Journal of Computer Applications ISSN 1001-9081 2017
计算机应用,2017 CODEN JYIIDU http://www.joca.cn

最新录用 doi:

基于预测模型的轨迹数据压缩方法

陈 煜，蒋伟*，周继恩

(中国银联股份有限公司 银联科技事业部，上海 201201)

(*通信作者电子邮箱 jiangwei1@unionpay.com)

摘 要: 针对目前路网环境下海量轨迹数据压缩效率低下的问题，提出了一种基于预测模型的轨迹数据压缩方法(CTPM)。通过将轨迹数据的时间信息和空间信息分别进行压缩，使得压缩后的轨迹数据在空间维度上无损，并且在时间维度上误差有界，以此提高压缩效率。在空间方面，首先利用部分匹配预测(PPM)算法通过轨迹已经行驶的部分路段对其下一时刻可能的位置进行预测;然后通过删除预测成功的路段来减少轨迹数据的存储代价。在时间方面，首先利用轨迹通行状况具有周期性的特点，构建了不同时间区间的通行速度统计模型，来预测移动对象进入下一路段所需要的时间;然后删除预测时间误差小于给定阈值的路段数据来进行压缩处理。实验结果显示，与已有的基于路网的并行轨迹压缩算法(PRESS)相比，CTPM 的空间压缩比和时间压缩比平均分别提高了 43%和 1.5%，同时时间压缩误差减小了 9.5%。实验结果表明所提算法在提高压缩比的同时有效的降低了压缩时间和压缩误差。

关键词: 时序数据; 时空数据库; 轨迹; 轨迹压缩; 预测模型

中图分类号:TP392(各种专用数据库) **文献标志码**: **A**

Compression method for trajectory data based on prediction model

CHEN Yu, JIANG Wei*, ZHOU Ji-En

(Science and Technology Department , China Unionpay, Shanghai 201201, China)

Abstract: A Compression method for Trajectory data based on Prediction Model (CTPM) was proposed to solve the low compression efficiency of massive trajectory data in road network environment. The method compresses the time information and spatial information of the trajectory data respectively so that the compressed trajectory data is lossless in the spatial dimension and the error is bounded in the time dimension. In terms of space, the Prediction by Partial Matching algorithm (PPM) was used to predict the possible position of the next moment by the part of the trajectory that has been driven. And then the predicted road segments was deleted to reduce the storage cost. In terms of time, the traffic speed model of different time intervals was constructed by using the features of the traffic condition to predict the required time for moving objects to enter the next section. And then the compression process was performed by deleting the time information when the error was smaller than the given threshold. In the comparison experiments with Paralleled Road-network-based trajectory comprESSion (PRESS), the average compression ratio of CTPM was increased by 43% in spatial and 1.5% in temporal, and the temporal error was decreased by 9.5 %. The experimental results show that the proposed algorithm can effectively reduce the compression time and compression error while improving the compression ratio.

Keywords: time series data; spatio-temporal database; trajectory; trajectory compression; prediction model

0 引言

随着定位技术的发展，人们可以很容易地通过全球定位系统(Global Positioning System，GPS)设备获得移动对象的实时位置信息，从而得到移动物体的运行轨迹。由于移动对象的运行状况具有连续性，需要不断地获取 GPS 采样点以便更好地通过离散的采样点拟合移动对象的运行轨迹。然而较高的采样频率会产生大量的 GPS 记录，给轨迹数据的存储、管理和分析带来巨大的挑战[1]。为了解决这个问题，大量的轨迹压缩算法通过对数据的时空特性进行分析，通过保存那

收稿日期: 2017-06-09; 修回日期:2017-09-13。

作者简介: 陈煜(1972—)，男，浙江乐清人，高级工程师，硕士研究生，非 CCF 会员，主要研究方向：大数据、人工智能；蒋伟（1990-），男，甘肃酒泉人，硕士，主要研究方向：数据管理、数据分析；周继恩(1976－)，男，江苏吴江人，高级工程师，博士，非 CCF 会员，主要研究方向：大数据、人工智能。

些包含重要信息的采样点[13]，如转向点、速度剧烈变化点、车辆停止点等，来达到压缩轨迹的目的。

近几年来，随着城市路网结构数据的不断完善，结合路网结构的轨迹压缩变得可行。为了降低轨迹数据的误差，首先对原始轨迹进行路网匹配(Map-Matching)预处理[2-3]。由于路网结构信息往往是静态的并且结构相对固定，因此利用路网结构信息对轨迹进行约束，不仅可以有效地降低轨迹数据的数据规模,同时可以通过路网信息降低因 GPS 设备精度造成的数据误差[14]。

