件数が小さい場合でも，全体を再実行しなければならないという課題があった．

本研究では，LCM 法の 3 つのオプションの中から，全ての頻出パターンを列挙するLCMFreqアルゴリズムに着目し，LCM-ZDDを用いて生成したZDDによるパターン頻度表を用いて追加データ処理に対応できるIncr-LCMFreq-ZDD法を提案する．データマイニングのベンチマークデータとして使用されるFIMI [4]を用いた実験でIncr-LCMFreq-ZDD法の正確性を確認し，効率を測定する．

２．準備

$I = \{1,2,\ldots,n\}$をアイテムの集合，$\boldsymbol{\mathcal{T}}$をI上のトランザクション（Iの部分集合）からなるデータベースとする．Iのパターン（部分集合）Pに対して，Pを含む$\boldsymbol{\mathcal{T}}$のトランザクションをPの出現とよぶ．Pの出現の集合を $T(P) = \{T \mid P \subseteq T,\ T \in \boldsymbol{\mathcal{T}}\}$と表記し，$P$の出現集合とよぶ．$P$の出現集合の大きさを$P$の頻度といい，$\mathrm{freq}(P)$と表記する．与えられた閾値$\theta$に対して，$\mathrm{freq}(P) \geq \theta$を満たす$P$は頻出パターンと呼ばれる．頻度表は，与えられたトランザクション集合の中に，各パターンが何回出現したかを，パターンごとに数えて表にしたものである．

本研究では，頻出パターンを列挙するだけでなく，どのパターンが何回出現するかを頻度表として出力することを考える．元のデータでパターンの頻度表を求め，少量のデータが後から追加された時に，追加後の頻出パターンの頻度表を効率良く求めることが，本研究の主要な課題である．

３．ZDDを用いたパターン頻度表の表現

ZDD [2]は，大規模な組合せ集合を効率的に表す有向非巡回グラフによるデータ構造である．ZDDを使うことにより，組合せ集合を圧縮することができ，多くの場合に効率良く集合演算を実行できる．

文献 [5]ではZDDを用いたパターン頻度表の表現が提案されている．頻度を 2 進数で表し，2 進数の桁ごとにZDDを割り当てる．割り当てられたZDDについて，各組合せ集合の頻度の 2 進数の桁が1(true)であれば，その組合せ集合を含むとする．そうでなければ含まないとする．

追加データを頻度表に反映させるには，トランザクションごとに2進数に1を加えるインクリメント演算に相当する集合演算を繰り返し行うことで実現できる（以下では逐次加算法と呼ぶ）．または，追加データをまとめて LCM-ZDD 法を実行して ZDDによる頻度表を作り，2 進数の加算アルゴリズムに相当する集合演算を用いて，2 つの頻度表を併合する手法も考えられる（以下では併合法と呼ぶ）．

４．提案手法

まずはγ緩和-頻出パターンを定義する．ここでは，追加データのトランザクション件数の最大値γをあらかじめ想定できるものとする．与えられた閾値θに対して$\mathrm{freq}(P) \geq \theta - \gamma$を満たす$P$を$\gamma$緩和-頻出パターンと呼ぶ．通常の頻出パターンは必ずγ緩和-頻出パターンでもある．

データが追加される前の元データに対する 1 回目の実行では，LCM-ZDD 法で頻出パターンを全列挙するZDDを求めて出力すると同時に，γ緩和-頻出パターンのZDD頻度表を生成して保存しておく．頻出パターンの列挙と，γ緩和-頻出パターンの頻度表の生成は，LCM-ZDD 法の再帰呼び出しアルゴリズムの中で同時に行うことが可能で，その計算時間は$\theta - \gamma$を閾値として頻度表を構築する時間と大差ないと見込まれる．

追加データが入った後の実行では，ZDDによる頻度表に使用する演算に応じて，併合法と逐次加算法

A Method of Additional Data Processing for Frequent Itemset Mining with ZDD-Based Pattern Histograms

† Graduate School of Informatics, Kyoto University

表１　実験結果

Database	アイテム数	データ件数	アイテム出現率	元データ件数	最小頻度 θ（出現率）	頻出パターン数	従来手法 (msec)	併合法 (msec)	逐次加算法 (msec)
accidents	468	340,183	0.07	329,200	48%	10333	2202	139856	391289
chess	75	3,196	49.33	3,000	80%	11238	117	64	65
connect	129	67,557	33.33	65,300	92%	15020	298	1156	347
kosarak	41,270	990,002	0.02	958,000	0.23%	11343	2267	メモリオーバー	
mushroom	119	8,124	19.33	7,800	23%	13440	47	34	30
pumsb*	2,088	49,046	2.42	47,400	42%	14392	263	4036	458
pumsb	2,113	49,046	3.50	47,400	86%	13070	438	51437	1030
retail	16,469	88,162	0.06	85,300	0.08%	10496	1367	428045	398101

