(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 111523577 A
(43)申请公布日 2020.08.11

(21)申请号 202010285620.X

(22)申请日 2020.04.13

(71)申请人 南京烽火星空通信发展有限公司
地址 210019 江苏省南京市建邺区云龙山路88号烽火科技大厦A栋26F

(72)发明人 刘宇 耿鑫 李国栋 李维

(74)专利代理机构 南京经纬专利商标代理有限公司 32200
代理人 楼高潮

(51)Int.Cl.
G06K 9/62(2006.01)
G06F 16/26(2019.01)

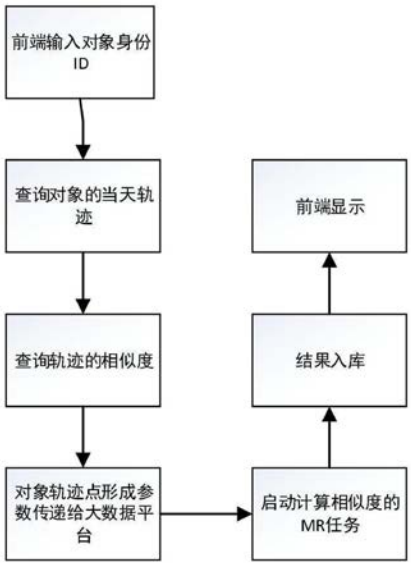
权利要求书2页 说明书8页 附图1页

(54)发明名称

一种基于改进的LCSS算法的海量轨迹相似度计算方法

(57)摘要

本发明公开了一种基于改进的LCSS算法的海量轨迹相似度计算方法,包括:基于采集设备采集移动对象实时位置信息,得到轨迹数据;对得到的轨迹数据进行校验;将经过校验的轨迹数据接入HBASE数据库;通过web前端输入要查询的对象轨迹的相似轨迹及其具体日期,根据对象ID查询出对象当天轨迹;将对象当天轨迹T_0以参数形式下发给大数据平台,大数据平台调度任务后启动MR任务计算相似度。本发明改进现有LCSS算法在计算轨迹点相似度的时候出现对时间阈值选取敏感的问题;解决了大数据集情况下,轨迹相似度计算的实时性问题;实现了挖掘并展示对象的关联轨迹的效果。



CN 111523577 A

1.一种基于改进的LCSS算法的海量轨迹相似度计算方法,其特征在于,所述方法包括如下步骤:

步骤一、通过采集设备采集对象实时位置信息,得到轨迹数据;

步骤二、对得到的轨迹数据进行校验,判断获取的位置信息中必要字段是否有值且格式正确,如果无值或非格式正确那么舍弃该条数据;

步骤三、将经过校验的轨迹数据接入HBASE数据库;

步骤四、通过web前端输入要查询的对象轨迹的相似轨迹及其具体日期,根据对象ID查询出对象当天轨迹;

步骤五、将对象当天轨迹T_0以参数形式下发给大数据平台,大数据平台调度任务后启动MR任务计算相似度。

2.如权利要求1所述的一种基于改进的LCSS算法的海量轨迹相似度计算方法,其特征在于,步骤五中所述启动MR任务计算相似度的具体步骤包括:

501、将对象轨迹数据中的各个轨迹点规约到对象身上,形成完整的轨迹,在展示端以轨迹身份证号为key,对象经度、纬度、时间为value进行输出;

502、得到对象的轨迹后,对轨迹点在时间上进行分段;

若时间段内有多个轨迹点,则针对时间、经度、纬度分别求中心点,直至使得在时间段内只有一个轨迹点,形成处理后的轨迹T_1;

对轨迹T_1和轨迹T_0在时间上进行求交运算,各自取公共的出现时间的轨迹点;

运用LCSS+算法对轨迹相似度进行计算,以对象ID为key,轨迹相似度为value进行输出;

503、求得与指定对象轨迹相似度大于设定阈值的k个对象,求解完成后输出结果入存储库。

3.如权利要求2所述的一种基于改进的LCSS算法的海量轨迹相似度计算方法,其特征在于,步骤502所述LCSS+算法中,时空轨迹运算的所需要的基本定义具体包括:

