制約充足確率を用いた
数値属性の平均値に関する系列パターンマイニング

Sequential pattern mining with constraints for mean value of numeric attribute by constraint satisfaction probability

北原洋一[1*] 折原良平[1] 櫻井茂明[1] 植野研[1]

Youichi Kitahara, Ryohei Orihara, Shigeaki Sakurai and Ken Ueno

[1]株式会社東芝研究開発センター
[1] Corporate Research & Development Center, Toshiba Corporation

Abstract: This paper presents a pattern mining method with constraints for mean value of numeric attribute by constraint satisfaction probability. From the standpoint of usability, it is sometimes required that a method quickly mine patterns satisfying the constraint, although it cannot enumerate them completely. Constraint satisfaction probability is a ratio of constraint satisfaction pattern to the possible super pattern. We attempt to decide search priority by constraint satisfaction probability in sequential pattern mining. In our experimental evaluation, we show that pattern mining by constraint satisfaction probability is a little more effective than simple heuristics.

1 はじめに

近年，計算機環境の普及により，大量データの蓄積が容易になり，このような大量データから，有用な情報を抽出するため，データマイニングが行われるようになった．POS データを用いた購買分析などは適用事例の一つである．購買分析などで行われるデータマイニングでは，頻出アイテムセットや頻出パターンを発見し，これらから有用なルールや情報を抽出する．

頻出パターンを利用したルール抽出がよく行われる要因の一つに，効率的なアルゴリズムの存在が挙げられる．頻出パターンを効率的に発見するアルゴリズムとしては，Apriori や PrefixSpan がよく知られている[1][2]．これらのアルゴリズムでは，パターンの支持度が有する逆単調性を利用することで枝刈りによる探索空間の削減を行っているため，効率的に高頻出なパターンを発見することができる．

しかし，頻出アイテムセットや頻出パターンだけでなく，より柔軟な条件を満たすパターンを発見したい場合もある．例えば，記述統計で用いられる指標を用いて，統計的に意味のあるパターンを発見できれば, 各種の分析で有効に活用することができる．これは，パターンが満たすべき統計的な条件を制約として表すことにより，制約に基づくパターンマイニング問題として扱うことができる．

制約が，支持度のように逆単調性を有している場合は，探索空間を削減することが可能である．しかし，制約が非逆単調であったり，逆単調性を有していても枝刈りが十分有効に働かなかったりする場合には，効率的な処理が困難になる．柔軟な指定を用いたデータマイニングを可能にするためには，このような逆単調性が有効に働かないときでも，制約を満たすパターンを効率的に発見する手法が必要である．

本研究の目的は，逆単調性が有効に働かない制約に基づくパターンマイニングを効率的に行うことである．特に本稿では，実用性の観点から，系列パターンの最終アイテムセットに付与される数値属性データに着目し，この数値属性データの平均値が満たす条件を制約としたときのパターンマイニングを扱う．

本稿は，６章から構成されている．２章では，問題設定について述べる．３章では，パターンマイニングについて説明する．４章では，制約充足確率を利用したマイニング方法を示す．５章では，実験結果を示し，６章では，まとめをする．

2 問題設定

2.1 データ

本研究で想定しているデータは，時系列データベースに記憶されているものとする．また，このデータベースには，連続値データと離散値データが含まれているものとする．

*連絡先：(株)東芝研究開発センターシステム技術ラボラトリー
〒210-8582 神奈川県川崎市幸区小向東芝町 1 番地
E-mail: youichi.kitahara@toshiba.co.jp

このようなデータは，個々の離散データをアイテムとし，同一時間のデータをアイテムセットとしてまとめ，順に並べ替えることで，一般的な系列パターンマイニング問題で扱われるデータと類似した形式に変換される．一般に用いられている系列データセットと異なるのは，計測データを表す数値属性データも付与されている点にある．図 1 は，系列データの例である．各アイテムセットの末尾に付与されている数値が数値属性である．

ID	系列
1	(ab:0.23)(ac:0.57)(b:0.8)
2	(cd:0.83)(ab:0.48)(b:0.51)
3	(bd:0.83)(cd:0.38)(ab:0.51)
4	(d:0.71)(bd:0.28)(c:0.21)

