文章编号: 1001- 2486(2010) 05- 0075- 05

基于查表的空间填充曲线映射算法*

吴国福, 窦 强, 窦文华
(国防科技大学 计算机学院, 湖南 长沙 410073)

摘 要: 空间填充曲线是进行数据降维处理的典型方法。首先给出对角线空间填充曲线的映射规则, 该规则使得在高维情况下存在唯一曲线, 并且每一维度上的格点数目不受限制。然后提出等势面的概念, 推导出不同等势面上格点数量的递推关系。在此基础上, 给出基于查表的对角线空间填充曲线映射算法, 该算法执行快、可扩展性好, 其时间复杂度随维度的增加呈线性增长。

关键词: 空间填充曲线; 对角线; 等势面; 降维
中图分类号: O244 文献标识码: A

Table-based Space-filling Curve Generation

WU Guo-fu, DOU Qiang, DOU Wen-hua
(College of Computer, National Univ. of Defense Technology, Changsha 410073, China)

Abstract: Space-filling curves are classical ways to reduce the dimensions of data. This paper first presents the mapping rules of the diagonal space-filling curve. Unique curve which exists under the rules and the number of grids on each dimensionality is not constrained. Then a new conception about equipotential surface was given, and the relationship between the number of grids on different equipotential surface was dirived. Table-based space-filling curve generation algorithms on the basis of the realtionship were presented. It is found that the algorithms have low running time and high scalability.

Key words: space-filling curve; diagonal; equipotential surface; dimension reduction

空间填充曲线[1] (space-filling curve) 技术是建立多维数据空间和一维数据空间相互映射的典型方法, 被广泛应用于多个研究领域[2-3]。文献[4- 7] 针对不同的空间填充曲线提出了不同的映射算法, 这些算法难以扩展到更高维空间。本文提出对角线空间填充曲线快速映射算法。根据对角线空间填充曲线的特点提出等势面的概念, 分析获得等势面格点数量之间的递推关系, 进而给出基于查表的高效快速映射算法。

1 空间填充曲线定义

1890 年, 皮亚诺给出从单位长度到单位正方形的连续满射函数, 即皮亚诺曲线。1891 年, 希尔伯特首次给出从单位长度到单位 d 维立方体映射的几何构造。随后大量空间填充曲线被陆续提出。

将单位 d 维立方体切分成等大格子, 将单位线段切分成同样数量的子段, 并在小格子与子段之间建立一一对应关系。通过无限切分, 即可用单位长度将单位 d 维立方体“填满”。

定义 1 如果映射 $f:[0, 1]\rightarrow[0, 1]^d$ 是从单位长度到单位 d 维立方体的连续满射函数, 则称映射 f 为 d 维空间填充曲线[1]。

在计算机应用中, 根据应用特点只需将格点切分成指定大小即可。我们假设每一维度切分的格点数目为 g, 则整个单位空间被切分成 g^d 个格点。给每个格点赋予一个整数坐标, 则我们可以得到离散化的空间填充曲线, 即 d 维整数区间 $[0, g-1]^d$ 到整数区间 $[0, g^d-1]$ 的双射函数 f'。在下文的论述中将不区分连续的和离散化的空间填充曲线, 读者请根据上下文自行判断。

* 收稿日期: 2010- 04- 20
基金项目: 国家自然科学基金资助项目(60633050)
作者简介: 吴国福(1980—), 男, 博士生。

空间填充曲线依次穿越 d 维空间每个格点刚好一次,格点的访问顺序建立了映射关系。不同的空间填充曲线区别在于格点的访问顺序不同,根据穿越曲线的路径走向自相似性[2],空间填充曲线分为两类:递归的空间填充曲线和非递归的空间填充曲线。图1显示了几种不同的2维空间填充曲线,每一维上的格点数目为8。其中(a)、(b)属于递归的空间填充曲线,(c)、(d)属于非递归的空间填充曲线。更多的关于空间填充曲线的介绍参考文献[1]。

