マルコフ連鎖モデルに基づく動的な移動ヒストグラム構築手法

Dynamic Mobility Histogram Construction Based on Markov Chains

町田 陽二♡ 石川 佳治♢ 北川 博之♠

Yoji MACHIDA Yoshiharu ISHIKAWA Hiroyuki KITAGAWA

リアルタイムに移動する多数のオブジェクトの移動状況の集約・分析のため，我々は移動データを要約する移動ヒストグラムの構築手法の開発を進めている．この手法では，マルコフ連鎖モデルに基づき移動統計量を集約する．移動ヒストグラムの論理モデルとしてはデータキューブを採用し，OLAP 的な分析を支援する．一方，物理モデルとしては木構造を採用し，ストリーム的に配信される移動状況データを限られたメモリ内で効率よく要約する．本稿では提案手法の詳細について述べ，性能を実験により評価する．

For the accumulation and analysis of a large collection of moving object trajectories, our group focuses on the research on a mobility histogram to summarize moving object trajectories. The histogram is based on a mobility statistics model called the Markov chain model. We provide a mobility histogram datacube-like logical representation and support an OLAP-style analysis. As its physical structure, we introduce a tree structure that efficiently works in a limited memory space. We describe the details of the method and evaluate its performance based on experiments.

1. はじめに

大量の移動オブジェクトの移動状況をリアルタイムに追跡し，分析・予測に役立てるため，本研究グループでは，移動データを要約する**移動ヒストグラム**（mobility histogram）の概念を提案し，ストリーム的に配信されてくる移動状況データを効率よく集計する動的なヒストグラム構築手法の研究を進めている．このアプローチでは，マルコフ連鎖モデルに基づいて移動パターンの移動統計量を集約する．インクリメンタルなヒストグラムの更新，および，コンパクトかつ精度のよい実現手法が課題となる．

[4] では，基本データ構造（本稿では素朴な方式と呼ぶ) の提案を行った．本稿では，素朴な方式の概要を述べ，その改善方式として，コンパクトな近似方式，および，近似方式の精度を改善する方式（ビットマップ併用方式）の提案を行い 3 方式について比較実験を行う．

2. 関連研究

ヒストグラムは，データを要約するための一手法であり，データベースの分野でも，特に問合せ最適化や近似問合せにおいて盛んに研究開発が進められている [3]．[2, 8] では，蓄積された移動データに対する問合せ選択率を効率的に推定するための研究が行われている．[7] では，過去，現在，未来の移動状況に対する問合せに対応できるよう，時空間ヒストグラムを用いて移動情報を集約する研究が行われている．一部の過去の移動状況と現在の移動情報状況の集約情報のみを主記憶上に持つことで，頻出する問合せに対応できるような実装を行っている．

本研究が対象とするヒストグラムは，特殊な制約はあるが多次元のヒストグラムに分類される．移動軌跡データのリアルタイムの集積を実現するためには，インクリメンタルな更新に対応するため，特に効率性が求められる．[2, 8, 7] と比較した本研究の他の特徴としては，1) マルコフ連鎖モデルに基づく移動統計量の表現，2) 複数の解像度を用いて移動状況を集積する点，3) ストリーム的に得られる移動軌跡データのリアルタイムな処理である．

3. マルコフ連鎖モデルに基づいた移動統計量

時空間データ分析における**マルコフ連鎖モデル**（Markov chain model）は，ある地域から別の地域へある期間内にどの程度の人口が移動したなどの，移動オブジェクトの時空間的な移動傾向の把握に用いられる [9]．まず，準備として，移動パターンの表現のために用いる，2 次元平面の番号付け手法について述べる．この手法は 2 次元平面を 1 次元で順序付けする手法の一つである **Z-ordering** [6] に基づいている．

2 次元空間が各次元ごとに 2^P 個ずつ，$R = 2^{2P}$ 個のセルに分割されているとする (図 1 は $P = 2$ の場合)．この分割のことを**レベル P の分割**と呼ぶ．各セルには，$2P$ ビットのセル番号を付与する．セル番号の上付き数字は，空間分割レベルを表す．この概念を用いることで空間分割の粒度を指定できる．レベルの異なる分割を対比して，粗い分割の方を上位の空間分割，細かい分割の方を下位の空間分割と呼ぶ．図 1 は，時刻 $t = \tau$ でセル 9 にいたオブジェクトが，次の時刻 $t = \tau + 1$ でセル 12 に，そして $t = \tau + 2$ の時点でセル 6 に移動した状況を示している．

