4A1-2in2

トランザクションストリーム上の頻出飽和アイテム集合系列の抽出に関する基礎的考察

A Fundamental Study on Mining Frequent Closed Sequential Patterns from a Transaction Stream

仁科 拓巳 *1 Takumi Nishina　山本 泰生 *2 Yoshitaka Yamamoto　岩沼 宏治 *2 Koji Iwanuma

*1山梨大学大学院医工農学総合教育部工学専攻コンピュータ理工学コース
Computer Science and Engineering Course, Integrated Graduate School of Medicine, Engineering and Agricultural Sciences, University of Yamanashi

*2山梨大学大学院総合研究部
Graduate Faculty of Interdisciplinary Research, University of Yamanashi

This paper studies an online mining of frequent closed itemset sequence from a transactional stream (FCSM-TS). In order to avoid huge amounts of calculation for mining closed itemset sequences from a stream, we propose a fast algorithm using triple indexing mechanisms. The first indexing is called *IDList*, which records, for each item, a list of sequence identifiers such that the item appears in the last itemset of the sequence. The second is a bucket-based indexing that distributes each closed sequence into a bucket according to its frequency count. The third is an ordinary hash table where each closed sequence is used as a hash key. In this paper, we give some experimental study and also show several results of some evaluation. In addition, we also investigate a new essential difficulty of mining closed sequences from a stream, which never occurs in the case of mining closed itemsets.

1. はじめに

近年，Web ニュースや掲示板，ブログなど膨大なデータが存在し，無限に増加し続けている．このように，常に高速に流れ続けるデータを**ストリームデータ**という．

ストリームデータの収集が行われるようになったことで，ストリームデータから有用な情報を高速に抽出する技術が盛んに研究されている．先行研究では，既に求められている左飽和集合系列と新規に到着したウィンドウとの左極大共通系列を求めることで，効率よく頻出アイテム集合系列を抽出する手法が提案された [山本 16]．しかし，左極大共通系列の抽出には多大な計算コストがかかる．

そこで本研究では頻度表中の左飽和集合系列を末尾に含まれるアイテムごとに索引化した IDList と見積もり頻度ごとに索引化したバケット機構等を用いる三重索引を新たに構築し，効率的に左飽和集合系列を求める計算手法を提案する．また，左飽和集合系列の抽出に関する考察を述べる．

2. 準備

2.1 系列

$I = \{i_1, i_2, \cdots, i_n\}$ を**アイテム**の全体集合とするとき，I の部分集合 $X(X \subseteq I)$ を**アイテム集合**と定め，$X = (i_1 i_2 \cdots i_m)$ と記述する．**系列** α はアイテム集合 $s_j \subseteq I$ の列であり $\langle s_1 s_2 \cdots s_n \rangle$ と記述する．2 つの系列 $\alpha = \langle s_1 s_2 \cdots s_n \rangle$，$\beta = \langle t_1 t_2 \cdots t_m \rangle$ に対し，$s_1 \subseteq t_{j_1}, s_2 \subseteq t_{j_2}, \cdots, s_n \subseteq t_{j_n}, (1 \leq j_1 < j_2 < \cdots < j_n = m)$ を満たすとき α を β の**右部分系列**，β を α の**左拡大系列**と呼び，$\alpha \preceq \beta$ と記述する．また，他の系列が β を右部分系列として真に含まないとき，β を**左極大系列**と呼ぶ．例えば，系列 $\alpha = \langle (a)(bc) \rangle$, $\beta = \langle (d)(a)(bc) \rangle$ としたとき，α は β の右部分系列，β は α の左拡大系列である．一方で，$\gamma = \langle (a)(bc)(b) \rangle$ に対して α の末尾集合 (bc) が γ の末尾集合 (b) に含まれないため，α は γ の右部分系列ではない．

連絡先: 山梨大学大学院工学専攻 (修士) コンピュータ理工学コース, 〒 400-8510 山梨県甲府市武田 4-3-11,
E-mail: t13cs042@yamanashi.ac.jp

2.2 トランザクション

単位時間内に到着するアイテム集合を**トランザクション**と呼び，トランザクションの列 $\langle t_1 t_2 \cdots t_n \rangle$ を**トランザクションストリーム**と呼ぶ．抽出する右部分系列の最大の長さを k としたとき，時刻 i における**ウィンドウ** $\mathrm{win}(i)$ を $\langle t_{\mathrm{e}(i)} \cdots t_i \rangle$ と定義する．ただし $\mathrm{e}(i)$ は以下のように定義する．

$$\mathrm{e}(i) = \begin{cases} i-k+1 & if\ i \geq k \\ 1 & otherwise \end{cases}$$