Hu 等[4]提出的 Nonmaterialized 算法首次提出了利用路网结构对轨迹数据进行压缩。该算法首先将轨迹采样点数据通过路网匹配算法映射在路网上，然后通过 Nonmaterialized 算法将映射过得轨迹数据转化为路口序列，以达到压缩轨迹的目的。Lerin 等[5]对 Nonmaterialized 算法进行改进，通过最短路径(Shortest Path)算法和链接算法来对轨迹数据进行进一步压缩。主要思想是:若轨迹行驶路径与最短路径一致时，只保留头尾两个路段;若行驶轨迹每次都沿着最小转角的路段行驶，那么也只用头尾两条路段来代替整段轨迹。两条路段间的最短路径可以通过 Dijkstra 算法[6]或者 A-Star 算法[7]计算得到，利用计算来对数据进行压缩。然而上述这些方法都没有考虑时间信息，无法知道压缩后的轨迹在某个时间点处于哪一个位置。

Kellaris 等[8-9]提出的路网匹配轨迹压缩 (Map-Matched Trajectory Compression，MMTC)算法把轨迹压缩问题转化成了函数最优化问题，主要思想是找到一条最适合的轨迹，使得这条轨迹尽可能地用最短路径表示，并且近似轨迹与原始轨迹相似度较高。MMTC 算法用最小描述长度 (Minimal Description Length，MDL) 模型来最优化压缩率和相似度组成的目标函数。在 MMTC 算法中，MDL 模型由两个部分组成，分别是 L(H)和 L(D/H)，L(H)表示压缩后的轨迹长度，L(D/H)表示压缩后的近似轨迹与原始轨迹的误差。根据 MDL 原则，需要找到一条路径使得 L(H) + L(D/H)最小，那么这条路径就满足目标条件。MMTC 算法可以在压缩率和准确率两个方面得到很好的权衡，然而它同样没有考虑时间约束。

Song 等[10]提出基于路网的并行轨迹压缩(Paralleled Road-Network-Based Trajectory Compression，PRESS)算法，在 MMTC 算法的基础上将轨迹的时间信息和空间信息分别进行压缩。在空间信息方面，PRESS 首先将 GPS 轨迹点映射在路网上，通过最短路径算法省略中间的路段，随后通过对一部分数据集进行训练，找到频繁的子轨迹。利用子轨迹的频繁度对轨迹进行霍夫曼编码，以此达到二次压缩的目的。这使得轨迹数据在空间方面是无损的，即压缩后的轨迹和原始轨迹是经过同一条路径的。在时间信息方面，PRESS 通过提出时间同步路网距离(Time Synchronized Network Distance，TSND)和路网同步时间距离(Network Synchronized Time Difference，NSTD)两个误差度量，将时间压缩的误差控制在设定的误差范围，由此对轨迹数据的时间信息进行有损压缩。

然而上述方法，如 MMTC 算法[8-9]和 PRESS 算法[10]都是基于一种假设，即假设轨迹总是沿着路网中最短路径行驶的，因此这两种算法通过将轨迹的运动轨迹以最短路径的形式表示，从而达到轨迹压缩的目的。然而，在现实生活中，由于受到交通状况的限制，如交通信号灯、道路拥堵状况、道路施工和交通事故等，大部分移动物体并不能按照行程最短的路径行驶。

为了解决上述问题，本文基于路网信息结构提出了基于预测模型的轨迹数据压缩方法(CTPM)。由于移动对象的运动受到交通状况的限制，使得轨迹数据往往具有重复性和周期性。比如，由于受到上班高峰期的影响，从住所到上班地点通常会行驶在用户偏好的路径上，这条路径往往是通过对历史交通状况的判断而选择的一条最优路径，同时这条路径往往会在工作日内每天重复。因此，通过对历史轨迹数据进行学习，得到轨迹运行的预测模型，通过预测模型对轨迹数据进行压缩，将预测成功的采样点删除，保留那些预测失败的采样点作为轨迹关键点存储。

1 相关背景及问题定义

CTPM 算法整体上分为学习和压缩两个阶段。在学习阶段，大量历史的 GPS 轨迹数据作为输入，经过噪点过滤等预处理步骤后进行路网匹配，通过路网匹配算法[2-3]，可以将欧氏空间的轨迹数据映射在路网结构上从而得到路网轨迹。然后对路网轨迹的空间信息和时间信息分别进行学习，生成预测模型。利用该预测模型，将轨迹数据时间信息和空间信息分别进行压缩，使得压缩后的轨迹数据在空间维度上无损，且时间维度上误差有界。与 PRESS 相似，将空间和时间这两种特征维度完全不同的信息分别考虑，从而取得更好的压缩效果。下面给出本文相关概念的形式化定义。