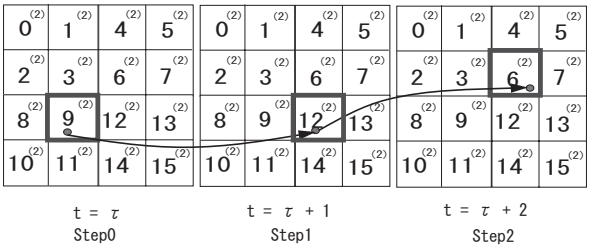


図 1: マルコフ連鎖モデルの概念
Fig. 1: The Notion of Markov Chain Model

4. 移動ヒストグラムの論理構造

移動ヒストグラムは遷移シーケンスの集約を行うことで構築される．集約情報を表現する移動ヒストグラムの論理表現として，本手法では**データキューブ**を用いる．n 次のマルコフ連鎖の場合，ヒストグラムを $n + 1$ 次元のデータキューブとして構成する．$n = 2$ の場合のデータキューブ表現を図 2 に示す．これはレベル $P = 1$ の分割の場合であり，2 次元平面を $R = 2^{2P} = 4$ 個のセルに分割している．Step 0, 1, 2 という各次元は，それぞれマルコフ連鎖のステップに相当する．たとえば $1 \rightarrow 1 \rightarrow 2$ という遷移シーケンスが生成されると，対応するキューブセルに 1 が加算される．

5. 移動ヒストグラムの物理構造

データキューブ形式のヒストグラムの直接的な実装はオーバヘッドが大きい．有限なメモリを効率的に使うため，物理構造として木構造を用いる．

♡ 正会員 筑波大学理工学研究科．現在，(株) 日立情報システムズ
y-machida@kde.cs.tsukuba.ac.jp
♢ 正会員 名古屋大学情報連携基盤センター
ishikawa@itc.nagoya-u.ac.jp
♠ 正会員 筑波大学システム情報工学研究科／計算科学研究センター
kitagawa@cs.tsukuba.ac.jp

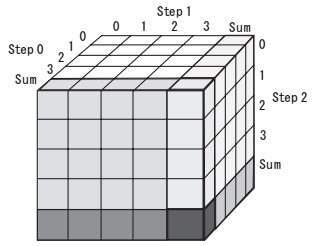


図 2: ヒストグラムの論理構造
Fig. 2: Logical Histogram

5.1 素朴な実現方式

移動ヒストグラムは四分木（quad-tree）と k-d 木 [6] を統合した木構造で表現する．ここでは，基本的なアイデアを示すために，素朴な方式について説明する．

入力の移動軌跡データにおいて，$2^{(P)} \rightarrow 10^{(P)} \rightarrow 12^{(P)}$ のように，分割レベル P でのセルの番号付けがなされていると想定する．そこで，P がこれ以上高い精度でのデータ表現にならない場合，分割レベル P を**最大空間分割レベル**と呼び，m で表記する．

木の各内部ノードは 0 個以上 4 個以下の子ノードを有し，リーフノードはそのノードに対応する移動軌跡の数を集積するためのカウントを有する．ここでは，2 次の遷移シーケンス $a^{(m)} \rightarrow b^{(m)} \rightarrow c^{(m)}$ が木に挿入されることを考える．$a^{(m)}$, $b^{(m)}$, $c^{(m)}$ はセル番号を表す．$a^{(m)}$ をステップ 0 のセル，$b^{(m)}$ をステップ 1 のセル，$c^{(m)}$ をステップ 2 のセルと呼ぶ．基本的には以下のような処理が行われる．

