基于事件的位置不确定移动对象连续概率 Skyline 查询

付世昌 [1] 董一鸿 [1] 唐燕琳 [1, 2] 陈华辉 [1] 钱江波 [1]

摘 要 Skyline 查询是基于位置服务 (Location based service, LBS) 的一项重要操作, 其目的是发现数据集中不被其他点支配的点的集合. 移动对象在运动过程中, 其位置信息具有不确定性, 导致各数据点间的支配关系不稳定, 从而影响 Skyline 操作. 本文针对以位置不确定移动对象为查询点的 Skyline 查询进行研究, 首先, 定义了查询点移动时各对象间支配概率, 提出了支配概率和 Skyline 概率的微元计算方法. 在此基础上, 提出一种面向不确定移动对象进行连续概率 Skyline 查询的有效算法 U_CPSC. 该算法首先快速计算初始时刻的 p-Skyline 集合; 然后, 定义了两类可能引起 p-Skyline 变动的事件, 通过对这些事件的跟踪计算快速更新 p-Skyline 集合, 无需在移动对象的每一运动时刻去遍历整个数据集, 实现了对 p-Skyline 的连续更新操作, 大大减少了算法的查找和计算开销, 提高了运算效率; 最后, 提出一种静态算法 U_SPSC, 与 U_CPSC 进行了对比试验, 实验结果证明了算法的有效性.

关键词 概率 Skyline, 不确定数据, 移动对象, 支配概率

DOI 10.3724/SP.J.1004.2011.00836

Continuous Probabilistic Skyline Queries for Moving Objects with Uncertainty Based on Event

FU Shi-Chang[1] DONG Yi-Hong[1] TANG Yan-Lin[1, 2] CHEN Hua-Hui[1] QIAN Jiang-Bo[1]

Abstract Skyline queries are an important operator of location based service (LBS), which aim to find all data that are not dominated by any others. The uncertainty of moving objects makes the dominant relationship of data instable, which will affect skyline operator. In this paper, skyline inquires for moving objects with uncertainty are studied. Firstly, the dominant probability between two moving objects is defined. Then it is proposed how to compute the dominant probability and skyline probability by differential element method. A novel effective algorithm U_CPSC is presented to handle continuous probabilistic skyline queries for uncertain moving objects based on these definitions. The initial p-Skyline set is firstly searched by rapid computing. Secondly, two types of events affecting p-Skyline are defined to track and update p-Skyline set continuously instead of re-computing the whole dataset each time. A static algorithm U_SPSC is proposed to compare with U_CPSC. Experiments have positive results that show effectiveness of the proposed algorithm.

Key words Probabilistic skyline, uncertain data, moving objects, dominant probability

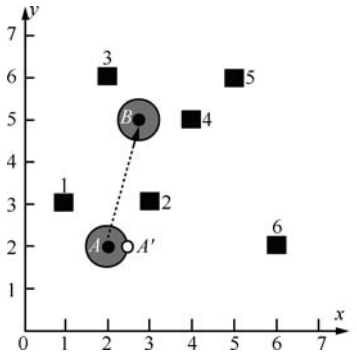
Skyline 查询[1] 是位置服务中的一项重要操作, 其返回数据库中不被其他点支配的点. 它反映目标数据集的整体轮廓且有利于用户查询数据集中感兴趣的目标, 在多目标决策、数据挖掘和数据库可视化等方面有着潜在的应用.

目前已有的针对移动对象的 Skyline 查询都认为移动对象的位置是确定的. 如图 1 所示, 假设图中的点代表的是旅馆的空间位置 (x, y), 表格显示旅馆的价格信息. 不失一般性, 考虑属性值越小越好的情况. 将手机用户作为一个查询点, 图 1 中箭头所指的方向是从时刻 t_1 到时刻 t_2 手机用户从 A 移动到 B. 在时刻 t_1, 查询点位于 $A(2, 2)$, Skyline 集合为 $\{1, 3, 5\}$; 随着查询点的移动, 在时刻 t_2, 查询点位于 $B(3, 5)$, 查询点与各旅馆间的距离由于查询点的移动发生了改变, Skyline 集合也随之变化为 $\{3, 4, 5\}$.

在现实应用中对移动对象的更新操作是间断进行的, 受技术手段 (如 GPS 技术) 的限制, 两次操作之间, 对移动对象位置采集的信息存在一定误差, 即该信息存在若干不确定性. 而且移动对象处于不断运动的过程中, 其在每一时刻的具体位置与采集信息存在误差, 导致采集的数据无法准确表达对象的实际位置, 从而无法准确地表示各数据点间的支配关系. 仍然观察图 1, 由于采集到的位置信息存在误差, 如果手机用户真实的初始位置不是位于 A, 而是位于 $A'(2.5, 2)$, Skyline 集合将变为 $\{1, 2, 3, 5, 6\}$, 与在位置 A 相比 Skyline 集增加了 2 和 6 两个目标点. 这是由于手机用户的位置不确定导致了 Skyline

收稿日期 2010-04-09 录用日期 2011-03-02
Manuscript received April 9, 2010; accepted March 2, 2011
国家自然科学基金 (60973047, 60803021), 浙江省自然科学基金 (Y1080490, Y1091189), 浙江省公益技术应用研究项目 (2010C33149), 宁波市自然科学基金 (2010A610098) 资助
Supported by National Natural Science Foundation of China (60973047, 60803021), Natural Science Foundation of Zhejiang Province (Y1080490, Y1091189), Science and Technology Project of Zhejiang Province (2010C33149), and Natural Science Foundation of Ningbo (2010A610098)
1. 宁波大学信息科学与工程学院 宁波 315211 2. 浙江大学计算机科学与技术学院 杭州 310027
1. College of Information Science and Engineering, Ningbo University, Ningbo 315211 2. College of Computer Science and Technology, Zhejiang University, Hangzhou 310027

集的不精确. 因此, 比较合理的方法是采用移动对象的不确定模型. 以不确定区域 (图 1 中的灰色区域) 表示移动对象的可能位置, 以概率密度函数 $pdf(\boldsymbol{x})$ 表示其真实位置的概率. 在这个例子中, 为了简化问题, 以 $R = 0.5$ 的圆形区域表示移动对象的不确定区域, 并假设 $pdf(\boldsymbol{x})$ 为均匀分布. 移动对象作为查询点, 沿直线作匀速运动. 本文提出了以位置不确定的移动对象作为查询点进行 Skyline 查询的连续更新算法, 根据该方法重新对时刻 t_1 和 t_2 手机用户的 Skyline 集合进行计算. 采用不确定模型后, 通过计算得到: 在时刻 t_1 查询点位于 A 区域时, 各数据点的 Skyline 概率分别是 $\{1.0, 0.5, 1.0, 0, 1.0, 0.5\}$; 在时刻 t_2 查询点位于 B 区域时, 各数据点的 Skyline 概率分别是 $\{0, 0, 1.0, 0.78, 1.0, 0\}$. 假设给定概率阈值 $p = 0.3$, 则大于阈值 p 的数据点 $\{1, 2, 3, 5, 6\}$ 构成时刻 t_1 的 p-Skyline 集, 时刻 t_2 的 p-Skyline 集合为 $\{3, 4, 5\}$.



Hotel	(x, y)	Price
1	(1, 3)	4
2	(3, 3)	6
3	(2, 6)	2
4	(4, 5)	5
5	(5, 6)	1
6	(6, 2)	3

图 1　移动环境下的 Skyline 点

Fig. 1　Skylines in mobile environment

本文研究了位置不确定的移动对象在移动过程中连续概率 Skyline 计算问题. 主要贡献是:

1) 文献 [2−5] 研究了离散的不确定数据的概率 Skyline 查询, 其前提条件是对象间的支配概率相互独立. 本文首次研究连续型不确定数据的概率 Skyline 查询, 对象间的支配概率不独立, 因此, 文献 [2−5] 提出的定义和方法都不适用. 本文充分考虑不确定区域对支配关系的影响, 提出了不确定数据支配概率和 Skyline 概率的微元计算方法.

2) 充分分析各数据点的空间位置对支配关系的影响, 定义了两类可能引起 p-Skyline 集合变动的 Event 事件, 提出了一个有效的不确定移动对象连续概率 Skyline 查询算法 (Continuous probabilistic skyline computation algorithm for uncertain moving object, U_CPSC), 实现对位置不确定移动对象的 p-Skyline 进行连续更新计算, 并在算法中提出三条剪枝规则实现查询优化.

3) 进行了大量的实验研究. 由于本文首次研究连续型不确定数据的连续概率 Skyline 查询, 目前还没有类似的算法, 因此, 提出了一种静态算法 (Static probabilistic skyline computation algorithm for uncertain moving object, U_SPSC), 与 U_CPSC 算法进行了对比实验, 实验结果证明了 U_CPSC 算法的有效性.

本文第 1 节总结 Skyline 计算的相关工作; 第 2 节介绍移动对象不确定模型的相关预备知识; 第 3 节研究了不确定移动对象的连续概率 Skyline 更新策略, 包括支配概率和概率 Skyline 查询的定义, 提出移动对象不确定环境下支配概率和 Skyline 概率的计算方法, 定义了影响概率 Skyline 集合的两种类型的触发事件 Event, 并提出剪枝策略进行算法优化; 第 4 节详细描述了静态算法 U_SPSC 和动态更新算法 U_CPSC, 并对其进行复杂度分析; 第 5 节通过实验验证上述方法的有效性; 第 6 节对全文进行总结和展望.

1　相关工作

Borzsonyi 等[1] 首次将 Skyline 操作引入数据库系统, 提出了 BNL 和 D & C 两种计算方法. 文献 [6] 将原始数据集编码成位图 (Bitmap), 提出了一种基于比特位运算的改进方法, 同时还提出了一种基于索引 (Index) 的方法, 这两种方法均能够在扫描全部数据集之前开始输出 Skyline 结果, 因此, 适合在线计算. Kossmann 等[7] 基于 R-tree 索引结构提出的 NN (Neural network) 算法是渐近、公平的, 而且能够接受外界交互信息, 具有良好的用户友好性. Papadias 等[8] 分析了 NN 方法在 I/O 和存储方面存在的不足, 并对之进行了改进, 提出了 BBS 算法, 这是一种公平、公正的算法, 在计算时间、占用空间、输入输出上都有很好的效率和扩展性, 被认为是当前最优的集中式 Skyline 计算算法.

Skyline 查询也得到了国内学者的广泛关注. 周红福等[9] 提出了一种基于高维空间的在线高效子空间 Skyline 算法 — Csky 算法. 该算法充分利用了一个新颖数据结构 InvertS 的特征, 能够高效地计算出任意子空间上的 Skyline. 孙圣力等[10] 研究了滑动窗口中子空间 Skyline 的计算, 其利用网格索引策略, 采用自顶向下的方式通过两个阶段增量式返回目标子空间上的结果, 开发的剪枝策略和启发式优化算法显著提高了全空间 Skyline 的维护以及子空间 Skyline 的计算效率. 苏亮等[11] 对数据流上自适应的稀疏 Skyline 查询进行了研究, 通过数据维度之间的相关性来自适应地调整查询质量的两个在线算法来实现 Skyline 操作. 文献 [12] 首次对概率数据流上的 Skyline 计算问题进行了研究, 提出了概率定界、逐步求精、提前淘汰与选择补偿等启发式规则对概率数据流上 Skyline 查询算法 (Skyline over probabilistic data stream, SOPDS), 从时间和空间

两方面进行了系统的优化. 黄震华等[13] 利用多维数据对象的同胚评估和偏序格加权技术研究 Skyline 查询的并行处理, 并提出相应的并行处理 Skyline 查询的有效算法 (Parallel algorithm for processing skyline queries, PAPSQ) 算法.

