高效前缀约简的三维Hilbert空间填充曲线编解码算法

贾连印[①②] 范 瑶[①] 丁家满*[①] 李晓武[①] 游进国[①]

[①](昆明理工大学信息与自动化学院 昆明 650500)

[②](云南省计算机应用重点实验室 昆明 650500)

摘 要：3维Hilbert空间填充曲线(3D HSFC)的编码和解码效率对空间查询处理、图像处理等领域的应用举足轻重。现有的3维编解码算法独立编解码每一个点，忽略了Hilbert曲线的局部保持特性。为了提高编解码效率，该文设计了高效的3D状态视图，并提出一种新的前缀约简的3D HSFC编码算法(PR-3HE)和前缀约简3D HSFC解码算法(PR-3HD)，这两个算法通过公共前缀的定义和识别、公共前缀约简及多种优化技术来最小化需要编码的阶数，从而提高3D HSFC的编解码效率。理论上证明：当编码或解码一个k阶的窗体(窗体内总共含有$2^k \times 2^k \times 2^k$个点)时，PR-3HE平均每个点的编码阶数不超过2，PR-3HD平均解码阶数不超过8/7。相对于传统的基于迭代的方法，编解码时间复杂度从$O(k)$降低到了$O(1)$。实验结果表明，该文算法在模拟数据集和真实数据集上的表现显著优于现有算法。

关键词：3维Hilbert空间填充曲线；3维状态视图；前缀约简；3D HSFC编码算法；3D HSFC解码算法

中图分类号：TN911.2; TP301.6 **文献标识码**：A **文章编号**：1009-5896(2024)02-0633-10

DOI: 10.11999/JEIT230013

3D Hilbert Space Filling Curve Encoding and Decoding Algorithms Based on Efficient Prefix Reduction

JIA Lianyin[①②] FAN Yao[①] DING Jiaman[①] LI Xiaowu[①] YOU Jinguo[①]

[①](*Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming* 650500, *China*)

[②](*Key Laboratory of Computer Technology Application of Yunnan Province, Kunming* 650500, *China*)

Abstract: The encoding and decoding efficiency of 3D Hilbert Space Filling Curve (3D HSFC) is key for the application of spatial query processing, image processing. The existing 3D encoding and decoding algorithms encode and decode each point independently, ignoring the local preservation characteristic of Hilbert curve. To improve the efficiency of encoding and decoding, an efficient 3D state view is designed in this paper, and a new Prefix Reduction 3D HSFC Encoding Algorithm (PR-3HE) and Prefix Reduction 3D HSFC Decoding Algorithm (PR-3HD) are proposed. These two algorithms minimize the orders to be processed through the definition and identification of common prefix, common prefix reduction and various optimization techniques, thus improving 3D HSFC encoding and decoding efficiency. Theoretical proof is provided in this paper, demonstrating that when encoding or decoding a k-order window (all $2^k \times 2^k \times 2^k$ points in a window), PR-3HE passively encodes no more than 2 orders for each coordinate on average, while PR-3HD passively decodes no more than 8/7 orders for each Hilbert code on average. The encoding and decoding time complexity can be reduced from $O(k)$ to $O(1)$. Experimental results show on both synthetic and real dataset the benefits of our algorithms over the other counterparts.

Key words: 3D Hilbert Space Filling Curve (3D HSFC); 3D state view; Prefix reduction; 3D HSFC encoding algorithm; 3D HSFC decoding algorithm

收稿日期：2023-01-12；改回日期：2023-06-01；网络出版：2023-06-20

*通信作者： 丁家满 jiamanding@kust.end.cn

基金项目：国家自然科学基金(62262035, 62262034, 62062046)

Foundation Items: The National Natural Science Foundation of China (62262035, 62262034, 62062046)

1 引言

Hilbert空间填充曲线(Hilbert Space Filling Curve, HSFC)是空间填充曲线家族一个典型代表，其可将高维空间点和1维空间点进行映射和逆映射，同时保持其在原始空间的局部性，相对于Z曲线等其他空间填充曲线，其具有跳变小和空间局部性好等优点。

HSFC广泛应用于空间查询处理[1,2]、图像处理[3]、点云压缩[4]等领域，编解码算法是Hilbert曲线应用的基础。与目前的研究工作多着重于2D HSFC的编解码算法不同[5-8]，本文着重于3D HSFC编解码。相对而言，3D HSFC编解码远较2D HSFC复杂[9]，该问题的研究对3D图像处理[10]、3D空间索引[11]、人工智能[12,13]等领域有着重要的意义。

早期3D HSFC编解码算法多为基于递归算法，其时间复杂度多为$O(k^2)$(k为HSFC的阶数)。目前的算法以效率更高的迭代算法为主，根据其采用的核心技术的不同，其又可分为基于位操作的算法[14]、基于演化规则的算法[15]、基于状态视图的算法[16,17]等。考虑到基于状态视图的算法简单、直观、高效、易于实现，因此其近年来受到广泛的关注。Zhang等人[16]基于状态视图提出一种适用于不规则边界的3D HSFC编解码算法，Jia等人[17]基于状态视图，提出一种适用于数据向原点偏斜分布的3D HSFC编解码算法。 然而，现有算法通常独立地对每个点进行编解码，而忽略了点之间的关系，从而导致效率低下。本文发现相邻点通常具有一定的公共前缀，充分利用这些公共前缀可提高3D HSFC编码效率。基于此，设计高效的3D状态视图，并提出一种新的前缀约简的3D HSFC编码算法(PR-3HE)和对应的解码算法(PR-3HD)，这两个算法通过公共前缀识别、公共前缀约简等技术提高了3D HSFC的编解码效率。在编码阶数为k的窗体时，相对传统算法需对每个数据点编解码k阶，PR-3HE对每个点平均仅需编码2阶，PR-3HD对每个点平均仅需解码8/7阶，复杂度降低到$O(1)$。实验结果表明，本文算法显著优于现有算法。