1. $a^{(m)}$，$b^{(m)}$，$c^{(m)}$ をバイナリ表記して，まず，$a^{(m)}$ の上位 2 ビットを取り出す．その内容が 00 (= $0^{(1)}$)，01 (= $1^{(1)}$)，10 (= $2^{(1)}$)，11 (= $3^{(1)}$) のいずれかであるかに応じて，対応する辺を辿り，子ノードへと達する．ただし，そのような辺がない場合には，新たに辺を作成する．
2. $b^{(m)}$，$c^{(m)}$ の順に上位 2 ビットを取り出し，その値に応じて同様に子ノードにアクセスする．これまでの処理は空間分割レベル 1 のおおまかな遷移に対応する．
3. 次に，$a^{(m)}$ の次の 2 ビット，$b^{(m)}$ の次の 2 ビット，$c^{(m)}$ の次の 2 ビットをそれぞれ取り出し同様の処理を行う．このステップは空間分割レベル 2 に対応する．
4. このようなステップを繰り返し，$2m$ ビットずつを処理した時点でリーフノードに至る．このリーフノードまで至る過程において，各ノードのカウントを 1 ずつインクリメントする．

図 3 に遷移シーケンスを追加する例を示す．点線はノードを辿るような遷移シーケンスがまだ到着していないことを表しており，実際にはそれ以下の部分木は存在しない．実線は，ノードを辿るような遷移シーケンスが過去に挿入されたことを表している．

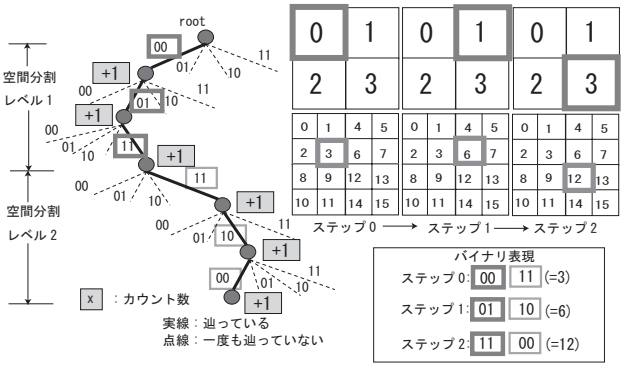


図 3: ヒストグラムの物理構造 ($3^{(2)} \rightarrow 6^{(2)} \rightarrow 12^{(2)}$，$m = 2$)
Fig. 3: Physical Histogram ($3^{(2)} \rightarrow 6^{(2)} \rightarrow 12^{(2)}$，$m = 2$)

5.2 問合せ処理

与えられた遷移シーケンスについて，その出現数を答えるための問合せ処理方式について説明する．$m = 2$ の場合について，例を用いて処理の概要を説明する．アルゴリズムの詳細については [5] を参照願いたい．

1. 移動軌跡の各セルのレベルが最大空間分割レベルと一致する場合：$3^{(2)} \rightarrow 6^{(2)} \rightarrow 9^{(2)}$ を考える．遷移シーケンスの挿入と同様に，各ステップをバイナリ表記して各レベルごとに 2 ビットずつ辿っていき，目的のカウントを返す．
2. 移動軌跡の各セルのレベルが最大空間分割レベルよりも粗い場合：$1^{(1)} \rightarrow 1^{(1)} \rightarrow 9^{(2)}$ を考える．1 と同様に，各ステップをバイナリ表現にして，各レベルごとに 2 ビットずつ辿っていくが，空間分割レベル 1 の 01 → 01 → 10 の後，{00, 01, 10, 11} → {00, 01, 10, 11} → 01 の 16 通りの経路をすべて辿り各カウントを調べ，総和をとる．
3. 移動軌跡のセルのレベルが最大空間分割レベルよりも大きい場合：$3^{(2)} \rightarrow 25^{(3)} \rightarrow 9^{(2)}$ を考える．$3^{(2)} \rightarrow 25^{(3)} \rightarrow 9^{(2)}$ は，空間分割レベル 2 までの木では表現できないので，カウントを近似的に推定する．まず 25 をバイナリ表記すると，(011001) となる．この上位 4 ビットが $25^{(3)}$ の上位空間に対応する．すなわち，その上位空間は $3^{(2)} \rightarrow 6^{(2)} \rightarrow 9^{(2)}$ である．$3^{(2)} \rightarrow 6^{(2)} \rightarrow 9^{(2)}$ のカウントはヒストグラムのノードに保持されている．$6^{(2)}$ の下位空間には $\{24^{(3)}, 25^{(3)}, 26^{(3)}, 27^{(3)}\}$ があるので，4 等分したカウントで近似する．