近年来, 对于动态 Skyline 查询的研究逐步展开. Huang 等[14] 研究了移动对象环境下的连续 Skyline 计算问题. 该文假设包括查询点在内的所有数据点均以平滑可预测的方式进行移动, 导致 Skyline 集合持续发生变化, 提出了一个连续跟踪算法 (Continuous skyline query, CSQ). 作者深入分析了点的空间位置对支配关系的贡献, 限定了导致 Skyline 集合发生变化的点的存在区域, 并提供了一种高效的动态维护方法. 文献 [15] 研究了位置和速度等多维动态属性情况下的连续 Skyline 计算, 通过对候选对象集的空间支配关系的分析, 确定任意时刻的 Skyline 集.

Pei 等[2] 首次对多事例数据开展了概率 Skyline 查询的研究, 定义了不确定数据的概率 Skyline 模型, 认为不确定数据对象以概率的形式存在于 Skyline 集合当中, 而概率 Skyline 集合包含着所有概率不小于给定阈值 p 的不确定数据对象, 并提出了自底向上和自顶向下两种算法来求解 Skyline 集. 文献 [3] 提出了聚集 R 树 (Aggregate R tree, AR) 对不确定数据进行管理, 从而维护 Skyline 集合. 文献 [4] 提出划分的方法对离散的不确定数据进行 Skyline 计算. 文献 [5] 研究了高斯分布和高斯混合模型下的 Skyline 概率.

总而言之, Skyline 计算呈现出从集中到分布、从静态到动态、从精确到近似的发展趋势. 与已有工作不同, 本文采用不确定模型对移动对象连续概率 Skyline 查询进行研究.

2 预备知识

假设 $S = \{\boldsymbol{s}_1, \boldsymbol{s}_2, \cdots, \boldsymbol{s}_N\}$ 是 N 个被监控的目标对象的集合, 每个对象的当前状态表示为 $\langle id, x_1, x_2, \cdots, x_d\rangle$, 其中 id 为对象的唯一标识, $x_i\,(i = 1, 2, \cdots, d)$ 是描述对象的 d 个属性, 组成了一个 d 维空间, 每个对象的当前状态都对应该空间的一个点.

定义 1. 支配[1]. 设两个对象 $\boldsymbol{s}_1$, $\boldsymbol{s}_2$ 的属性分别为 $(x_1, x_2, \cdots, x_d)$ 和 $(y_1, y_2, \cdots, y_d)$, 若 $\forall i$, $x_i \le y_i$ 都成立, 且 $\exists j$ 满足 $x_j < y_j\,(1 \le i, j \le d)$, 则称 $\boldsymbol{s}_1$ 支配 $\boldsymbol{s}_2$, 表示为 $\boldsymbol{s}_1 \prec \boldsymbol{s}_2$.

定义 2. Skyline 集合[1]. 所有不被任何其他点支配的点的集合称为 Skyline 集合. 其中, 每个点称为 Skyline 点.

定义 3. 不确定区域[16]. 移动对象 M 在时刻 t 的不确定区域是一个封闭区域: $Ur(t)$, M 只可能处于 $Ur(t)$ 之内的任何位置.

定义 4. 概率密度分布[17]. 移动对象 M 在不确定区域 $Ur(t)$ 内的概率密度分布函数记为 $pdf(\boldsymbol{x}, t)$, $\boldsymbol{x} \in Ur(t)$, 它描述了移动对象 M 在时刻 t 处于位置 $\boldsymbol{x}$ 的概率.

定义 5. 移动速度[17]. 移动对象 M 在 m 维空间时刻 t 的移动速度为 $\boldsymbol{v}_M(t) = (v^M_{d1}(t), v^M_{d2}(t), \cdots, v^M_{dm}(t))$.

不确定移动对象的实例如图 2 所示, 其中, 中心圆圈表示移动对象 M 在数据库中实际记录的位置, 而 M 有可能出现在其不确定区域 $Ur(t)$ (灰色区域) 中的任何位置, 其具体分布与 $pdf(\boldsymbol{x}, t)$ 相关.

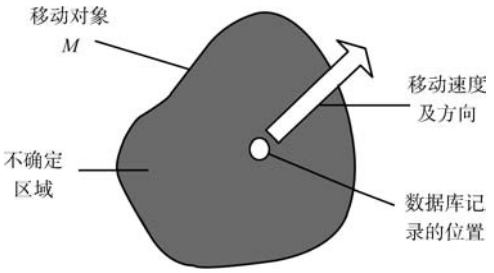


图 2 移动对象的不确定模型实例

Fig. 2 Example of uncertain moving object

3 不确定移动对象连续概率 Skyline 更新策略

通过对本文引用实例 (图 1 所示) 的数据观察可以发现: 只有距离随移动对象的移动而不断变化, 而其他因素并没有随移动对象的移动发生变化. 因此, 将 d 维空间 $D = (D_1, D_2, \cdots, D_d)$ 中各维空间分 k 维静态维属性 $D_{s1}, D_{s2}, \cdots, D_{sk}$ 和 m 维动态维属性 $D_{d1}, D_{d2}, \cdots, D_{dm}\,(d = k+m)$. 对于移动环境下的任意一个目标对象 $\boldsymbol{s}_i = (x^i_1, x^i_2, \cdots, x^i_d)^{\mathrm{T}}$, 其中, 静态维属性记为矢量 $\boldsymbol{s}^S_i = (x^i_{s1}, x^i_{s2}, \cdots, x^i_{sk})^{\mathrm{T}}$, 动态维属性记为矢量 $\boldsymbol{s}^d_i = (x^i_{d1}, x^i_{d2}, \cdots, x^i_{dm})^{\mathrm{T}}\,(d = k + m)$. 如上述实例中, 旅馆的价格、环境、服务质量是静态属性, 而空间位置是动态属性. 不确定移动对象 M 的不断移动, 导致数据集中的点 $\boldsymbol{s}_i\,(i = 1, 2, \cdots, N)$ 与 M 之间的距离不断变化. 为了简化问题, 以下的描述中我们假设动态维属性仅包含移动对象的空间位置属性.

3.1 支配概率

文献 [2] 首次将不确定数据间的支配关系表示为概率的形式, 并提出了数据间支配概率的计算方法, 但是对不确定移动对象来讲, 这种支配关系计算方法显然不再适用. 本文中提出了如下支配概率的表示方式.

定义 6. 支配概率. $\boldsymbol{s}_i$、$\boldsymbol{s}_j$ 是数据集 S 中两数据点, $\boldsymbol{s}^S_i$ 和 $\boldsymbol{s}^d_i$ 分别是 $\boldsymbol{s}_i$ 的静态维属性矢量和动态维

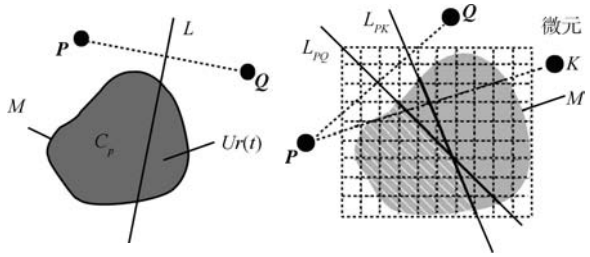
属性矢量, $\boldsymbol{s}_j^S$ 和 $\boldsymbol{s}_j^d$ 分别是 $\boldsymbol{s}_j$ 的静态维属性矢量和动态维属性矢量, 则 $\boldsymbol{s}_i$ 支配 $\boldsymbol{s}_j$ 的概率表示为

$$Pr(\boldsymbol{s}_i \prec \boldsymbol{s}_j) = \int_{Ur(t)} pdf(\boldsymbol{x}) \times \left(\begin{cases} 1, & \text{若 } \boldsymbol{s}_i^S \prec \boldsymbol{s}_j^S \text{ 且 } |\boldsymbol{s}_i^d - \boldsymbol{x}| < |\boldsymbol{s}_j^d - \boldsymbol{x}| \\ 0, & \text{否则} \end{cases} \right) \mathrm{d}\boldsymbol{x} \tag{1}$$

其中, $|\boldsymbol{s}_i^d - \boldsymbol{x}|$ 表示数据点 $\boldsymbol{s}_i$ 到不确定区域 $Ur(t)$ 中位置点 $\boldsymbol{x}$ 的距离.

如图 3 (a) 所示, 任意两个数据点 $\boldsymbol{P}$ 和 $\boldsymbol{Q}$, $\boldsymbol{P} = (x_1^P, x_2^P, \cdots, x_d^P)^{\mathrm{T}} = (x_{s1}^P, x_{s2}^P, \cdots, x_{sk}^P, x_{d1}^P, x_{d2}^P, \cdots, x_{dm}^P)^{\mathrm{T}}$, $\boldsymbol{Q} = (x_1^Q, x_2^Q, \cdots, x_d^Q)^{\mathrm{T}} = (x_{s1}^Q, x_{s2}^Q, \cdots, x_{sk}^Q, x_{d1}^Q, x_{d2}^Q, \cdots, x_{dm}^Q)^{\mathrm{T}}$, 其中, $d = k + m$, 不确定移动对象 M 具有不确定区域 $Ur(t)$ 和分布函数 $pdf(\boldsymbol{x}, t)$, $\boldsymbol{x} \in Ur(t)$. 支配概率的计算方法描述为: L 是连接 $\boldsymbol{P}$、$\boldsymbol{Q}$ 两点的线段 $|\boldsymbol{PQ}|$ 的垂直平分线, C_p 是 $\boldsymbol{P}$ 与 $Ur(t)$ 在 L 同侧的区域. 数据点 $\boldsymbol{P}$ 对 $\boldsymbol{Q}$ 的支配概率表示为

$$Pr(\boldsymbol{P} \prec \boldsymbol{Q}) = \frac{\int_{C_P} pdf(\boldsymbol{x}, t)\mathrm{d}\boldsymbol{x}}{\int_{Ur(P)} pdf(\boldsymbol{x}, t)\mathrm{d}\boldsymbol{x}} \tag{2}$$



(a) 支配概率实例　(b) Skyline 概率实例

(a) Example of dominant probability　(b) Example of skyline probability

图 3　支配概率和 Skyline 概率的表示方式

Fig. 3　Representations of dominant probability and skyline probability

对支配概率的求解分为以下几种情况:

1) $\exists$ 两静态维 $i, j, 1 \le i \le k, 1 \le j \le k, i \ne j$, 如果 $x_{si}^P < x_{si}^Q$ 和 $x_{sj}^P > x_{sj}^Q$ 成立, 则 $Pr(\boldsymbol{P} \prec \boldsymbol{Q}) = Pr(\boldsymbol{Q} \prec \boldsymbol{P}) = 0$.

2) $\forall$ 静态维 $i, 1 \le i \le k$, 如果 $x_{si}^P \ge x_{si}^Q$, 则 $Pr(\boldsymbol{P} \prec \boldsymbol{Q}) = 0$.