2.3 頻度

本稿では，系列の頻度を求める尺度として**系列末尾頻度**を利用する [Iwanuma 05]．系列末尾頻度とは，頻度の重複数え上げを行わず，左拡大系列に対して逆単調性を持つ尺度である．

ストリーム $S = \langle t_1 t_2 \cdots t_n \rangle$，系列 $\alpha = \langle s_1 s_2 \cdots s_m \rangle$ としたとき，時刻 n における α の頻度 $\sup(\alpha, n)$ を以下のように定義する．

$$\sup(\alpha, n) = \sum_{i=1}^{n} \mathrm{include}(\mathrm{win}(i), \alpha)$$

ただし，$\mathrm{include}(\mathrm{win}(i), \alpha)$ は以下のように求められる．

$$\begin{cases} 1 & if\ \alpha \text{ が } \mathrm{win}(i) \text{ の右部分系列} \\ 0 & otherwise \end{cases}$$

2.4 左飽和集合系列

$\alpha \neq \beta$ かつ $\alpha \preceq \beta$, $\sup(\alpha, n) = \sup(\beta, n)$ となる系列 β が存在しないとき系列 α を**左飽和集合系列**と呼ぶ．また，α と β の**右共通系列** γ を $\gamma \preceq \alpha$ かつ $\gamma \preceq \beta$ となる右部分系列と定め，右共通系列の中で左極大な系列を**左極大共通系列**と呼ぶ．一般に α と β の左極大共通系列は複数あるので，その集合を $\mathrm{int}(\alpha, \beta)$ と表記する．

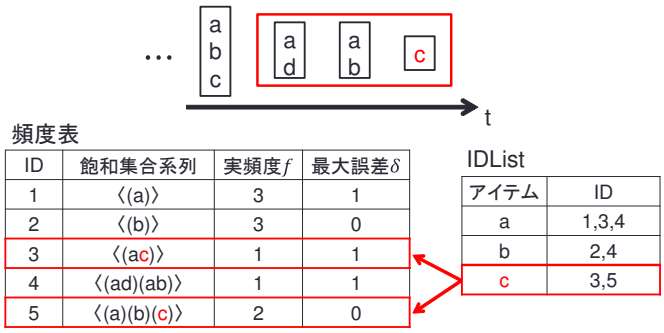

図 1: 頻度表と IDList

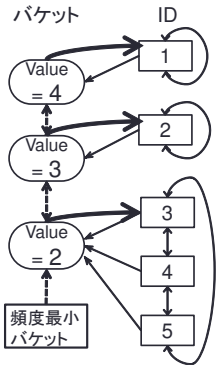

図 2: バケット機構

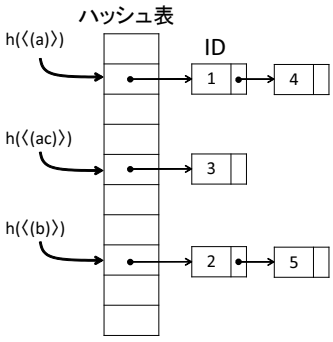

図 3: ハッシュ表

3. 既存手法

トランザクションストリームから全ての頻出なアイテム集合系列を素朴に求めることは，組み合わせ爆発により困難である．例えば，平均トランザクション長を k，ウィンドウ幅を l としたとき，アイテム集合系列の組み合わせは $(2^k)^l - 1$ となる．そこで先行研究 [山本 16] では，左飽和集合系列のみを求めることで組み合わせ爆発の抑制を行っている．左飽和集合系列はアイテム集合系列の圧縮表現であり，系列末尾頻度の逆単調性により全てのアイテム集合系列とその頻度値が復元可能である．また，頻度の少ない系列を積極的に消去する近似計算 [Manku 03][Metwally 05] を行うことで，プログラム内部の頻度記録の表の爆発的増加を防いでいる．

3.1 左飽和集合系列の抽出

先行研究 [山本 16] では，飽和アイテム集合を求める漸近的交差法 [Borgelt 11] を左飽和集合系列の抽出に拡張している．左飽和集合系列を保持する表 TS を**頻度表**と呼ぶ．頻度表中の左飽和集合系列とウィンドウとの右共通系列を求めることで，新規に登録される左飽和集合系列を求めている．時刻 i の頻度表 $TS(i)$ を求める漸化式は以下のように定めている．