定义1:时空轨迹中的轨迹点由可三元属性构成,分别是时间,经度,纬度,时空轨迹中的第i个时空轨迹点记为:$P_i=(t_i,log_i,lat_i)$,t_i代表轨迹中第i个轨迹点所处时间,log_i代表第i个轨迹点的经度,lat_i代表第i个轨迹点的纬度;

定义2:时空轨迹是由轨迹中的所有轨迹点构成,一条轨迹i记为:$T_i=\{P_{i1},P_{i2},\cdots,P_{in}|in=len(T_i)\}$;

定义3:轨迹T中的时间序列记为ts,$ts=\{P_1(t),P_2(t),\cdots,P_n(t)|\forall i>j,P_i,P_j\epsilon T,P_j(t)<P_i(t)\}$;

定义4:时空轨迹T的最小时间记为$MinT=\{P_i(t)|\forall P_j\in T,i!=j,P_j(t)>P_i(t)\}$,时空轨迹T的最大时间记为$MaxT=\{P_i(t)|\forall P_j\in T,i!=j,P_j(t)<P_i(t)\}$;

定义5:时空轨迹T_a的时间序列ts_a和时空轨迹T_b的时间序列ts_b之间的交集定义为:$ts_a\cap ts_b=\{P(t_i)|MaxT_a\geq t\geq MinT_a\cap MaxT_b\geq t\geq MinT_b\}$;

定义6:时空轨迹T_a和时空轨迹T_b之间的交集定义为:$T_a\cap T_b=\{P_i|P_i\in T_a||P_i\in T_b,P_i(t)\in(ts_a\cap ts_b)\}$;

定义7:若时空轨迹T_a的最小时间$MinT_a$和最大时间$MaxT_a$与时空轨迹T_b的最小时间$MinT_b$和最大时间$MaxT_b$满足$MinT_b \leqslant MibT_a \leqslant MaxT_a \leqslant MaxT_b$,则有$T_a \in_t T_b$。

4.如权利要求2所述的一种基于改进的LCSS算法的海量轨迹相似度计算方法,其特征在于,步骤502中,轨迹点在位置上包括经度和纬度,在比对轨迹点时需要计算其相隔距离,两个轨迹点的距离为:

$$dis(lat1,log1,lat2,log2) = 2R * \arcsin\left(sqrt(sin^2\left(\frac{lat1-lat2}{2}\right) + \cos(lat1) * \cos(lat2) * sin^2\left(\frac{log1-log2}{2}\right)\right);$$

其中,lat1代表轨迹点1的经度,llog1代表轨迹点1的纬度,类似的lat2,log2代表轨迹点2的经度纬度;

若$dis(P_{ai}(lat),P_{ai}(log),P_{bj}(lat),P_{bj}(log)) < \varepsilon$,则认为这两个轨迹点在空间上是一对空间相似点。

5.如权利要求4所述的一种基于改进的LCSS算法的海量轨迹相似度计算方法,其特征在于,两个轨迹点的相似度计算规则为:

如果两个轨迹点在空间上都不相似,那么两个轨迹点就不是相似的;

如果两个轨迹点在空间上相似,两个轨迹点时间上越接近,那么这两个轨迹点越相似,否则就越不相似;

两个轨迹点P_{ai}、P_{bj}的相似值设为:

$$sim(P_{ai},P_{bj}) = \frac{1}{1+(\frac{P_{ai}(t)-,P_{bj}(t)}{t})^2};$$

当轨迹点P_{ai}和轨迹点P_{bj}在时间上差距小于t时,轨迹点之间的相似性为1;

当轨迹点在时间上差距大于t时,轨迹点相似度向0靠近。

6.如权利要求4所述的一种基于改进的LCSS算法的海量轨迹相似度计算方法,其特征在于:所述LCSS+算法在比较两条轨迹的时候,目标是使轨迹点相似性之和最大;

定义dp[i][j]为轨迹T_a从1到i-1编号组成子序列,轨迹T_b从1到j-1编号组成子序列,它们所包含的最大的相似性;