3) $\forall$ 静态维 $i, 1 \le i \le k$, 如果 $x_{si}^P \le x_{si}^Q$, 则比较 $\boldsymbol{P}$ 与 $\boldsymbol{Q}$ 的动态维空间属性. 若 L 与当前时刻移动对象 M 的不确定区域 $Ur(t)$ 无交集, 如果 $\boldsymbol{P}$ 与 M 在 L 同侧, 则 $Pr(\boldsymbol{P} \prec \boldsymbol{Q}) = 1$, 否则为 0; 若 L 与 $Ur(t)$ 有交集, L 将 $Ur(t)$ 划分成两部分, 我们将 C_P 称为数据点 $\boldsymbol{P}$ 对 $\boldsymbol{Q}$ 的有效支配区域, 利用式 (2), 计算 $Pr(\boldsymbol{P} \prec \boldsymbol{Q})$. 如果 $Pr(\boldsymbol{P} \prec \boldsymbol{Q}) = 1$, 则称数据点 $\boldsymbol{P}$ 完全支配数据点 $\boldsymbol{Q}$.

3.2　概率 Skyline

定义 7. Skyline 概率. $\boldsymbol{s}_i$ 是 S 中一数据点, 则 $\boldsymbol{s}_i$ 不被集合 S 中任何其他数据点所支配的概率即为点 $\boldsymbol{s}_i$ 的 Skyline 概率. 公式化为

$$Pr(\boldsymbol{s}_i) = Pr(\underset{\boldsymbol{s}_j \in S}{\wedge} \boldsymbol{s}_i ! \prec \boldsymbol{s}_i) = \int_{Ur(t)} pdf(\boldsymbol{x}) \cdot \left(\prod_{\boldsymbol{s}_j \in S} \left\{ \begin{array}{ll} 0, & \text{若 } \boldsymbol{s}_j^S \prec \boldsymbol{s}_i^S \text{ 且 } |\boldsymbol{s}_j^d - \boldsymbol{x}| < |\boldsymbol{s}_i^d - \boldsymbol{x}| \\ 1, & \text{否则} \end{array} \right. \right) \mathrm{d}\boldsymbol{x} \tag{3}$$

$\boldsymbol{s}_j ! \prec \boldsymbol{s}_i$ 表示 $\boldsymbol{s}_i$ 不被 $\boldsymbol{s}_j$ 所支配, $\boldsymbol{s}_i$ 的 Skyline 概率的有效支配区域即 $\boldsymbol{s}_i$ 不被其他数据点支配的有效支配区域的交集, 其有效支配区域与移动对象不确定区域的比值即为 $\boldsymbol{s}_i$ 的 Skyline 概率. 如图 3 (b) 所示, 数据点 $\boldsymbol{P}$ 的有效支配区域是 $\boldsymbol{P}$ 不被 $\boldsymbol{Q}$ 和 $\boldsymbol{K}$ 支配的有效区域的交集, 即图 3 (b) 中斜线阴影部分.

定义 8. 概率 Skyline. 给定阈值 $p \in [0, 1]$, 对集合 S 中的任意数据点 $\boldsymbol{s}_i$, 如果 $Pr(\boldsymbol{s}_i) \ge p$, 则 $\boldsymbol{s}_i$ 属于概率 Skyline (p-Skyline) 集合, 即

$$sky(p) = \{\boldsymbol{s}_i \in S | Pr(\boldsymbol{s}_i) \ge p\} \tag{4}$$

对数据点 $\boldsymbol{s}_i$ 来讲, 其有效支配区域在现实的计算过程中由于受各数据点对其支配区域的影响, 它的边界函数和精确区域面积是很难计算的, 因此, 本文采用了概率估计的方法, 来计算其有效支配区域的大小. 具体计算方法为: 首先, 将整个移动对象的不确定区域划分成多个长度为 δ 的微元 (如图 3 (b) 所示), 然后, 测试微元是否位于 $\boldsymbol{s}_i$ 的有效支配区域内. $\boldsymbol{s}_i$ 的 Skyline 概率即落入 $\boldsymbol{s}_i$ 有效支配区域内微元的面积总和与整个不确定区域内微元面积总和的比值.

3.3　移动环境下查询点和数据点间距离计算

对于记录点位于 $\boldsymbol{x}_0 = (x_{d1}^i, x_{d2}^i, \cdots, x_{dm}^i)$, 速度为 $\boldsymbol{v}_i(t) = (v_{d1}^i(t), v_{d2}^i(t), \cdots, v_{dm}^i(t))$ 的不确定移动对象 M ($\boldsymbol{x}_0$ 表示数据库中记录的位置), 与点 $\boldsymbol{P} = (x_{d1}^p, x_{d2}^p, \cdots, x_{dm}^p)$, 速度为 $\boldsymbol{v}_p(t) = (v_{d1}^p(t), v_{d2}^p(t), \cdots, v_{dm}^p(t))$ 的数的欧氏距离函数可以表达为关于时间 t 的函数[18]:

$$\mathrm{dist}(\boldsymbol{x}, \boldsymbol{P}, t) = \sqrt{a_{\boldsymbol{P}} t^2 + b_{\boldsymbol{P}} t + c_{\boldsymbol{P}}}, \quad \boldsymbol{x} \in Ur(t) \tag{5}$$

其中, $a_{\boldsymbol{P}}$, $b_{\boldsymbol{P}}$, $c_{\boldsymbol{P}}$ 是系数, 由它们的起始位置和速度决定, $a_{\boldsymbol{P}} = \sum_{k=1}^{m} [v_{dk}^i(t) - v_{dk}^p(t)]^2$, $b_{\boldsymbol{P}} = 2 \sum_{k=1}^{m}$

$[(x^i_{dk}- x^p_{dk})(v^i_{dk}(t) - v^p_{dk}(t))]$, $c_{\boldsymbol{P}} = \sum_{k=1}^{m}(x^i_{dk}- x^p_{dk})^2$. 当 $\boldsymbol{P}$ 静止时, $a_{\boldsymbol{P}}$、$b_{\boldsymbol{P}}$、$c_{\boldsymbol{P}}$ 仍然由此公式决定, 此时 $v^p_{dk}(t) = 0, \forall k$. $\mathrm{dist}(\boldsymbol{x}, \boldsymbol{P}, t)$ 表示时刻 t 数据点 $\boldsymbol{P}$ 到不确定区域 M 中任一点 $\boldsymbol{x}$ 的距离函数; 最小距离函数 $\mathrm{Mindist}(\boldsymbol{x}, \boldsymbol{P}, t)$ 表示 $\boldsymbol{P}$ 到整个不确定区域 M 的最小距离函数; 最大距离 $\mathrm{Maxdist}(\boldsymbol{x}, \boldsymbol{P}, t)$ 函数表示 $\boldsymbol{P}$ 到整个不确定区域 M 的最大距离函数.

定理 1. 任意两数据点 $\boldsymbol{P}$, $\boldsymbol{Q}$, 与沿直线匀速运动的移动对象 M 建立的欧氏距离曲线函数间最多有一个交点.

证明. 由式 (5) 知, 对任意两数据点 $\boldsymbol{P}$, $\boldsymbol{Q}$, 建立其与 M 的距离函数表示为: $\mathrm{dist}(\boldsymbol{x}, \boldsymbol{P}, t) = \sqrt{a_{\boldsymbol{P}}t^2 + b_{\boldsymbol{P}}t + c_{\boldsymbol{P}}}$, $\mathrm{dist}(\boldsymbol{x}, \boldsymbol{Q}, t) = \sqrt{a_{\boldsymbol{Q}}t^2 + b_{\boldsymbol{Q}}t + c_{\boldsymbol{Q}}}$, $\boldsymbol{x} \in Ur(t)$. 由于数据点 $\boldsymbol{P}$、$\boldsymbol{Q}$ 是静止的, $\boldsymbol{v_p}(t) = \boldsymbol{v_q}(t) = 0$, 则 $\boldsymbol{a_P} = \boldsymbol{a_Q} = \sum_{k=1}^{m} v^i_{dk}(t)^2$. 由此求得, 若 $\mathrm{dist}(\boldsymbol{x}, \boldsymbol{P}, t)$, $\mathrm{dist}(\boldsymbol{x}, \boldsymbol{Q}, t)$ 有交点, 则 $b_{\boldsymbol{P}}t + c_{\boldsymbol{P}} = b_{\boldsymbol{Q}}t + c_{\boldsymbol{Q}}$, 即 $t = \frac{c_{\boldsymbol{Q}} - c_{\boldsymbol{P}}}{b_{\boldsymbol{P}} - b_{\boldsymbol{Q}}}$. 若 $b_P = b_{\boldsymbol{Q}}$, 则无交点; 若 $b_{\boldsymbol{P}} \neq b_{\boldsymbol{Q}}$, 则仅有一个交点.

□

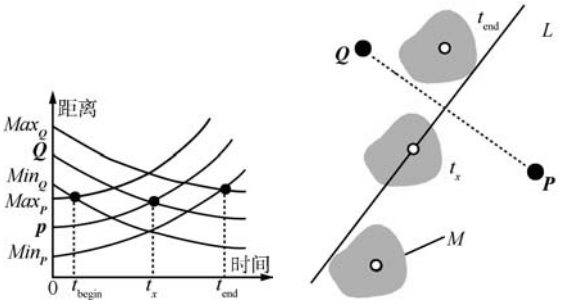
3.4 影响 p-Skyline 集合的 Event

移动对象移动过程中, 其与数据点间的距离不断变化, 造成数据点间支配关系的变化, 如图 4 (a) 所示, $f(\boldsymbol{P})$, $f(\boldsymbol{Q})$ 分别是不确定移动对象 M (在此表示数据库记录的位置) 与数据点 $\boldsymbol{P}$ 和 $\boldsymbol{Q}$ 建立的欧氏距离曲线函数, $Max_{\boldsymbol{P}}$, $Min_{\boldsymbol{P}}$ 分别是 M 与 $\boldsymbol{P}$ 的最大、最小距离曲线函数, $Max_{\boldsymbol{Q}}$, $Min_{\boldsymbol{Q}}$ 分别是 M 与 $\boldsymbol{Q}$ 的最大、最小距离曲线函数, t_x 是曲线 $\boldsymbol{P}$, $\boldsymbol{Q}$ 的相交时刻, t_{begin} 是曲线 $Max_{\boldsymbol{P}}$, $Min_{\boldsymbol{Q}}$ 的相交时刻, t_{end} 是曲线 $Min_{\boldsymbol{P}}$, $Max_{\boldsymbol{Q}}$ 的相交时刻. 图 4 (b) 是对应于图 4 (a) 的各个曲线相交时刻移动对象不确定区域的位置信息. 假设 $\forall x^{\boldsymbol{P}}_{si} \geq x^{\boldsymbol{Q}}_{si} (1 \leq i \leq k)$, 在时刻 t_{begin} 之前 $\boldsymbol{P}$ 与查询点 M 距离比 $\boldsymbol{Q}$ 近, $Pr(\boldsymbol{Q} \prec \boldsymbol{P}) = 0$; 到 t_x 时, M (在此表示数据库记录的位置) 位于 $|\boldsymbol{PQ}|$ 垂直平分线 L 上, $Pr(\boldsymbol{Q} \prec \boldsymbol{P}) = 0.5$; 时刻 t_{end} 之后, M 的整个不确定区域都与 $\boldsymbol{Q}$ 在 L 的同侧, 此时, $Pr(\boldsymbol{Q} \prec \boldsymbol{P}) = 1$. 由此可以发现, 在数据点 $\boldsymbol{P}$, $\boldsymbol{Q}$ 对应距离曲线交叉过程中, 两者之间的支配概率发生了变化, 从而可能引起 p-Skyline 集合从中间某个时刻开始发生变化.