2 基础定义和必要的准备

2.1 3D 1阶HSFC的基本状态

Zhang等人[16]和Jia等人[17]采用包含12个基本状态的状态视图，由于基本状态类型的缺失，其在构造3阶及以上的HSFC时，易导致曲线出现跳变，从而降低Hilbert曲线的内聚性。为避免这一问题，本文根据如图1所示的24种3D 1阶基本状态来构建状态视图，基本状态的编号依次为0～23。

2.2 3D HSFC

3D HSFC以递归的方式将空间逐步分割为8个子空间，并通过特定顺序连接这些子空间生成不同的3维Hilbert空间填充曲线。图2展示了状态0的1阶Hilbert曲线中1阶编码与坐标的映射关系，括号外为Hilbert编码，括号内为空间坐标。图3展示了使用分层方式，从24个状态中选取合适的状态，构造出的2阶Hilbert曲线的示意图。

2.3 基础符号与定义

本文中，3维空间中任意一个坐标p表示为$p=(X_p,Y_p,Z_p)$，其中$X_p=(x_p^1,x_p^2,\cdots,x_p^k)_2$，$Y_p=(y_p^1,y_p^2,\cdots,y_p^k)_2$和$Z_p=(z_p^1,z_p^2,\cdots,z_p^k)_2$分别代表$p$的

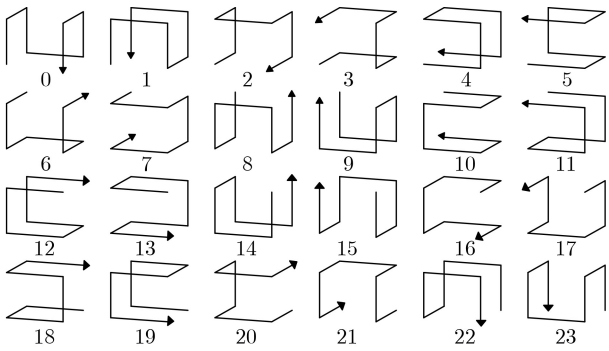

图 1 24种基础状态

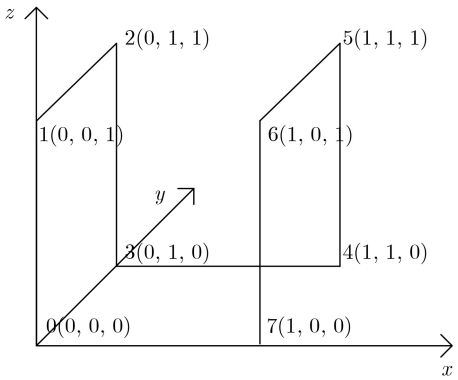

图 2 1阶Hilbert曲线

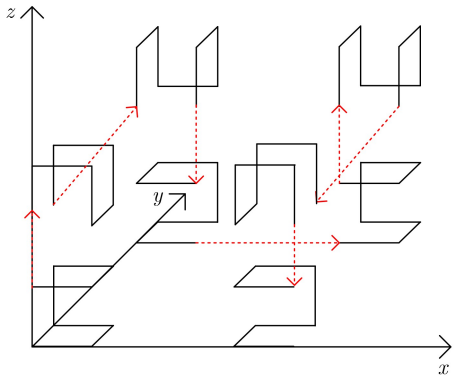

图 3 2阶Hilbert曲线

横、纵、竖3个坐标分量。每个坐标分量用k位2进制表示，这里k也称作Hilbert曲线的阶数。本文规定，阶次从左向右递增，x_p^i表示坐标分量X_p的第i阶，$p^i=(x_p^i,y_p^i,z_p^i)_2$表示坐标$p$的第$i$阶，$\overrightarrow{X_p^i}=(x_p^1,x_p^2,\cdots,x_p^i)_2$表示坐标分量$X_p$的前$i$阶前缀，$\overrightarrow{p^i}=(\overrightarrow{X_p^i},\overrightarrow{Y_p^i},\overrightarrow{Z_p^i})_2$表示坐标的前$i$阶前缀。给定坐标$p$，计算其对应Hilbert编码值$H_p=(h_p^1,h_p^2,\cdots,h_p^{3k})_2$的过程称之为编码。反之，给定编码值$H_p$，计算其对应的坐标$p$的过程称之为解码。$H_p^i=(h_p^{3i-2}h_p^{3i-1}h_p^{3i})_2$对应的3位二进制表示编码$H_p$的第$i$阶，$\overrightarrow{H_p^i}=(h_p^1,h_p^2,\cdots,h_p^{3i})_2$表示编码$H_p$的前$i$阶前缀。

3　编码算法

3.1　3D编码状态视图

在编码阶段，设计了用于存储1阶坐标到Hilbert编码的映射视图CHM和用于存储1阶坐标到下一阶状态的映射视图CSM。给定图1所示的24个基本状态，构建的CHM和CSM分别如表1和表2所示。这两个表的表头中的3位数值分别代表了1阶坐标分量x, y, z的二进制值，而表中的数值则分别表示对应的1阶Hilbert编码(表1)或下一阶状态(表2)。

本文将CHM和CSM设计为两个4维数组。给定第i阶坐标(x^i,y^i,z^i)和第i阶的状态，可分别根据$H^i=\mathrm{CHM}[s^i][x^i][y^i][z^i]$, $s^{i+1}=\mathrm{CSM}[s^i][x^i][y^i][z^i]$计算第$i$阶的Hilbert编码$H^i$和第$i+1$阶状态$s^{i+1}$。本文设定初始默认状态$s^1=0$，即将状态0作为第1阶Hilbert曲线的默认状态。

3.2　编码算法

3.2.1　PR-3HE算法

PR-3HE算法的核心包括公共前缀识别、公共前缀编码约简和约简优化3部分。为便于描述，引入包含p, q和r3个连续编码点的简单模型来对后文算法进行说明，此处p表示已编码点，q表示当前编码点，r表示未编码点。