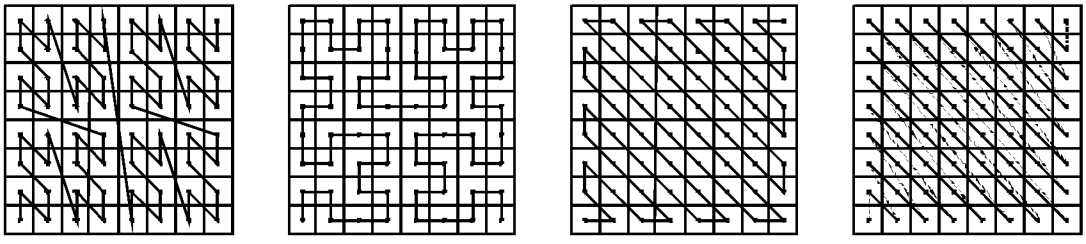

图1 2维空间填充曲线
Fig. 1 2-dimension space-filling curve

2 对角线空间填充曲线快速映射算法

本节研究对角线空间填充曲线映射算法。假定多维空间的维度为 d,每一维度上有 g 个格点。

2.1 对角线空间填充曲线映射规则

首先规定对角线空间填充曲线访问规则,给定格点 $C=(c_1, c_2, \cdots, c_d)$ 和 $C'=(c'_1, c'_2, \cdots, c'_d)$,令其访问序号分别为 p、q,$k=\sum_{i=1}^{d} c_i$,$k'=\sum_{i=1}^{d} c'_i$,则 $p<q$ 当且仅当下列条件之一成立:

(1) $k<k'$

(2) $k=k'$,存在 i,$c_i<c'_i$,当 $j<i$ 时,$c_j=c'_j$

2.2 格点访问顺序的计算

二维对角线空间填充曲线的访问顺序直观,容易求解;对于高维空间($d>2$)中的格点 $C=(c_1, c_2, \cdots, c_d)$,访问顺序不再直观。下面介绍格点序号的计算方法,快速算法的具体描述在2.4小节给出。

首先引入等势面的概念,等势面是指坐标元素之和相等的所有格点集合,用符号 E_i 表示,下标 i 表示等势面高度,即格点坐标元素之和,显然 $i\geqslant 0$。对于格点 $C=(c_1, c_2, \cdots, c_d)$,令 $k=\sum_{i=1}^{d} c_i$,由映射规则知格点 C 的访问顺序 $SFC\_Order(C)$ 决定于 E_i 中格点的数量 $|E_i|$ ($i<k$) 和 C 在 E_k 中的访问顺序 $E_k\_Order(C)$,即

$$SFC\_Order(C)=\sum_{i=0}^{k-1}|E_i|+E_k\_Order(C) \tag{1}$$

下面分别推导 E_k 中格点数量 $|E_k|$ 和 C 在 E_k 中顺序 $E_k\_Order(C)$ 的计算方法。

使用函数 $f(k, g, d)$ 表示等势面 E_k 中格点的数量。由于 k 是格点的坐标元素之和,所以 k 的有效取值范围是 $[0,\ d\cdot(g-1)]$,我们规定

$$f(k, g, d)=0, \text{如果 } k<0 \tag{2}$$

$$f(k, g, d)=0, \text{如果 } k>d\cdot(g-1) \tag{3}$$

当 $d=1$ 时每个等势面只有一个格点,即

$$f(k, g, 1)=1,\ (k\in[0,\ g-1],\ k,\ g\in N) \tag{4}$$

由于格点的坐标元素值不小于零,故 $k=0$ 当且仅当格点坐标每个元素值都为0,也就是说等势面 E_0 中仅有一个元素,即

$$f(0, g, d)=1,\ (g, d\in N) \tag{5}$$

等势面 E_k 中格点 $C=(c_1, c_2, \cdots, c_d)$ 的坐标元素值之和为 k, 首先考虑元素 c_d 的取值, 当 c_d 取 0 时其他 $d-1$ 个元素之和必须为 k, 即 $\sum_{i=1}^{d-1} c_i = k$, 即当 $c_d=0$ 时共有 $f(k, g, d-1)$ 个格点在 E_k 中, 同理当 c_d 取 $i \in [0, g-1]$ 时, 共有 $f(k-i, g, d-1)$ 个格点在 E_k 中, 因此可得

$$f(k,g,d)=f(k-0,g,d-1)+f(k-1,g,d-1)+f(k-2,g,d-1)+\cdots+f(k-g+1,g,d-1) \tag{6}$$