図 1　データの例

パターン分析の目的は様々であるが，トラブルに至った経緯の分析や，最終的な顧客満足度等の分析を今回は想定する．これらは，特定時期を基点としたパターンである．最終的な状態に至った経緯の分析を想定し，系列パターンの最終アイテムセットに付与されている数値属性を基点としたパターンマイニング問題を扱う．

なお，本稿記載の実験では，数値属性は 0 から 1 までの実数に制限した．そこで，以下の議論においても，数値属性はこの範囲に制限されるものとする．

2.2　制約

よく用いられる制約では，パターンの着目している指標が特定の閾値より大きい，もしくは，小さいことを条件として指定することが多い．しかし，実際には，極端なケースではないが，一般的なケースでもないような状況を分析することもある．そこで，今回は数値属性の平均値が特定範囲にならなければならないという制約を満たすパターンを発見する問題を扱うことにする．

本稿で扱う制約を次のように定義する．探索対象の系列パターン S を含む系列データ集合から，S と系列データをマッチングさせたときに，S の最終アイテムセットの位置に対応する系列データのアイテムセットの数値属性を抽出するとき，抽出された数値属性の集合を $\{m_l(S)\}$ と表記する．S の頻度を $f(S)$ とし，数値属性の平均値に関する閾値を t_l，t_u で表すとき，制約を

$$C_A(S,t_l,t_u)=\left\{ S \middle| t_l < \frac{\sum_{k=1}^{f(S)} m_k(S)}{f(S)} < t_u \right\} \quad \text{--- (1)}$$

と表現する．

例えば，$t_l = 0.3, t_u = 0.45$ のとき，図 1 のデータにおいて系列パターン<(d)(b)>が制約を満たすかを考える．この場合，ID＝2 が 0.51，ID=3 が 0.51，ID=4 が 0.28 であるから，それらを平均して 0.43 となり制約を満たす．なお，最終アイテムセットに着目しているため，パターンマッチングは最終アイテムセットから行う．そのため，ID＝2 では，１番目と２番目のアイテムセットではなく，１番目と３番目のアイテムセットと照合するために，数値属性は 0.48 ではなく 0.51 となる．

3　パターンマイニング

3.1　頻出パターンマイニング

パターンマイニングの中でもっともよく用いられているのが，頻出パターンマイニングである．頻出パターンマイニングでは，アイテムセット，もしくは，系列データセットから，頻出するアイテムの組み合わせ，もしくは，系列パターンを発見する．

頻出パターンマイニングがよく用いられる要因のひとつに効率的なアルゴリズムの存在が挙げられる．頻出アイテムセットマイニングでは，Apriori アルゴリズムや fp-growth などが知られている．また，系列パターンマイニングでも，AprioriAll や GSP，PrefixSpan などがある．

これらのアルゴリズムでは，いずれもパターンの頻度が有する性質を利用して効率的な処理を行っている．系列パターン S について，$S' \supseteq S$ となる系列パターン S' を S の上位パターンと呼ぶ．このとき，S と S' の出現頻度には

$$f(S) \geq f(S') \quad \text{--- (2)}$$

という関係が成立する．なお，系列パターン同士の含有関係では，順序が保持されているものとする．最小頻度 minsup より出現頻度の高いパターンを発見するときには，(2)より $f(S) < \mathrm{minsup}$ であれば $f(S') < \mathrm{minsup}$ であることが直ちに言えるから，S が非頻出パターンであれば S' に関する処理を省略することができる．これは，枝刈りと呼ばれる．これによって，探索空間が縮小されるため，効率的に頻出パターンを発見することが可能となる．

3.2 制約に基づくパターンマイニング

パターンマイニングには，特定の制約条件を満たすパターンを発見するというアプローチもある[3][6]. 制約に基づくパターンマイニングにおける制約条件には，アイテム制約や時間制約，正規表現制約等多くの種類がある．本研究も，この制約に基づくパターンマイニングのひとつである．

制約に基づくパターンマイニングでは，制約ごとに多様なアプローチがなされているが，逆単調性と呼ばれる性質を利用して効率化を図ることが多い．ここでは，パターン S に関する制約 $C(S)$ の否定 $\overline{C(S)}$ を用いて，逆単調性を次のように定義する．

$$\overline{C(S)} \Rightarrow \overline{C(S')} \quad \text{--- (3)}$$

性質(3)を満たす制約の例としては，頻出パターンマイニングで用いられる頻度に関する制約

$$C_f(S,t) = \left\{ S \middle| f(S) < \mathrm{minsup} \right\} \quad \text{--- (4)}$$