(1) $TS(1) = \mathrm{win}(1)$

(2) $TS(i+1) = TS(i) \cup \{\mathrm{win}(i+1)\} \cup$
$\{\beta \mid \beta \in \mathrm{int}(\alpha, \mathrm{win}(i+1)), \alpha \in TS(i)\}$

左飽和集合系列となるのは右共通系列の中で，左極大なものだけである．先行研究 [山本 16] では，頻度表中の左飽和集合系列と新規ウィンドウの各アイテム集合の共通部分を表にした共通部分表を作成し，左極大共通系列となる経路を探索することで，効率よく左極大共通系列のみを求める手法を提案している．

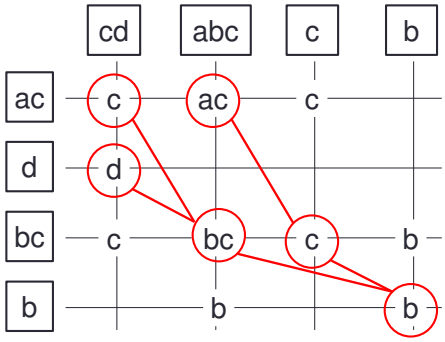

図 4: 共通部分表の例

$\alpha = \langle(ac)(d)(bc)(b)\rangle, \mathrm{win}(i+1) = \langle(cd)(abc)(c)(b)\rangle$ としたときの共通部分表の例を図 4 に示す．このとき，表中の赤線で示す経路が左極大共通系列であり，$\langle(b)(c)(ac)\rangle$, $\langle(b)(bc)(d)\rangle$, $\langle(b)(bc)(c)\rangle$ が求められる．

3.2 低頻度系列の ϵ-消去規則

ϵ-近似処理により，低頻度の左飽和集合系列を頻度表から積極的に削除することで，頻度表の大きさを抑制し，左飽和集合系列の抽出計算の高速化を行う．[山本 16] では，[Metwally 05] の Space-Saving 法を利用しているが，本研究では，より簡明な処理を目的として，**Lossy-Counting（LC）法** [Manku 03] の **ϵ-消去処理**を拡張した近似処理を行う．

ϵ-近似処理を行うために，許容誤差を ϵ，左飽和集合系列を α としたとき，頻度表 TS には 3 項組 $\langle \alpha, f(\alpha), \delta(\alpha)\rangle$ を登録する．ここで，$f(\alpha)$ は α が登録された時点 t_c から現時刻 t までの出現回数であり，$\delta(\alpha)$ は登録された時点 t_c での出現頻度の最大誤差となる $\epsilon \cdot (t_c - 1)$ である．$f(\alpha) + \delta(\alpha)$ を α の**見積もり頻度**と呼ぶ．ϵ-消去処理とは，各時刻 t において，$f(\beta) + \delta(\beta) \leq \epsilon \cdot t$ 以下となる左飽和集合系列 β を削除する処理である．このとき，β の真の頻度 $\sup(\beta, t)$ に対して，以下ようなの誤差保証ができる．

$$f(\beta) \leq \sup(\beta, t) \leq f(\beta) + \delta(\beta)$$

このように，頻度誤差限界を保証しながら，低頻度の左飽和集合系列を頻度表から積極的に削除できる．

4. 提案手法

共通部分表を利用した左極大共通系列の探索や，LC 法による近似処理には多大な計算コストがかかる．本研究では，図 3 のような従来の左飽和集合系列をキーとしたハッシュ表に加え，共通部分計算の絞り込みを行う末尾要素の索引 (図 1 参照) と，低頻度の系列を求める見積もり頻度の索引 (図 2 参照) を加えた三重索引を新たに構築し，計算を効率化する手法を提案する．

4.1 IDList

頻度表中の全ての左飽和集合系列とウィンドウとの左極大共通系列を求める共通部分計算は，コストが非常に大きい．そこでアイテムの出現状況を記録する索引を用いて計算対象を絞り込むことを考える．

系列の末尾のアイテム集合に共通部分がないとき，系列末尾頻度の定義により頻度は 0 であるため，左極大共通系列を求める必要がない．例えば，新規ウィンドウ $\langle(ad)(ab)(c)\rangle$ が到着したとき，頻度の数え上げを行うのは，末尾に (c) を含む系列