当轨迹T_a的轨迹点P_{ai}和轨迹T_b的轨迹点P_{bi}满足

$dis(P_{ai}(lat),P_{ai}(log),P_{bj}(lat),P_{bj}(log)) \leqslant \varepsilon$时,

$$dp[i][j] = \max\{dp[i-1][j-1] + \frac{1}{1+(\frac{P_{ai}(t)-,P_{bj}(t)}{t})^2}, dp[i][j-1], dp[i-1][j]\};$$

当$dis > \varepsilon$时,dp[i][j]=max{dp[i][j-1],dp[i-1][j]};

当i=0或者j=0时,dp[i][j]=0。

7.如权利要求1所述的一种基于改进的LCSS算法的海量轨迹相似度计算方法,其特征在于,所述步骤一中,所述对象实时位置信息包括经度、纬度、身份ID或时间;采集时对数据进行格式校验和数据过滤。

一种基于改进的LCSS算法的海量轨迹相似度计算方法

技术领域

[0001] 本发明公开了一种基于改进的LCSS算法的海量轨迹相似度计算方法，涉及互联网信息技术领域。

背景技术

[0002] 在现有的对象轨迹信息定位处理方法中，由于采集设备采集移动对象位置信息时，会出现采样不均匀问题，导致在计算对象之间的轨迹相似度的时候，要从全局最优的角度上来匹配轨迹之间的相似度。

[0003] 现有技术中传统的LCSS算法，在轨迹点比对时，会出现对时空轨迹点时间差阈值选取时敏感性的问题。同时，由于对象轨迹数据量较大，若在计算轨迹相似度的时候在单机上运行算法，运算速度会显得比较慢，计算所需要的时间较长，导致系统的实时性较差。

发明内容

[0004] 本发明所要解决的技术问题是：针对现有技术的缺陷，提供一种基于改进的LCSS算法的海量轨迹相似度计算方法，通过特定算法对移动对象轨迹信息进行相似度计算，基于该算法可以得到对象的潜在时空关系对象，进而完善对象的时空关系图谱。本发明中，改进的LCSS算法定义为LCSS+，该算法从使轨迹点相似性之和最大为目标，克服了对于时间阈值敏感的问题，在平均情况下优于LCSS算法，比LCSS算法更加精确。同时，本发明利用Hadoop集群充分发挥机器的运算能力，使用MapReduce作为其计算框架，将计算任务分而治之，从而缩短计算所需要的时间，并设计实现了分布式的LCSS+算法，进一步的提升系统的实时性。

[0005] 本发明为解决上述技术问题采用以下技术方案：一种基于改进的LCSS算法的海量轨迹相似度计算方法，包括如下步骤：

[0006] 步骤一、通过采集设备采集移动对象实时位置信息，得到轨迹数据；

[0007] 步骤二、对得到的轨迹数据进行校验，判断获取的位置信息中必要字段是否有值且格式正确，如果无值或非格式正确那么舍弃该条数据；

[0008] 步骤三、将经过校验的轨迹数据接入HBASE数据库；

[0009] 步骤四、通过web前端输入要查询的对象轨迹的相似轨迹及其具体日期，根据对象ID查询出对象当天轨迹；

[0010] 步骤五、将对象当天轨迹T_0以参数形式下发给大数据平台，大数据平台调度任务后启动MR任务计算相似度。

[0011] 进一步的，步骤五中所述启动MR任务计算相似度的具体步骤包括：

[0012] 501、将对象轨迹数据中的各个轨迹点规约到对象身上，形成完整的轨迹，在展示端以轨迹身份证号为key，对象经度、纬度、时间为value进行输出；

[0013] 502、得到对象的轨迹后，对轨迹点在时间上进行分段；

[0014] 若时间段内有多个轨迹点，则针对时间、经度、纬度分别求中心点，直至使得在时

间段内只有一个轨迹点，形成处理后的轨迹T_1；

[0015] 对轨迹T_1和轨迹T_0在时间上进行求交运算，各自取公共的出现时间的轨迹点；

[0016] 运用LCSS+算法对轨迹相似度进行计算，以对象ID为key，轨迹相似度为value进行输出；

[0017] 503、求得与指定对象轨迹相似度大于设定阈值的k个对象，求解完成后输出结果入存储库。

[0018] 进一步的，步骤502所述LCSS+算法中，时空轨迹运算的所需要的基本定义具体包括：

[0019] 定义1：时空轨迹中的轨迹点由可三元属性构成，分别是时间，经度，纬度，时空轨迹中的第i个时空轨迹点记为：$P_i=(t_i,log_i,lat_i)$，t_i代表轨迹中第i个轨迹点所处时间，log_i代表第i个轨迹点的经度，lat_i代表第i个轨迹点的纬度；