综合分析移动对象不确定区域对支配关系的影响, 本文定义了两种可能影响 p-Skyline 集合的 Event 类型:

定义 9. Up event. 对于集合 S 中满足 $\forall x^{\boldsymbol{P}}_{si} \geq x^{\boldsymbol{Q}}_{si} (1 \leq i \leq k)$ 的数据点 $\boldsymbol{P}$, $\boldsymbol{Q}$, 如果 $\mathrm{dist}(\boldsymbol{x}, \boldsymbol{P}, t)$ 与 $\mathrm{dist}(\boldsymbol{x}, \boldsymbol{Q}, t)$, $\boldsymbol{x} \in Ur(t)$ 在某一时刻 t_x 相交, 对任意两时刻 $t_1, t_2 \in [t_{\mathrm{begin}}, t_{\mathrm{end}}]$, 假设 $t_1 < t_2$, 若 $Pr(\boldsymbol{Q} \prec \boldsymbol{P})|_{t_1} > Pr(\boldsymbol{Q} \prec \boldsymbol{P})|_{t_2}$ 成立, 则 $\boldsymbol{P}$, $\boldsymbol{Q}$ 产生一个 Up event $(\boldsymbol{P}, \boldsymbol{Q})$, 表示为 $\langle \boldsymbol{P}, \boldsymbol{Q}, t_{\mathrm{begin}}, t_{\mathrm{end}}, Up \rangle$. 其中, t_{begin} 是 $\mathrm{Mindist}(\boldsymbol{x}, \boldsymbol{P}, t)$ 与 $\mathrm{Maxdist}(\boldsymbol{x}, \boldsymbol{Q}, t)$ 的交叉时刻, t_{end} 是 $\mathrm{Maxdist}(\boldsymbol{x}, \boldsymbol{P}, t)$ 与 $\mathrm{Mindist}(\boldsymbol{x}, \boldsymbol{Q}, t)$ 的交叉时刻.

定义 10. Down event. 对于集合 S 中满足 $\forall x^{\boldsymbol{P}}_{si} \geq x^{\boldsymbol{Q}}_{si} (1 \leq i \leq k)$ 的数据点 $\boldsymbol{P}$, $\boldsymbol{Q}$, 如果 $\mathrm{dist}(\boldsymbol{x}, \boldsymbol{P}, t)$ 与 $\mathrm{dist}(\boldsymbol{x}, \boldsymbol{Q}, t)$, $\boldsymbol{x} \in Ur(t)$ 在某一时刻 t_x 相交, 对任意两时刻 $t_1, t_2 \in [t_{\mathrm{begin}}, t_{\mathrm{end}}]$, 假设 $t_1 < t_2$, 若 $Pr(\boldsymbol{Q} \prec \boldsymbol{P})|_{t_1} < Pr(\boldsymbol{Q} \prec \boldsymbol{P})|_{t_2}$ 成立, 则 $\boldsymbol{P}$、$\boldsymbol{Q}$ 产生一个 Down event $(\boldsymbol{P}, \boldsymbol{Q})$, 表示为 $\langle \boldsymbol{P}, \boldsymbol{Q}, t_{\mathrm{begin}}, t_{\mathrm{end}}, Down \rangle$. 其中, t_{begin} 是 $\mathrm{Maxdist}(\boldsymbol{x}, \boldsymbol{P}, t)$ 与 $\mathrm{Mindist}(\boldsymbol{x}, \boldsymbol{Q}, t)$ 的交叉时刻, t_{end} 是 $\mathrm{Mindist}(\boldsymbol{x}, \boldsymbol{P}, t)$ 与 $\mathrm{Maxdist}(\boldsymbol{x}, \boldsymbol{Q}, t)$ 的交叉时刻.



(a) 距离曲线相交实例 (b) 不确定区域变化示意

(a) Example of intersection of the distance curve (b) Example of the change of uncertain area

图 4 支配关系变化

Fig. 4 Transformation of dominant relationship

3.5 剪枝策略

由 event 定义可知, 每一数据点都要与对其存在支配可能性的数据点计算可能产生的 Event, 对于大规模数据集来讲, 计算产生的 Event 数量是很庞大的, 如果对每个 Event 都进行更新计算, 计算量同样庞大. 通过对 Event 的研究发现, 有些 Event 对 p-Skyline 集合无任何影响, 举例来说, 当一个 Down event$(\boldsymbol{self}, \boldsymbol{peer})$ 结束后, 无论移动对象当前处于什么位置, 至少有一个数据点 $\boldsymbol{peer}$ 完全支配数据点 $\boldsymbol{self}$, 那么与 $\boldsymbol{self}$ 相关且其 t_{begin} 在此时刻之后的 Event 对 p-Skyline 集合的变动不起任何作用. 本文提出以下三种剪枝规则, 可以将对 p-Skyline 集合无任何影响的 Event 剪枝. 具体描述如下:

剪枝规则 1. 数据点 $\boldsymbol{s}_i$, $\boldsymbol{s}_j$ 产生 Down event$(\boldsymbol{s}_i, \boldsymbol{s}_j)$, 则 $\forall$ event$(\boldsymbol{s}_i, \boldsymbol{s}_k)$ $(k \neq i, j$ 且 $0 < k < N)$, 如果 event$(\boldsymbol{s}_i, \boldsymbol{s}_k)$. $t_{\mathrm{begin}} \geq$ event$(\boldsymbol{s}_i, \boldsymbol{s}_j)$. t_{end}, 则 event$(\boldsymbol{s}_i, \boldsymbol{s}_k)$ 属于无效 Event.

证明. 对于任意 Down event$(\boldsymbol{s}_i, \boldsymbol{s}_j)$, 在时刻 t_{end}, $Pr(\boldsymbol{s}_j \prec \boldsymbol{s}_i) = 1$. 由式 (3) 得 $Pr(\boldsymbol{s}_i) = 0$, 由定理 1 得 $\boldsymbol{s}_i$, $\boldsymbol{s}_j$ 对应的距离曲线只有一个交点, 则在时刻 t_{end} 之后, $\boldsymbol{s}_i$ 与 $\boldsymbol{s}_j$ 之间的支配关系将不再发

生改变, 即 $Pr(\boldsymbol{s}_j \prec \boldsymbol{s}_i) = 1$ 恒成立, $Pr(\boldsymbol{s}_i)$ 始终为 0, 因此, event$(\boldsymbol{s}_i, \boldsymbol{s}_k)$ 的更新计算对 $Pr(\boldsymbol{s}_i)$ 无任何影响, 属于无效 Event. □

剪枝规则 2. 任意满足 $Pr(\boldsymbol{s}_j \prec \boldsymbol{s}_i) = 1$ 的 UP event$(\boldsymbol{s}_i, \boldsymbol{s}_j)$, 如果 $\exists$ event$(\boldsymbol{s}_i, \boldsymbol{s}_k)$ $(0 < k < N$ 且 $k \neq i)$ 且 event$(\boldsymbol{s}_i, \boldsymbol{s}_k).t_{\text{begin}} <$ event$(\boldsymbol{s}_i, \boldsymbol{s}_j).t_{\text{begin}}$, 则在 event$(\boldsymbol{s}_i, \boldsymbol{s}_j).\ t_{\text{begin}}$ 之前对 event$(\boldsymbol{s}_i, \boldsymbol{s}_k)$ 的计算属于无效计算; 如果 event$(\boldsymbol{s}_i, \boldsymbol{s}_k).t_{\text{end}} \leq$ event$(\boldsymbol{s}_i, \boldsymbol{s}_j).t_{\text{begin}}$, 则 event$(\boldsymbol{s}_i, \boldsymbol{s}_k)$ 属于无效 Event.

证明. 根据式 (3) 和定理 1, 由于 $Pr(\boldsymbol{s}_j \prec \boldsymbol{s}_i) = 1$, 则可得出在时刻 event$(\boldsymbol{s}_i, \boldsymbol{s}_j).t_{\text{begin}}$ 之前 $Pr(\boldsymbol{s}_i) = 0$ 恒成立. 在 event$(\boldsymbol{s}_i, \boldsymbol{s}_j).t_{\text{begin}}$ 之前对 event$(\boldsymbol{s}_i, \boldsymbol{s}_k)$ 的更新计算, 根据式 (3) 可以得出 $Pr(\boldsymbol{s}_k \prec \boldsymbol{s}_i)$ 的变化对 $Pr(\boldsymbol{s}_i)$ 无任何影响, 因此, 其计算属于无效计算; 同理, 如果 event$(\boldsymbol{s}_i, \boldsymbol{s}_k)$ 在 event$(\boldsymbol{s}_i, \boldsymbol{s}_j).t_{\text{begin}}$ 之前结束, 此 Event 对 $Pr(\boldsymbol{s}_i)$ 无任何影响, 因此, 属于无效 Event. □

剪枝规则 3. 假设 Up event$(\boldsymbol{s}_i, \boldsymbol{s}_m)$ 是所有与数据点 $\boldsymbol{s}_i$ 相关且 $Pr(\boldsymbol{s}_j \prec \boldsymbol{s}_i) = 1 (1 \leq j \leq d)$ 的 Up event 中 t_{begin} 最大的 Event, 记其 t_{begin} 为 $start\_T$; 假设 Down event$(\boldsymbol{s}_i, \boldsymbol{s}_n)$ 是所有与数据点 $\boldsymbol{s}_i$ 相关的 Down event 中 t_{end} 最小的 Event, 记其 t_{end} 为 $finish\_T$; 如果 $finish\_T \leq start\_T$, 则所有与 $\boldsymbol{s}_i$ 相关的 Event 都是无效 Event.

证明. 由剪枝规则 2 得, 在 $start\_T$ 之前对 ∀event 的更新计算都是无效计算; 由剪枝规则 1 得, 在 $finish\_T$ 之后的 ∀event 都是无效 Event, 对其更新计算也属于无效计算, 当 $finish\_T \leq start\_T$ 时, 很显然对与 $\boldsymbol{s}_i$ 相关 Event 的更新计算都属于无效计算, 即这些 Event 都是无效的. □

4 算法实现策略

在算法中利用全局双向链表 L_{sp} 存储当前全部的 p-Skyline 点, $DataType\langle id, position, Info[d], start\_T, finish\_T\rangle$ 表示数据点信息, $flag(\boldsymbol{self}, \boldsymbol{peer})$ 标识两数据点间可能性支配关系, $\langle \boldsymbol{self}, \boldsymbol{peer}, t_{\text{begin}}, t_{\text{end}}, type\rangle$ 表示 Event, Q_e 存储可能影响 p-Skyline 集合的 Event, Q_{handle} 用于存储连续跟踪计算的 Event.