のみであり，それ以外の系列では頻度の数え上げを行わない．そのため，末尾に共通部分が存在する左飽和集合系列とのみ，共通部分の計算を行えばよい．そこで頻度表中の各左飽和集合系列の ID を，末尾のアイテム集合に含まれるアイテムごとに記録した IDList を保持する．

IDList の例を図 1 に示す．図 1 では，新規ウィンドウ $W = \langle (ad)(ab)(c) \rangle$ が到着したとき，W の末尾のアイテム集合は (c) であるため，IDList のアイテム c を参照し，頻度表中の ID が 3，5 の左飽和集合系列とのみ左極大共通系列を求める計算を行う．IDList による絞り込みのアルゴリズムを以下に示す．

Algorithm 1 IDList による絞り込みのアルゴリズム

Input: $S = \langle A_1, A_2, \cdots A_N \rangle$: ストリーム, l: ウィンドウ幅,
TS: 頻度表, $IDList$: 末尾アイテムごとの ID の索引
Output: 左極大共通系列の集合

```
α ← win(t) の末尾集合
for each i ∈ α do
    ID_set ← ID_set ∪ IDList[i]
        ▷ win(t) の末尾アイテムを末尾に含む系列の ID の集合
end for
for id ∈ ID_set do
    A ← TS[id]
    C_set ← inter(A, win(t))
        ▷ A と win(t) の左極大共通系列の集合
end for
return C_set
```

このように，系列の末尾に共通部分が存在する左飽和集合系列とのみ計算を行うことにより，不要な計算を減らし，効率的に左極大共通系列を求めることができると考えられる．

4.2 バケット機構

LC 法の ϵ-消去処理を行う際，低頻度の左飽和集合系列を頻度表から求める必要がある．そこで見積もり頻度ごとの索引を用いた削除処理の効率化を提案する．

図 2 の左側のように，左飽和集合系列 α の ID とその見積もり頻度 V を記録したバケットを保持し，頻度表中の α と同じ見積もり頻度を持つ左飽和集合系列の ID の双方向リストを保持する．また，バケットの双方向リストと見積もり頻度最小のバケットへのポインタ作成する．

図 1 の頻度表と図 2 のバケット機構に対して，許容誤差 $\epsilon = 0.1$，$t = 20$ としたときの動作例を示す．処理を単純化するため ϵ-消去処理の削除対象を $\lfloor \epsilon \cdot t \rfloor$ とする．ϵ-消去処理により，頻度表から見積もり頻度が $\lfloor \epsilon \cdot t \rfloor$ 以下の左飽和集合系列の削除を行う．頻度最小バケットが保持する左飽和集合系列の見積もり頻度が 2 であるため，頻度表から頻度最小バケットが保持する左飽和集合系列の双方向リストを探索し，$\lfloor \epsilon \cdot t \rfloor$ 以下の見積もり頻度を持つ左飽和集合系列を削除する．そして頻度最小バケットを見積もり頻度 3 の左飽和集合系列を保持するバケットに更新する．バケット機構による削除処理のアルゴリズムを以下に示す．

Algorithm 2 バケット機構による削除処理のアルゴリズム

Input: TS: 頻度表, ϵ: 許容誤差, t: 時刻,
B: バケット, p : 最小頻度バケットのポインタ

```
if B[p] の上限値 < ⌊ε · t⌋ then
    for each D ∈ B[p] do
        del_seq(D, TS)
            ▷ 最小頻度バケット内の系列をすべて削除
    end for
    p ← 新たな最小頻度バケットへのポインタ
end if
```

このように，見積もり頻度による索引を作ることで，ソート処理などを行わずに，低頻度の左飽和集合系列を頻度表から定数時間で求め，高速に削除処理を行うことができる．

5. 評価実験

IDList とバケット機構を実装し，既存手法のヒープ木での実装との比較実験を行った．実験には疎なデータセット retail と密なデータセット mushroom を使用した．データセットの詳細を表 1 に示す．実験結果とその考察を以下に示す．

表 1: データセット

データ種類	データ長	アイテム種類数	最大長	平均長
retail	88162	16469	76	10.3
mushroom	8124	118	43	43.0

削除処理の実行時間の比較を図 5 と 6 に示す．疎なデータ，密なデータどちらのデータセットに対しても，バケット機構により削除処理の効率化ができていることが分かる．また，頻度表のサイズが大きいほど，ヒープ木での実装に比べ，より効率化ができていることが分かる．