定义 1 路网。路网结构被定义为一个有向图 $G(V,E)$，其中：V 是节点集合，E 是边集合。每条边 e 上的权重，记做 $w(e)$，可以表示该路段的路径长度，通行时间或速度限制。

定义 2 轨迹。一条轨迹表示一个移动对象在对应时间维度下通过的路径，同时包含时间和空间两方面信息。在传统的轨迹表示方法中，一条轨迹被表示为 GPS 采样点 $p_i=(x_i,y_i,t_i)$ 的序列的形式，即 $t=<p_1,p_2,...,p_m>$，其中 (x_i,y_i) 表示物体在时间 t_i 时的欧氏空间位置坐标。

定义 3 路网轨迹。假设所有的移动对象均受到路网 G 的约束，那么通过路网匹配算法，轨迹数据可以通过路网结构进行表示，即路网轨迹 $t=<(e_1,t_1),(e_2,t_2)...(e_r,t_r)>$，其中：$e_i \in E$ 表示轨迹在路网 G 中的某个路段，t_i 表示轨迹进入该路段 e_i 时的时间戳。

与 PRESS 算法类似，将路网轨迹 t 分别表示为空间方面的路段序列 $t^s=<e_1,e_2,...,e_n>$ 和时间序列 $t^t=<t_1,t_2,...,t_n>$ 通过两种完全不同的模型分别对轨迹数据进行压缩，以达到更高的压缩效率。在空间方面，不同于传统的基于最短路径的压缩方法，CTPM 算法的压缩过程分为训练和压缩两个阶段。在训练阶段通过轨迹历史数据构建部分匹配预测(Prediction by Partial Matching，PPM)模型，通过轨迹已经行驶的部分路段对其下一时刻可能行驶的路段进行预测。在压缩阶段，利用生成的预测模型对轨迹下个可能的路段进行预测，若预测成功则将该路段删除，若预测失败则保留该路段信息，以减少轨迹数据的存储代价。在解压过程中，通过预测模型，将压缩后的轨迹完全还原，使其在空间上无损压缩。在时间方面，通过历史轨迹数据计算路网中不同时间区间下道路的通行速度，结合轨迹当前阶段的速度信息和道路长度信息，来计算轨迹时间信息，以此得到预测的轨迹时间序列，最后通过对比原始轨迹与预测轨迹之间时间的相似度来保留关键的时间变化路段，使得预测轨迹的满足一定误差阈值。下面，分别从空间和时间两个方面来详细地介绍本文方法。

2 空间轨迹压缩

将轨迹的空间信息表示为路段的序列，即 $t^s=<e_1,e_2,...,e_n>$。空间轨迹无损压缩的目的是通过预建的字典 T 将 t^s 压缩为一个更短的序列。由于轨迹的长度、起点和终点具有随机性，传统的 0 阶区间编码算法如霍夫曼编码(Huffman Coding)、LZ 系列算法等需要构建较大的字典树导致压缩比降低。然而轨迹数据由于受到交通状况的约束，其可能的行驶的路段往往依赖于之前行驶的路段。例如若车辆驶入高架路段，其行驶轨迹往往无法改变直到驶出该路段。因此利用具备 k 阶马尔科夫模型的 PPM 算法，通过轨迹已经行驶的部分路段来预测轨迹下一时刻可能行驶的位置。整个空间压缩过程可以分为两个阶段：一个是通过历史数据生成 PPM 预测模型；二是通过 PPM 模型对轨迹进行压缩。下面对 PPM 预测模型进行描述。

2.1 部分匹配预测模型(PPM)