4.1 静态算法 U_SPSC

算法 Initial_PSkyline() 用于计算初始时刻的 p-Skyline, 描述如下:

Algorithm 1. Initial_PSkyline()
Output. The initial p-Skyline set.
1. For any point $\boldsymbol{P}$ in S;
2. For each point $\boldsymbol{Q}$ in S;
3. If $\boldsymbol{Q}$ has the potential to dominate $\boldsymbol{P}$;
4. flag ($\boldsymbol{Q}$,$\boldsymbol{P}$)=1; //静态支配关系成立;
5. Compute Pr ($\boldsymbol{Q} \prec \boldsymbol{P}$);
6. Else
7. flag ($\boldsymbol{Q}$,$\boldsymbol{P}$)=0;
8. Pr ($\boldsymbol{Q} \prec \boldsymbol{P}$)= 0;
9. Compute skyline probability Pr ($\boldsymbol{P}$); //计算数据点的 p-Skyline 概率;
10. If Pr($\boldsymbol{P}$) $\geq p$;
11. Insert $\boldsymbol{P}$ into L_{sp}.

算法 U_SPSC 在移动对象移动过程中的每一时刻, 重新遍历数据点, 对存在可能性支配关系的任意两数据点, 重新计算其支配关系, 更新每个数据点的 Skyline 概率, 并与给定阈值 p 比较, 不断更新 p-Skyline 集合.

Algorithm 2. U_SPSC
Input. M is the uncertain moving object, p is the probabilistic threshold.
Output. The p-Skyline in maintance.
1. Initial_PSkyline();
2. For every time during the M's moving;
3. For any two points $\boldsymbol{P}$, $\boldsymbol{Q}$ in S;
4. If flag ($\boldsymbol{Q}$, $\boldsymbol{P}$)==1;
5. Compute Pr ($\boldsymbol{Q} \prec \boldsymbol{P}$);
6. Compute Pr ($\boldsymbol{P}$) for every point;
7. According to p and Pr ($\boldsymbol{P}$) update L_{sp}.

4.2 连续更新算法 U_CPSC

U_CPSC 算法的基本思想是提前计算不确定移动对象移动过程中, 可能引起 p-Skyline 集合变动的 Event, 通过对这些 Event 的跟踪计算从而不断更新 p-Skyline 集合. 首先计算初始时刻的 p-Skyline 集合 (和 U_SPSC 算法相同); 接着, 算法 CreatEvent 计算所有可能影响 p-Skyline 的 Event; 最后, 算法 HandleEvent 对 Q_{handle} 中所有 Event 进行跟踪计算, 并及时更新 Skyline 集合, 直到队列 Q_{handle} 为空为止.

Algorithm 3. U_CPSC
Input. M is the uncertain moving object, p is the probabilistic threshold.
Output. the p-Skyline for M's starting position and in maintance.
1. Initial_PSkyline();
2. CreatEvent();
3. HandleEvent().

4.2.1 CreateEvent 算法

CreateEvent 算法用于计算可能影响 p-Skyline 集合的 Event. 对数据集 S 中数据点 $\boldsymbol{P}$, 首先将其与数据点 $\boldsymbol{Q}$ 比较, 若 $\boldsymbol{Q}$ 存在支配 $\boldsymbol{P}$ 的可能性, 建立与移动对象 M 之间的欧氏距离函数 dist$(\boldsymbol{x}, \boldsymbol{P}, t)$, dist$(\boldsymbol{x}, \boldsymbol{Q}, t)$, 计算两距离曲线的交叉时刻 t_x, t_{begin} 和 t_{end}, 根据 Event 的定义, 判定 $\langle \boldsymbol{P}, \boldsymbol{Q}, t_{\text{begin}}, t_{\text{end}}, type\rangle$ 的类型 $type$, 并更新数据点 $\boldsymbol{P}$ 的 $start\_T$ 和 $finish\_T$; 然后根据剪枝规则 1 得, 对任意 Event, 如果其 $t_{\text{begin}} \geq finish\_T$,

则该 Event 为无效事件, 由剪枝规则 3 得, 如果 $t_{\text{end}} \leq start\_T$, 则与 $\boldsymbol{P}$ 相关的所有 Event 都是无效事件, 利用剪枝规则 1 和剪枝规则 3 对 Event 剪枝, 然后将有效 Event 插入 Q_e. 算法描述如下.

Algorithm 4. CreateEvent()

Input. M is the uncertain moving object.

Output. Upcoming events that may affect the p-Skyline.

1. For any two point $\boldsymbol{P}$, $\boldsymbol{Q}$ in S;
2. If flag($\boldsymbol{P}$, $\boldsymbol{Q}$)= 1;
3. Compute a, b, c in term of q for $\boldsymbol{P}$, $\boldsymbol{Q}$ and the intersection time t_x;
4. Judge the event type of ($\boldsymbol{P}$, $\boldsymbol{Q}$) and compute crossing time t_x;
5. If $t_{\text{begin}} \geq P_{finish\_T}$;
6. Continue;
7. If $t_{\text{end}} \geq P_{start\_T}$;
8. Delete all events that are related to $\boldsymbol{P}$;
9. Else
10. Insert$\langle \boldsymbol{P}, \boldsymbol{Q}, t_{\text{begin}}, t_{\text{end}}, type \rangle$ into Q_e.

4.2.2 HandleEvent 算法

该算法主要功能是对可能影响 p-Skyline 集合的 Event 进行跟踪计算, 并不断更新 p-Skyline 集合. 将 Event 从 Q_e 中移出, 然后插入 Q_{handle}, 当 Event 的 $t_{\text{begin}} > time$ 时, 更新计算 Q_{handle} 中所有事件: 根据剪枝规则 2, 如果当前处理 Event($\boldsymbol{P}$,$\boldsymbol{Q}$) 的 $t_{\text{begin}} \geq start\_T$, 则更新计算 $Pr(\boldsymbol{Q} \prec \boldsymbol{P})$ 和 $Pr(\boldsymbol{P})$, 若 $Pr(\boldsymbol{P}) \geq p$ 且 $\boldsymbol{P}$ 不在 L_{sp} 中, 将其加入到 L_{sp}; 若 $Pr(\boldsymbol{P}) < p$ 且 $\boldsymbol{P}$ 在 L_{sp} 中, 将其从 L_{sp} 删除; 如果当前时刻 $time$ 大于 t_{end}, 将 Event 从 Q_{handle} 中删除. 递归运行上述操作, 直到 Q_e 与 Q_{handle} 都为空为止.

Algorithm 5. HandleEvent()

Input. M is the uncertain moving object.

Output. the p-Skyline at any time in maintenance.

1. While(Queue(Q_e)! = NULL)
2. e = Deueue(Q_e);
3. While $e.t_{\text{begin}} \leq time$
4. EnQueue(Q_{handle},e);
5. e = Deueue(Q_e);
6. Updata(Q_{handle});
7. time = $e.t_{\text{begin}}$.

4.3 算法性能分析

U_CPSC 中频繁的操作是根据当前移动对象的不确定区域和各数据点的空间位置计算任意两点间的支配概率, 这是影响算法性能的重要因素. 本文将支配概率的计算分成两步: 首先, 比较数据点 $\boldsymbol{P}$, $\boldsymbol{Q}$ 间的静态维属性, 增设 Flag 标志位用于标识数据点间的可能性支配关系, 为 0 表示 $\boldsymbol{P}$ 不可能支配 $\boldsymbol{Q}$, $Pr(\boldsymbol{P} \prec \boldsymbol{Q}) = 0$ 永远成立; 为 1 表示 $\boldsymbol{P}$ 可能支配 $\boldsymbol{Q}$, 再考虑动态维属性, 根据当前移动对象的不确定区域位置, 利用式 (2) 计算两者之间的支配概率. Flag 还有一重要应用, 对于 $\forall event(\boldsymbol{P},\boldsymbol{Q})$, $flag(\boldsymbol{Q},\boldsymbol{P}) = 1$ 恒成立, 因此, 当 $flag(\boldsymbol{Q},\boldsymbol{P}) = 0$ 时, 可以不去比较计算两者间可能产生的 Event, 减少计算量.

针对计算产生的 Event 有些对 p-Skyline 集合无任何影响, 本文提出了三种剪枝策略, 可以将对 p-Skyline 集合变动无关的 Event 剪枝, 以保证剩余 Event 发生时对数据点的 Skyline 概率都会产生一定影响. 通过对这些 Event 的跟踪计算就可以快速地更新 p-Skyline 集合, 而不必在每一更新时刻时去遍历整个数据集, 减少了查询和更新操作.

下面分析 U_CPSC 算法时间复杂度. 对于数据集 S 中任意数据点 $\boldsymbol{s}_i$, $\boldsymbol{s}_j$ 来讲, 需要比较计算相互间的可能性支配关系, 比较的次数为: $(N-1)+(N-2)+\cdots+1 = N(N-1)/2$, 时间复杂度为 O($d \cdot N(N-1)/2$). 根据式 (3), 对任意数据点 $\boldsymbol{s}_i$ 来讲, 其 Skyline 概率是不被其他 $N-1$ 个数据点支配的有效区域的交集, 即每个数据点的 Skyline 概率可以在 O($N-1$) 时间内完成, 用概率估计法对数据点的 Skyline 概率求解需要 O(δ^2) (其中 δ^2 是微元的个数), 因此在 O($\delta^2 \cdot (N-1)$) 时间内可以完成对 Skyline 概率的计算. 计算 Skyline 概率之后, 还要与给定阈值 p 比较其是否属于 p-Skyline 集合, 同理, p-Skyline 集合的计算可以在 O(N) 时间内完成. 则初始时刻求解 p-Skyline 集合所花费的最多时间开销为 O($d \cdot (N-1)N/2 + \delta^2 \cdot (N-1) + N$).

CreateEvent 算法中, 任意两数据点间存在可能性支配的情况最多为 $(N-1)N/2$, 对其建立与移动对象的欧氏距离曲线函数、计算曲线函数的交叉时间、判定产生 Event 的类型及其有效性可以在 O(k_1) 时间内完成, 则 CreateEvent 最多可在 O($k_1 \cdot (N-1)N/2$) 时间内完成. HandleEvent 算法中, 由于最多可产生 Event 个数为 $(N-1)N/2$, 对每个 Event 需要在有效时刻内进行跟踪计算, 最坏情况下, 每个 Event 的有效时刻区间为 [0, Maxtime], 开始时刻都是在 0 时刻, 在每个时刻对 Event 更新计算的时间为 O(k_2), 则对 Event 跟踪计算可在 O($k_2 \cdot \text{Maxtime} \cdot (N-1)N/2$) 时间内完成. 综上所述, U_CPSC 算法的平均运算时间为: O($d \cdot (N-1)N/2 + \delta^2 \cdot (N-1) + N + k_1 \cdot (N-1)N/2 + k_2 \cdot \text{Maxtime} \cdot (N-1)N/2$), 等价于 O($N^2$).

5 实验结果与分析

5.1 数据集和不确定对象的分布

目前关于概率 Skyline 查询的研究都是针对离散数据, 尚未有对不确定移动对象概率 Skyline 查询的研究工作, 因此, 本文提出一种静态算法 U_SPSC, 通过与 U_CPSC 性能进行比较, 验证 U_CPSC 的

有效性.