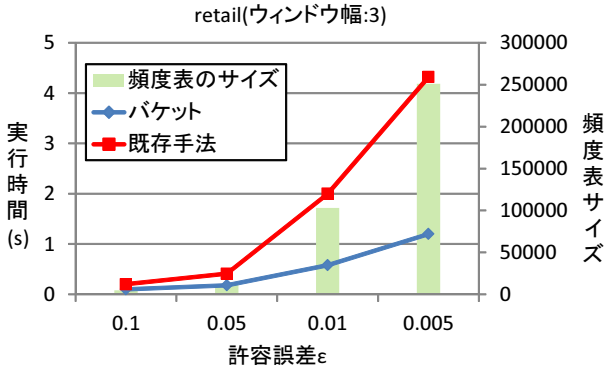

図 5: 削除処理の比較: retail

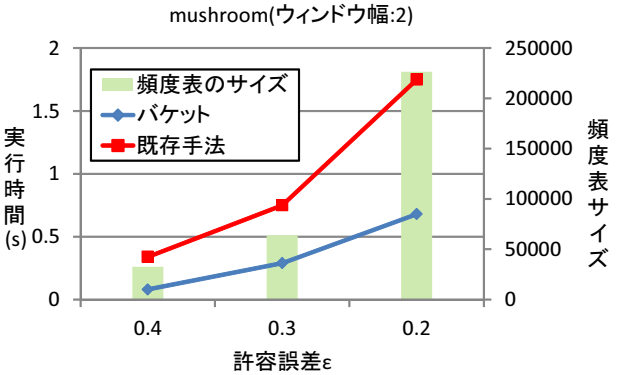

図 6: 削除処理の比較: mushroom

次に，共通部分計算の実行時間の比較を図 7 と 8 に示す．retail に対しては，IDList での絞り込みにより効率化ができていることが分かる．retail では，IDList により共通部分計算の対象を 5 割弱に絞りこむことができるが，絞り込みの処理のオーバーヘッドにより，実行時間は 6 割ほどにしか短縮できていないことが分かる．また，mushroom では全てのトランザクションに出現するアイテムが存在するため，IDList による共通部分計算の対象の絞り込みができない．そのため，絞り込

みの処理により，ヒープ木での実装より実行時間が大きくなっていることが分かる．mushroom は非常に密なデータであるため，このような結果になったと考えられる．

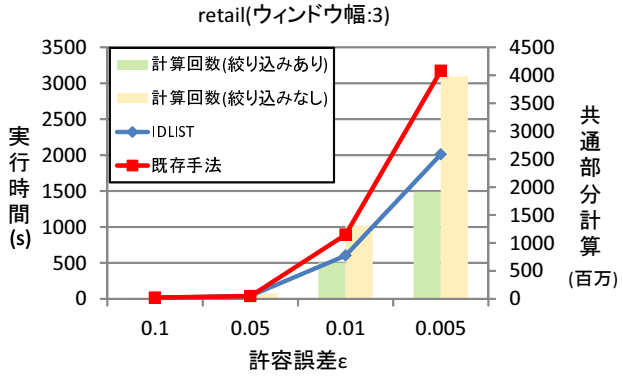

図 7: 共通部分計算の比較: retail

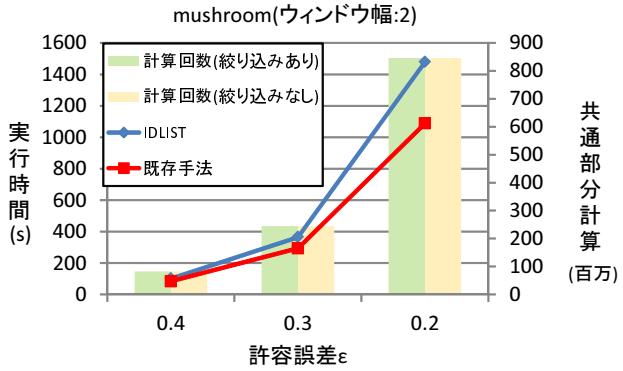

図 8: 共通部分計算の比較: mushroom

6. 考察

先行研究 [山本 16] では，左飽和集合系列のみ抽出を行うアルゴリズムが提案されたが，左飽和集合系列ではない系列が頻度表に登録される場合があることが分かった．

新規ウィンドウ 〈(c)(a)(b)〉

時刻7における頻度表

ID	飽和系列	頻度
1	〈(a)〉	3
2	〈(a)(b)〉	3
3	〈(a)(b)(b)〉	1
4	〈(b)(b)(a)〉	1
5	〈(b)(a)(b)〉	1
6	〈(b)(b)〉	2
7	〈(a)(b)(c)〉	1
8	〈(b)(c)(a)〉	1
9	〈(b)(a)〉	2