図 4 に問合せ処理の例を示す．上記 1 に相当し，結果は 4 となる．

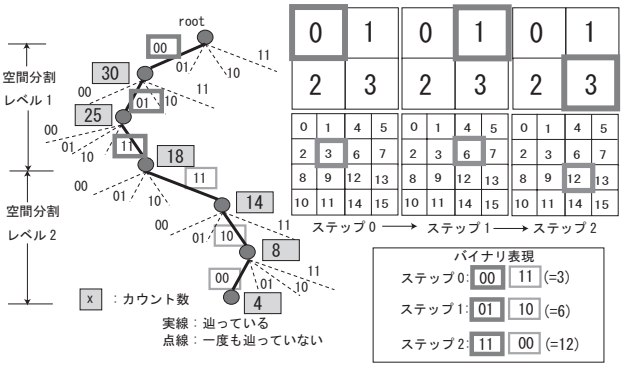


図 4: 問合せ処理例（$3^{(2)} \rightarrow 6^{(2)} \rightarrow 12^{(2)}$，$m = 2$）
Fig. 4: Query Processing ($3^{(2)} \rightarrow 6^{(2)} \rightarrow 12^{(2)}$，$m = 2$)

素朴な方式はシーケンスのカウントを正確に表現できるが，格納コストの面で問題がある．そこで，与えられたノードの数の上限 N のもとで近似的にヒストグラムを構築する方式を以下に示す．

5.3 近似方式による構築

近似方式では，移動シーケンスの追加に対し適応的に木を展開する．2 次の遷移シーケンスを集積する例を示す．アルゴリズムの詳細は [5] に示している．

初期時点ではルートノードのみを構築し，移動シーケンスを順次受け付ける．現在，最初の 100 件のシーケンスが得られた時点であるとし，各シーケンスがステップ 0 においてレベル 0 のどのセル内に位置するかを調べたとき，

セル $0^{(0)}$: 22, セル $1^{(0)}$: 23, セル $2^{(0)}$: 27, セル $3^{(0)}$: 28

という分布だったとする．この場合，カウント数に大きな差が見られないため，木を細分化するメリットがないと考える．そこで，引き続き移動シーケンスの集積を続ける．一方，カウント数が

セル $0^{(0)}$: 10, セル $1^{(0)}$: 20, セル $2^{(0)}$: 50, セル $3^{(0)}$: 20

という場合には，カウントにばらつきが大きいことから，木を細分化し，ルートノードの下に 4 つの子ノードを作成する．これにより，パターンが集中した領域を中心に木構造を展開する．ルー

トノードが展開されると，次にはルートノードの子ノードについてカウント数の分布を追跡する．ただし，今回はステップ 1 のセルに着目する．このような処理を繰り返すことで，ヒストグラムの最大数 N に達するまで，ばらつきが大きい領域を中心に細分化を続ける．N に達した後は，ヒストグラムの木構造を固定して，カウントのインクリメントだけを行う．

問題となるのが，具体的なばらつきの検出方式である．ここでは，

- 統計的に明確な指標に基づいていること
- 効率的に実現できること

の 2 つを重視する．そのため，本研究では χ^2 適合度検定 [10] を用いる．対象となる領域を 4 分割したときのカウントを，

x_{00}	x_{01}
x_{10}	x_{11}

のように x_{00}，x_{01}，x_{10}，x_{11} とおく．ここでの帰無仮説は，「各セルにデータが入る確率は一様に 1/4 である」であり，χ^2 値は，

$$\bar{x} = \frac{x_{00} + x_{01} + x_{10} + x_{11}}{4} \qquad \chi^2 = \sum_{c \in \{00,01,10,11\}} \frac{(x_c - \bar{x})^2}{\bar{x}}$$

で求められる．この値は自由度 4 – 1 = 3 の χ^2 分布に従う．たとえば有意水準 0.05 の場合，χ^2 統計の表から 7.815 であるため，$\chi^2 > 7.815$ になった時点で分布が一様ではないと判断できる．

ただし，χ^2 適合度検定を用いる場合は，データが少ない場合に注意が必要である．特に上式において x_c の値が非常に小さい場合が問題となる．本手法では，データのカウント数が小さい場合には χ^2 適合度検定ではなく，ロバストなノンパラメトリックな手法である 2 項検定の拡張手法を用いる [5].

