索引付けされた移動軌跡データからの効率的な移動統計量抽出法

An Efficient Mobility Statistics Extracting Method for Indexed Spatio-Temporal Datasets

塚本 祐一[1] 石川 佳治[2] 北川 博之[3]

Yuichi TUKAMOTO Yoshiharu ISHIKAWA Hiroyuki KITAGAWA

空間情報利用の急速な進展および携帯機器などの普及から，時空間データベースの研究分野では，移動するオブジェクトやユーザの移動状況をデータベースに蓄積し効率よく管理するための研究が盛んに進められている．本稿では，蓄積された移動情報データから移動統計量を高速に抽出し，対話的な移動状況の分析を支援するための手法を提案する．セルの集合に分割された空間上を時間の経過につれてオブジェクトが移動する状況を捉えるための統計量として，マルコフ連鎖モデルが存在する．本研究では，空間索引R-木に蓄積された移動オブジェクトの移動軌跡データから，マルコフ連鎖モデルにおける遷移確率を効率的に求めるための手法を示す．遷移確率の導出を空間索引を用いた制約充足問題の処理に帰着させ，空間索引の内部情報を利用して効率的に処理を行う点が特徴となっている．

With the recent progress of spatial information technologies and mobile computing technologies, spatio-temporal databases which store information on moving objects including vehicles and mobile users have gained a lot of research interests. In this paper, we propose an algorithm to extract mobility statistics from indexed spatio-temporal datasets for the interactive analysis of huge collections of moving object trajectories. We focus on a mobility statistics value called the Markov transition probability, which is based on a cell-based organization of a target space and the Markov chain model. The proposed algorithm efficiently computes the specified Markov transition probabilities with the help of a spatial index R-tree. We reduce the statistics computation task to a kind of constraint satisfaction problem that uses a spatial index, and utilize internal representation of R-tree in an efficient manner.

[1] 学生会員 筑波大学大学院システム情報工学研究科
yuichi@kde.is.tsukuba.ac.jp
[2] 正会員 筑波大学電子・情報工学系
ishikawa@is.tsukuba.ac.jp
[3] 正会員 筑波大学電子・情報工学系
kitagawa@is.tsukuba.ac.jp

1. はじめに

今日では，地図データの電子化やGPSデータなどの空間測位技術の進展などにより，空間情報利用の拡大が急速に進んでいる．また，携帯電話やモバイルPCなどの普及により，モバイルユーザを対象としたデータ管理技術がより重要となってきている．これを受け，時空間データベース（spatio-temporal database）の研究分野では，移動するオブジェクトやユーザのためのデータ提供技術の研究が盛んに進められている[1]．

大量の移動オブジェクトに関する移動情報の蓄積や問合せを目的とした移動オブジェクトデータベースでは，問合せの効率化に関する研究が重要な課題の1つとなっている．

加えて，時空間データベースからの統計情報の抽出に関しても近年いくつか提案がなされている．時空間データに関する統計量は，データベース処理の効率化のみならず，蓄積された移動状況データをもとに移動分析（mobility analysis）を行う際においても有用である[2]．時空間データの利用拡大により，移動分析のための統計量を効率よく抽出することがより重要となると考えられる．

以上の背景をもとに，本研究では時空間データベースから移動オブジェクトの移動状況に関する統計情報を抽出する手法を提案する．移動に関する統計量として，本研究ではマルコフ連鎖（Markov chain）モデルに基づく移動統計量を考える．

時空間データ分析におけるマルコフ連鎖モデルは，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，人や物の時空間的な移動傾向を把握するために用いられる[2]．この統計情報により，ある時点である位置にいるオブジェクトが次の時点でどこに移る可能性が高いといった予測が可能となる．

本研究では特に，移動オブジェクトの移動軌跡が空間索引R-木に蓄積されている状況を想定し，R-木から効率的にマルコフ連鎖の遷移確率を推定する手法を示す．R-木から遷移確率を推定する問題は，一種の制約充足問題（constraint satisfaction problem）として定式化できる．

2. マルコフ過程モデルに基づく移動統計量

図1に示すように，空間がセルに分割されているとする．各セルは矩形でなくてはならないが，サイズは必ずしも均一でなくてよいとする．各セルにはセル番号が付けられていて，番号によりセルを特定できるものとする．図1は，時刻$t=\tau$でセルc_0にいたオブジェクトAが，次の時刻$t=\tau+1$でセルc_1に，そして$t=\tau+2$の時点でセルc_2に移動した状況を示している．

