doi: 10.12068/j.issn.1005-3026.2017.07.002

基于排序树索引的轨迹压缩方法

林树宽, 张培鹤, 刘晓强, 乔建忠
(东北大学 计算机科学与工程学院, 辽宁 沈阳 110169)

摘 要: 原始采集的GPS轨迹数据通常非常庞大,导致对其的传输、存储和处理变得越来越困难,需要对原始GPS轨迹数据进行压缩.现有的基于方向的轨迹压缩方法存在可容忍误差难以确定、计算代价大、压缩效果较差等问题.针对这些问题,提出了基于排序树索引的轨迹压缩方法,借助于排序树索引,在轨迹压缩的过程中进行有效的剪枝,提高了轨迹压缩的效率.同时,对压缩轨迹中轨迹点的去留起决定作用的指标——线段误差进行了重新定义,提高了轨迹压缩的效果.大量真实数据集上的实验验证了所提的轨迹压缩方法的有效性和高效性.

关 键 词: 轨迹压缩;排序树索引;轨迹线段;线段误差;压缩轨迹误差
中图分类号: TP 391 **文献标志码**: A **文章编号**: 1005-3026(2017)07-0918-05

Trajectory Compression Method Based on Sort Tree Index

LIN Shu-kuan, *ZHANG Pei-he*, *LIU Xiao-qiang*, *QIAO Jian-zhong*
(School of Computer Science & Engineering, Northeastern University, Shenyang 110169, China. Corresponding author: ZHANG Pei-he, E-mail: 825966816@qq.com)

Abstract: Raw GPS trajectories are usually long and the data volume is large. The transmission, storage and processing of trajectory data are becoming more and more difficult. So, raw GPS trajectories data need to be compressed. However, those problems still exist in the current DPTC (direction-preserving trajectory compression) methods, such as that error tolerance is difficult to determine, computation cost is high, and compression effect is bad. For these problems, the trajectory compression method based on sort tree index was proposed. With the help of the sort tree index, effective pruning in the trajectory compression process improved the efficiency of trajectory compression. Meanwhile, the segment error was redefined, which played a decisive role in determining whether a trajectory point is preserved in the compressed trajectory or not, and the compression effect was enhanced. The extensive experimental results on real data sets show the effectiveness and efficiency of the proposed method.

Key words: trajectory compression; sort tree index; trajectory segment; segment error; compressed trajectory error

随着GPS设备的广泛应用和位置采集技术的提高,轨迹数据无处不在.轨迹数据的分析与挖掘可用于对交通情况进行分析、为旅行者推荐最佳的旅游路径、对社会关系或者用户行为进行分析等.原始轨迹数据通常非常庞大,导致传输、存储和处理的开销非常大.此外,许多应用并不需要给出高密集度的位置信息[1].因此,通常情况下,在原始轨迹数据传输、存储和处理之前要对其进行压缩.

目前,大部分的轨迹压缩方法是保留位置的轨迹压缩(position-preserving trajectory compression, PPTC)[2-9],主要考虑轨迹的位置信息以决定轨迹点是否保留.该类方法只保留了轨迹的位置信息,而忽略了轨迹的方向信息,压缩效果不是十分理想.近期,Long等[10-11]提出了一种新的压缩轨迹的框架——保留方向的轨迹压缩(direction-preserving trajectory compression, DPTC),其主要思想是在轨迹压缩过程中,保留

收稿日期: 2017-01-19
基金项目: 国家自然科学基金资助项目(61272177).
作者简介: 林树宽(1966-),女,吉林长春人,东北大学教授; 乔建忠(1964-),男,辽宁兴城人,东北大学教授,博士生导师.

轨迹的方向信息，同时限制轨迹位置的丢失. 从压缩效果上看，DPTC 要优于 PPTC[10]. 本文主要针对保留方向的轨迹压缩方法进行研究.