实验运行环境为 Intel Core 2 Duo, 2.93 GHz CPU, 3 GB RAM, Windows XP 的 PC 机, VC++2008 开发平台. 实验中数据的动态维属性是两维的数据空间坐标 (x, y), 静态维属性是广泛采用的三种遵循不同分布的模拟数据: 独立数据 Indep、正相关数据 Corr 和反相关数据 Anti. 以两维静态数据为例, 实验数据分布在 500×500 的数据空间上. 实验中分别采用两种不确定区域形状: 圆和椭圆; 两种概率密度函数分布: 均匀分布和高斯分布; 组合后形成 4 种不确定移动对象的分布状态: 圆 - 均匀分布 (Circle-uniform)、圆 - 高斯分布 (Circle-Gaussian)、椭圆 - 均匀分布 (Ellipse-uniform) 和椭圆 - 高斯分布 (Ellipse-Gaussian), 分别对算法的性能进行验证分析. 其中, $Circle_R$ 表示采用圆形模型, R 表示圆形区域半径; $Ellipse_{R_L,R_S}$ 表示椭圆模型, R_L、R_S 分别表示椭圆的长轴和短轴半径, 为了符合实际情况, 将移动对象的运动方向设置为长轴的方向. 实验采用的二维高斯密度函数为: $pdf(x,y) = \frac{1}{2\pi\sigma_x\sigma_y} e^{\frac{1}{2}[\frac{(x-\mu_x)^2}{\sigma_x^2}+\frac{(y-\mu_y)^2}{\sigma_y^2}]}$, 其中, $\mu_x = \mu_y = 0$, $\sigma_x = \sigma_y = 1$, $\boldsymbol{v} = (v_x, v_y)$ 表示移动对象的移动速度, 概率阈值 $p = 0.3$.

5.2 数据规模 N 对算法影响

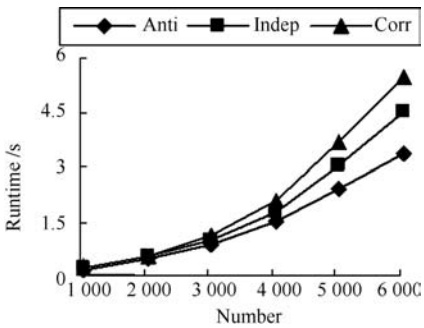
图 5 显示了在 Circle-uniform 模型下, 动态增量更新算法 U_CPSC 和静态算法 U_SPSC 的对比实验, 参数 $d = 2$, $Circle_R = 10$, $Ellipse_{R_L,R_S} = (10, 8)$, $\boldsymbol{v} = (v_x, v_y) = (20, 25)$. 显而易见, U_CPSC 的运行时间远少于静态算法 U_SPSC, 因为对每一更新时刻, U_CPSC 不必重新遍历整个数据集, 而是仅对与 Event 相关的数据点进行更新操作, 减少了遍历与计算的时间开销, 与 U_SPSC 相比较, U_CPSC 的平均运算时间约是其 5 %.

图 5 还可以看出, 随着数据规模 N 的增大, 算法平均处理时间不断增加, 这是因为数据规模的扩大导致支配概率计算次数增加, 需要处理的 Event 事件的增多, 从而使得运行时间增加, 图 6 显示了这种变化.

在数据空间中任取一点 $Position(x, y)$, 假设三种数据集中此位置点上都存在数据点 $\boldsymbol{P}$, 由 Event 的定义知, 数据点 $\boldsymbol{Q}$ 与数据点 $\boldsymbol{P}$ 能产生 event$(\boldsymbol{P}, \boldsymbol{Q})$ 的首要条件是 $\forall X_{si}^{\boldsymbol{P}} \geq x_{si}^{\boldsymbol{Q}} (1 \leq i \leq k)$, 则 $\boldsymbol{P}$ 需要与数据集中满足上述条件的任意数据点 $\boldsymbol{Q}$ 比较计算可能产生的 Event, 由三种测试数据的分布示意可知, 满足上述要求的数据点数量 Corr 是最多的, Anti 最少, 因此, Corr 数据类型中需比较计算的次数较其他两种数据类型要增多, 产生的 Event 数量也随之增大, 图 6 (a) 验证了上述分析 (圆 - 均匀分布模型). U_CPSC 虽然产生大量 Event, 但并不是所有 Event 都对 p-Skyline 产生影响, 如图 6 (b) 所示, 运用剪枝规则后, 将对 p-Skyline 无影响的 Event 剪枝, 与剪枝前相比较 Event 减少了约 600 倍, 体现了剪枝规则的有效性.

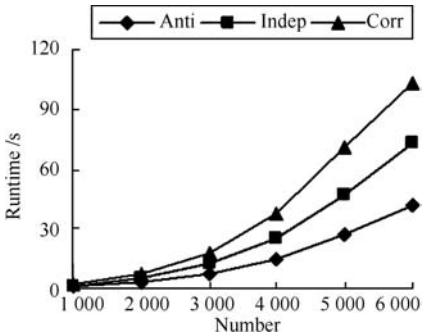
图 7 描述了 U_CPSC 在 4 种不同不确定区域分布模型下算法平均运算时间的影响. 从图 7 中可以看出, 椭圆与圆、高斯分布与均匀分布模型相比较, 随着区域的不规则程度的增大, 算法的平均处理时间增加. 在图 7 (a) ~ 7 (c) 的三个数据集的实验中, Ellipse-Gaussian 型的平均处理时间最大, Circle-uniform 型的平均处理时间最少.

图 8 显示了数据规模 N 的变化对 p-Skyline 集合的影响. 由于 Anti 数据类型的不规则程度远远大于其他两种数据类型, 其 p-skyline 尺寸约是 Corr 的 3 倍, 而 Indep 处于二者之间. 在 $N < 1\,000$ 时, p-Skyline 尺寸大小变化较大, 随着 N 的不断增大, p-Skyline 尺寸大小将趋向平稳.



(a) U_CPSC 算法的运算时间

(a) Runtime for U_CPSC algorithm

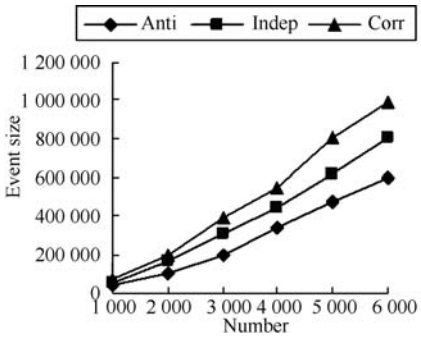


(b) U_SPSC 算法的运算时间

(b) Runtime for U_SPSC algorithm

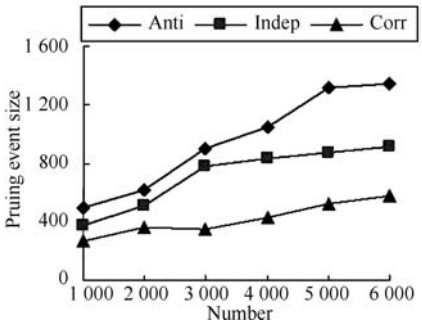
图 5　数据规模在圆 - 均匀分布 (Circle-uniform) 模型中对算法运行时间的影响

Fig. 5　Effect of dataset size on algorithm runtime in circle-uniform model



(a) 剪枝前 Event 规模

(a) Event size before pruning

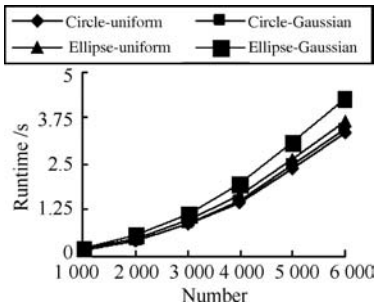


(b) 剪枝策略对 Event 影响

(b) Effect of pruning techniques on event

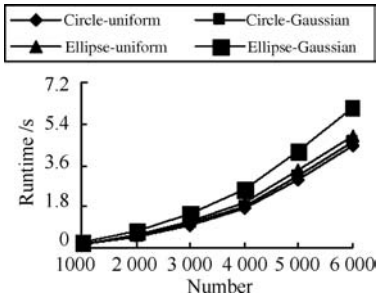
图 6 数据规模在圆 - 均匀分布 (Circle-uniform) 模型中对 Event 的影响

Fig. 6 Effect of dataset size on event size in circle-uniform model



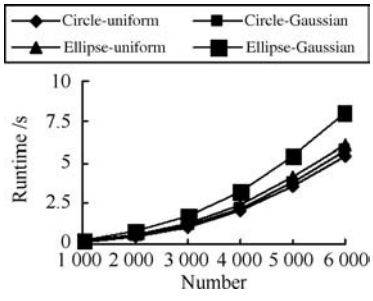
(a) 反相关数据

(a) Anti data set



(b) 独立数据

(b) Indep data set



(c) 正相关数据

(c) Corr data set

图 7 数据规模在不同模型中对算法运行时间的影响

Fig. 7 Effects of dataset size on algorithm runtime in different models

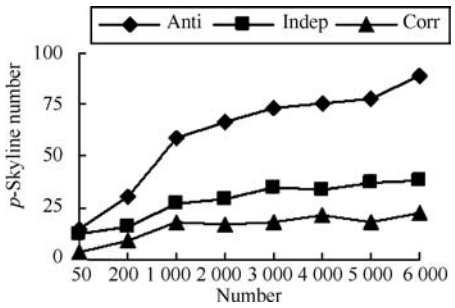


图 8 数据规模在圆 - 均匀分布模型中对 p-Skyline 规模的影响

Fig. 8 Effect of dataset size on p-Skyline size in circle-uniform model

5.3 不确定区域对算法影响

本实验考察移动对象不确定区域对算法性能影响. $N = 3\,000$, 静态维度 $d = 2$, $\boldsymbol{v} = (v_x, v_y) = (20, 25)$.

不确定区域半径 R 越大, 任意两数据点 $Pr(\boldsymbol{Q} \prec \boldsymbol{P}) \neq 0$ 的概率越大, 产生的 Event 数量增多, 如图 9 所示. 同理, $Pr(\boldsymbol{Q} \prec \boldsymbol{P}) \neq 1$ 的概率也增大. 当 $Pr(\boldsymbol{Q} \prec \boldsymbol{P}) = 1$ 时, 由式 (3) 可直接得到 $Pr(\boldsymbol{P}) = 0$; $Pr(\boldsymbol{Q} \prec \boldsymbol{P}) \neq 1$ 概率增大, 即 $Pr(\boldsymbol{P}) \neq 0$ 的概率增大, 对 $Pr(\boldsymbol{P})$ 的计算量增大, 从而增加了算法的处理时间; 对于任意 Event 而言, R 的增大, 导致 $|t_{\text{end}} - t_{\text{begin}}|$ 增大, 更新算法中跟踪计算的时间增长. 因此, 随着 R 的不断增大, 两种算法的平均处理时间都不断增大, 图 10 验证了上述分析. 图 11 显示了在不同不确定区域, 算法 U_CPSC 平均处理时间随 R 的增大而增大. 其中, 横轴 R 表示圆形区域的半径, 或者椭圆形区域的长轴半径, 椭圆的参数 $Ellipse_{R_L,R_S} = \{(2, 1.5), (6, 4), (10, 8), (14, 10), (18, 15)\}$.

