【VLDB2010勉強会】

Session 22: Moving Object Databases

担当: 眞野 将徳(名古屋大学)

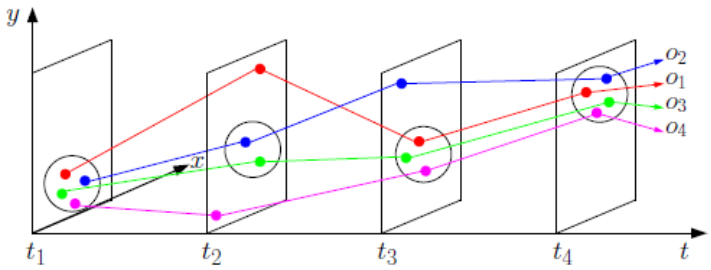
論文一覧

- **[1] Swarm: Mining Relaxed Temporal Moving Object Clusters**
  - Zhenhui Li (University of Illinois at Urbana-Champaign, United States of America), Bolin Ding (University of Illinois at Urbana-Champaign, United States of America), Jiawei Han (University of Illinois at Urbana-Champaign, United States of America), Roland Kays (New York State Museum, United States of America)
- **[2] An Adaptive Updating Protocol for Reducing Moving Object Databases Workload**
  - Su Chen (National University of Singapore, Republic of Singapore), Beng Chin Ooi (National University of Singapore, Republic of Singapore), Zhenjie Zhang (National University of Singapore, Republic of Singapore)
- **[3] Shortest Path Computation on Air Indexes**
  - Georgios Kellaris (Singapore Management University, Republic of Singapore), Kyriakos Mouratidis (Singapore Management University, Republic of Singapore)

Swarm: Mining Relaxed Temporal Moving Object Clusters

図は[1]より引用

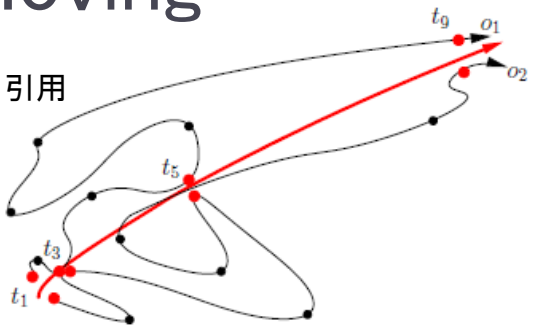
- 移動オブジェクトのクラスタリング
  - 既存手法は常に一緒に行動していなければならない
  - もう少し緩いクラスタリングもあるのでは
- swarm (O, T), $|O| \geq min_o$, $|T| \geq min_t$
  - O:移動オブジェクト集合
  - T:タイムスタンプ集合



- (例)
  - o_1(赤)、o_3(緑)、o_4(桃)はt_1,t_3,t_4で同一グループにいる
  - $min_o = 2$, $min_t = 3$ なら
    ($\{o_1, o_3, o_4\}$, $\{t_1, t_3, t_4\}$)はswarm