文献[10]中的 DPTC 主要研究的是在压缩轨迹的误差不超过可容忍误差的条件下，其尺度最小化，被称为最小尺度问题（min-size problem）. 最小尺度问题只适合于用户对可容忍误差已知的情况. 但是，大多数情况下用户无法明确地指定合理的可容忍误差. 文献[11]中的 DPTC 主要研究的是在给定压缩轨迹尺度的条件下，压缩误差最小，被称为最小误差问题（min-error problem），该问题是最小尺寸问题的一个对偶问题，无需用户指定可容忍误差[11]. 虽然，最小误差问题提高了压缩后轨迹的可用性，但是，当前最小误差问题的解决方法时间代价较大，且压缩效果不理想，无法达到用户的需求. 为了减小最小误差问题的时间代价，改善压缩效果，本文提出了基于排序树索引的轨迹压缩方法，借助于排序树索引，在轨迹压缩过程中进行有效的剪枝，提高了轨迹压缩的时间效率. 同时在压缩过程中，重新定义了对压缩轨迹误差有重要影响的线段误差，改进了压缩效果.

1　问题定义

GPS 原始轨迹是由一系列时空采集点构成的集合，$T=\{(p_1,t_1),(p_2,t_2),\cdots,(p_n,t_n)\}$ 表示移动对象在时间点 $t_i(i\in[1,n])$ 位于轨迹点 p_i，其中，$t_1<t_2<\cdots<t_n$，$p_i=(x_i,y_i)$ 是以经纬度表示的轨迹点. 轨迹中包含的轨迹点数称为轨迹长度. 为表达方便，本文将移动对象的轨迹表示为 $T=\{p_1,p_2,\cdots,p_n\}$.

定义 1　轨迹线段：轨迹点 p_i 与 p_j $(1\leqslant i<j\leqslant n)$ 的连线称为连接 p_i 与 p_j 的轨迹线段，表示为 p_ip_j，其中，p_i 称为线段的左端点，p_j 称为线段的右端点.

p_i 与 p_j 之间所有连接相邻轨迹点的轨迹线段集合表示为 $e[p_i:p_j]$，即 $e[p_i:p_j]=\{p_ip_{i+1},p_{i+1}p_{i+2},\cdots,p_{j-1}p_j\}$.

例 1　在图 1 所示的原始轨迹 $T=\{p_1,p_2,\cdots,p_{10}\}$ 中，轨迹长度为 10，p_1p_2，p_1p_4 都是轨迹线段，$e[p_1:p_2]=\{p_1p_2\}$，$e[p_1:p_4]=\{p_1p_2,p_2p_3,p_3p_4\}$.

定义 2　线段方向：给定轨迹 T 的一条线段 p_ip_j，其方向是在以 p_i 为原点，以经度方向为 x 轴，以纬度方向为 y 轴的坐标系中，沿 x 轴正方向逆时针旋转到向量 $\overrightarrow{p_ip_j}$ 的角度，表示为 $\mathrm{dir}(p_ip_j)$.

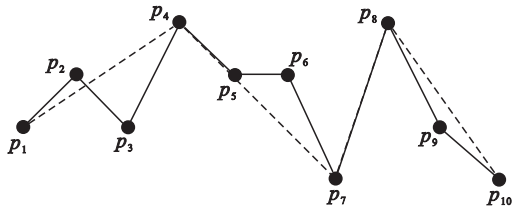

图 1　轨迹线段示例

Fig. 1　An example of trajectory segment

对于线段集合 $e[p_i:p_j]$ 中所有的线段，其方向的集合表示为 $\mathrm{dir}[p_i:p_j]$，即 $\mathrm{dir}[p_i:p_j]=\{\mathrm{dir}(p_ip_{i+1}),\mathrm{dir}(p_{i+1}p_{i+2}),\cdots,\mathrm{dir}(p_{j-1}p_j)\}$.

例 2　图 2 中，线段 p_1p_2 的方向为 $\mathrm{dir}(p_1p_2)=\alpha$，$\mathrm{dir}[p_1:p_4]=\{\alpha,\beta,\gamma\}$.