Senft 等[11]提出了一种基于前缀字典树的数据压缩(Suffix Tree based Data Compression, STDC) 算法，该算法是 PPM 模型的一种实现，其主要思想是在输入序列中利用序列元素的顺序和频度来预测一下元素可能出现的概率。PPM 需要指定待构造的变阶马尔科夫模型的阶数 k，通过训练数据集构造一棵字典树 T。若模型阶数为 k，则 PPM 构造的字典树的最大深度为 $k+1$。字典树 T 的每一个节点由一个两元组*(c,num)*构成，其中：c 为训练序列中出现的元素，*num* 为元素 c 出现的频度。字典树中的每一条从根节点到叶子节点的路径表示序列中的一个子序列，字典树的根节点不表示任何信息，表示一个空序列。在构造字典树的过程中，算法将训练序列分割成长度为 k 的子序列集合，依据子序列构造字典树结构，每一条路径为一个子序列。每个子序列会使得该路径下所有元素频度 *num+1*。在构造完字典树之后，通过式(1)来计算序列 s 之后出现元素 c 的概率：

$$p(c|s)=num(sc)/(|Q_s|+\sum^{c'\in Q} num(sc')) \quad (1)$$

其中：序列 s 的长度为 *k-1*；*num(sc)*为长度为 k 的序列 *sc* 出现的频度；Q_s 是以 s 为前缀的所有序列集合。

通过一个例子来说明 PPM 算法。假设输入序列为 ACBAEADACBA，假设马尔科夫模型阶数 *k=3*，则该序列会被划分为子序列集合 ACB、CBA、AEA、EAD、ADA、ACB、CBA，其构建的字典树如图 1 所示。利用该字典树，可以通过公式(1)来计算序列 AC 后出现 B 的概率为 $p(B|AC)=2/(1+2)=0.667$。

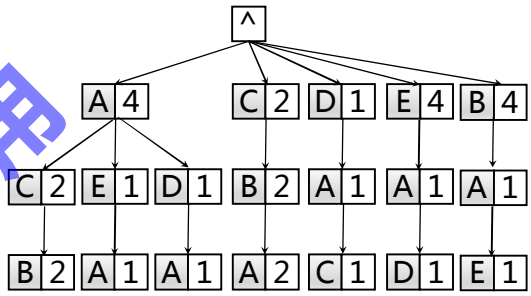


图 1 PPM 算法举例

Fig. 1 Example of PPM Algorithm

2.2 基于预测的空间轨迹压缩算

通过大量的历史轨迹数据对 PPM 模型进行训练，利用训练好的 PPM 模型可以通过轨迹之前行驶的路段来预测轨迹下一个位置。PPM 模型可以用式(2)表示：

$$y(t,D)=arg\,max\,P(st/t') \quad (2)$$

其中：t 为给定的待压缩的轨迹；D 为历史轨迹数据集；t' 是轨迹 t 中长度为 k 的子序列；*st* 为 t' 后的下一个路段。用字典树 T 表示 PPM 模型 y，树中的每个节点 *node* 包含路段信息 *rid*、孩子节点列表 *children*、频度信息 *num* 和预测下一个可能的路段信息 *pred*。

PPM 训练算法实现的伪代码如算法 1 所示。算法将路网 G，轨迹数据集 D 和 PPM 模型阶数 k 作为输入，将 PPM 模型的字典树 T 作为输出。对于每一条属于 D 的轨迹 t，将第 i 段轨迹 st_i 之前的 k 个轨迹段作为滑动窗口，同时增加这个滑动窗口构成的子序列中所有的元素的频度 *num*。对于一个节点 *nd* 来说，将频度最大的子节点作为其预测节点，直到所有轨迹段都被处理完。算法 1 空间复杂度为 $\Theta(\deg^k_{\max})$，其中：

$\deg_{max}$ 为路网 G 中任意一条边 e 的最大度数。算法 1 的时间复杂度为 $\Theta(|D|g|t|gk)$，其中：$|D|$为数据集 D 的大小；$|t|$为轨迹$|t|$的长度。

算法 1 PPM 训练算法

输入：路网 *G*，历史轨迹数据集 *D*，模型阶数 *k*

输出：表示 PPM 模型的字典树 *T*

1) 初始化字典树 *T*
2) **Foreach** $t \in D$ **Do**
3) **Foreach** $i \leftarrow 0$ **to** $|t|$ **Do**
4) /* 将轨迹段 st_i 之前的 *k* 个路段作为滑动窗口*/
5) $start \leftarrow i$
6) $end \leftarrow \min(i+k,|t|-1)$
7) $nd \leftarrow T.root$
8) **Foreach** $j \leftarrow [start,end]$ **Do**
9) /* 滑动窗口路段构成的路径的频度+1 */
10) $nd \leftarrow nd.children[st_j]$;
11) $nd.children[st_j+1]++$;
12) /*选出频度最大的节点作为预测节点*/
13) $nd.pred \leftarrow max(nd.children[].num)$
14) Return *T*