(1) 公共前缀的识别

PR-3HE需要利用坐标点之间的前缀关系，为此，分别给出坐标分量的最大公共前缀和坐标的最大公共前缀的定义。

表 1　CHM

状态	000	001	010	011	100	101	110	111
0	0	1	3	2	7	6	4	5
1	0	1	7	6	3	2	4	5
2	0	3	1	2	7	4	6	5
3	0	7	1	6	3	4	2	5
4	0	3	7	4	1	2	6	5
5	0	7	3	4	1	6	2	5
6	2	1	3	0	5	6	4	7
7	6	1	7	0	5	2	4	3
8	2	3	1	0	5	4	6	7
9	6	7	1	0	5	4	2	3
10	4	3	7	0	5	2	6	1
11	4	7	3	0	5	6	2	1
12	2	1	5	6	3	0	4	7
13	6	1	5	2	7	0	4	3
14	2	3	5	4	1	0	6	7
15	6	7	5	4	1	0	2	3
16	4	3	5	2	7	0	6	1
17	4	7	5	6	3	0	2	1
18	2	5	1	6	3	4	0	7
19	6	5	1	2	7	4	0	3
20	2	5	3	4	1	6	0	7
21	6	5	7	4	1	2	0	3
22	4	5	3	2	7	6	0	1
23	4	5	7	6	3	2	0	1

表 2　CSM

状态	000	001	010	011	100	101	110	111
0	5	1	13	0	13	22	5	0
1	3	0	7	23	7	1	3	1
2	4	19	3	2	19	4	16	2
3	1	9	2	17	9	1	3	3
4	2	21	21	2	5	4	10	4
5	0	15	15	0	4	11	5	5
6	6	7	12	11	6	20	11	12
7	21	6	1	9	7	7	9	1
8	8	18	9	10	8	10	14	18
9	15	3	8	7	9	7	9	3
10	8	23	23	8	10	10	4	11
11	6	17	17	6	11	5	11	10
12	12	13	12	18	6	17	17	6
13	19	12	13	13	0	15	15	0
14	14	20	14	16	15	16	8	20
15	9	5	15	13	14	13	15	5
16	14	22	16	16	22	14	2	17
17	12	11	17	3	11	12	17	16
18	18	18	19	12	8	23	23	8
19	13	19	18	19	2	21	21	2
20	20	20	14	22	21	6	22	14
21	7	21	4	19	20	21	19	4
22	20	22	16	22	16	0	20	23
23	18	23	10	1	10	23	18	22

定义1 坐标分量X_p和X_q的最大公共前缀P_{X_p,X_q}：P_{X_p,X_q}定义为$\overrightarrow{X_p^n}$，此处n为满足$\overrightarrow{X_p^n}=\overrightarrow{X_q^n}$的最大值，即$n$满足$\overrightarrow{X_p^n}=\overrightarrow{X_q^n}$且$\forall n'>n$，$\overrightarrow{X_p^{n'}}\neq\overrightarrow{X_q^{n'}}$。同理，可类似定义$Y_p$和$Y_q$的最大公共前缀$P_{Y_p,Y_q}$，$Z_p$和$Z_q$的最大公共前缀$P_{Z_p,Z_q}$。本文用$|P_{X_p,X_q}|$, $|P_{Y_p,Y_q}|$, $|P_{Z_p,Z_q}|$分别表示3组对应坐标分量的最大公共前缀长度。

定义2 点p和q的最大公共前缀$P_{p,q}$：$P_{p,q}$定义为$\overrightarrow{p^n}$，满足$n=\min(|P_{X_p,X_q}|,|P_{Y_p,Y_q}|,|P_{Z_p,Z_q}|)$。$|P_{p,q}|$表示点$p$和$q$之间的最大公共前缀长度。

(2) 公共前缀编码约简

在识别2个坐标的公共前缀之后，如何利用公共前缀来提高编码效率是本文需要解决的核心问题。PR-3HE算法依托如下的定理。

定理1 点p和q对应的前$|P_{p,q}|$阶编码值相同，第$|P_{p,q}|+1$阶状态相同。

证明 对$|P_{p,q}|=0$的情形，显见定理成立。对$|P_{p,q}|\geq 1$的情形，因$s_p^1=s_q^1=0$，$p^1=q^1$，故$H_p^1=H_q^1$，$s_p^2=s_q^2$。依次类推，在$2\leq i\leq |P_{p,q}|$时，因$s_p^i=s_q^i$，$p^i=q^i$，得$H_p^i=H_q^i$，$s_p^{i+1}=s_q^{i+1}$，故易见$\overrightarrow{H_p^{|P_{p,q}|}}=\overrightarrow{H_q^{|P_{p,q}|}}$。 证毕

由定理1可见，在p已编码的情况下，可直接将p的前$|P_{p,q}|$阶编码值$\overrightarrow{H_p^{|P_{p,q}|}}$作为$q$的前$|P_{p,q}|$阶编码值$\overrightarrow{H_q^{|P_{p,q}|}}$，从而避免对$q$的$|P_{p,q}|$阶前缀重复编码，实现公共前缀编码约简。

尽管$\overrightarrow{H_p^{|P_{p,q}|}}$可用于$q$的编码，但考虑到$|P_{q,r}|$和$|P_{p,q}|$通常不等，故$\overrightarrow{H_p^{|P_{p,q}|}}$不能直接用于$r$的编码。如何更好地利用$p$的已编码信息来服务$r$及后续未编码点是本文需要解决的关键问题。为此，本文设计了一个长度为k的记忆数组M，依次存储p的各阶状态，进而支持r的编码。本文定义M_r为M可用于编码r的长度，其值为$|P_{p,q}|(|P_{p,q}|\leq|P_{q,r}|$时)或$|P_{q,r}|(|P_{p,q}|>|P_{q,r}|$时)。

(3) 公共前缀编码约简的优化

记忆数组M解决了p的已编码信息对r编码的支持，然而其同样存在以下3个需要解决的问题：(a) $|P_{q,r}|>|P_{p,q}|$时，q和r比p和q有更多的公共前缀，M不能用于从$|P_{p,q}|+1$阶到$|P_{q,r}|$阶的编码；(b) r之后的坐标和p的公共前缀可能非常小或没有公共前缀，M中静态存储的p的状态信息无法用于r之后点的编码；(c) 坐标点编码顺序对公共前缀的长度有重要的影响，如何最大化利用M提高编码效率？