5.4 ビットマップ併用方式による構築

近似方式のバリエーションとして，ビットマップを併用する方式を考える．ここでいうビットマップは，粗い解像度の論理ヒストグラムを実体化したものである．ただし，キューブセルの値はバイナリであるとし，値が 1 である場合，対応する遷移シーケンスが 1 つ以上存在することを表す．後述の実験では，空間分割レベル 3 のビットマップ（32KB 相当）を利用した．

6. 評価実験

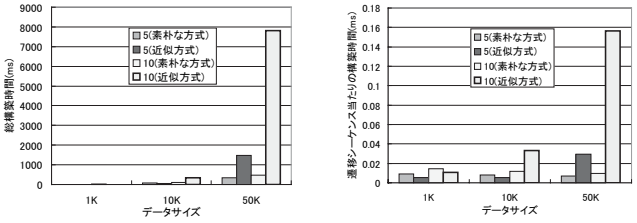
6.1 実験データと実験環境

本実験では，Brinkoff により作成された移動オブジェクトデータ生成ソフトウェアにより生成されたデータを利用する [1]．このシステムは実際の市街地の道路ネットワーク上を自動車などが移動する際の移動の状況をシミュレーションするものである．今回扱うデータは，このシステムで提供されているドイツ Oldenburg 市の市の中心部の道路ネットワークをもとに生成している．パラメータ設定の詳細は [5] にある．実験では，Pentium4 3.2GHz の CPU，1 GB のメモリをの PC を使用して実験を行った．

6.2 ヒストグラム構築時間

図 5 に構築時間を示す．図 5 中の 5, 10 は，ヒストグラムの構築の最大空間分割レベルを，1K は 1000 件，10K は 1 万件，50K は 5 万件のデータ量を表す．

素朴な方式については，データサイズに構築時間がほぼ比例することが見てとれる．一方，近似方式ではデータサイズが大きくなるにつれ，総構築時間が増加している（ビットマップ併用方式も同様）．近似方式では，頻度分布ばらつきの判定を行うため，データサイズが大きいと χ^2 適合度検定を行う頻度も増えていることが原因と考えられる．また，ノードの分割処理にも時間を要していると考えられる．ただし，現状でも遷移シーケンス当たりの構築時間は 0.16 ms と小さいため，まだ余裕がある数値ともいえる．



(a) 構築時間　(b) シーケンス当たりの構築時間

図 5: 構築時間
Fig. 5: Construction Time

6.3 ヒストグラムのサイズ

ヒストグラムのサイズを図 6 に示す．どの方式も，サイズの増え方はデータ数に比例している．素朴な方式のヒストグラムサイズは他の方式に比べて非常に大きいことがわかる．すなわち，サイズに関しては近似方式とビットマップ併用方式が有効といえる．

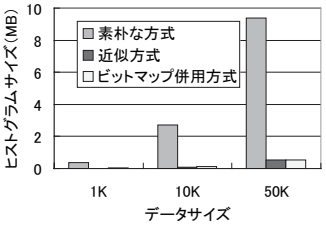
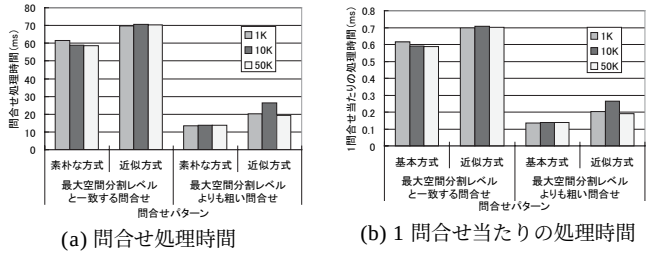