基于 PPM 模型的压缩算法实现的伪代码如算法 2 所示。算法将训练好的 PPM 模型的字典树 *T*、轨迹 t 和阶数 *k* 作为输入，输出一条压缩轨迹 t'。首先，算法将 t 中的前 *k-1* 条路段加入压缩轨迹 t' 中。从 t 中第 *k* 条路段开始，将其前 *k-1* 条路段作为滑动窗口，找到该滑动窗口对于序列路径的最后一个节点，并判断该节点的预测结果 *nd.pred* 与第 *k* 条轨迹段 st_i 是否一致，若预测错误，则将 st_i 加入压缩轨迹 t' 中，直到轨迹中所有的路段被处理完成。通过算法 2 可以对轨迹在空间上进行无损压缩，即对于任意一条压缩后的轨迹都可以通过算法 1 生成的字典树 *T* 将其还原。同时算法 2 具有较好的性能，其时间复杂度为 $\Theta(|t|gk)$。

算法 2 基于 PPM 模型的压缩算法

输入：PPM 模型 *T*，待压缩的轨迹 t ，模型阶数 *k;*

输出：压缩后的轨迹 t'。

1) 初将轨迹 t 中前 *k-1* 条路段加入 t'
2) **Foreach** $i \leftarrow k$ **To** $|t|$ **Do**
3) /* 将路段 st_i 之前的 *k-1* 个路段作为滑动窗口*/
4) $start \leftarrow max(0,i-k),end \leftarrow i-1$;
5) $nd \leftarrow T.root.children[st_{start}]$;
6) **While** $|end-start|>1$ **And** $nd.children[st_{start+1}]$ **Do**
7) $start \leftarrow start+1$
8) $nd \leftarrow T.root.children[st_{start}]$
9) /* 当预测失败时，将该路段加入压缩轨迹*/
10) **If** $nd.pred!=st_i$ **Then**;
11) $t' \leftarrow t' \cup st_i$;
12) Return t'

3 时间轨迹压缩

轨迹的时间信息表示为路段与时间的二元组序列，即 $t=<(e_1,t_1),(e_2,t_2),...,(e_n,t_n)>$。对于一个路网 *G* 来说，往往可以很容易获得路网上任意一条路段 *e* 的路径长度，用 *d(e)*来表示。因此可以通过轨迹的通行速度来计算轨迹的时间信息，即 $t_{i+1}=t_i+d(e_{i+1})/v_{i+1}$，其中：$v_{i+1}$ 表示移动物体在路段 e_{i+1} 时的通行速度。因此，时间轨迹压缩算法通过对历史轨迹数据进行统计，预测每条轨迹在各自路段的通行速度，从而得到预测的轨迹时间序列。通过对比预测轨迹与原始轨迹，将预测误差较大的时间点保留，将误差较小的时间点删除，以达到轨迹压缩的目的。轨迹的时间维度压缩会造成时间信息的损失，因此需要先对轨迹的时间距离进行度量。由于轨迹的空间信息压缩是无损的，只需要对比轨迹在各自路段上时间信息距离，其形式化定义如下。

定义 4 路网同步时间距离。给定原始轨迹 $t=<(e_1,t_1),(e_2,t_2),...,(e_n,t_n)>$ 和预测后的轨迹序列 $t'=<(e_1,t_1')>,(e_2,t_2')>,...,(e_n,t_n')>$，其路网同步时间距离 $NSTD(t,t')=\sum_{i=1}^{n}|t_i-t_i'|$，其中：$t_i$ 和 t_i' 分别为轨迹 t 、t' 进入路段 e_i 的时间戳；n 为轨迹 t 和 t' 的长度，即 $n=|t|=|t'|$。

对于轨迹时间信息的压缩，预测轨迹的通行速度是至关重要的，即给定一条轨迹段 (e_i,t_i)，需要预测移动对象在下一个路段 e_{i+1} 时的通行速度。从两个方面考虑轨迹的速度信息：当前轨迹速度和历史速度。轨迹速度是假设移动对象作匀速运动，则可以用整条轨迹的平均速度表示轨迹在每个路段上的速度，其值为 $v_t=\sum_{i=0}^{n}d(e_i)/(t_n-t_0)$。然而由于交通条件的限制，轨迹在不同的路段上通行所用的时间是不同的，例如轨迹从狭窄的双车道进入四车道高架桥时速度往往会有很大提升。由于轨迹数据具有明显的周期性，比如上班高峰期道路的通行速度较低，而夜晚道路通行速度较高，因此可以用历史轨迹数据统计路网中不同时段不同路段的通行速度，即历史速度。统计路网中每一条路段在一定的时间间隔(如 30 分钟)内的平均通行速度，如图 2 所示，上午 8 : 00