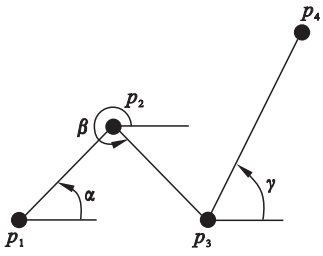

图 2　线段方向示例

Fig. 2　Examples of segment directions

对于方向 α 和 β，其偏差 $\mathrm{diff}(\alpha,\beta)$ 可描述为从 α 到 β 或者从 β 到 α 逆时针旋转的最小角度，即

$$\mathrm{diff}(\alpha,\beta)=\min\{|\alpha-\beta|,2\pi-|\alpha-\beta|\}. \quad (1)$$

定义 3　线段误差：线段 p_ip_j 与原始轨迹 T 之间的误差 $\varepsilon(p_ip_j)$ 定义为方向 $\mathrm{dir}(p_ip_j)$ 与方向集合 $\mathrm{dir}[p_i:p_j]$ 中所有方向之间角差和的均值，即

$$\varepsilon(p_ip_j)=\frac{\sum_{i\leqslant k<j}\mathrm{diff}(\mathrm{dir}(p_ip_j),\mathrm{dir}(p_kp_{k+1}))}{|\mathrm{dir}[p_i:p_j]|}. \quad (2)$$

其中，$|\mathrm{dir}[p_i:p_j]|$ 为集合 $\mathrm{dir}[p_i:p_j]$ 的基数.

线段误差是决定轨迹点是否保留在压缩轨迹中的重要指标，本文的上述定义充分体现了轨迹线段偏离原始轨迹的程度，有助于改善轨迹压缩的效果.

定义 4　压缩轨迹误差：给定原始轨迹 $T=\{p_1,p_2,\cdots,p_n\}$，其压缩轨迹 $\mathrm{CT}=\{p_{s_1},p_{s_2},\cdots,p_{s_m}\}$ $(m\leqslant n)$ 的误差 $\varepsilon(\mathrm{CT})$ 定义为 CT 中所有相邻轨迹点间的线段误差总和，即

$$\varepsilon(\mathrm{CT})=\sum_{1\leqslant k<m}\varepsilon(p_{s_k}p_{s_{k+1}}). \quad (3)$$

压缩轨迹中包含的轨迹线段数量与原始轨迹

包含的轨迹线段数量的比率称为压缩轨迹尺度. 通常,在进行轨迹压缩时由用户指定最大的压缩轨迹尺度,称为压缩轨迹尺度阈值,表示为 τ.

本文的目标是对原始 GPS 轨迹 T 进行压缩,在满足尺度要求的前提下,使得压缩后的轨迹 CT 与原始轨迹 T 之间的误差最小. 具体的问题定义描述见定义 5.

定义 5 轨迹压缩:给定原始轨迹 T 和压缩轨迹尺度阈值 τ,轨迹压缩即是发现 T 的压缩轨迹 CT,满足:①$|\mathrm{CT}| \leqslant \tau \times (n-1)$($|\mathrm{CT}|$ 表示压缩轨迹 CT 所包含的轨迹线段数量,n 为原始轨迹长度);②$\varepsilon(\mathrm{CT})$ 最小. 同时满足上述两个条件的轨迹称为最终压缩轨迹.

2 基于排序树索引的轨迹压缩

本文通过建立排序树索引进行轨迹压缩. 排序树中每个节点 node 的结构如图 3 所示.

Segment	Parent	Error	Stack

图 3 排序树中节点的结构
Fig. 3 Structure of nodes in sort tree

其中,Segment 是以轨迹线段 p_ip_j 表示的节点标志(根节点除外,其标识为 null),因此,节点 node 又称为节点 p_ip_j;Parent 是指向其父节点的指针,在基于排序树索引进行轨迹压缩的过程中,需要依据节点的 Parent 域向上回溯,以完成路径搜索过程;Error 是节点 node 中线段 Segment 的误差 ε(Segment),在基于排序树索引的轨迹压缩中用于计算压缩轨迹误差;Stack 是由 node 的子节点构成的排序栈,栈中的元素是由其子节点的线段及其误差构成的二元组(Segment, ε(Segment)),栈中元素从栈顶到栈底按照误差非递减的顺序排序. 借助于排序栈可按顺序生成 node 的子节点,因此,该树结构本文称为排序树.

排序树的建立应满足以下条件:

1) 根节点的子节点是以轨迹起点 p_1 为左端点的线段;

2) 每个节点的子节点按照其排序栈中的顺序建立;

3) 父节点线段的右端点与其子节点线段的左端点相同.