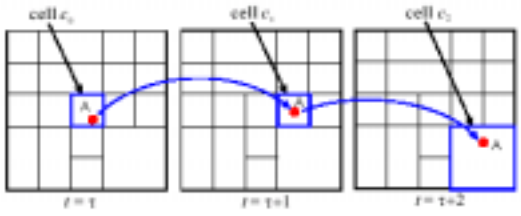

図1　マルコフ過程モデルの概念

Aのように，セルc_0に現在いるオブジェクトが次の時点でセルc_1に移動する確率$\Pr(c_1|c_0)$を時空間データベース中のデータを用いて予測したいとすると，

$$\Pr(c_1|c_0)=\frac{\sum_{t=0}^{T-1}|\,\mathrm{objs}(c_0,t)\cap\mathrm{objs}(c_1,t+1)\,|}{\sum_{t=0}^{T-1}|\,\mathrm{objs}(c_0,t)\,|} \qquad (1)$$

と計算できる．ただし，$\mathrm{obj}(c_i,t)$は，時刻t にセルc_iの領域に含まれているオブジェクトの集合を返す関数である．この式では，各時点$t=0,\cdots,T-1$でc_0にいたオブジェクトの総数を分母とし，そのようなオブジェクトのうち，次の時点でc_1に移ったオブジェクトの総数を分子としている．

確率$\Pr(c_1|c_0)$は，現在の状態（セルc_0）に依存して次の状態（セルc_1）を予測するという意味で，1次のマルコフ過程の遷移確率に相当する．これを一般化すると，n次のマルコフ過程による遷移確率，すなわち，単位時間ごとに$c_0,c_1,\cdots,c_{n-1}$とセルを移動してきたオブジェクトが次の時点でセルc_nを訪れる確率 $\Pr(c_n|c_0,\cdots,c_{n-1})$と表され，その推測値は

$$\Pr(c_n|c_0,\mathrm{K},c_{n-1})=\frac{\sum_{t=0}^{T-1}|\,\mathrm{I}_{i=0}^{n}\mathrm{objs}(c_i,t+i)\,|}{\sum_{t=0}^{T-1}|\,\mathrm{I}_{i=0}^{n-1}\mathrm{objs}(c_i,t+i)\,|} \qquad (2)$$

という一般形で与えられる．なお，上の議論では移動オブジェクトの移動が定常的（stationary）な過程に従い，時刻によって遷移確率が変化しないと仮定している．

マルコフ遷移確率を効率的に求めることができるならば，移動オブジェクトが次の時点でどのセルに移動する可能性が高いかという，一種の経路予測を行うことが可能となる．また，ある時刻$t=\tau$における移動オブジェクト集合の移動状況データがまとめて与えられたとき，$t=\tau+1,2,\ldots$ における移動状況がどうなるかをシミュレーションすることも可能となる．

3. 空間索引による索引付け

3.1 移動軌跡データのための索引手法

2節で述べたような推定を行うには，時刻t にセルc 内に存在したオブジェクトの集合$\mathrm{obj}(c,t)$をいかに効率よく見つけるかが鍵となる．そのためには，索引の適切な利用が不可欠である．本研究では，空間索引として一般的なR-木の利用を想定し，それに特殊な拡張を必要としない手法を特に検討する．R-木による移動軌跡データの管理に関してすでに提案されているアプローチとしては，3D R-木（時間の次元を加えて3次元で軌跡を表す）[3]やSTR-木（移動軌跡の検索を目的とする）[4]がある．

3.2 索引付けの具体例

以下では，移動軌跡を点の集合として離散的に表現する場合の，本研究のアルゴリズムが想定する索引付け方式について述べる．

例として，1次元空間でのオブジェクトの移動の例を考える．図2は，オブジェクトA，Bが時刻$t=0$から$t=T\ (=8)$まで x 軸上を移動する様子を示している．移動経路は曲線で表現されている．実際の移動軌跡はこのように複雑であるが，計算機上での表現には何らかの近似が必要である．図2に示した経路上の点は，各時刻において移動軌跡をサンプリングしたものである．この近似により，各オブジェクトの経路は時刻と位置のペアの列で表現できる．このような表現手法を**点による表現**と呼ぶ．GPS により一定期間ごとに移動オブジェクトの位置を検出する場合は，このような表現をすることが自然である．