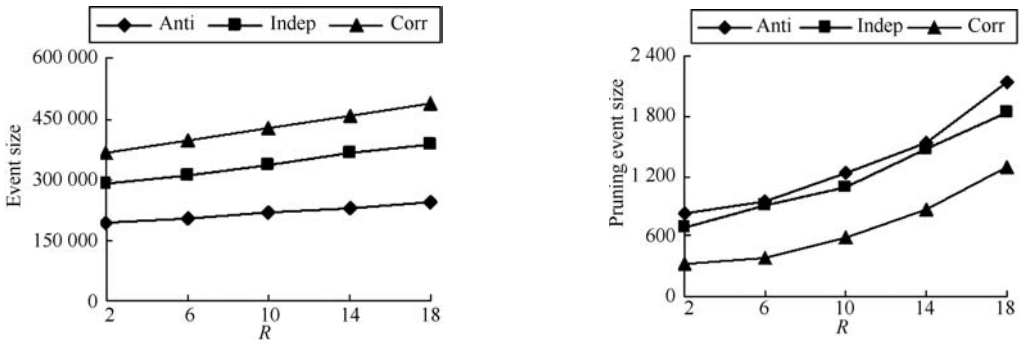
5.4 静态维度对算法影响

本实验考察静态维度对算法效率的影响. $N = 3\,000$, $Circle_R = 10$, $Ellipse_{R_L,R_S} = (10, 8)$, $\boldsymbol{v} = (v_x, v_y) = (20, 25)$. 在 Circle-uniform 模型下, 算法的平均处理时间 (图 12 所示) 和 Event 队列规模 (图 13 (a) 所示) 都随 d 的增大而不断减少. 对任意两点 $\boldsymbol{P}$, $\boldsymbol{Q}$, 当 d 增大时, 则存在两静态维 i, j 使得 $x_{si}^{\boldsymbol{P}} < x_{si}^{\boldsymbol{Q}}$ 和 $x_{sj}^{\boldsymbol{P}} < x_{sj}^{\boldsymbol{Q}}$ 成立的概率增大, 导致数据点间存在动态支配的概率减少, Event 队列规模减少, 因此, 对支配概率和 Event 计算的次数

减少, 算法的平均处理时间降低. d 增大使得其他数据点对 P 存在静态支配的概率降低, 则 P 不被其他点支配的概率增大, 因此, 在剪枝后对 p-Skyline 产生影响的 Event 数量随 d 的增加而不断增大, 图 13 (b) 验证了上述分析.

图 14 显示在 4 种不同的分布模型下, 随着 d 的增加, U_CPSC 算法的平均处理时间不断减少, 相比之下, 仍然是 Ellipse-Gauss 型所花的时间最多, Circle-uniform 型所花的时间最少. 随着维度的增大, 4 种算法的时间开销差距减小, 其根本原因在于 d 的增加, 造成数据间存在支配可能性的概率减低, 计算次数减少.

对数据点 $\boldsymbol{P}$ 而言, 随着维度的增加, 存在数据点 $\boldsymbol{Q}$ 使得 $\forall x_{si}^{\boldsymbol{P}} \leq x_{si}^{\boldsymbol{Q}} (1 \leq i \leq k)$ 成立的概率减少, 即 $\boldsymbol{P}$ 成为 p-Skyline 点的概率增大, 这里采用 Circle-uniform 型进行了验证, 如图 15 所示.

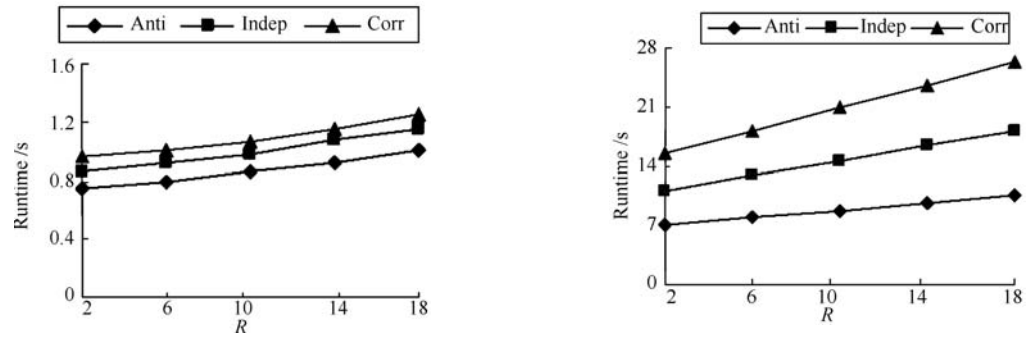


(a) 剪枝前 Event 规模　　(b) 剪枝策略对 Event 影响

(a) Event size before pruning　　(b) Effect of pruning techniques on event

图 9　R 在圆形 - 均匀分布模型中对 Event 的影响

Fig. 9　Effect of R on event size in circle-uniform model

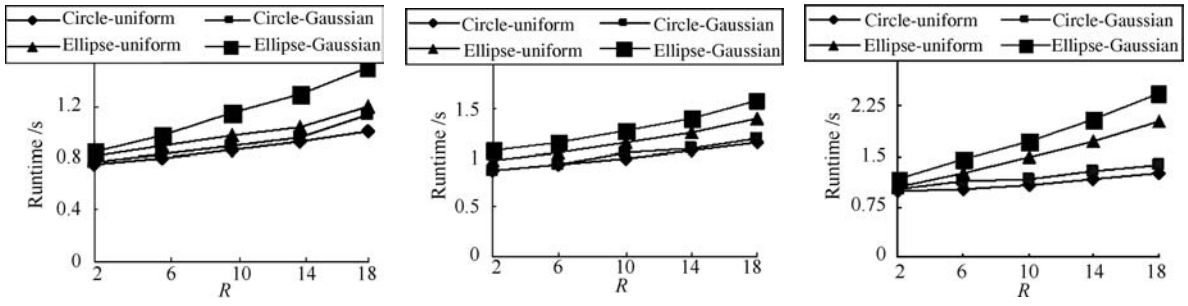


(a) U_CPSC 算法运算时间　　(b) U_SPSC 算法运算时间

(a) Runtime for U_CPSC algorithm　　(b) Runtime for U_SPSC algorithm

图 10　R 在圆形 - 均匀分布模型中对算法运行时间的影响

Fig. 10　Effect of dataset size on algorithm runtime in circle-uniform model

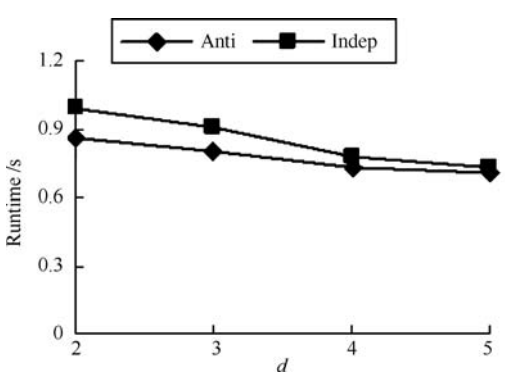


(a) 反相关数据　　(b) 独立数据　　(c) 正相关数据

(a) Anti data set　　(b) Indep data set　　(c) Corr data set

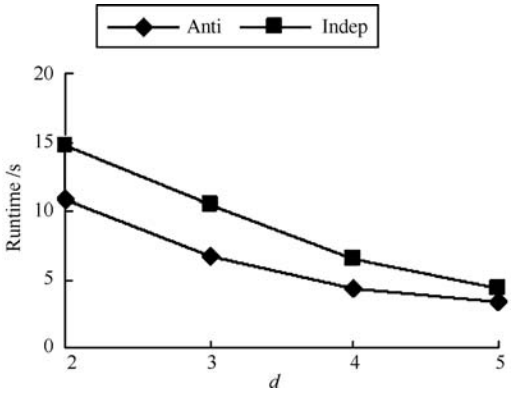
图 11　R 在不同模型中对算法运行时间的影响

Fig. 11　Effects of R on algorithm runtime in different models



(a) U_CPSC 算法的运算时间

(a) Runtime for U_CPSC algorithm

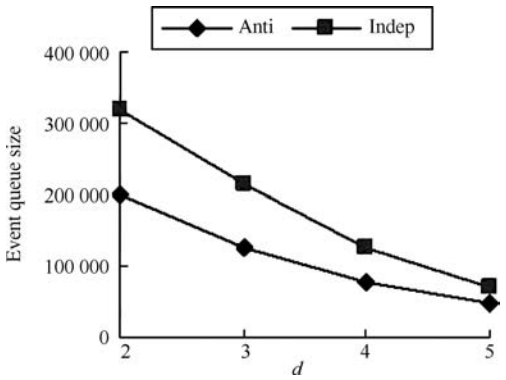


(b) U_SPSC 算法的运算时间

(b) Runtime for U_SPSC algorithm

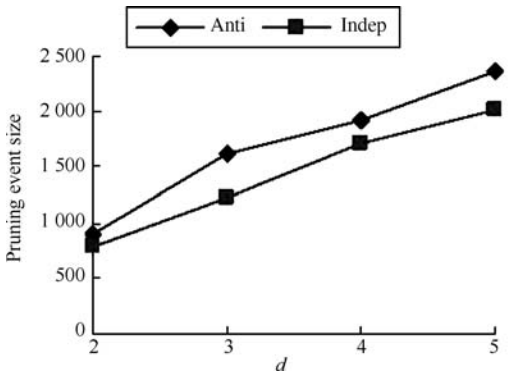
图 12 静态维度在圆形-均匀分布模型中对算法运行时间的影响

Fig. 12 Effect of static dimensionality on algorithm runtime in circle-uniform model



(a) 剪枝前 Event 规模

(a) Event size before pruning

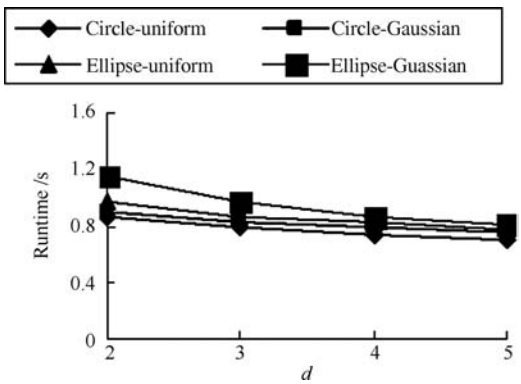


(b) 剪枝策略对 Event 影响

(b) Effect of pruning techniques on event

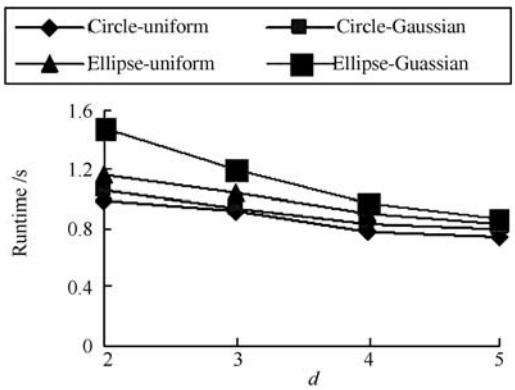
图 13 静态维度在圆形-均匀分布模型中对 Event 的影响

Fig. 13 Effect of static dimensionality on event size in circle-uniform model



(a) 反相关数据

(a) Anti data set



(b) 独立数据

(b) Indep data set

图 14 静态维度在不同模型中对算法运行时间的影响

Fig. 14 Effect of static dimensionality on algorithm runtime in different models

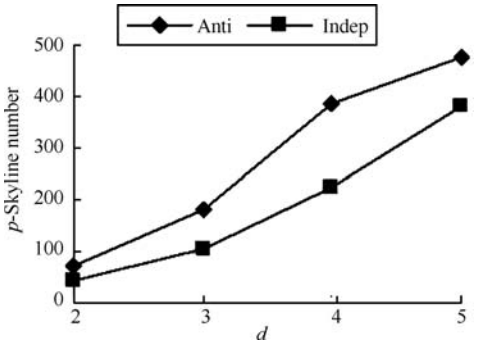


图 15　静态维度在圆形 - 均匀分布模型中对 p-Skyline 规模的影响

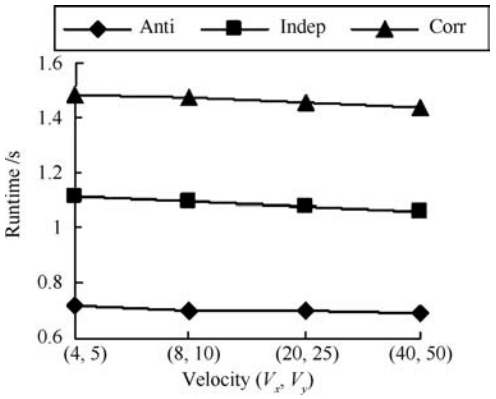
Fig. 15　Effect of static dimensionality on p-Skyline size in circle-uniform model