对问题(a)和(b)，为克服静态存储p的状态的不足，引入M的动态更新机制。在编码q时，动态地将M的从$|P_{p,q}|+2$到$|P_{q,r}|+1$位置的状态更新为q的对应阶的状态，从而使得M可支持r从1到$|P_{q,r}|$阶的编码，即M_r总等于$|P_{q,r}|$。该更新是动态的，在编码r时同样对M进行更新，故M可支持r之后坐标的编码。

图4展示了当$k=8$时，$|P_{q,r}|\geq|P_{p,q}|$和$|P_{q,r}|<|P_{p,q}|$ 2种情形下编码示例。图中假定p, q和r为同一竖线上的3点且$X_p=X_q=X_r=10010110_2$，$Y_p=Y_q=Y_r=01101001_2$。坐标栏某个点对应的绿色方框内的部分代表该点与前邻点的公共前缀，记忆数组栏蓝色方框内的部分表示需要更新的阶，Hilbert编码栏红色方框内的部分表示需要迭代编码的阶。图4(a)中，因$|P_{q,r}|=7\geq|P_{p,q}|=3$，故需对记忆数组的第5～8阶进行更新，而图4(b)中，因$|P_{q,r}|=6<|P_{p,q}|=7$，故无需更新操作。

考虑到两个坐标越接近，其倾向于有更多的公共前缀，故引入蛇形扫描的机制依次编码坐标点。以图6所示的3D窗体为例，蛇形扫描先从$x=0$面开始，自下至上依次编码$y=0$列的所有点，然后

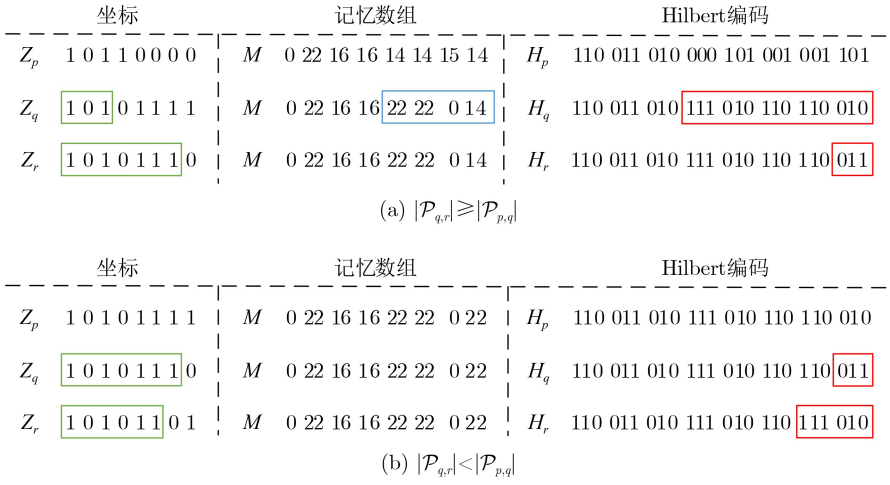

图 4 编码示例

自上至下编码$y=1$列的所有点，直至$y=2^k-1$列，从而编码完$x=0$面。然后按类似的规则编码其他面。通过蛇形编码，连续编码的点为距离接近的位置点，有着较多的公共前缀，因此可尽可能地降低M的更新次数，最大限度地提高编码效率。

编码当前点q的基本流程如图5所示，对应的编码算法如算法1所示。

3.2.2 编码效率分析

迭代编码次数是影响编码效率的关键因素，以3阶3D窗体为例，PR-3HE部分需要迭代编码的次数如图6所示。此外，给出PR-3HE编码效率的详细分析如下：

定理2　在对一个k阶窗体进行编码时，PR-3HE算法对每一个点平均编码阶数小于2。

证明　将整个窗体分为5部分：a_1(第1个点需要迭代编码的次数，图6中用黄色菱形标记)，a_2(除去a_1，在$y=0$，$z=0$这一条直线上的点需要迭代编码的次数，图6中用绿色正方形标记)，a_3(在$\mathrm{mod}(x,2)=1$，$y=2^k-1$，$z=0$这一条直线上的点需要迭代编码的次数，图6中用红色三角形标记)，a_4(除a_1，a_2，a_3之外进入每一列的第1个点需要迭代编码的次数，图6中用蓝色六边形标记)，a_5(其余坐标点需要迭代编码的次数，图6中其余未标记部分)，各部分编码次数计算如下：

算法1 PR-3HE

输入：H_p：p的编码；$|P_{p,q}|$：p和q的公共前缀长度；
　　q：当前点；r：后邻点；k：阶数

输出：H_q：当前点q的Hilbert编码；$|P_{p,r}|$：q和r的公共前缀长度；

1. $H_q \leftarrow \overrightarrow{H_p^{|P_{p,q}|}}$
2. $|P_{p,r}| \leftarrow$计算q和r的最大公共前缀长度
3. $s \leftarrow M[|P_{p,q}|+1]$
4. FOR $i \leftarrow |P_{p,q}|+1$ to k
5. 　$H_q = H_q << 3|\mathrm{CHM}[s][x_q^i][y_q^i][z_q^i]$
6. 　$s = \mathrm{CSM}[s][x_q^i][y_q^i][z_q^i]$
7. 　IF $i <= |P_{q,r}|$
8. 　　$M[i+1]=s$
9. 　END IF
10. END FOR

$a_1=k; a_2=\sum_{j=1}^{k} j\times 2^{k-j}=2^{k+1}-k-2; a_3=2^{k-1}; a_4=2^k\times(2^{k+1}-k\text{-}2); a_5=2^{2k}\times(2^{k+1}-k-2)$。

计算得k阶窗体需要编码的次数为：$a_1+a_2+a_3+a_4+a_5=2^{3k+1}+2^{k-1}-k\times 2^{2k}-k\times 2^k-2$。

故平均编码阶数为$\frac{2^{3k+1}+2^{k-1}-k\times 2^{2k}-k\times 2^k-2}{2^{3k}}<2$。　　证毕

4　解码算法

4.1 解码状态视图

解码阶段设计用于存储Hilbert编码值到1阶坐标的映射视图HCM和用于存储Hilbert编码值到下一阶状态的映射视图HSM。给定图1所示的24个基本状态，构建的HCM和HSM如表3和表4所示。