Swarm: Mining Relaxed Temporal Moving Object Clusters

図は[1]より引用

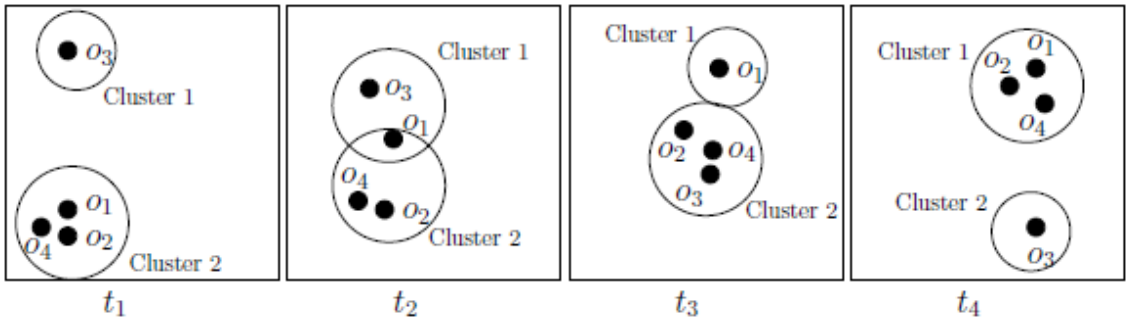


- 軌跡でも同様の定義ができる
- 右の二つの軌跡
  - min_o = 2, min_t = 3 なら($\{o_1, o_2\}$, $\{t_1, t_3, t_5, t_9\}$)はswarm
- ObjectGrowth
  - オブジェクト空間における深さ優先探索
  - 最大時間集合Tmax(O)
    - あるオブジェクト集合Oについて、Oが同一のクラスタであるタイムスタンプの最大集合 (例:上の軌跡のTmax({o1, o2}) = {t1, t3, t5, t9}
  - 操作
    - 前方枝刈り
      - Tmax(O) ≦ min_tならば、それ以下の部分木を刈り取る
    - 後方枝刈り
      - (O, Tmax(O)) にいくつかのオブジェクトを加えて発見済みになるなら刈り取り
    - 前方閉塞チェック
      - 一度swarmであると思われたものが、さらによいswarmの部分集合でないかチェックする

例

min_o=2, min_t=2

図は[1]より引用



- Oにオブジェクトを深さ優先で追加
- それについて$T_{max}(O)$を求める
- min_o,min_tを満たす{O, $T_{max}(O)$}がswarm

O: {}
Tmax(O): {t1, t2, t3, t4}

①

O: {o1}
Tmax(O): {t1, t2, t3, t4}

⑦ 以下同様に探索

②

④からo4を除いたもの
前方閉鎖チェック ×

O: {o1, o2}
Tmax(O): {t1, t2, t4}

⑤

O: {o1, o3}
Tmax(O): {t2}
前方枝刈り

⑥

O: {o1, o4}
Tmax(O): {t1, t2, t4}

④からo2を除いたもの
後方枝刈り

③

O: {o1, o2, o3}
Tmax(O): {}

Tmax(O)の要素数がmin_t未満
前方枝刈り

④

O: {o1, o2, o4}
Tmax(O): {t1, t2, t4}

min_oとmin_tを満たす
swarm

O: {o1, o2, o3, o4}
Tmax(O): {}

さらに深いところは当然満たさない

前方枝刈り
　Tmax(O) ≦ min_tならば、それ以下の部分木を刈り取る
後方枝刈り
　(O, Tmax(O)) にいくつかのオブジェクトを加えて 発見済みswarmになるなら刈り取り
前方閉塞チェック
　一度swarmであると思われたものが、さらによいswarmの部分集合でないかチェックする

An Adaptive Updating Protocol for Reducing Moving Object Databases Workload

図は[2]より引用

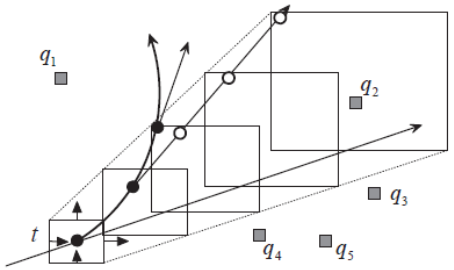
- 時空間データベースにおける良い索引構造
- STSR: Spatio-Temporal Safe Region
  - 位置と速度について、可能性のある値を示す時空間領域

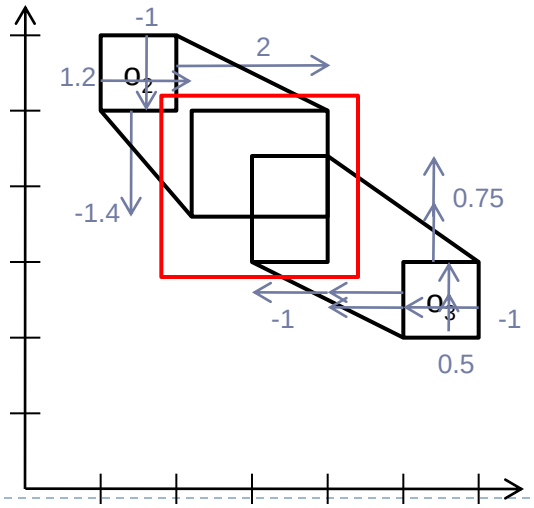
[LR.x⊢, LR.x⊣], LR.y⊢, LR.y⊣]

reference time

expiry time

STSR	LR	VR	t_r	t_e
R(o2)	[1, 2] x [5, 6]	[1.2, 2] x [-1.4, -1]	2	5
R(o3)	[5, 6] x [2, 3]	[-1, -1] x [0.5, 0.75]	1	5