[0020] 定义2：时空轨迹是由轨迹中的所有轨迹点构成，一条轨迹i记为：$T_i=\{P_{i1},P_{i2},\cdots,P_{in}|in=len(T_i)\}$；

[0021] 定义3：轨迹T中的时间序列记为ts，$ts=\{P_1(t),P_2(t),\cdots,P_n(t)|\forall i>j,P_i,P_j\epsilon T,P_j(t)<P_i(t)\}$；

[0022] 定义4：时空轨迹T的最小时间记为$MinT=\{P_i(t)|\forall P_j\in T,i!=j,P_j(t)>P_i(t)\}$，时空轨迹T的最大时间记为$MaxT=\{P_i(t)|\forall P_j\in T,i!=j,P_j(t)<P_i(t)\}$；

[0023] 定义5：时空轨迹T_a的时间序列ts_a和时空轨迹T_b的时间序列ts_b之间的交集定义为：$ts_a\cap ts_b=\{P(t_i)|MaxT_a\geq t\geq MinT_a\cap MaxT_b\geq t\geq MinT_b\}$；

[0024] 定义6：时空轨迹T_a和时空轨迹T_b之间的交集定义为：$T_a\cap T_b=\{P_i|P_i\in T_a||P_i\in T_b,P_i(t)\in(ts_a\cap ts_b)\}$；

[0025] 定义7：若时空轨迹T_a的最小时间$MinT_a$和最大时间$MaxT_a$与时空轨迹T_b的最小时间$MinT_b$和最大时间$MaxT_b$满足$MinT_b\leqslant MinT_a\leqslant MaxT_a\leqslant MaxT_b$，则有$T_a\in_t T_b$。

[0026] 进一步的，轨迹点在位置上包括经度和纬度，在比对轨迹点时需要计算其相隔距离，两个轨迹点的距离为：

[0027]
$$dis(lat1,log1,lat2,log2)=2R*\arcsin\left(\mathrm{sqrt}\left(sin^2\left(\frac{lat1-lat2}{2}\right)+\cos(lat1)*\cos(lat2)*sin^2\left(\frac{log1-log2}{2}\right)\right)\right);$$

[0028] 其中，lat1代表轨迹点1的经度，llog1代表轨迹点1的纬度，类似的lat2，log2代表轨迹点2的经度纬度；

[0029] 若$dis(P_{ai}(lat),P_{ai}(log),P_{bj}(lat),P_{bj}(log))<\varepsilon$，则认为这两个轨迹点在空间上是一对空间相似点。

[0030] 进一步的，两个轨迹点的相似度计算规则为：

[0031] 如果两个轨迹点在空间上都不相似，那么两个轨迹点就不是相似的；

[0032] 如果两个轨迹点在空间上相似，两个轨迹点时间上越接近，那么这两个轨迹点越相似，否则就越不相似；

[0033] 两个轨迹点P_{ai}、P_{bj}的相似值设为：

[0034] $$\mathrm{sim}(P_{ai},P_{bj})=\frac{1}{1+(\frac{P_{ai}(t)-,P_{bj}(t)}{t})^2};$$

[0035] 当轨迹点P_{ai}和轨迹点P_{bj}在时间上差距小于t时，轨迹点之间的相似性为1；

[0036] 当轨迹点在时间上差距大于t时，轨迹点相似度向0靠近。

[0037] 进一步的，所述LCSS+算法在比较两条轨迹的时候，目标是使轨迹点相似性之和最大；

[0038] 定义dp[i][j]为轨迹T_a从1到i-1编号组成子序列，轨迹T_b从1到j-1编号组成子序列，它们所包含的最大的相似性；

[0039] 当轨迹T_a的轨迹点P_{ai}和轨迹T_b的轨迹点P_{bi}满足

[0040] dis(P_{ai}(lat)，P_{ai}(log)，T_{bj}(lat)，P_{bj}(log))≤ε时，

[0041] $$\mathrm{dp[i][j]}=\max\{\mathrm{dp[i-1][j-1]}+\frac{1}{1+(\frac{P_{ai}(t)-,P_{bj}(t)}{t})^2},\mathrm{dp[i][j-1]},\mathrm{dp[i-1][j]}\};$$