给定第i阶的Hilbert编码和第i阶状态s^i，可根据 $x^iy^iz^i=\mathrm{HCM}[s^i][H^i]$ 和$s^{i+1}=\mathrm{HSM}[s^i][H^i]$ 分别获得第i 阶的坐标值和第$i+1$ 阶状态s^{i+1} 。

4.2 解码算法

PR-3HD算法

同编码算法，PR-3HD的核心包含公共前缀识别、公共前缀解码约简、公共前缀解码优化3部分。

公共前缀识别部分，H_p和H_q的最大公共前缀P_{H_p,H_q}定义为H_p和H_q的前m阶编码值$\overrightarrow{H_p^m}$，其中m为满足$\overrightarrow{H_p^m}=\overrightarrow{H_q^m}$的最大的$m$。公共前缀解码约简部分，考虑到$H_p$和$H_q$对应的前$|P_{p,q}|$阶坐标值相同，则第$|P_{p,q}|+1$阶的状态相同，故在$p$已解码的情况下，可直接将$H_p$的前$|P_{p,q}|$阶作为$H_q$的前$|P_{p,q}|$阶坐标，从而避免对$H_q$的$|P_{p,q}|$阶前缀重复解码。公共前缀解码约简优化部分，引入动态更新机制提供对H_r及后续编码值的解码支持。考虑到两个编码值越相近，其倾向于有更多的公共前缀，故按编码值从小到大依次对编码值解码，从而最大限度地提高解码效率。完整的PR-3HD算法如算法2所示。

对解码效率分析，按编码类似的思路，可证明对于一个k阶窗体，PR-3HD对每个点的平均解码次数少于8/7。限于幅面，此处未给出详细的证明过程。

5　实验

5.1 实验环境

实验主要硬件平台：CPU：Intel(R)Core(TM) i7-7700 CPU@3.60 GHz, RAM：16 GB (2400 MHz)，系统环境为Windows 11，编译器为Visual Studio 2019。

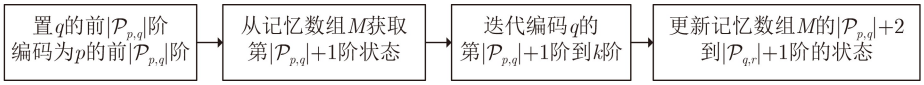

图 5 编码当前点q的流程图

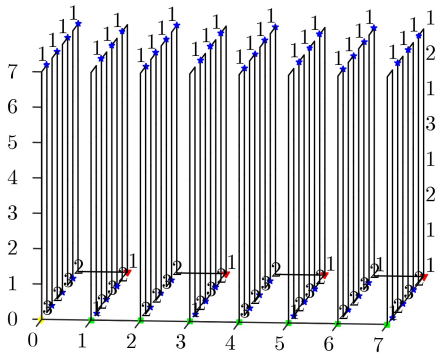

图 6 迭代编码的次数(3阶)

表 3 HCM

状态	0	1	2	3	4	5	6	7
0	000	001	011	010	110	111	101	100
1	000	001	101	100	110	111	011	010
2	000	010	011	001	101	111	110	100
3	000	010	110	100	101	111	011	001
4	000	100	101	001	011	111	110	010
5	000	100	110	010	011	111	101	001
6	011	001	000	010	110	100	101	111
7	011	001	101	111	110	100	000	010
8	011	010	000	001	101	100	110	111
9	011	010	110	111	101	100	000	001
10	011	111	101	001	000	100	110	010
11	011	111	110	010	000	100	101	001
12	101	001	000	100	110	010	011	111
13	101	001	011	111	110	010	000	100
14	101	100	000	001	011	010	110	111
15	101	100	110	111	011	010	000	001
16	101	111	011	001	000	010	110	100
17	101	111	110	100	000	010	011	001
18	110	010	000	100	101	001	011	111
19	110	010	011	111	101	001	000	100
20	110	100	000	010	011	001	101	111
21	110	100	101	111	011	001	000	010
22	110	111	011	010	000	001	101	100
23	110	111	101	100	000	001	011	010

5.2 实验数据集

本文采用了以下两个模拟数据集和一个真实数据集。

窗体数据集：分别设置阶数k=6, 7, 8, 9, 10，生成不同窗体大小的数据集。

离散数据集：非重复地随机取点坐标的数据集。生成以下两种类型的离散数据集：(1)固定坐标数量N=1500万，阶数$k=8$, 12, 16, 20的离散坐标数据集；(2)固定阶数$k=8$，坐标数量N=100万，

表 4 HSM

状态	0	1	2	3	4	5	6	7
0	5	1	0	13	5	0	22	13
1	3	0	1	7	3	1	23	7
2	4	3	2	19	4	2	16	19
3	1	2	3	9	1	3	17	9
4	2	5	4	21	2	4	10	21
5	0	4	5	15	0	5	11	15
6	11	7	6	12	11	6	20	12
7	9	6	7	1	9	7	21	1
8	10	9	8	18	10	8	14	18
9	7	8	9	3	7	9	15	3
10	8	11	10	23	8	10	4	23
11	6	10	11	17	6	11	5	17
12	17	13	12	6	17	12	18	6
13	15	12	13	0	15	13	19	0
14	16	15	14	20	16	14	8	20
15	13	14	15	5	13	15	9	5
16	14	17	16	22	14	16	2	22
17	12	16	17	11	12	17	3	11
18	23	19	18	8	23	18	12	8
19	21	18	19	2	21	19	13	2
20	22	21	20	14	22	20	6	14
21	19	20	21	4	19	21	7	4
22	20	23	22	16	20	22	0	16
23	18	22	23	10	18	23	1	10