- t = 3 のときのSTSR
- LRからVR$(t-t_r)$だけのばす
- 範囲問合せを対象とする
  - 問合せ結果の予測ができる

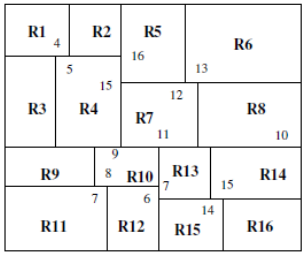
更新

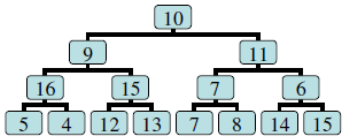
- 効率的な運用のためには更新をどのようにおこなうかも重要
- 能動的更新
  - 各オブジェクトはタイムスタンプごとにSTSRと異なる移動をしていないかチェックする
    - あたらしい移動になったらそのことをサーバに通知
    - サーバはSTSRを更新し、新しいSTSRをオブジェクトに伝える
- 受動的更新
  - 範囲問合せがされたときに発生
  - 予測範囲問合せでは結果の対象となるオブジェクトが、実際には範囲外にいた時にそれにあわせて更新する

Shortest Path Computation on Air Indexes

- 無線環境における最短経路検索
  - ロードネットワークをグラフにし、v_sからv_tへの最短経路
  - 二つの手法を紹介
- EB法（楕円境界法）
  1. ノード群を領域に分解する
     - kd木を利用
  2. リージョン間の最大/最小距離
     - n×n配列で事前に計算しておく
  3. リージョンと距離情報を順番にブロードキャスト
  - $R_s(\in v_s)$から$Rt(\in v_t)$の最短経路
    - Min(Rs, R)+Min(R, Rt) ≦ Max(Rs, Rt)なRは通る可能性あり
  - 例：R1からR5への最短経路検索
    - Max(R1, R5) = 7
    - R2を通るとmin(R1, R2)+min(R2, R5) = 1 + 1 ≦ 7 ・・・ R2は通るかも
    - R3を通るとmin(R1, R3)+min(R3, R5) = 6 + 2 ＞ 7 ・・・ R3は通らない

図は[3]より引用





R1	R2	R3
R4	R5	R6

	R1		R2		R3		R4		R5		R6	
	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max
R1			1	5	6	8	1	4	3	7	8	9
R2	1	5			1	3	2	5	1	2	2	4
R3	4	6	1	3			5	8	2	4	1	4
R4	1	3	2	4	5	8			2	3	4	7
R5	3	6	1	3	2	4	1	3			1	2
R6	7	9	2	4	1	2	5	6	1	3		

Shortest Path Computation on Air Indexes

- NR法
  - R_mのインデックスA^mはn×n配列
  - $A^m_{Ri,Rj}$はR_iからR_jへの最短経路で通る領域R_{nxt}を指す
  - 領域情報とインデックスは順番にブロードキャストされる
  - R_1からR_{25}への探索時に受け取ったA^{12}が右のものだとする
    - 最短経路はR_{13}を経由するかも
    - R_{13}と隣接インデックスA^{14}を取得
    - A^{14}にはR_{14}も経由するかもと
      - R_{14}とA^{15}を取得
    - A15のインデックス
    - R_{19}とA^{20}がブロードキャストされるまで待つ
  - これを繰り返すと必要な領域が判る
  - NR法はEB法より良い性能を示す

図は[3]より引用

12th	R_1	R_2	…	R_{24}	R_{25}
R_1		R_1		R_{13}	R_{13}
R_2	R_1			R_{13}	R_{13}
⋮					
R_{24}	R_{13}	R_{13}			R_{24}
R_{25}	R_{13}	R_{13}		R_{24}	

15th	R_1	R_2	…	R_{24}	R_{25}
R_1		R_1		R_{19}	R_{19}
R_2	R_1			R_{19}	R_{19}
⋮					
R_{24}	R_{19}	R_{19}			R_{24}
R_{25}	R_{19}	R_{19}		R_{24}	

	R_1		R_2		R_3		R_4		R_5
	R_6		R_7		R_8		R_9		R_{10}
	R_{11}		R_{12}		R_{13}		R_{14}		R_{15}
	R_{16}		R_{17}		R_{18}		R_{19}		R_{20}
	R_{21}		R_{22}		R_{23}		R_{24}		R_{25}