時刻8における頻度表

ID	飽和系列	頻度
1	〈(a)〉	3
2	〈(a)(b)〉	4
3	〈(a)(b)(b)〉	1
4	〈(b)(b)(a)〉	1
5	〈(b)(a)(b)〉	1
6	〈(b)(b)〉	2
7	〈(a)(b)(c)〉	1
8	〈(b)(c)(a)〉	1
9	〈(b)(a)〉	2
10	〈(c)(a)(b)〉	1
11	〈(b)〉	3

図 9: 左飽和集合系列が正しく抽出できない例

ストリーム $S = \langle a, b, b, a, b, c, a, b \rangle$，ウィンドウ幅 3 としたときの動作例を示す．時刻 7 における頻度表は図 9 の左のようになる．新規ウィンドウ $\langle (c)(a)(b) \rangle$ が到着すると，左極大共通系列 $\langle (b) \rangle$, $\langle (a)(b) \rangle$ が求められ，時刻 8 における頻度表は図 9 の右のようになる．ここで，$\langle (b) \rangle$ は $\langle (a)(b) \rangle$ の右部分系列となるため，左飽和集合系列ではない．

このような現象が起きる根本的要因は，左極大共通系列が左飽和であるとは限らないためである．これは，アイテム集合の飽和性を考えた場合には生じない現象であり，飽和アイテム系列の処理の難しさを示す本質的な差異と考えられる．

左飽和集合系列のみの抽出を行うためには，左極大共通系列の左飽和性の確認を行う必要がある．しかし，単純に全ての左極大共通系列の左飽和性を確認するには多大な計算コストがかかってしまう．そこで左極大共通系列の頻度，アイテム数及び末尾集合のアイテムの三つを利用した左飽和性の確認を行う対象の絞り込みを行うことを予定している．

自身が左飽和であるのは，自身より頻度が等しいか大きい左拡大系列が存在しない場合である．即ち，系列 α の飽和性の確認は，$\sup(\alpha) \leq \sup(\beta)$ かつ α よりもアイテム種類数や出現延べ数が多く，α の末尾集合が β の末尾集合に含まれるような系列 β に限定して調べればよいことになる．

このように，左飽和性の確認を行う対象を絞り込むことで，効率よく左飽和集合系列のみを求めることを今後の課題とする．

7. まとめ

本稿では，左飽和集合系列の抽出の効率化を行うための IDList とバケット機構等での三重索引の提案を行い評価実験で有用性を示した．また，左飽和集合系列の抽出に関する考察を述べた．左飽和集合系列のみ抽出を行うアルゴリズムの設計と実装及び Space-Saving 法 [Metwally 05] と Lossy-Counting 法 [Manku 03] を融合した近似解法の検討を今後の課題とする．

謝辞

本研究は一部，JSPS 科学研究費補助金（No.16K00298）および JST さきがけの援助を受けている．

参考文献

[山本 16] 山本泰生, 山内夏美, 岩沼宏治: 漸近交差法に基づくオンライン頻出系列パターンマイニング, 人工知能学会研究会資料, SIG-FPAI-B503, pp.80-85 (2016)

[Iwanuma 05] K. Iwanuma, R. Ishihara, Y. Takano and H. Nabeshima: Extracting frequent subsequences from a single long data sequence: a novel anti-monotonic measure and a simple on-line algorithm, *Proc. of ICDM'05*, pp.186–193 (2005).

[Manku 03] G.S Manku and R. Motwani: Approximate frequency counts over data streams. *Proc Intl. Conf. on VLDB'02*, pp.346-357 (2002)

[Metwally 05] A. Metwally, D. Agrawal and A.E. Abbadi: Ecient computation of frequent and Top-k elements in data streams, *Proc. of ICDT'05*, pp.398-412 (2005)

[Borgelt 11] C. Borgelt, X. Yang, R. Nogales-Cadenas, P. Carmona-Saez, and A. Pascual-Montano: Finding Closed Frequent Item Sets by Intersecting Transactions: *Proc. of Intl. Conf. on EDBT'11*, pp.367-376 (2011).