由上述条件可以看出,当叶节点为轨迹终点时,从根节点到叶节点的路径将形成一条压缩轨迹.

为了描述基于排序树索引的轨迹压缩过程,先给出以下定义.

定义 6 累计误差:若节点 node 的线段为 p_ip_j,其累计误差定义为从根节点至节点 node 的路径上所有线段误差的累加和,表示为 ε^{s}(node)或 $\varepsilon^{\mathrm{s}}(p_ip_j)$.

定义 7 节点高度:若节点 node 的线段为 p_ip_j,其高度定义为从根节点至节点 node 的路径上所有线段的计数.

定义 8 最小误差:在当前找到的所有压缩轨迹中,最小的压缩轨迹误差称为最小误差,表示为 ε_0.

基于排序树索引的轨迹压缩过程主要由以下三步构成:首先,针对给定的原始轨迹,创建一个误差空间 $\boldsymbol{E}$;然后,根据误差空间 $\boldsymbol{E}$,为每条线段创建由其子节点及其误差构成的排序栈;最后,创建排序树索引,返回最终压缩轨迹.

1) 创建误差空间 $\boldsymbol{E}$. 误差空间 $\boldsymbol{E}$ 是以每个轨迹点 $p_k(1 \leqslant k \leqslant n)$ 为行、列的上三角矩阵. 矩阵元素 $\boldsymbol{E}[i,j]$ 为线段 $p_ip_j(1 \leqslant i < j \leqslant n)$ 的误差 $\varepsilon(p_ip_j)$.

创建误差空间的目的是为了下一步建立子节点排序栈,误差空间所占用的内存可在子节点排序栈建立之后进行释放.

2) 子节点排序栈的建立. 根据误差空间 $\boldsymbol{E}$ 中的误差对每条线段创建排序栈,用以存储其子节点线段及其误差,使其子节点可按照误差大小有序地建立.

根据排序树建立的条件,对于节点 p_ip_j,其子节点是以 p_j 为左端点的线段,其子节点排序栈如图 4 所示.

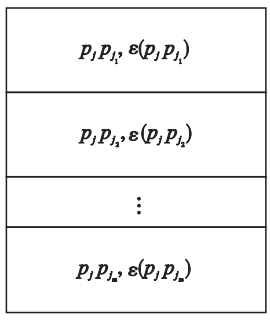

图 4 排序树节点 p_ip_j 的子节点排序栈
Fig. 4 Child-node sort stack of node p_ip_j in sort tree

其中,$j < j_1, j_2, \cdots, j_m \leqslant n$ 且 $\varepsilon(p_jp_{j_1}) \leqslant \varepsilon(p_jp_{j_2}) \leqslant \cdots \leqslant \varepsilon(p_jp_{j_m})$.

特别地,根节点的排序栈存储的是以轨迹起点 p_1 为左端点的线段及其误差.

3) 排序树索引的创建. 创建排序树的过程

即是发现最终压缩轨迹的过程. 以排序树为索引，进行轨迹压缩的步骤为

步骤 1　初始化根节点，其数据域 Segment，Parent，Error 和 Stack 分别初始化为 Null，Null，0 和由以轨迹起点为左端点的线段及其误差构成的排序栈，并将根节点作为当前节点 node.

步骤 2　当前节点 node 的排序栈不为空时，弹出其栈顶，建立 node 的子节点 child. 若子节点 child 中的线段不为 p_ip_n（其中，$i<n$，p_n 为原始轨迹终点），则以 child 作为当前节点 node，重复执行步骤 2；否则，若子节点 child 中的线段为 p_ip_n，则根节点至 child 的路径形成一条压缩轨迹，此时，需要进行以下处理：

①计算累计误差 ε^s(child)；

②将最小误差 ε_0 更新为 min(ε_0，ε^s(child))；

③继续以 node 为当前节点，重复步骤 2.

步骤 3　若当前节点 node 的排序栈为空，则对当前节点进行回溯，将其父节点作为当前节点 node，重复执行步骤 2.

步骤 4　当回溯到根节点且其排序栈为空时，整个过程结束. 此时，与最小误差 ε_0 相对应的压缩轨迹即为最终压缩轨迹.