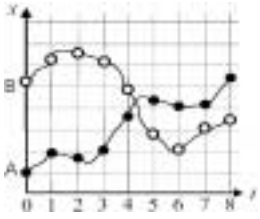

図2　点による移動軌跡の表現

点による表現を用いる場合，移動オブジェクトo のある時刻$t=\tau\ (\tau=0,1,\cdots,T)$における位置が$[x_1,\cdots,x_d]$という$d$ 次元空間上の点で表される場合，$d+1$ 次元空間のR-木を構築し，各時刻t に対するo の情報を$[x_1,\cdots,x_d,t]$という$d+1$ 次元ベクトルと表現し，o のオブジェクトIDとペアでR-木に挿入すると索引が構築できる．これは3D-R木の一般化である．こうすることで，点による表現に対して時刻t においてセルc に含まれている移動オブジェクトのIDを検索することが可能となる．

4. 素朴な遷移確率推定アルゴリズム

4.1 問題の定式化

ここで，時空間索引を用いて，式(2)に示したn 次のマルコフ遷移確率の推定を行うことを考える．ただし，特定のセルの組合せ$c_0,\cdots,c_n$ に対する遷移確率の推定ではなく，以下のように問題を一般化する．

遷移確率の推定問題

$n+1$ 個のセルの集合 $C_0=\{c_{0,1},\cdots,c_{0,|c0|}\},\cdots,C_n=\{c_{n,1},\cdots,c_{n,|cn|}\}$が与えられたとき，任意のセルの組み合わせ$(c_0,c_1,\cdots,c_n)\in C_0\times\cdots\times C_n$に対し，$\Pr(c_n|c_0,\cdots,c_{n-1})$の値が未定義でなければその値を出力する．

□

4.2 素朴なアルゴリズム

ここでまず，特定のセルの組み合わせ$c_0,\cdots,c_n$について$\Pr(c_n|c_0,\cdots,c_{n-1})$の推定を行うことを考える．この確率は以下の2つの集合S, Q を求めることで計算できる．

(1) ある時刻$t=\tau$ ($\tau=0,1,\cdots,T-n$)においてセルc_0にいて，かつ，$t=\tau+1$においてセルc_1にいて，$\cdots$，かつ$t=\tau+n-1$においてセルc_{n-1}にいたオブジェクトの集合S

(2) S に含まれるオブジェクトのうち，$t=\tau+n$ においてセルc_nにいたオブジェクトの集合Q

先の推定問題に対する素朴なアルゴリズムはこのアイデアをもとに導けるが，R-木の検索数がT に比例するという性質がある[6]．

5. 効率的な遷移確率推定アルゴリズム

5.1 制約充足解の探索アルゴリズム

5.1.1 メインルーチン

$\Pr(c_n|c_0,\cdots,c_{n-1})$の推定を以下の3ステップで行う．

(1) 単位時間ごとに$c_0,\cdots,c_{n-1}$に存在したオブジェクトの総数をR-木を用いてカウント

(2) 単位時間ごとに$c_0,\cdots,c_n$に存在したオブジェクトの総数をR-木を用いてカウント

(3) ステップ2のカウント数÷ステップ1のカウント数により$\Pr(c_n|c_0,\cdots,c_{n-1})$を計算

R-木の探索はステップ(1)，(2)でそれぞれ1回ずつ，合計2回しか行われないため，先の素朴な手法に比べ効率的な処理が達成できることになる．メインルーチンのアルゴリズムを図3に示す．*Scount*, *Qcount*は文字列キーのハッシュ変数であり，$c_0\#\cdots\#c_n$は$c_0,\cdots,c_n$をそれぞれ文字列化した後，それらを連接してできる文字列である．関数FC_countは次に説明する．

```
Procedure FC_Estimation(n, root, level, (C0,...,Cn), max_dist)
Input: n: マルコフ遷移の次数, root: R-木のルートノード
       level: R-木のレベル数, (C0,…,Cn)：セル集合のリスト
       max_dist: 単位時間に移動可能な最大距離
Output: 値が未定義でない Pr(cn|c0,…,cn-1)の推定値のリスト
1. for j := 0 to n do nodes[j] := root;
2. Scount := FC_count(n–1, level, nodes, (C0,…,Cn-1), max_dist);
3. Qcount := FC_count(n, level, nodes, (C0,…,Cn), max_dist);
4. foreach (c0,…,cn)   C0×⋯×Cn do
5.   if Scount{c0#⋯#cn-1} > 0 then
6.     output(c0,…,cn, Qcount{c0#...#cn}/Scount{c0#...#cn-1});
```

図3 メインルーチン

5.1.2 集計処理関数

集計処理を行う関数FC_count（図4）について説明する．この関数は，R-木を用いた空間的制約の充足解の探索手法[5]を拡張したものである．このアプローチでは，R-木上をルートからリーフ方向に制約を満たす解を探していく．その際，バックトラックや枝刈りをしながら充足解をもれなく探索する．