算法2 PR-3HD

输入：X_p, Y_p,Z_p：p的坐标分量；$|P_{Hp,Hq}|$：Hp和Hq的最大公共前缀长度；H_q, H_r：q和r的Hilbert编码；k：阶数

输出：q的坐标分量；$|P_{Hq,Hr}|$：Hq和Hr的最大公共前缀长度；

1. X_q,Y_q,Z_q←$(\overleftarrow{X_p^{|P_{H_p,H_q}|}},\overleftarrow{Y_p^{|P_{H_p,H_q}|}},\overleftarrow{Z_p^{|P_{H_p,H_q}|}})$
2. $|P_{H_q,H_r}|$←计算Hq和Hr的最大公共前缀长度
3. s←$M[|P_{H_p,H_q}|+1]$
4. FOR i←$|P_{H_p,H_q}|$+1 to k
5. $x_q^i y_q^i z_q^i = \mathrm{HCM}[s][H_q^i]$
6. $X_q = X_q << 1|x_q^i$
7. $Y_q = Y_q << 1|y_q^i$
8. $Z_q = Z_1 << 1|z_q^i$
9. $s = \mathrm{HSM}[s][H_q^i]$
10. IF $i <= |P_{H_q,H_r}|$
11. $M[i+1] = s$
12. END IF
13. END FOR

200万, 400万, 800万, 1 200万, 1 600万, 2000万的随机坐标数据集。

GeoLife数据集：该数据集为微软亚洲研究院GeoLife项目收集的北京市GPS轨迹数据[18]数据集，包含182个用户在2007年4月至2012年8月间的17 621条轨迹。轨迹的总距离为1 292 951 km，共有24 876 978个坐标点，每个坐标点包含纬度、经度和海拔信息。由于数据集中存在分布于北京市之外的异常点，本文选取位于北京市内的GPS数据作为实验数据，选取坐标范围为$x \in [39.524\,420, 40.324\,200]$, $y \in [115.903\,388, 116.903\,388]$, $z \in [-100, 100]$，总计选择了15 871 978个坐标点。

这3个数据集中，窗体数据集代表数据点紧密相邻的情形，即同一水平线或垂直线上的相邻两点距离为1；离散数据集中的数据点随机分布，相邻两点的平均距离往往远大于窗体数据；而GeoLife数据则介于二者之间，相邻两点的横坐标之间、纵坐标之间可能具有较长的公共前缀。

将本文算法与MOORE[14], LI[15], ZHANG[16], JFK[17]等算法进行比较。为便于描述，为各编码算法增加后缀“-3HE”，为解码算法增加后缀“-3HD”。对所有基于状态视图算法统一采用24个状态视图。

5.3 编码效率对比

离散数据集、窗体数据集、GeoLife数据集上不同编码算法的编码效率对比分别如图7、图8、图9所示。其中窗体数据集的纵坐标为对数坐标。在GeoLife数据集中采用其原始轨迹顺序进行编码。

由图7(a)可见，PR-3HE在离散数据集上的表现优于其余算法，但随着阶数k增大，PR-3HD算法相对其他算法的优势有所降低。其原因在于随着k的增大，Hilbert空间不断增大，数据集倾向于更稀疏，因此PR-3HE需编码的阶数增加。由图7(b)可见，当固定$k=16$时，PR-3HE的效率而随着N增大而不断增加。原因是当阶数k保持不变时，坐标数量在不断增加，坐标点之间有更多共同的阶，从而使得PR-3HE的与其他算法的优势不断扩大。

由图8可见，所有算法的编码时间均随着阶数增加而增加。相较而言，PR-3HE是所有算法中效率最高的，当k=10时，PR-3HE编码仅需65.012 s，而ZHANG-3HE需要104.829 s，PR-3HE比ZHANG-3HE效率提升61.24%，比MOORE-3HE快4.063倍，比LI-3HE快3.512倍。通过进一步分析发现，PR-3HE在6, 7, 8, 9, 10阶下的平均编码阶数分别为1.905, 1.945, 1.969, 1.982和1.990，与定理2分析一致，而ZHANG-3HE的编码阶数则为