表２　chess での実験結果の詳細

追加データ件数(元データとの比)	従来手法 (msec)	併合法 (msec)	逐次加算法 (msec)
3 (960:1)	117 (58+59)	64 (62+2)	65 (62+3)
8 (360:1)	168 (84+84)	103 (94+9)	99 (94+5)
16 (180:1)	142 (69+73)	225 (84+141)	111 (84+27)
33 (90:1)	153 (76+77)	504 (138+366)	233 (138+95)

表３　mushroom での実験結果の詳細

追加データ件数(元データとの比)	従来手法 (msec)	併合法 (msec)	逐次加算法 (msec)
8 (960:1)	47 (24+23)	34 (28+6)	30 (28+2)
21 (360:1)	50 (24+26)	45 (24+21)	32 (24+8)
34 (180:1)	60 (30+30)	109 (30+79)	65 (30+35)
86 (90:1)	82 (31+51)	165 (34+131)	144 (34+110)

がある．いずれの方法でも最終結果は同じであり，データを追加した後のデータベースに対する頻出パターンの頻度表が得られる．

５．実験

SAPPOROBDDパッケージ[1]を用いて提案手法と従来手法を比較した．実験では Apple M1 MacBook（主記憶 8GB）を使用した．実験の例題として FIMI ベンチマークデータ [4]を使用するが，このベンチマークでは元データと追加データの設定はないので，トランザクションのリストを後で示す比率で元データと追加データに分けて実験を行った．

表 1 に実験結果を示す．ここでは元データと追加データの件数の比を 960:1 としている．最小頻度 θ の設定は頻出パターン数がおよそ 10000 になるように調整して実験を行った．追加データに対して差分を用いずに LCM-ZDD 法を単純に再実行した場合（従来手法），提案手法で併合法を使って頻度表を更新した場合（併合法），および逐次加算法を使って頻度表を更新した場合（逐次加算法）の 3 通りの実行時間（msec）を示す．全ての場合について最終結果は同じ頻度表となる．なお，実行時間は，入力時間＋探索時間を計測した. 出力は ZDD としてメモリに保存されているので，出力時間は含まない．表１に示す実験結果では，アイテム出現率が高い例題 chess と mushroom では提案手法が有効だった. また，アイテム数が多くアイテム出現率が低い例題では，提案手法の実行時間は増大する傾向があり，例題 kosarak ではメモリオーバーになった．

表 2, 表 3 で，例題 chess と mushroom での実験結果の詳細を示す．最小頻度 θ は表 1 と同じ値とし，元データと追加データの件数の比率を 960:1 から 90:1 までの間で，提案手法が従来手法より大幅に遅くなるまで追加データ件数を増やした時の計算時間を調べた．括弧の中の数値は，それぞれ元データでの実行時間とデータを追加後の実行時間である．

６．おわりに

本研究では, 追加データに対応する Incr-LCMFreq-ZDD 法を提案し，実験で効率を評価した．提案手法では元データでの実行で頻出パターンだけではなく，γ緩和-頻出パターンを求めて保存しておく必要がある．γ緩和-頻出パターン数が頻出パターンより多いため，元データでの実行では従来手法よりやや計算時間を要する．追加データの比率が小さい場合は，元データの頻出パターン数とγ緩和-頻出パターン数の差が小さくなるため，提案手法が有利になると考えられる．

本研究の一部は，JSPS 科研費 JP20H00605, JP20H05794, JP20H05964 の助成を受けたものです．

参考文献

[1] T. Uno, T. Asai, Y. Uchida, H. Arimura, "LCM: An Efficient Algorithm for Enumerating Frequent Closed Item Sets," In Proc. of FIMI, 2003.
[2] S. Minato, "Zero-Suppressed BDDs for Set Manipulation in Combinatorial Problems," In Proc. of DAC, pp. 272-277, 1993.
[3] S. Minato, T. Uno, H. Arimura, "LCM over ZBDDs: Fast Generation of Very Large-Scale Frequent Itemsets Using a Compact Graph-Based Representation," In Proc. of PAKDD, pp. 234-246, 2008.
[4] B. Goethals, M.J. Zaki, Frequent itemset mining dataset repository.In Proc. of FIMI, 2003, http://fimi.uantwerpen.be/data/.
[5] 湊真一, 有村博紀, 「ゼロサプレス型二分決定グラフを用いたトランザクションデータベースの効率的解析手法:データ工学論文特集」, 電子情報通信学会論文誌 D, 情報・システム, J89(2): 172-182, 2006.

[1] https://github.com/Shin-ichi-Minato/SAPPOROBDD