这样 d 维等势面 E_k 格点数量可用 $(d-1)$ 维上多个等势面格点数量之和表示。同样 d 维等势面 E_{k-1} 格点数量可以表示为

$$f(k-1, g, d)=f(k-1-0, g, d-1)+f(k-1-1, g, d-1)+\cdots+f(k-1-g+1, g, d-1) \tag{7}$$

式(6)、(7)相减得到函数 $f(k, g, d)$ 的递推关系, $f(k, g, d)-f(k-1, g, d)=f(k-0, g, d-1)-f(k-1-g+1, g, d-1)$, 即

$$f(k, g, d)=f(k-1, g, d)+f(k, g, d-1)-f(k-g, g, d-1) \tag{8}$$

使用式(2)、(4)、(5)、(8), 可以依次求解各个维度上不同高度的等势面中所含有的格点数量。

确定格点 C 的访问顺序还需要求解格点 $C=(c_1, c_2, \cdots, c_d)$ $(k=\sum_{i=1}^{d} c_i)$ 在等势面 E_k 中的访问顺序。求解思路如下: 假设格点 $X=(x_1, x_2, \cdots, x_{d-1}, x_d)$ 位于等势面 E_k 中(即 $\sum_{i=1}^{d} x_i = k$), 并且 X 在 E_k 中的访问顺序 p 小于 C 在 E_k 中的访问顺序 q, 根据映射规则(2)求解满足条件的 X 数量。先考虑 x_1 的取值, 当 $x_1 \in [0, c_1-1]$, $x_i \in [0, g-1]$ $(i \in [2, d])$ 时, 因为 $x_1 < c_1$, 根据规则(2)得知 $p<q$, 这种情况下 X 的数量 $N_1=\sum_{j=0}^{c_1-1} f(k-j, g, d-1)$。固定 $x_1=c_1$, 考虑 x_2 的取值, 当 $x_2 \in [0, c_2-1]$, $x_i \in [0, g-1]$ $(i \in [3, d])$ 时, 因为 $x_1=c_1$ 且 $x_2<c_2$, 由规则(2)得知 $p<q$, 这种情况下 X 的数量 $N_2=\sum_{j=0}^{c_2-1} f(k-c_1-j, g, d-2)$。以此类推可以求得 $N_i=\sum_{j=0}^{c_i-1} f(k-\sum_{t=1}^{i-1} c_t-j, g, d-i)$, $(i \in [1, d-1])$, 显然 $N_d=0$。上述推导过程中各个 N_i 之间没有重叠且涵盖所有可能 X 的取值, 所以满足条件的 X 数量为 $N=\sum_{i=1}^{d} N_i$, 则 C 在等势面 E_k 中的顺序为 N(因为从 0 开始计数), 即

$$E_k\_Order(C)=\sum_{i=1}^{d} N_i=\sum_{i=1}^{d}\sum_{j=0}^{c_i-1} f(k-\sum_{t=1}^{i-1} c_t-j, g, d-i) \tag{9}$$

结合式(1)、(9)得格点 $C=(c_1, c_2, \cdots, c_d)$ 在对角线空间填充曲线中的访问顺序为

$$SFC\_Order(C)=\sum_{i=0}^{k-1} f(i, g, d)+\sum_{i=1}^{d}\sum_{j=0}^{c_i-1} f(k-\sum_{t=1}^{i-1} c_t-j, g, d-i) \tag{10}$$

等势面中格点数量表可离线构造, 函数 f 值通过查表获得, 根据式(10)可快速确定格点 C 在对角线空间填充曲线中的访问顺序。

2.3 格点坐标的计算

给定格点 C 的访问序号 $SFC\_Order(C)$, 逆向求解格点坐标。首先根据格点的序号 $SFC\_Order(C)$ 确定格点所处的等势面高度 k。因为 $0 \leqslant E_k\_Order(C) < |E_k|$(序号从 0 开始计数), 根据式(1)得

$$\sum_{i=0}^{k-1} |E_i| \leqslant SFC\_Order(C) < \sum_{i=0}^{k} |E_i| \tag{11}$$