到 8 : 30 路段 e_1 的平均时速为 25km/h。可以通过轨迹前一段时间 t_i 和下一路段 e_{i+1} 来得到轨迹在 t_{i+1} 时刻路段 e_{i+1} 上的历史速度 $v_g(e_{i+1})$。

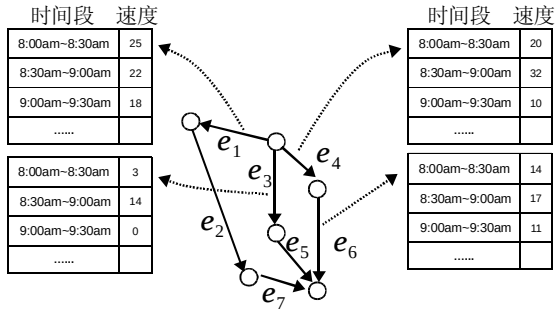


图 2 时间相关的速度路网举例

Fig. 2 Example of the time-dependent speed road netword

算法 3 描述了算法实现的伪代码。算法的输入为原始轨迹 t 、时间距离值 q 、起始路段 s 和结束路段 e，算法的输出为压缩轨迹 t'。算法首先获得起点为 s 终点为 e 的子轨迹段 $t_{s,e}$，通过预测速度得到该预测轨迹 $t_{s,e}'$。通过挨个寻找预测误差最大的路段，删除该路段后似的原始轨迹 $t_{s,e}$ 和预测轨迹 $t_{s,e}'$ 的路网同步时间距离(Network Synchronized Time Difference, NSTD)最小。若删除后的 *NSTD* 仍然大于误差阈值 q，则将路段 *min* 作为分割点，将轨迹切割成两条子轨迹重新进行计算，否则说明子轨迹 $t_{s,e}$ 可以用预测轨迹 $t_{s,e}'$ 误差有界表示。算法的执行过程与 Douglas-Peucker 算法[12]相似，其时间复杂度为 $\Theta(|D|g|t|^2)$，其中：$|t|$ 为轨迹平均路段数目，$|D|$ 为轨迹数据集 D 中的轨迹数量。

算法 3 时间信息压缩算法(Time-compression)

输入：原始轨迹 t，时间距离阈值 q，起始路段 s，结束路段 e

输出：压缩轨迹 t'

1) $t_{s,e} = segmentation(t, s, e)$
2) 根据速度得到预测轨迹 $t_{s,e}'$
3) $\min \leftarrow start$
4) /*选择删除预测轨迹中预测误差最大的路段*/
5) **Foreach** $i \leftarrow start$ **To** end **Do**
6) $t_i' \leftarrow t_i$
7) **If** $NSTD(t_i, t_i') < NSTD(t_{min}, t_{min}')$ **Then**
8) $\min \leftarrow i$
9) $dist \leftarrow NSTD(t_{min}, t_{min}')$
10) **If** $dist > q$ **then**
11) Time-compression(t, q, s, max)
12) Time-compression(t, q, max, s)
13) **Else**
14) 删除 t 中 $s+1$ 到 $e-1$ 的轨迹段
15) 将保留的轨迹段加入 t'
16) **Return** t'

4 实验结果与分析

通过一个真实轨迹数据集来验证压缩算法的有效性。实验所用的数据集为 2016 年 4 月 1 日上海市的出租车轨迹数据集，该数据集包含 45902 条轨迹数据，其平均长度为 5692m，平均采样率为 15s，数据集可视化情况如图 3 所示。实验使用 Java 编程语言实现所有的算法。所有实验的硬件平台均为 Intel Core(TM) i5@3.2GH CPU 和 8GB RAM 的 PC 机，操作系统为 64 位 Window7 系统，JDK 版本为 64 位 1.8.0 20。将轨迹的压缩比定义为原始轨迹 t 和压缩轨迹 t' 所需存储空间的比值，即 $c = |t|/|t'|$，作为算法压缩性能的评价指标。

图 3 轨迹数据集可视化

Fig. 3 Visualization of trajectory dataset

首先分析 CTPM 的空间轨迹压缩算法的有效性。CTPM 在空间方面需要指定待构造的变阶马尔科夫模型的阶数 k，图 4 展示了该参数对压缩效果的影响。如图 4(a)所示，随着 k 值的增加，CTPM 算法的压缩比先增加后降低；当 $k=2$ 时，算法压缩比最高，其值为 12.25。图 4(b)展示了阶数 k 对压缩时间的影响，随着 k 值的增加，算法的匹配次数随之增加，所需要的压缩时间呈线性趋势增加。