在上述基于排序树索引的轨迹压缩过程中，本文依据下面的性质和定理进行安全剪枝，从而大大提高了基于排序树索引进行轨迹压缩的效率.

性质 1　若当前节点的高度为 $\tau\times(n-1)$，则其所有子孙节点均无需建立.

定理 1　若当前节点的子节点排序栈的栈顶误差超过最小误差 ε_0，则其所有子孙节点均无需建立.

证明略.

定理 2　在当前节点的所有子节点中，若有一个子节点，其所含轨迹线段的右端点已为轨迹终点，则当前节点的其他子节点均无需建立.

证明略.

3　实验及分析

为了验证所提出的基于排序树索引的轨迹压缩方法的性能，本文在两个真实的数据集（Geolife 和 T-Drive）上进行了实验，其中，GeoLife 包含了 182 个用户的 17 621 条轨迹数据，覆盖了中国近 30 个城市以及美国和欧洲的部分城市；T-Drive 包含 10 357 辆出租车采集的轨迹样本，包含有 1.5 亿个轨迹点.

所有实验程序的开发环境为 Intel（R）Core™ I5－2400 CPU @ 3.10 GHz，4GB 内存，操作系统为 Windows 7（64 位），编程语言为 JAVA.

实验将本文提出的基于排序树索引（sort tree，ST）的轨迹压缩方法与文献[11]中的动态规划（dynamic programming，DP）方法和误差搜索（error search，ES）方法分别从运行时间、空间占用、压缩轨迹误差，以及实际压缩效果 4 个方面进行比较.

图 5 给出了原始轨迹长度和压缩轨迹尺度阈值 τ 变化时，三种方法的运行时间比较. 从图 5 可以看出，三种方法的运行时间均随原始轨迹长度的增加而增大，因为原始轨迹越长，需要处理的轨迹点数越多. 从图 5 还可以看出，ST 方法相比于 DP 和 ES 方法，轨迹压缩所需的时间更短，因为 ST 的轨迹压缩在借助于排序树索引的基础上，从三方面（见性质 1、定理 1 和定理 2）进行了安全剪枝，显著减少了发现最终压缩轨迹的时间.

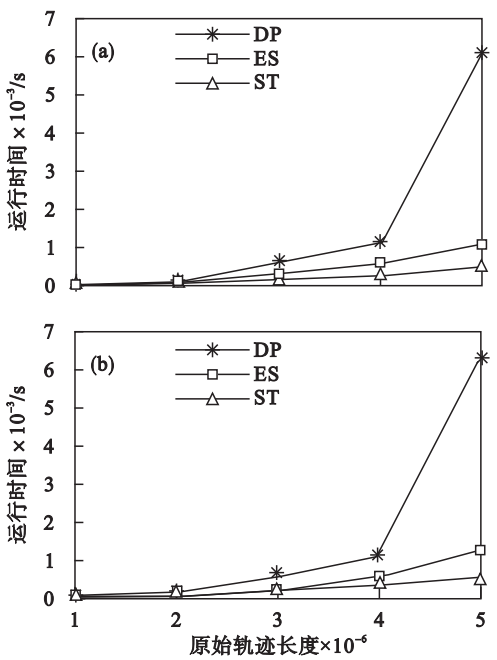

图 5　原始轨迹长度对运行时间的影响

Fig. 5　Effect of raw trajectory length on running time

（a）—Geolife；（b）—T-Drive.

图 6 给出了两个数据集上压缩轨迹误差随原始轨迹长度的变化情况. 文献[11]中 DP 和 ES 方法的压缩误差相同，因此，图 6 只给出了 ST 与 ES 方法的压缩误差比较. 由图 6 可以看出，ST 与 ES 方法的压缩轨迹误差都随原始轨迹长度呈正相关增长，但 ES 方法的压缩误差高于 ST 方法，且随着原始轨迹长度不断增加，二者差距越来越大. 原

因在于,在轨迹压缩过程中,原始轨迹中的哪些轨迹点能保留在压缩轨迹中,有一个重要的指标就是线段误差,本文对线段误差重新进行了定义,ST 方法的线段误差计算方式相比于 ES 方法能更好地体现线段偏离原始轨迹的偏差,压缩轨迹中的保留点可更好地表达原始轨迹,因此,压缩轨迹误差低于 ES 方法,且原始轨迹越长,优势越明显.