がある．

逆単調性(3)が成立する制約に基づくパターンマイニングでは，頻出パターンマイニングと同様に，$C(S)$ が満たされないことが判明したら，上位パターン S' の処理を行わないようにすることで，探索空間を削減することができる．そのため，多くの研究では，問題を逆単調な制約に帰着させることで，効率化を図ることが多い．

しかしながら，記述統計的な指標に関する制約に基づくパターンマイニングでは，制約が非逆単調である，もしくは，逆単調であっても枝刈りの効果が小さいことが多い．そこで，このような逆単調性を有効に利用できない制約を適切に扱うことのできる手法が必要になる．

このような問題に対するアプローチは，大きく分けて三種類がある．一つ目は，頻出パターンマイニングを用いて頻出パターンを抽出した後，条件に合致するパターンを抽出する方法である．二つ目は，制約の性質を利用して，逆単調な制約問題に帰着させる方法である．三つ目は，ヒューリスティック等を用いる方法である．

一つ目の頻出パターンマイニングを用いる方法は，シンプルで，よく利用される．しかし，データ量が多いと，探索効率が悪くなりやすい．

二つ目の逆単調な制約問題に帰着させる方法は，制約を満たすパターンを完全列挙するのに適している．このアプローチを採用したアルゴリズムには，AprioriSMP が挙げられる[4]. AprioriSMP では，制約を構成する関数の凸性を利用することで算出される，上位パターンが取り得る制約関数の上限値を用いることで，逆単調な制約と同様の効率的な処理を可能にさせている．ただし，扱う問題によっては上限値が高くなり，発見効率が低下することがある．

三つ目のヒューリスティックスを用いる方法は，比較的効率的に制約を満たすパターンを発見できる．しかし，制約を満たすパターンを完全に列挙する場合，処理量が著しく大きくなるという問題がある．現実的な問題では，完全列挙せずとも効率的に主要なパターンのみを発見できればよいケースも多い．そこで，本研究では，三つ目の方法を採用した．

数値属性の平均値制約に関するヒューリスティックスの例としては，探索候補のパターンの平均値と制約条件の閾値との距離が短いものから探索する方法がある．制約(1)の場合，制約を満たす範囲の中心からの距離

$$h(S) = \left| \frac{t_u - t_l}{2} - \frac{\sum_{k=1}^{f(S)} m_k(S)}{f(S)} \right| \quad \text{--- (5)}$$

が小さいものから探索する．

4 提案手法

4.1 制約充足確率

確率 p で成立する条件を $\underset{p}{\Rightarrow}$ と表し，制約を

$$\overline{C(S)} \underset{p}{\Rightarrow} \overline{C(S')} \quad \text{--- (6)}$$

と表現する．逆単調性(3)は，$\overline{C(S)}$ であれば 100% の確率で $\overline{C(S')}$ であることを示しており，これは p が 1.0 のケースである．逆単調ではない場合，p は 1.0 ではない．そこで，性質(6)を p が 1.0 ではないときに拡張することで，制約充足確率を導入する．

制約に関する性質，

$$\overline{C(S)} \underset{1-p_S}{\Rightarrow} \overline{C(S')} \quad \text{--- (7)}$$

が満たされるとき，確率 p_S を制約充足確率と呼ぶことにする．制約充足確率 p_S を導入することにより，逆単調性が確率 p_S で緩和されて成り立つ．これを利用すれば，S が制約 C を満たさない，もしくは，満たしにくいことが判明した時点で，S の上位パターン S' に関する処理を枝刈りしたり，探索優先度を下げたりすることができるため，効率的なマイニング処理が可能になる．