5.5　运动速度对算法影响

本实验验证运动速度对算法的影响, $N = 3\,000$, 静态维度 $d = 2$, Circle-uniform 型, $Circle_R = 10$. 图 16 说明随着速度的增加, 算法 U_CPSC 和 U_SPSC 的平均运算时间不断减少. 对算法 U_CPSC 而言, R 不变, 对 Event 更新计算时间 $|t_{\text{end}} - t_{\text{begin}}|$ 变小, 因此, 平均运算时间随速度增大不断降低, 但是算法 U_CPSC 的处理时间主要用在支配概率的计算上, 速度的变化并未带来支配概率的改变, 因此, 未对 U_CPSC 产生很大影响. 当速度增大到一定程度, 算法的平均运算时间将趋向于平稳. 而 U_SPSC 需要在移动对象的每一运动时刻去遍历计算整个数据集, 速度增大导致遍历计算次数减少, 平均运算时间也随之减少.

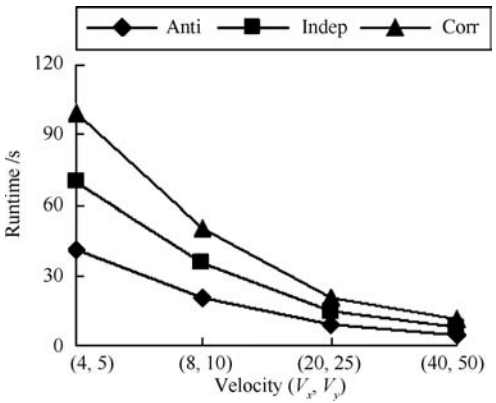
6　结束语

为了解决不确定移动对象的连续概率 Skyline 计算问题, 本文首先提出了在移动对象不确定条件下, 各数据点间支配概率的微元计算方法, 然后定义了两种类型的可能引起 p-Skyline 集合变动的 Event, 提出了对不确定移动对象进行连续概率 Skyline 计算的算法框架 U_CPSC, 通过对可能引起 p-Skyline 集合变动的 Event 的跟踪计算高效地维护 p-Skyline 集合. 提出了计算移动对象概率 Skyline 问题的静态算法 (U_SPSC) 并进行对比实验. 实验采用了 4 种不确定区域分布模型, 从数据的规模 (密度)、不确定区域的范围、静态维度和移动对象移动速度等几个方面进行了详细的分析探讨, 理论分析和实验结果证明了算法 U_CPSC 的有效性. 进一步的研究将关注查询点移动的同时, 目标对象也同时移动的概率 Skyline 查询, 即查询点和目标对象的位置都具有不确定性的条件下的概率 Skyline 查询.



(a) U_CPSC 算法的运算时间

(a) Runtime for U_CPSC algorithm



(b) U_SPSC 算法的运算时间

(b) Runtime for U_SPSC algorithm

图 16　速度对算法影响

Fig. 16　Effect of velocity

References

1 Borzsonyi S, Kossmann D, Stocker K. The skyline operator. In: Proceedings of the 17th International Conference on Data Engineering. Heidelberg, Germany: IEEE, 2001. 421−430

2 Pei J, Jiang B, Lin X, Yuan Y D. Probabilistic skylines on uncertain data. In: Proceedings of the 33rd International Conference on Very Large Data Bases. Vienna, Austria: VLDB Endowment, 2007. 15−26

3 Zhang W J, Lin X M, Zhang Y, Wang W, Yu J X. Probabilistic skyline operator over sliding windows. In: Proceedings of the 25th International Conference on Data Engineering. Shanghai, China: IEEE, 2009. 1060−1071

4 Atallah M J, Qi Y N. Computing all skyline probabilities for uncertain data. In: Proceedings of the 28th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems. New York, USA: ACM, 2009. 279−287

5 Bohm C, Fiedler F, Oswald A, Plant C, Wackersreuther B. Probabilistic skyline queries. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management. New York, USA: ACM, 2009. 651−660

6 Tan K L, Eng P K, Ooi B C. Efficient progressive skyline computation. In: Proceedings of the 27th International Conference on Very Large Data Bases. San Francisco, USA: Morgan Kaufmann Publishers, 2001. 301−310

7 Kossmann D, Ramsak F, Rost S. Shooting stars in the sky: an online algorithm for skyline queries. In: Proceedings of the 28th International Conference on Very Large Data Bases. Hong Kong, China: Morgan Kaufmann Publishers, 2002. 275−286

8 Papadias D, Tao Y, Fu G, Seeger B. Progressive skyline computation in database systems. *ACM Transactions on Database Systems*, 2005, **30**(1): 41−82

9 Zhou Hong-Fu, Gong Xue-Qing, Zheng Kai, Zhou Ao-Ying. CSky: an online efficient algorithm for subspace skyline computation in high dimensional space. *Chinese Journal of Computers*, 2007, **30**(8): 1409−1417
(周红福, 宫学庆, 郑凯, 周傲英. 基于高维空间的在线高效子空间 Skyline 算法 -CSky. 计算机学报, 2007, **30**(8): 1409−1417)

10 Sun Sheng-Li, Huang Zhen-Hua, Li Jin-Jiu, Guo Jian-Kui, Zhu Yang-Yong. Efficient computation of subspace skyline over data streams. *Chinese Journal of Computers*, 2007, **30**(8): 1418−1428
(孙圣力, 黄震华, 李金玖, 郭建奎, 朱扬勇. 数据流上高效计算子空间 Skyline 的算法. 计算机学报, 2007, **30**(8): 1418−1428)

11 Su Liang, Zou Peng, Jia Yan. Adaptive mining of sparse skyline over data stream. *Acta Automatica Sinica*, 2008, **34**(3): 360−366
(苏亮, 邹鹏, 贾焰. 数据流上自适应的稀疏 Skyline 挖掘. 自动化学报, 2008, **34**(3): 360−366)

12 Sun Sheng-Li, Dai Dong-Bo, Huang Zhen-Hua, Zhang Qi-Xun, Zhou Li-Xin. Algorithm on computing skyline over probabilistic data stream. *Acta Electronica Sinica*, 2009, **37**(2): 285−293
(孙圣力, 戴东波, 黄震华, 张齐勋, 周立新. 概率数据流上 Skyline 查询处理算法. 电子学报, 2009, **37**(2): 285−293)

13 Huang Zhen-Hua, Xiang Yang, Xue Yong-Sheng, Zhao Gang. An efficient method for parallel processing of skyline queries. *Acta Automatica Sinica*, 2010, **36**(7): 968−975
(黄震华, 向阳, 薛永生, 赵杠. 一种并行处理 Skyline 查询的有效方法. 自动化学报, 2010, **36**(7): 968−975)

14 Huang Z, Lu H, Ooi B C, Tung A K H. Continuous skyline queries for moving objects. *IEEE Transactions on Knowledge and Data Engineering*, 2006, **18**(12): 1645−1658

15 Lee M W, Hwang S W. Continuous skylining on volatile moving data. In: Proceedings of the IEEE International Conference on Data Engineering . Washington D. C., USA: IEEE, 2009. 1568−1575

16 Cheng R, Kalashnikov D V, Prabhakar S. Querying imprecise data in moving object environments. *IEEE Transactions on Knowledge and Data Engineering*, 2004, **16**(9): 1112−1127

17 Ding Xiao-Feng, Lu Yan-Sheng, Pan Peng, Hong Liang, Wei Qiong. U-Tree based indexing method for uncertain moving objects. *Journal of Software*, 2008, **19**(10): 2696−2705
(丁晓锋, 卢炎生, 潘鹏, 洪亮, 魏琼. 基于 U-tree 的不确定移动对象索引策略. 软件学报, 2008, **19**(10): 2696−2705)

18 Benetis R, Jensen C, Karciauskas G, Saltenis S. Nearest neighbor and reverse nearest neighbor queries for moving objects. In: Proceedings of the International Database Engineering and Applications Symposium. Edmonton, Canada: IEEE, 2002. 44−53

付世昌 宁波大学信息科学与工程学院硕士研究生. 主要研究方向为移动数据库, 数据挖掘.
E-mail: 2000shizi@163.com
(**FU Shi-Chang** Master student at the College of Information Science and Engineering, Ningbo University. His research interest covers mobile database and data mining.)

董一鸿 博士, 宁波大学信息科学与工程学院教授. 主要研究方向为移动数据库, 数据挖掘, 人工智能. 本文通信作者.
E-mail: dongyihong@nbu.edu.cn
(**DONG Yi-Hong** Ph. D., professor at the College of Information Science and Engineering, Ningbo University. His research interest covers mobile database, data mining, and artificial intelligence. Corresponding author of this paper.)

唐燕琳 浙江大学计算机科学与技术学院硕士研究生. 2010 年获得宁波大学计算机科学与技术专业学士学位. 主要研究方向为移动数据库, 数据挖掘.
E-mail: tangyanlin0210@163.com
(**TANG Yan-Lin** Master student at the College of Computer Science and Technology, Zhejiang University. She received her bachelor degree in computer science from Ningbo University in 2010. Her research interest covers mobile database and data mining.)

陈华辉 博士, 宁波大学信息科学与工程学院副教授. 主要研究方向为数据流, 数据挖掘.
E-mail: chenhuahui@nbu.edu.cn
(**CHEN Hua-Hui** Ph. D., associate professor at the College of Information Science and Engineering, Ningbo University. His research interest covers data stream and data mining.)

钱江波 博士, 宁波大学信息科学与工程学院副教授. 主要研究方向为数据库, 数据流.
E-mail: qianjiangbo@nbu.edu.cn
(**QIAN Jiang-Bo** Ph. D., associate professor at the College of Information Science and Engineering, Ningbo University. His research interest covers database and data stream.)