根据式(11)和等势面格点数量表可确定高度 k。根据式(1)和高度 k 确定格点 C 在等势面 E_k 中的访问顺序 $E_k\_Order(C)$, 然后依次计算格点 C 各个坐标元素值 c_i。首先求解坐标元素 c_1, 令 $N_1=$

$\sum_{j=0}^{c_1-1} f(k-j,g,d-1)$，由式(9)可得 $E_k\_Order(C) \geqslant N_1$；$f(k-c_1,g,d-1)$ 表示当 c_1 确定后，在保证高度 k 的前提下其它坐标元素取任意值的各种组合的数量，因此 $E_k\_Order(C) < N_1 + f(k-c_1,g,d-1)$（序号从0开始计数）。故

$$\sum_{j=0}^{c_1-1} f(k-j,g,d-1) \leqslant E_k\_Order(C) < \sum_{j=0}^{c_1} f(k-j,g,d-1) \tag{12}$$

令 $N_i = \sum_{j=0}^{c_i-1} f(k-\sum_{t=1}^{i-1} c_t - j, g, d-i)$，同理可推导得到

$$\sum_{j=0}^{c_i-1} f(k-j,g,d-1) \leqslant E_k\_Order(C) - \sum_{u=1}^{i-1} N_u < \sum_{j=0}^{c_i} f(k-j,g,d-1),\ i \in [1, d-1] \tag{13}$$

根据不等式(13)和等势面格点数量表可依次确定 $c_i(i \in |1, d-1|)$，由于 k 已知，c_d 也随之确定。

2.4 基于查表的快速映射算法

根据2.2、2.3小节的推导结果，本小节给出具体的对角线空间填充曲线映射算法及逆映射算法。等势面元素数量表是上述算法的基础，可提前离线构造。下面分别描述等势面格点数量表构造算法BuildTable，从多维空间到一维空间的映射算法SFC_N2ONE，以及从一维空间到多维空间的映射算法SFC_ONE2N。

2.4.1 等势面格点数量表构造算法

根据式(2)、(4)、(5)、(8)，可快速递推出表中各行元素，算法形式化描述见表1，其中参数 d_{max} 表示可能的最高维度。算法的执行时间主要集中在步骤(2)到步骤(5)的两层循环上，故其时间复杂度为 $O(g \cdot d_{max})$。

表1 表构造算法BuildTable

Tab.1 Table construction algorithm: BuidTable

$table_g = BuildTable(g, d_{max})$

(1) $table_g = zeros(d_{max}, (g-1)\cdot d_{max}+1)$; $table_g(1, 0:(g-1)) = 1$; $table_g(1:d_{max}, 1) = 1$; \\ *initialize*

(2) *for* $i = 2$ *to* d_{max} *do*

(3) *for* $j = 2$ *to* $i\cdot(g-1)$ *do*

(4) *if* $(j-g) < 0$ $table(i, j+1) = table(i, j+1-1) + table(i-1, j+1)$;

(5) *else* $table(i, j+1) = table(i, j+1-1) + table(i-1, j+1) - table(i-1, j+1-g)$.

2.4.2 映射算法 *SFC_N2ONE*

根据式(10)和参数 g 对应的表格 $table_g$，格点 C 的访问顺序求解算法SFC_N2ONE形式化描述见表2。算法中 k 的最大值为 $g \cdot d$，因此最坏情况下步骤(2)的执行时间为 $O(g \cdot d)$。外围循环变量 i 固定后，步骤(4)、(5)所示的内循环执行次数最多为 g，因此步骤(3)到步骤(6)的两层循环最坏情况下执行时间为 $O(g \cdot d)$。故算法SFC_N2ONE的时间复杂度为 $O(g \cdot d)$。

表2 映射算法SFC_N2ONE

Tab.2 Mapping algorithm: SFC_N2ONE

$seq = SFC\_N2ONE(C, d, table_g)$

(1) $k = sum(C)$; $semi\_k = 0$;

(2) $seq = sum(table_g(d, 0:(k-1)))$;

(3) *for* $i = 1$ *to* d *do*

(4) *for* $j = 0$ *to* $(C(i)-1)$ *do*

(5) $seq = seq + table_g(d-i, k-semi\_k-j)$;

(6) $semi\_k = semi\_k + C(i)$.