一般に制約充足確率はわからないので，制約に含まれる変量の逆単調性を利用することで推定するこ

とを試みる．制約そのものは逆単調性が有効でなくとも，制約を構成する条件式には，逆単調性を満たす構成要素が含まれていることが多い．そこで，逆単調性を満たす構成要素を変数とみなし，この変数から構成される空間におけるパターン S の上位パターン S' が取りうる領域を考えることで，制約充足確率を算出する．S' の存在確率が取りうる全領域で等しいと仮定すると，この空間において，パターン S の上位パターン S' が取りうる領域の体積を $V(S)$，制約を満たす S' が取りうる領域の体積を $V_C(S)$ とすると，制約充足確率 p_S は

$$p_S = \frac{V_C(S)}{V(S)} \quad \text{--- (8)}$$

となる．S' の存在確率の分布は一様とは限らず，データ特性とパターン S に依存して，多様な形状を取りうると考えられるが，これらを逐一推定することは困難であるから，取り扱いの容易性も考慮し，ここでは一様分布を仮定することにした．

制約充足確率の利用方法には，通常の頻出パターンマイニングと同様に枝刈りの閾値として利用する方法と，Top-K パターンマイニングのように探索優先度を決定するのに用いる方法とがある．いずれも，マイニングアルゴリズムの最小支持度や最小頻度による枝刈り処理を行う部分に，制約充足確率算出処理を追加することで実装できる．Top-K パターンマイニングのアルゴリズムとしては，支持度の高い順からパターンを発見する TopSequencesTraversal がある[5]. 今回の実験において，用いたアルゴリズムは，TopSequencesTraversal において，支持度を制約充足確率に置き換え，支持度の増加処理を省いたものとほぼ同じである．

4.2 平均値制約の制約充足確率

制約充足確率を算出するためには，まず逆単調な構成要素を決めなければならない．制約(1)に含まれる平均値は，数値属性の総和

$$M(S) \equiv \sum_{k=1}^{f(S)} m_k(S) \quad \text{--- (9)}$$

を用いて，

$$C_A(S, t_l, t_u) = \left\{ S \middle| t_l < \frac{M(S)}{f(S)} < t_u \right\} \quad \text{--- (10)}$$

と書くことができる．$f(S)$ は頻度であるから逆単調である．また，$M(S)$ は，一般には非逆単調であるが，平均値を算出するのに使う数値属性を有するアイテムセットを基点とし，そのアイテムセットにアイテムを付加することでパターンを成長させるアルゴリズムを用いることにより，逆単調になる．例えば，PrefixSpan のようなアルゴリズムを利用する場合，最初の prefix をそのような数値属性を有するアイテムセットにすることで，実現することができる．これらをまとめると

$$\begin{cases} M(S) \geq M(S') \\ f(S) \geq f(S') \end{cases} \quad \text{--- (11)}$$

となる．

次に，$M(S)$ と $f(S)$ から構成される空間における S' の取りうる領域の面積と，制約を満たす S' の取りうる面積の比を算出する．

S' の取りうる領域を正確に算出することで，制約充足確率もより適切な値になると考えられるが，制約充足確率を算出する時間自体が大きくなるという問題もある．パターンマイニングでは，パターンの探索数が非常に大きいため，マイニングの全処理時間は，１パターンの探索処理時間に強く依存する．そのため，制約充足確率が正確になることによる探索効率向上がもたらす処理時間の短縮効果が，１パターンの探索処理時間増大によって相殺される可能性もある．そこで，S' の取りうる領域の正確さに応じて三種類の制約充足確率 p_{s1}， p_{s2}， p_{s3} を考えることにする．

図 2 は，制約条件と S' の取りうる領域を図示したものである．$M(S)$ と $f(S)$ から構成される平面において，制約条件(1)は直線 $M(S) = t_u f(S)$ と $M(S) = t_l f(S)$ に挟まれる部分によって表すことができる．また，S' の取りうる範囲についても，直線もしくは曲線によって，この平面において表すことができる．

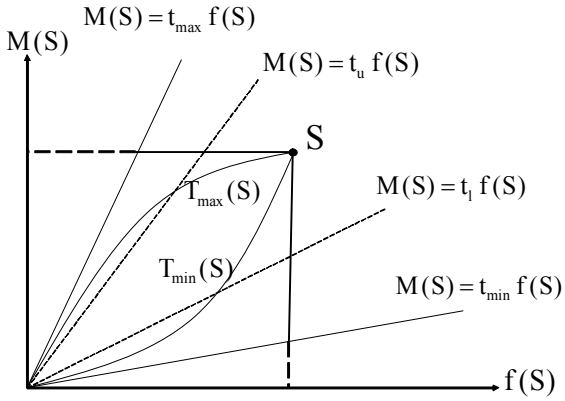

図 2 M(S)と f(S)の空間における S'の範囲

一つ目のもっともラフな制約充足確率 p_{S1} は，数

値属性の最大値が 1，最小値が 0 のとき，S'は直線 $M(S) = f(S)$ と $f(S)$ 軸で囲まれる範囲を取りうるとして算出する．このとき，p_{S1} は