4行目に現れるsp_overlap(C_j, v)関数は，セル集合C_j に含まれるセルのいずれかとv が交わるかどうかを判定する述語であり，解候補の絞込みに役立つ．

```
Function FC_count(n, level, nodes, (C0,…,Cn), max_dist)
Input : n：遷移の次数, level：現在処理しているR-木のレベル
        nodes：n+1要素のノード配列でnodes[j]はセルcjに対応
        (C0,…,Cn)：セル集合のリスト
        max_dist：単位時間に移動可能な最大距離
Output : count：集計結果を入れたハッシュ表
1.  for j := 0 to n do // 各制約に対する初期解集合を設定
2.    child_set := ∅;
3.    foreach v   nodes[j].children do
4.      if  sp_overlap(Cj, v) then  // v は Cj と空間的な交わりを持つ
5.        child_set := child_set ∪ {v};
6.    dom[0][j] := child_set;   // 子ノードの集合を代入
7.  end
8.  i := 0;  // 現在着目している制約に対するインデックス
9.  while true do
10.   if  dom[i][i].isempty  then  // 空集合になった
11.     if  i = 0 then return  count;  //手続きの終了
12.     else
13.       i −−;  continue;  //バックトラック
14.     end
15.   else
16.     new_val := get_next(dom[i][j]);  // 次の要素を取り出す
17.     inst[i].value := new_val;  //  Ci の制約に対する解の候補とする
18.     if level ≥ 1 then  //  inst[i]の値がとり得る有効な時区間を設定
19.       inst[i].trange := t_overlap(new_val.trange , [i, T–n+i]);
20.     else inst[i].trange := new_val.trange;
21.   end
22.   if  i = n then  // 現在の解候補で制約が充足された
23.     if level ≥ 1  then  // 非リーフノードの場合
24.       for k := 0  to n  do refs[i] := inst[k].value;
25.       FC_count(n, level–1, refs, {C0,…,Cn});
26.     else  // リーフノードの場合：充足解に対するカウントを増やす
27.       count{cell(inst[0].value)#...#cell(inst[n].value)}++;
28.     end
29.   else  // 制約充足の途中の場合
30.     if check forward(i, n, level, dom, inst, (Ci+1,…,Cn), max_dist) then
31.       i++;  // 充足解が存在した．次はCi+1の制約充足へ
32. end
```

図4 Forward Checking に基づく集計関数

19行目のt_overlap(P, Q)は，時区間の集合P, Qの交わりをとる関数である．たとえば，t_overlap({[1, 3], [4, 8], [10, 13]}, {[2, 6]}) = {[2, 3], [4, 6]}となる．関数check_forwardは次節で説明する．

関数中で使われる$(n+1)\times(n+1)$配列dom は集合を要素とする配列であり，$dom[i][j]$はセル集合C_i に関する制約条件の充足を検証している段階におけるセル集合C_j に関する解の候補の集合を保持する．なお，本アルゴリズムでは，指定された$C_0,\cdots,C_{n+1}$に対応する$n+1$個の制約条件の充足をこの順に処理していくと想定している．n+1要素の配列$inst$は，制約を充足するある解を探索している途中の状況で，部分的な探索解を保持するために用いている．i 番目の制約C_i を処理している状況では，$inst[0],\cdots,inst[i-1]$に部分解が入っている．

5.1.3 Forward Checkingの処理

関数check_forwardは, 現在のi番目の制約までの解候補をもとにi+1番目以降の制約を満たす解候補が存在するかどうかを調べることで, 解候補を調べる際に, いくつかの枝刈り条件に応じてあらかじめ解候補を絞り込むためのものである. この関数を用いることで, 解の構成要素を含まない枝を探索するということがなくなるため, 集計の効率化を図ることができる. このような処理を[5]では *forward checking* と呼んでいる.