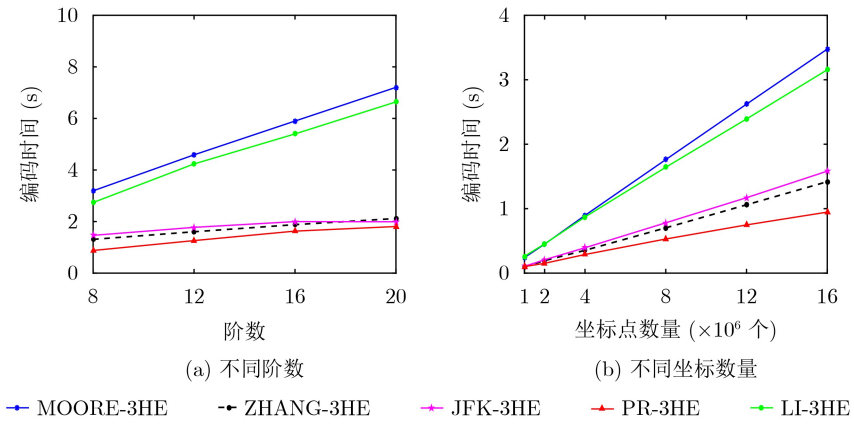

图 7 离散数据集上编码效率对比

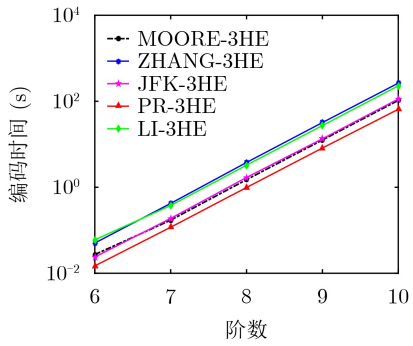

图 8 窗体数据集上编码效率对比

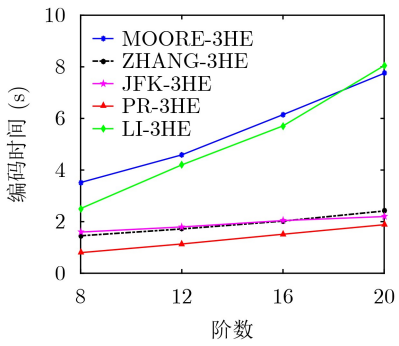

图 9 GeoLife数据集上编码效率对比

6, 7, 8, 9, 10。可以预见，随着阶数增加，PR-3HE相对其他算法的优势还将进一步增大。

由图9可见，PR-3HE在GeoLife数据集上的表现优于其余算法，当阶数k分别为8, 12, 16, 20时，PR-3HE算法编码分别需要0.799 s, 1.136 s, 1.513 s, 1.880 s，其运行速度比ZHANG-3HE最高快44.99%，比MOORE-3HE快3.403倍，比LI-3HE快3.279倍。在k为8, 12, 16, 20阶时，PR-3HE的平均编码阶数分别为：1.340, 3.901, 7.368, 11.156。相对而言PR-3HE在GeoLife上的优势不如在窗体数据集上明显，其原因是GeoLife为离散数据集，相邻编码点的公共前缀相对较窗体数据集小。此外，随着阶数的增加，坐标之间的距离逐步变大，数据集趋向稀疏，因此算法效率有所降低。

5.4 解码效率对比

离散数据集、窗体数据集、GeoLife数据集上不同解码算法解码效率对比分别如图10、图11、图12所示。

由图10(a)可见，PR-3HD在离散数据集上的表现优于其余算法，当k为8, 12, 16, 20阶时，PR-3HD的平均公共前缀分别为6.857 1, 7.240 5, 7.243 7, 9.0079。由图10(b)可见，当固定$k=8$时，PR-3HD算法的效率而随着坐标数量N的增大而不断增加。这是因为当k保持不变时，坐标数量在不断增加，编码值之间有更多共同的阶，从而使得PR-3HD的优势不断扩大。

由图11可见，PR-3HD是所有算法中效率最高的。在$k=10$时，PR-3HD解码仅需62.409 s，而ZHANG-3HD需要107.174 s。PR-3HD的运行速度比ZHANG-3HD快71.28%。比传统算法MOORE-3HD快4.27倍，比LI-3HD快3.42倍。可见，PR-3HD受益于前缀约简，具有更高的效率。

由图12可见，PR-3HD优于其他算法，当阶数k分别为8, 12, 16, 20时，PR-3HD仅需要0.872 2 s, 1.051 s, 1.367 s, 1.743 s的解码时间。其效率比ZHANG-3HD最高可提升39.62%，比MOORE-3HD可提升3.597 0倍，比LI-3HD提升2.309倍。进一步分析，在阶数k为8, 12, 16, 20阶时，PR-3HD的平均解码阶数分别为：0.061, 1.162, 4.182, 7.754。在GeoLife数据集上，PR-3HD相对其他算法仍有明显优势，但优势相对窗体数据集有所减少，其原因在于GeoLife为离散数据集，相邻编码值之间公共前缀相对窗体数据集有所降低。

6 结束语

本文针对现有算法单独编码的缺点，提出了时

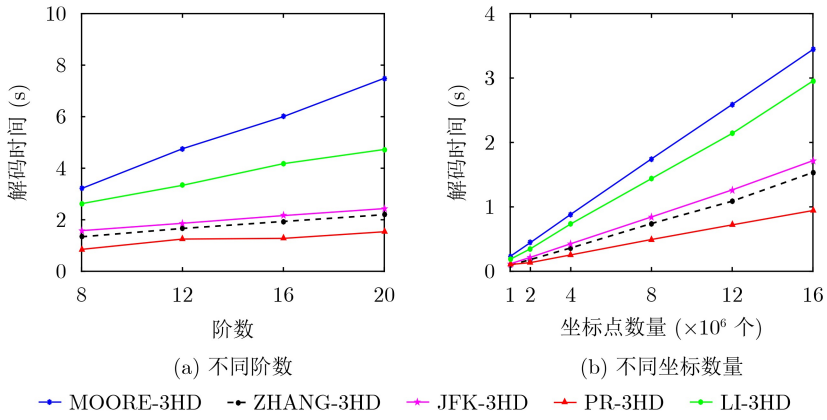

图 10 离散数据集上解码效率对比

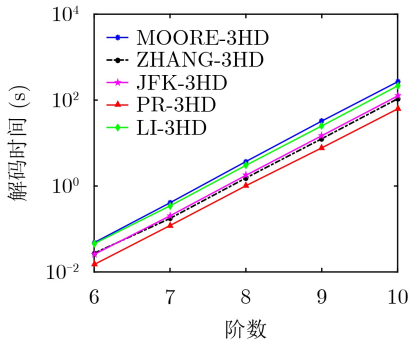

图 11 窗体数据集上解码效率对比

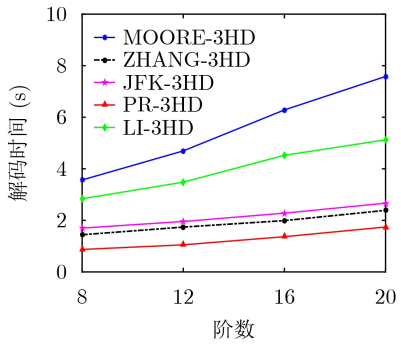

图 12 GeoLife数据集上解码效率对比

间复杂度为$O(1)$的3D编码算法PR-3HE和3D解码算法PR-3HD。通过利用相邻点的公共前缀信息，实现跳过公共前缀的编码和解码，通过引入了公共前缀的定义和识别、公共前缀约简及多种优化技术提高编解码效率。实验结果表明，本文算法相比MOORE算法，效率提高近4倍。下一步拟将本文算法扩展到高阶高维。此外，考虑到现实中数据分布多具有局部聚集特性，因此可将本文算法应用于该类场景。

参 考 文 献

[1] YAO Zhixin, ZHANG Jianqin, LI Taizeng, *et al*. A trajectory big data storage model incorporating partitioning and spatio-temporal multidimensional hierarchical organization[J]. *ISPRS International Journal of Geo-Information*, 2022, 11(12): 621. doi: 10.3390/ijgi11120621.

[2] LIU Zhaoman, WU Lei, MENG Weizhi, *et al*. Accurate range query with privacy preservation for outsourced location-based service in IOT[J]. *IEEE Internet of Things Journal*, 2021, 8(18): 14322–14337. doi: 10.1109/JIOT.2021.3068566.