$$W(S) \equiv \frac{M(S)}{f(S)}$$

$$p_{S1} = \begin{cases} \dfrac{(t_u - t_l)}{W(S)(2 - W(S))} & (W(S) \geq t_u \text{のとき}) \\ \dfrac{2 - t_u^{-1} W(S) - t_l W(S)^{-1}}{2 - W(S)} & \left(t_u > \dfrac{M(S)}{f(S)} > t_l \text{のとき} \right) \\ \dfrac{W(S)(t_l^{-1} - t_u^{-1})}{2 - W(S)} & \left(t_l \geq \dfrac{M(S)}{f(S)} \text{のとき} \right) \end{cases}$$

--- (12)

となる．算出式は比較的シンプルであるが，数値属性の最大値と最小値を固定しているため，S'の取りうる領域を過大評価しているという問題がある．

二つ目の制約充足確率 p_{S2} は，S の数値属性の最大値および最小値から得られる，S'の平均値の下界および上界を利用して算出される．S の数値属性の最大値を $t_{max}(S)$，最小値を $t_{min}(S)$ とすると，S'の平均値との間には次のような関係が成り立つ．

$$t_{min}(S) \leq \frac{M(S')}{f(S')} \leq t_{max}(S) \text{ --- (13)}$$

そこで，逐次 $t_{max}(S)$ と $t_{min}(S)$ を更新し，S'は直線 $M(S) = t_{max}\, f(S)$ と $M(S) = t_{min}\, f(S)$ に囲まれる範囲を取りうるとして p_{S2} を算出する．このとき p_{S2} は，

$$p_{S2} = \begin{cases} 0 \ (t_l \geq t_{max} \text{のとき}) \\ \dfrac{2 - t_{max}^{-1} W(S) - t_l W(S)^{-1}}{2 - t_{max}^{-1} W(S) - t_{min} W(S)^{-1}} & (t_u > t_{max} \geq t_l \wedge t_l > t_{min} \wedge W(S) \geq t_l \text{のとき}) \\ \dfrac{W(S)(t_l^{-1} - t_{max}^{-1})}{2 - t_{max}^{-1} W(S) - t_{min} W(S)^{-1}} & (t_u > t_{max} \geq t_l \wedge t_l > t_{min} \wedge t_l > W(S) \text{のとき}) \\ 1 (t_u \geq t_{max} > t_l \wedge t_{min} > t_l \text{のとき}) \\ \dfrac{W(S)^{-1}(t_u - t_{min})}{2 - t_{max}^{-1} W(S) - t_{min} W(S)^{-1}} & (t_{max} \geq t_u \wedge t_{min} \geq t_l \wedge W(S) \geq t_l \text{のとき}) \\ \dfrac{2 - t_u^{-1} W(S) - t_{min} W(S)^{-1}}{2 - t_{max}^{-1} W(S) - t_{min} W(S)^{-1}} & (t_{max} \geq t_u \wedge t_{min} \geq t_l \wedge t_l > W(S) \text{のとき}) \\ \dfrac{W(S)^{-1}(t_u - t_l)}{2 - t_{max}^{-1} W(S) - t_{min} W(S)^{-1}} & (t_{max} \geq t_u \wedge t_l > t_{min} \wedge W(S) \geq t_u \text{のとき}) \\ \dfrac{2 - t_u^{-1} W(S) - t_l W(S)^{-1}}{2 - t_{max}^{-1} W(S) - t_{min} W(S)^{-1}} & (t_{max} \geq t_u \wedge t_l > t_{min} \wedge t_u > W(S) \geq t_l \text{のとき}) \\ \dfrac{W(S)(t_l^{-1} - t_u^{-1})}{2 - t_{max}^{-1} W(S) - t_{min} W(S)^{-1}} & (t_{max} \geq t_u \wedge t_l > t_{min} \wedge t_l \geq W(S) \text{のとき}) \\ 0 \ (t_{min} \geq t_u \text{のとき}) \end{cases}$$