枝刈り条件としては, (1)対象としているオブジェクトIDと移動軌跡のオブジェクトIDが一致しているか, (2)移動軌跡が有効な時区間に含まれているか, (3)移動軌跡が対象とするセル内に含まれているか, (4)オブジェクトが空間的に単位時間で移動できる距離であるか, という4つがある.

```
Function check_forward(i, n, level, dom, inst, (C_{i+1},…C_n), max_dist)
Input: i：現在処理している制約の番号, n：遷移の次数
       level：現在処理している R-木のレベル
       dom：解候補集合の配列, (C_{i+1},…,C_n)：セル集合
       max_dist：単位時間に移動可能な最大距離
Output：現在の inst[0],…,inst[i]に対し, inst[i+1],…,inst[n]の候補があればtrue
        そうでない場合 false
Note：副作用として dom の値を変更
1.  for j := i + 1 to n do   // 未チェックの各制約について
2.    dom[i+1][j] := dom[i][j];   // 解の候補集合を初期化
3.    foreach v    dom[i+1][j] do
4.      if (level = 0) and (inst[0].value.id ≠ v.id) then goto 15;
5.        // v は別のオブジェクトに対する移動軌跡であった
6.      vrange := t_overlap(v.trange, [j, T–n+j]);
7.      if vrange = ⊥ then goto 15;   // vrange が空であった
8.      if t_overlap(shift(vrange, –j), inst[0].trange = ⊥) then goto 15;
9.        // 時間的な制約条件を満たさない
10.     if not sp_overlap(C_j, v) then goto 15;
11.       // v はセル集合 C_j の領域と空間的に重ならない
12.     if sp_dist(inst[i].value, v) > max_dist × (j – i) then goto 15;
13.       // inst[i].value から v までは j–i 時間内に移動できない
14.     continue;   // v は条件を満たした. 次の v のチェックへ進む
15.     dom[i+1][j] := dom[i+1][j] – {v};   // v を候補から削除
16.   end
17.   if dom[i+1][j] = ∅ then return false;   // 充足解の候補がない
18. end
19. return false;
```

図5 Forward Checkingのための関数

12行目に現れるsp_dist関数は, ノードのMBRと点オブジェクト, あるいは点オブジェクトどうしの最短距離を求める距離関数である. この関数の処理の結果, 配列domの値が副作用として変更される. その結果を図4に示した集計関数が利用する.

以上, アルゴリズムを図3〜5で示したが, 紙面数の都合から説明が足りない部分も多い. 特に, 図5の時間的制約による解候補の絞込み (6〜9行目) についてはアイデアを含め, 提案手法を詳しく紹介できなかった. 詳細については論文[6]を参照していただきたい.

6. まとめ

本稿では, 時空間データベースの情報からマルコフ過程モデルに基づく移動統計量を推定するためのアルゴリズムを示した. 空間索引の内部情報を利用し, 計数処理の効率化を図った点が特徴となっている. 今後の課題として, 1) 実験に基づく評価, 2) データベース中のデータの分布に合わせたセルの適合的な分割, 3) 非定常なマルコフ過程への拡張などが挙げられる.

[謝辞]

査読者の方々からは貴重なコメントをいただきました. ここに感謝いたします. 本研究の一部は, 日本学術振興会科学研究費基盤研究(B)(12480067), 若手研究(B)(14780316), および文部科学省科学研究費特定領域研究(14019009), 財団法人セコム科学振興財団研究奨励金による.

[文献]

[1] C.S. Jensen (ed.), “Special Issue: Indexing of Moving Objects”, *IEEE Data Engineering Bulletin*, 25(2), Jun. 2002.

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

[3] M.A. Nascimento, J.R.O. Silva, and Y. Theodoridis, “Evaluation of Access Structures for Discretely Moving Points”, *Proc. STDBM*, pp. 171–188, 1999.

[4] D. Pfoser, C.S. Jensen, and Y. Theodoridis, Novel “Approaches to the Indexing of Moving Object Trajectories”, *Proc. of VLDB*, 2000.

[5] D. Papadias, N. Mamoulis, and Vasilis Delis, “Algorithms for Querying by Spatial Structure”, *Proc. of VLDB*, 1998.

[6] 石川, 塚本, 北川, 索引付けされた移動軌跡データからの移動統計量の抽出法について, 第14回データ工学ワークショップ (DEWS 2003), 2003年.

塚本 祐一 Yuichi TSUKAMOTO
筑波大学大学院システム情報工学研究科在学中. 2002年筑波大学第三学群情報学類卒業. 時空間データベースの研究に従事. 日本データベース学会学生会員.

石川 佳治 Yoshiharu ISHIKAWA
筑波大学電子・情報工学系講師. 1994年筑波大学大学院博士課程工学研究科単位取得退学. 博士（工学）. 空間データベース, 文書データベース, 情報検索などに興味を持つ. ACM, IEEE-CS, 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, 各会員.

北川 博之 Hiroyuki KITAGAWA
筑波大学電子・情報工学系教授. 1980年東京大学大学院理学系研究科修了. 理学博士（東京大学）. 異種情報源統合, 文書データベース, WWWの高度利用等の研究に従事. 著者「データベースシステム」（昭晃堂),「Unnormalized Relational Data Model」(共著, Springer-Verlag)等. ACM, IEEE-CS, 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, 各会員.