2.4.3 映射算法 SFC_ONE2N

根据式(11)、(13)和参数 g 对应的表格 $table_g$，格点 C 的坐标求解算法SFC_ONE2N形式化描述见

表3。算法中 k 的最大值为 $g \cdot d$, 因此求解 k 最坏情况下执行时间为 $O(g \cdot d)$。坐标元素的最大值为 $g-1$, 因此求解单个坐标元素值最大执行次数为 g 次, 步骤(4)至步骤(8)最坏情况下执行时间为 $O(g \cdot d)$。故算法 SFC_ONE2N 的时间复杂度为 $O(g \cdot d)$。

表 3 映射算法 SFC_ONE2N

Tab.3 Mapping algorithm: SFC_ONE2N

$C = SFC\_ONE2N(seq, d, table_g)$

(1) $k = 0$; $semi\_k = 0$;

(2) $while\ (seq >= table_g(d, k))\ do$

(3) $seq = seq - table_g(d, k)$; $k = k + 1$;

(4) $for\ i = 1\ to\ (d-1)\ do$

(5) $value = 0$;

(6) $while\ (seq >= table_g(d-i, k-semi\_k-value))\ do$

(7) $seq = seq - table_g(d-i, k-semi\_k-value)$; $value = value + 1$;

(8) $C(i) = value$; $semi\_k = semi\_k + value$;

(9) $C(d) = k - semi\_k$.

表4比较了本文的算法与文献中类似算法的区别, 本文中的算法命名为 T_SFC, V_SFC 见文献[5], SFCGen 见文献[6], C_SFC 见文献[7]。

表 4 不同算法的对比

Tab.4 Comparison of different algorithm

算法	执行时间	扩展到高维	每维的格点数量 g	其它
T_SFC	$O(g \cdot d)$	容易	任意	需离线构造表格
V_SFC	$O(g \cdot d)$	困难	2 的幂次	需离线完成标号标记
SFCGen	$O(g \cdot d)$	困难	2 的幂次	需离线构造表格
C_SFC	$O(g^2)$	只针对 2 维	任意	直接执行

3 结 论

从多维数据空间到一维数据空间的映射为多维应用提供了预处理步骤, 映射的目标是用单一的数值代替多维空间中的向量。空间填充曲线是将多维空间映射到一维空间的典型方法, 在多个领域得到广泛使用。本文的主要工作包括以下两个方面: (1) 提出对角线空间填充曲线, 该曲线访问规则单一, 在高维情况下仅存在唯一的曲线; 同时每一维度上的格点数量不必受限于 2 的幂值, 可以是任意整数值。(2) 提出等势面的概念, 推导出等势面中格点数量之间的递推关系; 在此基础上给出基于查表的对角线空间填充曲线快速映射算法, 该算法高效, 容易扩展到高维。

参 考 文 献:

[1] Sagan H. Space Filling Curves[M]. Springer, Berlin, 1994.

[2] Valantinas J. On The Use of Space-filling Curves in Changing Image Dimensionality[J]. Information Technology and Control. 2005, 34(4): 345-354.

[3] 刘胤田, 唐常杰, 曾涛, 等. 基于空间填充曲线的数据分发区域匹配[J]. 系统仿真学报, 2007, 19(4).

[4] Bially T. Space_filling Curves: Their Generation and Their Application to Bandwidth Reduction[J]. IEEE Trans. on Information Theory, 1969, IT-15(6): 658-664.

[5] Bartholdi J, Goldsman P. Vertex-labeling Algorithms for the Hilbert Space-filling Curve[J]. Software Practice and Experience. 2001(31): 395-408.

[6] Jin G H, Crummey J M. SFCGen: A Framework for Efficient Generation of Multi-dimensional Space-filling Curves by Recursion[J]. ACM Transactions on Mathematical Software. 2005, 31(1): 120-148.

[7] Lee P Z, Huang C W, Shih T Y. On Cantor's Space-filling Curve[J]. Journal of TaiWan geographic information science. 2006(4): 13-26.