--- (14)

となる．S に応じて $t_{max}(S)$ と $t_{min}(S)$ が変化することが p_{S1} との大きな違いである．また，p_{S2} が 0 になる場合は，枝刈りを行うことで，探索空間を削減することができる．

三つ目の制約充足確率 p_{S3} は，S'の取りうる上界と下界を示す曲線 $T_u(S)$ と $T_l(S)$ で囲まれる範囲を，S'の取りうる領域とすることで算出される．曲線 $T_u(S)$ は，S の数値属性集合 $\{m_i(S)\}$ のうち，大きい値を順に取り出して和をとったものをプロットすることで示すことができる．曲線 $T_l(S)$ では，逆に小さい値から順に取り出して和をとったものをプロットする．曲線 $T_u(S)$ と $T_l(S)$ は S が定まらないと知ることができないため，逐次算出しなければならない．また，S'が取りうる領域の面積の算出についても，あらかじめ関数が判明しているわけではないので逐次数値積分を行って算出することになる．そのため，処理時間は p_{S1} や p_{S2} と比較して大きい．しかし，もっとも正確に S'の取りうる領域を算出することができる．

5 実験

5.1 実験条件

制約充足確率を用いたパターンマイニングの効果を確認するために，人工サンプルデータを用いて実験を行った．人工サンプルデータは，IBM が公開しているデータ生成プログラムが生成するサンプルデータに，一様乱数の数値属性を付与したものを用いた．実験は，制約を満たすパターンが 2,000 個発見されるまで行い，その時点における探索効率と処理時間を比較した．なお，発見パターン数/探索パターン数を探索効率と定義する．

アルゴリズムは，単純なヒューリスティックスである式(5)の $h(S)$，および，三種類の制約充足確率 p_{S1}，p_{S2}，p_{S3} を用いた．Top-K タイプのマイニングアルゴリズムを用いた．

実験では，制約条件，系列データ数，アイテムセット内アイテム数，アイテム種類数に着目し，それらが，各アルゴリズムの探索効率および処理時間に及ぼす影響を調べた．ベースとなるパラメーターは以下の通りである．

[ベースパラメーター]

最小支持度： 0.1 %

系列データ数： 2000

アイテムセット内アイテム数： 20

アイテムセット数： 20

アイテム種類数： 500

制約条件閾値： $t_u = 0.8,\ t_l = 0.65$

各実験においては，ベースパラメーターのうち，次のパラメーターを変化させて実行している．

[制約条件の実験]

制約条件閾値： $t_l = 0.6 \sim 0.74$ （0.1 刻み）

[系列データ数の実験]

系列データ数： 1000～5000 （1000 刻み）

[アイテムセット内アイテム数の実験]

アイテムセット内アイテム数： 5～30 （5 刻み）

[アイテム種類数の実験]

アイテム種類数： 100，500，1000

5.2 制約条件

制約条件に関する実験の探索効率を図 3 に，処理時間を図 4 に示す．

いずれの方法を用いても，制約条件が厳しくなると探索効率が低下し，処理時間が増加する傾向が見られる．特に，単純なヒューリスティックスと p_{S1} で顕著である． p_{S2} は，一部のケースで計算不能になるときが見られた． p_{S3} は，制約条件が厳しくなっても，探索効率の低下はわずかであり，処理時間の増加も小さい．このことは，探索しにくいパターンを発見するときほど， p_{S3} の制約充足確率が有効に機能していることを示唆しているように思われる．