図 6: ヒストグラムのサイズ
Fig. 6: Histogram Size

6.4 問合せ処理時間

2 次のマルコフ遷移の移動ヒストグラムに対し，問合せ 100 個をランダムに与えた際の全体の問合せ処理時間を調べる．ただし，1) 遷移シーケンスの各セルのレベルが最大空間分割レベルと一致する場合（例：$909876^{(10)} \rightarrow 397555^{(10)} \rightarrow 399468^{(10)}$），2) 各セルのレベルが最大空間分割レベルよりも粗い場合（例：$2^{(2)} \rightarrow 2^{(2)} \rightarrow 2^{(2)}$, $53662^{(9)} \rightarrow 66816^{(9)} \rightarrow 109748^{(9)}$) について検証する．問合せパターンごとの問合せ処理時間を図 7 に示す．



(a) 問合せ処理時間　(b) 1 問合せ当たりの処理時間

図 7: 問合せ処理時間
Fig. 7: Query Processing Time

どの手法も，データサイズに依存せずにほぼ一定の処理時間を要する．これは，問合せ処理が全般的に高速であることが原因であると考えられる．近似方式の方が，実際の作業が多いため，若干処理時間が大きくなっている．

6.5 精度

近似方式のヒストグラムの精度評価を行う．ACT_i を遷移シーケンス i の真のカウント値（素朴な方式による），EST_i を近似方式のカウント値とする．精度を表す指標を以下の相対誤差で定義する．

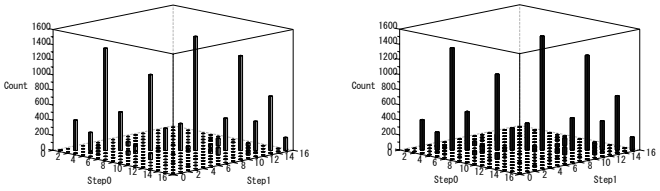
$$\sqrt{\frac{1}{(2^{2P})^{n+1}} \sum_{i=1}^{(2^{2P})^{n+1}} \left(\frac{ACT_i - EST_i}{ACT_i} \right)^2}$$

上式において，ACT_i が 0 になることもありうる（実際にはその遷移シーケンスが出現しなかった場合）．そこで，上記の精度指標式にラプラス推定による補正を以下のように行う．

$$\overline{ACT}_i = \frac{SeqC_i + 1}{S + B} \times S, \qquad \overline{EST}_i = \frac{SeqC'_i + 1}{S + B} \times S$$

$SeqC_i$，$SeqC'_i$ は，i 番目の遷移シーケンスのカウント，S $(= \sum SeqC_i)$ はデータサイズ，$B\,(= R^{n+1})$ はバケット数を表している．

素朴な方式と近似方式の精度評価の前に，それぞれの方式によるカウントを図 8 に示す．これは 1 次のマルコフ連鎖の遷移シーケンス 10,000 件を対象とした，空間分割レベル 2 の全問合せ（256 個）の結果である．素朴な方式と近似方式のカウントが見た目上はほぼ同様に分布していること，同一セル内（3 → 3 など）の遷移が頻繁に出現していることがわかる．



(a) 素朴な方式によるカウント　(b) 近似方式によるカウント

図 8: カウント数の分布 ($n = 1$, $S = 10000$, $P = 2$)
Fig. 8: Distribution of Counts

続いて，素朴な方式によるカウントから近似方式によるカウントを引いて絶対値をとったものを図 9 に示す．もともとの各遷移シーケンスのカウント値は，多いものでは千数百に達することを考えれば，誤差は十分小さいと考えられる．

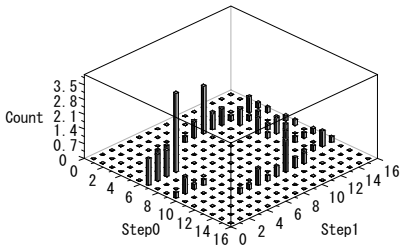
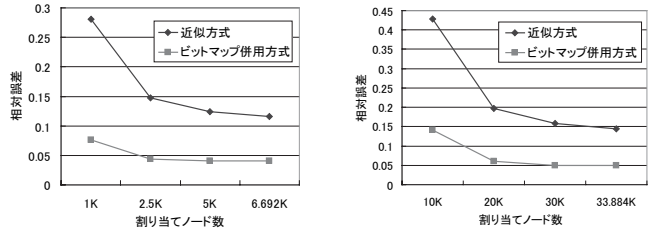