[0042] 当dis＞ε时，dp[i][j]＝max{dp[i][j-1]，dp[i-1][j]}；

[0043] 当i＝0或者j＝0时，dp[i][j]＝0。

[0044] 作为发明的进一步优选方案，步骤一中，所述对象实时位置信息包括经度、纬度、身份ID或时间；另外，采集时需要对数据进行格式校验和数据过滤。

[0045] 本发明采用以上技术方案与现有技术相比，具有以下技术效果：本发明改进现有LCSS算法在计算轨迹点相似度的时候出现对时间阈值选取敏感的问题；解决了大数据集情况下，轨迹相似度计算的实时性问题；实现了挖掘并展示对象的关联轨迹的效果。

附图说明

[0046] 图1是海量轨迹相似度计算流程图。

[0047] 图2是LCSS+算法状态转移矩阵求解过程示意图；

[0048] 图3是LCSS算法状态转移矩阵求解过程示意图。

具体实施方式

[0049] 下面详细描述本发明的实施方式，所述实施方式的示例在附图中示出，其中自始至终相同或类似的标号表示相同或类似的元件或具有相同或类似功能的元件。下面通过参考附图描述的实施方式是示例性的，仅用于解释本发明，而不能解释为对本发明的限制。

[0050] 下面结合附图对本发明的技术方案做进一步的详细说明：

[0051] 本发明提供了一种对时空轨迹相似度更为精确的计算方法LCSS+，并在海量的时空轨迹运算中，基于Hadoop平台设计了分布式的LCSS+算法，提升系统的实时性。

[0052] 海量轨迹相似度计算流程图如图1所示，所述方法包括：

[0053] 步骤一、通过采集设备对移动对象实时位置信息进行采集，采集内容主要有经度、纬度、身份ID、时间等，采集时对数据进行格式校验和数据过滤。

[0054] 步骤二、对于获取到的轨迹数据进行校验：判断获取的信息中必要字段是否有值且格式正确，如果无值或非格式正确那么舍弃该条数据。

[0055] 步骤三、轨迹数据接入HBASE数据库。

[0056] 步骤四、web前端输入要查询的对象轨迹的相似轨迹及其具体日期，可根据对象ID查询出对象当天轨迹。

[0057] 步骤五、将对象轨迹T_0以参数形式下发给大数据平台，大数据平台调度任务后启动MR任务计算相似度。

[0058] MR任务计算相似度的具体步骤包括：

[0059] a)由于对象轨迹点在数据库中存储时是以一个个离散的轨迹点。那么首先必须将各个轨迹点规约到人身上，形成完整的轨迹。则在Map端以轨迹身份证号为key，对象经度、纬度、时间为value输出。

[0060] b)在Reduce阶段，得到对象的轨迹后，对轨迹点在时间上进行分段，时间段可以通过配置文件设置，若时间段内有多个轨迹点，那么针对时间、经度、纬度分别求中心点，最后使轨迹点在时间段内只有一个轨迹点，形成处理后的轨迹T_1，随后对轨迹T_1和轨迹T_0在时间上进行求交运算，各自取公共的出现时间的轨迹点，随后运用LCSS+算法对轨迹相似度进行计算，最后以对象ID为key，轨迹相似度为value进行输出。

[0061] c)第二个MapReduce过程，主要是求得与指定对象轨迹相似度较大的k个对象，主要过程是求解topK的问题，求解完成后输出结果入存储库。

[0062] 步骤b)中，LCSS+算法由于时空轨迹的特殊性，以及后续算法的描述，现定义时空轨迹运算的所需要的基本定义，包括如下：

[0063] 定义1:时空轨迹中的轨迹点由可三元属性构成，分别是时间，经度，纬度，时空轨迹中的第i个时空轨迹点记为:$P_i=(t_i,log_i,lat_i)$，t_i代表轨迹中第i个轨迹点所处时间，log_i代表第i个轨迹点的经度，lat_i代表第i个轨迹点的纬度。