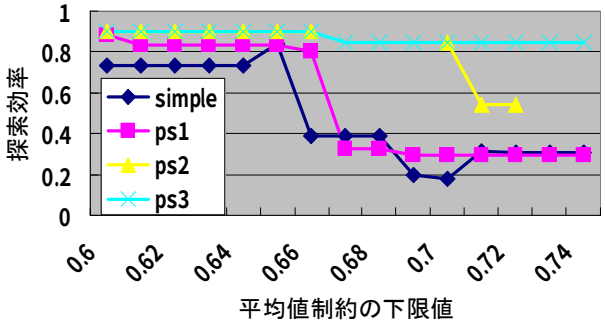

図 3 探索効率（制約条件）

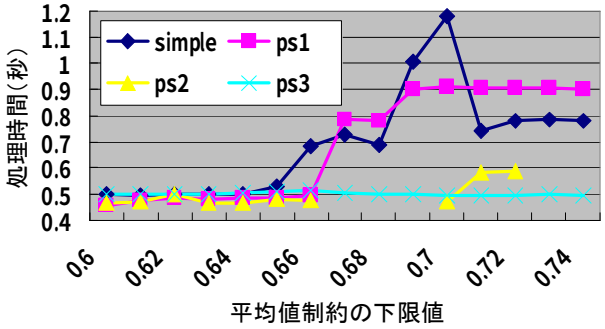

図 4 処理時間（制約条件）

5.2 系列データ数

系列データ数に関する実験の探索効率を図 5 に，処理時間を図 6 に示す．

探索効率と系列データ数とには，特別な傾向性は見られないが，系列データ数の増加にともない処理時間は線形に増加している．なお，系列データ数が 3000 のときに， p_{S3} 以外がすべて著しく探索効率が低下しているが，これはデータ特性によるものと思われる．このケースでも， p_{S3} には極度な探索効率の低下は見られない．

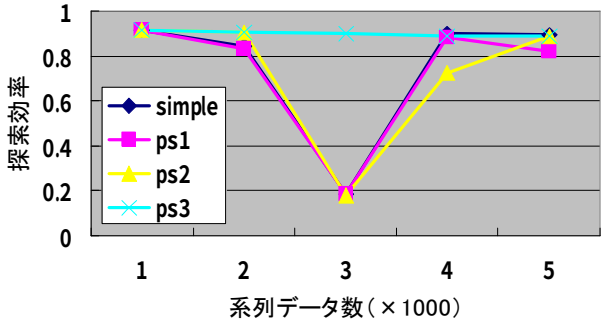

図 5 探索効率（系列データ数）

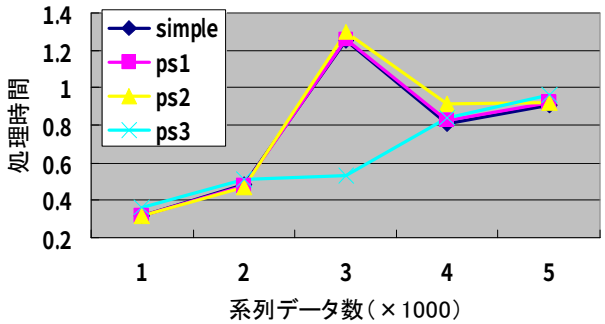

図 6 処理時間（系列データ数）

5.3 アイテムセット内アイテム数

アイテムセット内アイテム数に関する実験の探索効率を図 7 に，処理時間を図 8 に示す．

探索効率とアイテムセット内アイテム数とには，明瞭な傾向性は見られないが，わずかながら低下している．また，アイテムセット内アイテム数の増加にともない，処理時間が増加している．アイテムセット内アイテム数が 25 の場合，p_{S2} と p_{S3} で計算不能になっているが，h と p_{S1} では著しい探索効率の低下は見られるものの処理できている． p_{S3} であっても，処理可能であるとは限らない．