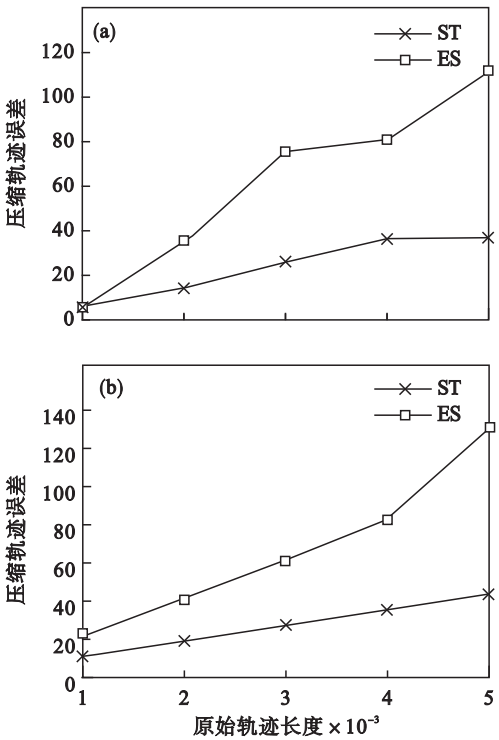

图 6 原始轨迹长度对压缩轨迹误差的影响

Fig. 6 Effect of raw trajectory length on compressed trajectory error

(a)—Geolife; (b)—T-Drive.

4 结 论

本文对考虑方向的轨迹压缩方法进行了研究. 针对现有的基于方向的轨迹压缩方法的不足,提出了基于排序树索引的轨迹压缩方法. 在轨迹压缩的过程中,对与轨迹点保留密切相关的指标进行了重新定义,并提出了排序树索引建立过程中安全剪枝的方法. 真实数据集上的实验表明,本文所提出的方法相比于已有的轨迹压缩方法,在运行时间和压缩轨迹误差方面均有较大提升,实际的压缩效果更好.

参考文献:

[1] Zheng Y. Trajectory data mining: an overview [J]. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2015, 6(3): 1 – 41.

[2] Douglas D H, Peucker T K. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature [J]. *Cartographica the International Journal for Geographic Information and Geovisualization*, 1973, 10(2): 112 – 122.

[3] Meratnia N, Rad B. Spatiotemporal compression techniques for moving point objects [C]//Proceedings of International Conference on Advances in Database Technology. Heraklion, 2004: 765 – 782.

[4] Keogh E, Chu S, Hart D, et al. An online algorithm for segmenting time series [C]//Proceedings of IEEE International Conference on Data Mining (ICDM). San Jose, 2001: 289 – 296.

[5] Yin H, Wolfson O. A weight-based map matching method in moving objects databases [C]//Proceedings of the 16th International Conference on Scientific and Statistical Database Management. Santorini Island, 2004: 437 – 438.

[6] Song R, Sun W, Zheng B, et al. PRESS: a novel framework of trajectory compression in road networks [J]. *Proceedings of the VLDB Endowment*, 2014, 7(9): 661 – 672.

[7] Schmid F, Richter K F, Laube P. Semantic trajectory compression [C]//Proceedings of International Symposium on Advances in Spatial & Temporal Databases. Munich, 2009: 411 – 416.

[8] Hershberger J, Snoeyink J. Speeding up the Douglas-Peucker line-simplification algorithm [C]//Proceedings of International Symposium on Spatial Data Handling. Beijing, 2000: 134 – 143.

[9] Muckell J, Hwang J H, Lawson C T, et al. Algorithms for compressing GPS trajectory data: an empirical evaluation [C]//Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems. San Jose, 2010: 402 – 405.

[10] Long C, Wong R C W, Jagadish H V. Direction-preserving trajectory simplification [J]. *Proceedings of the VLDB Endowment*, 2013, 6(10): 949 – 960.

[11] Long C, Wong R C W, Jagadish H V. Trajectory simplification: on minimizing the direction-based error [J]. *Proceedings of the VLDB Endowment*, 2014, 8(1): 49 – 60.