図 9: カウントの差分の絶対値
Fig. 9: Absolute Values of Count Differences

近似方式とビットマップ併用方式の相対誤差（ラプラス推定による補正含む）を図 10 に示す．これは，2 次のマルコフ連鎖の遷移シーケンス 10,000 件，および 50,000 件を対象とした空間分割レベル 3 の全問合せ（262,144 個）の結果である．割り当てノード数の 6.692K，33.884K は最大割り当てノード数である．



(a) $S = 10,000$, $n = 2$, $P = 3$　(b) $S = 50,000$, $n = 2$, $P = 3$

図 10: 精度
Fig. 10: Precisions

いずれの方式も，割り当てノードが増えるにつれて精度が向上している．近似方式の誤差率は 10% から 15% ぐらいに収束している．それに対して，ビットマップ併用方式の誤差率は 5% に収束し，近似方式に比べ精度がいいことがわかる．ビットマップの利用が精度向上に貢献することがわかる．

7. まとめ

本稿では，移動ヒストグラムの論理的な表現，および木構造による物理的な実装方式について述べた．実装方式として，素朴な方式，近似方式，ビットマップ併用方式の 3 種類のヒストグラムについて説明し，それらの性能を，ヒストグラムの構築時間，ヒストグラムのサイズ，問合せ処理時間，精度などについて評価した．評価実験から，特にビットマップ併用方式がサイズと精度のトレードオフの面で効果的であることが検証できた．

今後の課題として，実データを用いた本手法の有効性の検証などが考えられる．また，他の効率的な実装方式の開発，ビットマップを用いたカウントの推定手法の詳細化なども課題として挙げられる．

[謝辞]

本研究の一部は，日本学術振興会科学研究費 (15300027, 16500048)，および，文部科学省科学研究費特定領域研究 (16016205) による．

[文献]

[1] T. Brinkhoff. A framework for generating network based moving objects. *GeoInfomatica*, No. 6(2), pp. 153–180, 2002.

[2] Y. Choi and C. Chung. Selectivity estimation for spatio-temporal queries to moving objects. *Proc. ACM SIGMOD*, pp. 440–451, 2002.

[3] Y. Ioannidis. The history of histograms (abridged). *Proc. VLDB*, 2004.

[4] 町田陽二, 石川佳治, 北川博之. 移動軌跡ストリームデータのためのインクリメンタルなヒストグラム管理手法. 日本データベース学会 Letters, 4(2), pp. 45–48, 2005 年.

[5] 町田陽二, 石川佳治, 北川博之. マルコフ連鎖モデルに基づく動的な移動ヒストグラム構築手法. データ工学ワークショップ (DEWS), 2006 年.

[6] S. Shekhar and S. Chawla. *Spatial Databases*. Prentice Hall, 2002.

[7] J. Sun, et al. Querying about the past, the present, the future in spatio-temporal databases. *Proc. ICDE*, 2004.

[8] Y. Tao, et al. Selectivity estimation for predictive spatio-temporal queries. *Proc. ICDE*, 2003.

[9] G.J.G. Upton and B. Fingleton. *Spatial Data Analysis by Example, Volume II: Categorical and Directional Data*. John Wiley & Sons, 1989.

[10] 矢島美寛, 廣津千尋. 自然科学の統計学. 東京大学出版会, 1996 年.

町田 陽二　Yoji MACHIDA
2006 年筑波大学理工学研究科修了，同年，(株) 日立情報システムズ入社．時空間データベースに興味を持つ．日本データベース学会正会員．

石川 佳治　Yoshiharu ISHIKAWA
名古屋大学情報連携基盤センター教授．データベース，データ工学，情報検索等に興味を持つ．日本データベース学会，情報処理学会，電子情報通信学会，人工知能学会，ACM，IEEE CS 各会員．

北川 博之　Hiroyuki KITAGAWA
筑波大学システム情報工学研究科コンピュータサイエンス専攻教授．異種情報源統合，データマイニング，文書データベース，WWW の高度利用などの研究に従事．日本データベース学会，情報処理学会，電子情報通信学会，日本ソフトウェア科学会，ACM，IEEE CS 各会員．著書に「データベースシステム」（昭晃堂）など．