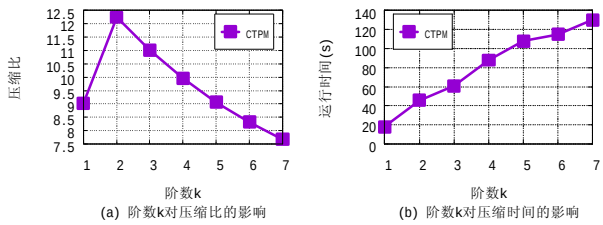
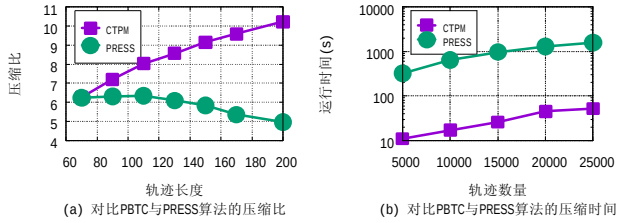


图 4 模型阶数 k 对压缩性能的影响

Fig. 4 The influence of k on compressive performance

下面对比 PRESS 算法与 CTPM 算法在空间维度下压缩性能。PRESS 算法与 CTPM 算法在空间方面均为无损压缩，两者的区别是 PRESS 算法通过使用最短路径算法来预测轨迹下一个位置，而 CTPM 算法通过对历史数据学习得到字典树来预测轨迹的下一个位置。将轨迹长度定义为一条轨迹包含轨迹段的数量，即$|t|$。对于不同的轨迹长度，两种算法的压缩性能对比如图 5(a)所示。由图 5(a)可以看出，随着轨迹长度的增加，CTPM 算法的压缩比也随之增加，而 PRESS 算法在轨迹长度为 70 时压缩性能与 CTPM 算法相似，二者压缩比均为 6.2；当轨迹长度在 70 到 110 之间时，PRESS 算法的压缩比随着轨迹长度的增加而增加；当轨迹长度超过 110 时，PRESS 算法的压缩比明显降低。不同的轨迹数量下，两种算法的压缩时间对比如图 5(b)所示。由图 5(b)可以看出，两者算法所需的压缩时间均随着轨迹数量的增加而增加，但由于 PRESS 需要计算任意两个路段的最短路径，因此其需要的压缩时间明显高于 CTPM 算法。

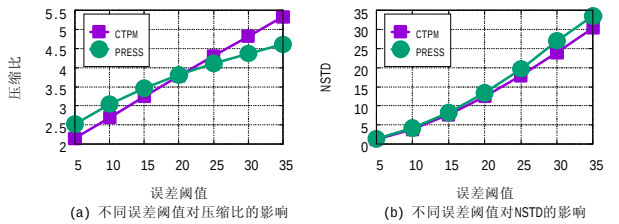


(a) 对比PBTC与PRESS算法的压缩比 (b) 对比PBTC与PRESS算法的压缩时间

图 5 CTPM 与 PRESS 算法空间压缩性能对比

Fig. 5 Comparison of spatial compression performance between CTPM and PRESS

PRESS 算法与 CTPM 算法在时间维度下均为有损压缩，本文使用路网同步时间距离 NSTD 来度量原始轨迹与压缩轨迹之间的误差。不同的误差阈值下，两种算法的压缩性能对比如图 6 所示。由图 6(a)可以看出，随着误差阈值的增加，两种算法的压缩比都随之增加；当误差阈值小于 20 时，PRESS 算法的压缩比略高于 CTPM 算法；当误差阈值大于于 20 时，CTPM 的压缩性能将优于 PRESS 算法。不同的误差阈值下，两种算平均路网同步时间距离 NSTD 的对比如图 6(b)所示。由图 6(b)可以看出，随着误差阈值的增加，两种算法的 NSTD 都随之增加。整体来看 CTPM 算法的误差略小于 PRESS 算法。



(a) 不同误差阈值对压缩比的影响 (b) 不同误差阈值对NSTD的影响

图 6 CTPM 与 PRESS 算法时间压缩性能对比

Fig. 6 Comparison of temporal compression performance between CTPM and PRESS

上述实验表明，在空间方面，CTPM 算法通过采用 PPM 模型代替计算耗时、准确率较低的最短路径算法，将压缩的重点由静态的路网结构转换到动态的历史轨迹数据，在降低压缩时间的同时有效的提高了压缩比。在轨迹平均长度为 70 到 200 的数据集中，CTPM 算法与 PRESS 算法相比压缩比提高了 43%。在时间方面，CTPM 算法通过采用速度预测模型有效的较低了平均压缩误差，在误差阈值 5 到 35 的参数设定下，CTPM 算法与 PRESS 算法相比压缩比提高了 1.5%，平均 NSTD 误差减小了 9.5%。