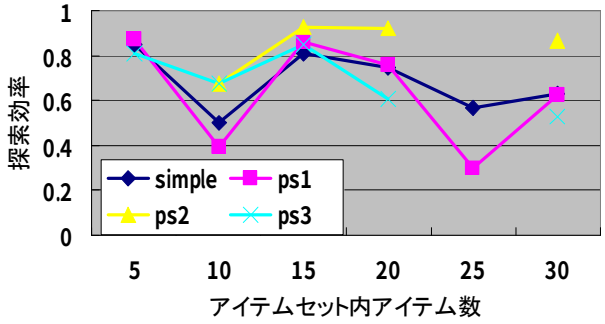

図 7　探索効率（アイテムセット内アイテム数）

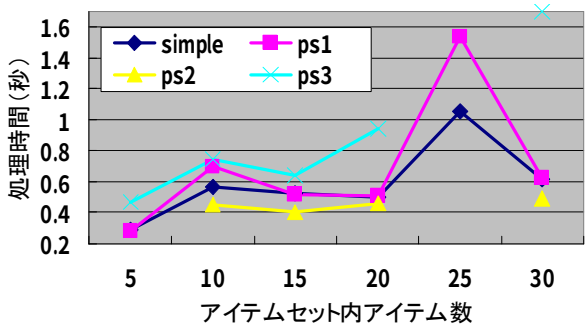

図 8　処理時間（アイテムセット内アイテム数）

5.4　アイテム種類数

アイテム種類数に関する実験の探索効率を図 9 に，処理時間を図 10 に示す．

探索効率は，全体的に安定しているが， p_{S3} はアイテム種類数が増加するとやや探索効率が低下しているように思われる．また，全般的に p_{S3} の探索効率が低い．今後，より多くの実験で，傾向を調べることが必要である．

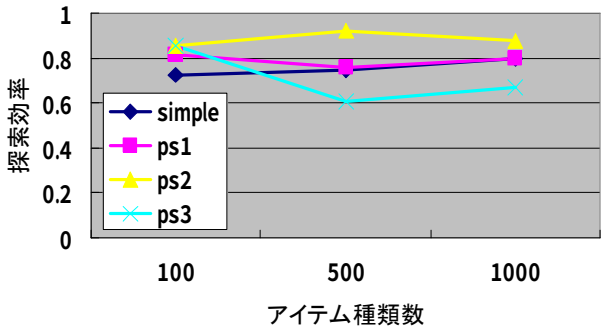

図 9　探索効率（アイテム種類数）

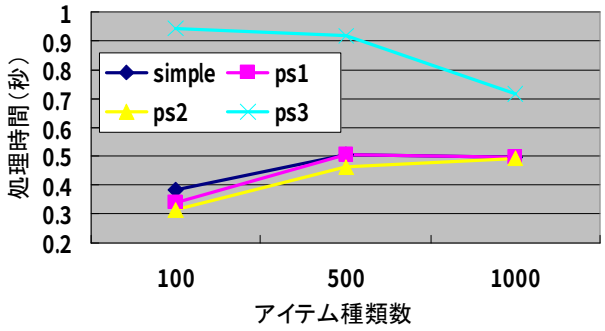

図 10　処理時間（アイテム種類数）

6　まとめ

本稿では，制約充足確率を利用して，数値属性の平均値に関する制約を満たすパターンを効率的に発見する手法を提案した．制約充足確率は，上位パターンの可動範囲の評価方法に応じて三種類を提案した．

簡単な実験の結果，数値積分を用いて可能な限り正確に算出された制約充足確率を用いたときに，もっとも探索効率が高い傾向が見られていた．また，特に，制約条件が厳しいときに，単純なヒューリスティックスなどと比較して効率的であった．このことは，正確な制約充足確率が，発見しにくいパターンを発見するときに適していることを示唆しているように思われる．しかしながら，パラメーターによっては計算不能になることもあり，安定しているとは言いがたい結果であった．

全般的に，実験結果は，個々のデータ特性に依存する部分が大きく，各パラメーターとの関係性が明瞭になったとはいえない．さらなる実験を実施し，制約充足確率の性質を明らかにすることが必要である．また，今回は最小支持度の条件を制約充足確率導出に利用していない．しかし，本来ならば最小支持度の条件も考慮すべきであるので，制約充足確率と最小支持度の関係も今後調べる予定である．

参考文献

[1] R. Agrawal et.al., Mining Sequential Patterns, Proc. Of the 11th Int. Conf. Data Engineering, pp.3-14 (1995)

[2] J. Pei et. al. PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth, Proc. Of the 12th IEEE Int. Conf. On Data Engineering (2001)

[3] R. Ng et.al., Exploratory Mining and Pruning Optimizations of Constrained Association Rules, Proc. Of the 1998 SIGMOD Conf. (1998)

[4] Morishita et.al., Traversing Lattice Itemset with Statistical Metric Pruning, Proc. Of PODS'00 (2000)

[5] P. Tzvetkov, TSP: Mining Top-K Closed Sequential Patterns, Proc. 2004 Int. Conf. on Data Mining, (2003)

[6] J. Pei et.al., Mining Sequential Patterns with Constraints in Large Databases, Proc. of the 2002 ACM CIKM Conf., (2002)