[0064] 定义2:时空轨迹是由轨迹中的所有轨迹点构成，一条轨迹i记为:$T_i=\{P_{i1},P_{i2},\cdots,P_{in}|in=len(T_i)\}$

[0065] 定义3:轨迹T中的时间序列记为ts，$ts=\{P_1(t),P_2(t),\cdots,P_n(t)|\forall i>j,P_i,P_j\epsilon T,P_j(t)<P_i(t)\}.$

[0066] 定义4:时空轨迹T的最小时间记为$MinT=\{P_i(t)|\forall P_j\in T,i!=j,P_j(t)>P_i(t)\}$,时空轨迹T的最大时间记为$MaxT=\{P_i(t)|\forall P_j\in T,i!=j,P_j(t)<P_i(t)\}$,

[0067] 定义5:时空轨迹T_a的时间序列ts_a和时空轨迹T_b的时间序列ts_b之间的交集定义为:$ts_a\cap ts_b=\{P(t_i)|MaxT_a\geq t\geq MinT_a\cap MaxT_b\geq t\geq MinT_b\}$

[0068] 定义6:时空轨迹T_a和时空轨迹T_b之间的交集定义为:$T_a\cap T_b=\{P_i|P_i\in T_a||P_i\in T_b,P_i(t)\in(ts_a\cap ts_b)\}$

[0069] 定义7:若时空轨迹T_a的最小时间$MinT_a$和最大时间$MaxT_a$与时空轨迹T_b的最小时间$MinT_b$和最大时间$MaxT_b$满足$MinT_b\leqslant MinT_a\leqslant MaxT_a\leqslant MaxT_b$，则有$T_a\in_t T_b$。

[0070] 由于轨迹点在位置上只包括经度和纬度，在比对轨迹点时需要计算其相隔距离，两个轨迹点的距离为：

[0071]
$$dis(lat1,log1,lat2,log2)$$
$$=2R*\arcsin\left(sqrt(sin^2\left(\frac{lat1-lat2}{2}\right)+\cos(lat1)*\cos(lat2)\right.$$
$$\left.*sin^2\left(\frac{log1-log2}{2}\right)\right);$$

[0072] 其中，lat1代表轨迹点1的经度，llog1代表轨迹点1的纬度，类似的lat2，log2代表轨迹点2的经度纬度。

[0073] 若dis(P_{ai}(lat)，P_{ai}(log)，P_{bj}(lat)，P_{bj}(log))＜ε，则认为这两个轨迹点在空间上是一对空间相似点。

[0074] 由于要在时空轨迹点还有时间属性，那么两个轨迹点的相似度计算规则，定义如下：如果两个轨迹点在空间上都不相似，那么两个轨迹点就不是相似的。如果两个轨迹点在空间上相似，两个轨迹点时间上越接近，那么这两个轨迹点越相似，否则就越不相似。

[0075] 则两个轨迹点P_{ai}、P_{bj}的相似值可以设为：

[0076] $$\mathrm{sim}(P_{ai},P_{bj})=\frac{1}{1+(\frac{P_{ai}(t)-,P_{bj}(t)}{t})^2};$$

[0077] 当轨迹点P_{ai}和轨迹点P_{bj}在时间上差距小于t时，轨迹点之间的相似性为1，而当轨迹点在时间上差距大于t时，轨迹点相似度会向0靠近，符合现实情况。

[0078] LCSS+算法在比较两条轨迹的时候，目标是使轨迹点相似性之和最大，LCSS+算法主要的思想是以轨迹点相似性和最大为目标来进行计算的，可以发现LCSS+也是一个动态规划的问题，定义dp[i][j]为轨迹T_a从1到i-1编号组成子序列，轨迹T_b从1到j-1编号组成子序列，它们所包含的最大的相似性和。