5 结语

本文针对目前路网环境下轨迹数据压缩的问题，提出了一种基于预测模型的轨迹数据压缩方法(CTPM)。在空间维度预测轨迹下一个可能的位置，，在时间维度预测轨迹进入下一路段的时间，通过删除预测正确的信息以达到数据压缩的目的。本文通过真实的出租车轨迹数据集来验证算法的有效性，实验表明在空间方面，CTPM 算法相对于传统的 PRESS 算法在降低压缩时间的同时大幅提高了压缩比。在时间方面，CTPM 算法可以有效地减少有损压缩带来的误差。然而，周期性的速度模型无法很好地描述某些突发性的交通状况，如车祸、道路施工等导致的堵车现象。在后续的工作中，可以通过结合实时路况信息来提升轨迹到达时间预测的准确性，在时间维度进一步优化压缩性能。

参考文献

[1] 高强，张凤荔，王瑞锦,等. 轨迹大数据:数据处理关键技术研究综述[J]. 软件学报, 2017, 28(4). (GAO Q, ZHANG F L, WANG R J, et al. Trajectory big data: a review of key technologies in data processing [J]. Journal of Software, 2017, 28(4))

[2] HASHEMI M, KARIMI H A. A critical review of real-time map-matching algorithms: current issues and future directions[J]. Computers Environment & Urban Systems, 2014, 48(8):153-165.

[3] BRAKATSOULAS S, PFOSER D, SALAS R, et al. On map-matching vehicle tracking data[C]// Proceedings of the 2005 International Conference on Very Large Data Bases. VLDB Endowment, 2005:853-864.

[4] CAO H, WOLFSON O. Nonmaterialized motion information in transport networks [J]. Lecture Notes in Computer Science, 2005, 3363:173-188..

[5] LERIN P M, YAMAMOTO D, TAKAHASHI N. Encoding travel traces by using road networks and routing algorithms [M]// Intelligent Interactive Multimedia: Systems and Services. Berlin: Springer Heidelberg, 2012:233-243.

[6] 张广林，胡小梅，柴剑飞,等. 路径规划算法及其应用综述[J]. 现代机械, 2011(5):85-90.(ZHANG G L,HU X M,CHAI J F, et al. Summary of path planning algorithm and its application [J]. Modern Machinery, 2011(5):85-90.)

[7] 张仁平，周庆忠，熊伟,等. A*算法改进算法及其应用[J]. 计算机系统应用, 2009, 18(9):98-100.(ZHANG R P, ZHOU Q Z, XIONG W. Updated A* algorithm and its application[J], Computer Systems & Applications, 2009, 18(9):98-100)

[8] KELLARIS G, PELEKIS N, THEODORIDIS Y. Trajectory compression under network constraints [C]// Proceedings of the 2009

International Symposium on Spatial and Temporal Databases. Berlin: Springer, 2009:392-398.
[9] KELLARIS G, PELEKIS N, THEODORIDIS Y. Map-matched trajectory compression [J]. Journal of Systems & Software, 2013, 86(6):1566-1579.
[10] 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.
[11] SENFT M. Suffix tree based data compression [J]. Lecture Notes in Computer Science, 2005, 3381:350-359. .
[12] TIENAAH T, STEFANAKIS E, COLEMAN D. Contextual douglas-peucker simplification [J]. Geomatica, 2015, 69(3):327-338.
[13] MUCKELL J, HWANG J H, PATIL V, et al. SQUISH: an online approach for GPS trajectory compression [C]// Proceedings of the 2011 International Conference on Computing for Geospatial Research & Applications. New York: ACM, 2011:1-8.
[14] KELLARIS G, PELEKIS N, THEODORIDIS Y. Trajectory compression under network constraints [C]// Proceedings of the 2009 International Symposium on Spatial and Temporal Databases. Berlin: Springer, 2009:392-398.

CHEN Yu, born in 1972, M.S. His research interests include big data, data mining and artificial intelligence.
JIANG Wei, born in 1990, M.S. His research interests include big data, data management and data analysis.
ZHOU Ji-En, born in 1976, Ph.D. His research interests include big data and artificial intelligence.