[3] ZHANG Xuncai, WANG Lingfei, ZHOU Zheng, *et al*. A chaos-based image encryption technique utilizing Hilbert curves and H-fractals[J]. *IEEE Access*, 2019, 7: 74734–74746. doi: 10.1109/ACCESS.2019.2921309.

[4] CHEN Jiafeng, YU Lu, and WANG Wenyi. Hilbert space filling curve based scan-order for point cloud attribute compression[J]. *IEEE Transactions on Image Processing*, 2022, 31: 4609–4621. doi: 10.1109/TIP.2022.3186532.

[5] 贾连印, 陈明鲜, 李孟娟, 等. 基于状态视图的高效Hilbert编码和解码算法[J]. 电子与信息学报, 2020, 42(6): 1494–1501. doi: 10.11999/JEIT190501.

JIA Lianyin, CHEN Mingxian, LI Mengjuan, *et al*. State view based efficient Hilbert encoding and decoding algorithms[J]. *Journal of Electronics & Information Technology*, 2020, 42(6): 1494–1501. doi: 10.11999/JEIT190501.

[6] BÖHM C, PERDACHER M, and PLANT C. A novel Hilbert curve for cache-locality preserving loops[J]. *IEEE Transactions on Big Data*, 2021, 7(2): 241–254. doi: 10.1109/TBDATA.2018.2830378.

[7] 贾连印, 孔明, 王维晨, 等. 数据偏斜分布下的二维Hilbert编解码算法[J]. 清华大学学报(自然科学版), 2022, 62(9): 1426–1434. doi: 10.16511/j.cnki.qhdxxb.2021.21.043.

JIA Lianyin, KONG Ming, WANG Weichen, *et al*. 2-D Hilbert encoding and decoding algorithms on skewed data[J]. *Journal of Tsinghua University* (*Science and Technology*), 2022, 62(9): 1426–1434. doi: 10.16511/j.cnki.qhdxxb.2021.21.043.

[8] 李绍俊, 钟耳顺, 王少华, 等. 基于状态转移矩阵的Hilbert码快速生成算法[J]. 地球信息科学学报, 2014, 16(6): 846–851. doi: 10.3724/SP.J.1047.2014.00846.

LI Shaojun, ZHONG Ershun, WANG Shaohua, *et al*. An algorithm for Hilbert ordering code based on state-transition matrix[J]. *Journal of Geo-Information Science*, 2014, 16(6): 846–851. doi: 10.3724/SP.J.1047.2014.00846.

[9] HAVERKORT H. How many three-dimensional Hilbert curves are there?[J]. *Journal of Computational Geometry*, 2017, 8(1): 206–281. doi: 10.20382/jocg.v8i1a10.

[10] WEISSENBÖCK J, FRÖHLER B, GRÖLLER E, *et al*. Dynamic volume lines: Visual comparison of 3D volumes through space-filling curves[J]. *IEEE Transactions on Visualization and Computer Graphics*, 2019, 25(1): 1040–1049. doi: 10.1109/TVCG.2018.2864510.

[11] WU Yuhao, CAO Xuefeng, and AN Zipeng. A spatiotemporal trajectory data index based on the Hilbert curve code[C]. IOP Conference Series: Earth and Environmental Science, Beijing, China, 2020: 012005. doi: 10.1088/1755-1315/502/1/012005.

[12] TSINGANOS P, CORNELIS B, CORNELIS J, *et al*. Hilbert sEMG data scanning for hand gesture recognition based on deep learning[J]. *Neural Computing and Applications*, 2021, 33(7): 2645–2666. doi: 10.1007/s00521-020-05128-7.

[13] 陈晓宏, 储飞黄, 方胜良, 等. 基于剖分网格改进A*算法的航迹规划研究[J]. 电光与控制, 2022, 29(7): 17–21. doi: 10.3969/j.issn.1671-637X.2022.07.004.

CHEN Xiaohong, CHU Feihuang, FANG Shengliang, *et al*. Trajectory planning based on A* algorithm improved by subdivision grid[J]. *Electronics Optics & Control*, 2022, 29(7): 17–21. doi: 10.3969/j.issn.1671-637X.2022.07.004.

[14] MOORE D. Fast Hilbert curve generation, sorting, and range queries[EB/OL]. https://github.com/Cheedoong/hilbert, 2021.

[15] 李晨阳, 张杨, 冯玉才. *N*维Hilbert编码的计算[J]. 计算机辅助设计与图形学学报, 2006, 18(7): 1032–1038. doi: 10.3321/j.issn:1003-9775.2006.07.024.

LI Chenyang, ZHANG Yang, and FENG Yucai. Calculation of *N*-dimensional Hilbert codes[J]. *Journal of Computer-Aided Design & Computer Graphics*, 2006, 18(7): 1032–1038. doi: 10.3321/j.issn:1003-9775.2006.07.024.

[16] ZHANG Jian and KAMATA S I. A generalized 3-D Hilbert scan using look-up tables[J]. *Journal of Visual Communication and Image Representation*, 2012, 23(3): 418–425. doi: 10.1016/j.jvcir.2011.12.005.

[17] JIA Lianyin, LIANG Binbin, LI Mengjuan, *et al*. Efficient 3D Hilbert curve encoding and decoding algorithms[J]. *Chinese Journal of Electronics*, 2022, 31(2): 277–284. doi: 10.1049/cje.2020.00.171.

[18] ZHENG Yu, XIE Xing, and MA Weiying. GeoLife: A collaborative social networking service among user, location and trajectory[J]. *IEEE Data Engineering Bulletin*, 2010, 33(2): 32–39.

贾连印：男，博士，副教授，研究方向为数据库、数据挖掘、信息检索等.

范 瑶：男，硕士生，研究方向为数据库、信息检索.

丁家满：男，硕士，教授，研究方向为数据挖掘、云计算等.

李晓武：男，博士，讲师，研究方向为数据库等.

游进国：男，博士，副教授，研究方向为数据库、数据挖掘等.

责任编辑：马秀强