[0079] 当轨迹T_a的轨迹点P_{ai}和轨迹T_b的轨迹点P_{bi}满足

[0080] dis(P_{ai}(lat)，P_{ai}(log)，P_{bj}(lat)，P_{bj}(log))≤ε时，

[0081] $$\mathrm{dp[i][j]}=\max\{\mathrm{dp[i-1][j-1]}+\frac{1}{1+(\frac{P_{ai}(t)-,P_{bj}(t)}{t})^2},\mathrm{dp[i][j-1]},\mathrm{dp[i-1][j]}\},;$$

[0082] 当dis＞ε时，dp[i][j]＝max{dp[i][j-1]，dp[i-1][j]}；

[0083] 当i＝0或者j＝0时，dp[i][j]＝0。

[0084] 算法的伪代码如下：

[0085] LCSS+(T_a，T_b)

[0086] 1 lena←len(T_a)

[0087] 2 lenb←len(T_b)

[0088] 3dp[ ][ ]←new dp[lena][lenb]

[0089] 4 for i←0 to lena

[0090] 5dp[i][0]←0

[0091] 6 forj←0 to lenb

[0092] 7dp[0][j]←0

[0093] 8 for i←1 to lena

[0094] 9 for j←1 to lenb

[0095] 10P_j←T_b(j)//轨迹b的第j个节点

[0096] 11P_i←T_a(i)

[0097] 12 if dis≤ε

[0098] 13 $\mathrm{dp[i][j]}\leftarrow\max\{\mathrm{dp[i-1][j-1]}+\frac{1}{1+(\frac{P_{ai}(t)-,P_{bj}(t)}{t})^2}$，dp[i][j-1]，dp[i-1][j]}，

[0099] 14 elsedp[i][j]←max{dp[i][j-1],dp[i-1][j]}

[0100] 15 return dp[lena][lenb]/(math.min{lena,lenb})。

[0101] 伪代码4-7行时间复杂度为:θ(n),8行到14行时间复杂度为θ(n^2),算法的平均时间复杂度为θ(n^2)。

[0102] 为了区别提出的LCSS+算法与传统时空轨迹相似度度量LCSS计算方法,下面给出一个具体实施例来说明传统LCSS算法在比对轨迹点时会出现对时间差阈值选取敏感性问题。

[0103] 如图2所示,箭头对标的是两个空间轨迹在空间位置上类似的情况下,其累计相似性和的转移情况,例如在B(9:32)和B(9:50)在空间位置上都是属于B点,在时间上相差19分钟,那么它的累计最优相似性和就是:$dp[1][2]+\frac{1}{1+(\frac{P_{ai}(t)-P_{bj}}{t})^2}$,dp[2][2],dp[1][3]之间的最大值,可以算出最大值为1.22,又在B(9:40)和B(10:02)在空间置上属于B点,时间上相差为22分钟,那么从dp[2][3]转移过来的累计最大相似性和为:1/(1+2.2*2.2)+1.22=1.39。而从dp[3][3]转移过来的累计最大相似性和为1.45.显然更大,故dp[3][4]的取值应为1.45。相似度为2.45/5=0.49。虽然两个轨迹在行进过程中在B点可以取两个公共B,但是位于两个公共B的轨迹点时间差比较大,还不如一个公共B,即B((9:40)和B(9:51)点所带来的相似性累计。

[0104] 再来给对应的LCSS算法在求解过程中的状态转移矩阵,如图3所示LCSS算法在求解状态矩阵的时候,只有两个公共相似点,第一对相似点为A(9:25)和A(9:30),第二对相似点为D(10:13)和D(10:15)。最终求解出来的最大相似性和为2,相似度为0.4,相比于LCSS+算法其因选取时间间隔敏感的问题就凸显出来,当选取10分钟作为间隔点的时候,LCSS算法可能会损失一些可能相似的公共点,比如B(9:40)和B(9:51),就因为相差11分钟就舍去公共点,从而其在轨迹相似度比较中损失了一些精度,最终可能导致其不能出现在对象的关系人中。

[0105] 分布式LCSS+算法主要是体现在MR程序的设计上,现给出程序伪代码:

[0106] 第一个MapReduce

Map阶段:

map()

key ← （身份ID）

value ← (经度，纬度，时间)

emit(key,value)

Reduce阶段

setup()//Reduce阶段初始化函数

$T_a \leftarrow Conf.get()$ //从conf对象里面获取轨迹T_a

col ← Collection() /新建一个集合

reduce(records)//reduce函数

for record ← records //将reduce函数聚集的记录加入到集合中

col add record

sort(col)//将轨迹点以时间维度进行排序

$T_b \leftarrow col$ //形成轨迹T_b

[0107]

$T_a^{\circ} = T_a \cap T_b$ //轨迹T_a在时间上与轨迹T_b求交

$T_b^{\circ} = T_b \cap T_a$ //轨迹T_b在时间上与轨迹T_a求交

res ←lcss+(T_a° , T_b°)//用

key ← （身份ID）

value ← (res)

emit(key,value) //输出结果

第二个MapReduce

Map阶段:

set ← TreeSet()//新建一个TreeSet用于排序

map(record)

if size(set)<k

set add record

else

set add record

set remove first

```
cleanup() //Map任务清理函数，只会调用一次
    for record in set
        key ← record(0)//身份ID
        value ← record(1)//相似度值
        emit(key,value)
```

[0109] Reduce阶段的伪代码与Map阶段类似，故不再复述。经过上述两个依赖的MapReduce任务就可以输出最大的k个轨迹相似度值的对象。

[0110] 上面结合附图对本发明的实施方式作了详细说明，但是本发明并不限于上述实施方式，在本领域普通技术对象所具备的知识范围内，还可以在不脱离本发明宗旨的前提下做出各种变化。以上所述，仅是本发明的较佳实施例而已，并非对本发明作任何形式上的限制，虽然本发明已以较佳实施例揭露如上，然而并非用以限定本发明，任何熟悉本专业的技术对象，在不脱离本发明技术方案范围内，当可利用上述揭示的技术内容做出些许更动或修饰为等同变化的等效实施例，但凡是未脱离本发明技术方案内容，依据本发明的技术实质，在本发明的精神和原则之内，对以上实施例所作的任何简单的修改、等同替换与改进等，均仍属于本发明技术方案的保护范围之内。

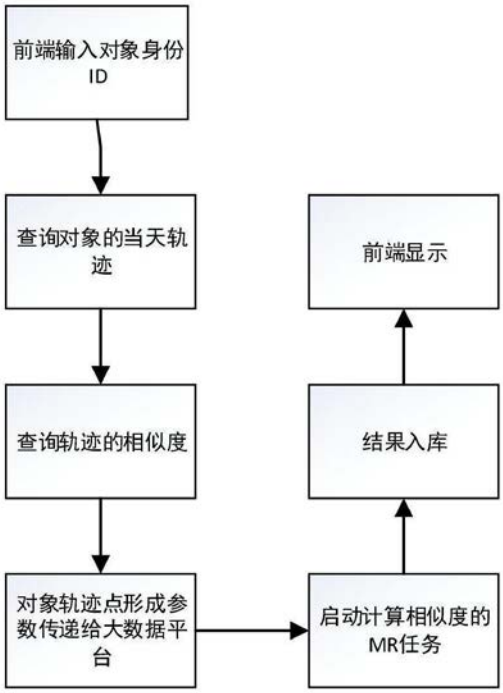


图1

	A(9:30)	F(9:40)	B(9:51)	B(10:02)	D(10:13)	B(10:24)
A(9:25)	1	1	1	1	1	1
B(9:32)	1	1	1.22	1.22	1.22	1.22
B(9:40)	1	1	1.45	1.45	1.45	1.45
C(10:03)	1	1	1.45	1.45	1.45	1.45
D(10:15)	1	1	1.45	1.45	2.45	2.45

图2

	A(9:30)	F(9:40)	B(9:51)	B(10:02)	D(10:13)	B(10:24)
A(9:25)	1	1	1	1	1	1
B(9:32)	1	1	1	1	1	1
B(9:40)	1	1	1	1	1	1
C(10:03)	1	1	1	1	1	1
D(10:15)	1	1	1	1	2	2

图3