Peano 曲线的演化生成算法及其推广*)

王能超[1] 陈宁涛[2] 施宝昌[1]

(华中科技大学并行计算研究所 武汉 430074)[1] (华中科技大学计算机学院 武汉 430074)[2]

摘 要 基于演化思想探讨了 Peano 空间填充曲线的非递归生成算法及其推广。通过基本元素的确定、变换和组合三个步骤完成曲线的绘制。算法简单、快速,而且有一定的通用性。

关键词 Peano 曲线,空间填充曲线,分形,演化

Peano Curves Generated with Evolutional Algorithm and its Generalization

WANG Neng-Chao[1] CHEN Ning-Tao[2] SHI Bao-Chang[1]

(Parallel Computing Institute, Huazhong University of Science & Technology, Wuhan 430074)[1]

(Computer Institute, Huazhong University of Science & Technology, Wuhan 430074)[2]

Abstract Based on the evolutionary concept, a non-recursive algorithm and its generalization are designed for Peano curves' generating, which includes three steps as the unit shapes' designing, transforming and combining to draw the curves. Experiment shows that the algorithm is simple and fast. It is significant that the algorithm presents a precept for drawing such space-filling curves as Peano curves.

Keywords Peano curve, Space-filling curve, Fractal, Evolution

1 引言

Peano 曲线族是闭合间隔单元 $I=[0,1]$到闭合矩形单元 $S=[0,1]^2$ 的连续映射,也是所有能够填满二维或更高维空间的连续分形曲线的总称,故又称为空间填充曲线,或者 FASS(space-filling, self-avoiding, simple and self-similar)曲线,它是由意大利数学家 Peano 于 1890 年提出的。这种奇怪的曲线曾使数学界大吃一惊,自然引起了广泛的注意,不久便找到了具有这样性质的其他曲线,如著名的 Hilbert 曲线,后来统称为 Peano 曲线[1]。这种曲线已在图像存储和检索、空间数据库索引等领域得到了成功的应用[2,3],因此研究 Peano 曲线有重要的理论意义和应用价值。

Peano 曲线的背后有深刻的数学思想,它与 Cantor 集有密切关系[1]。本文不打算讨论其数学基础,仅就曲线的生成算法进行研究。作为分形的重要研究对象,近年来的大部分算法都是基于递归技术生成曲线,如 L 系统方法。递归方法的明显缺点是内存需求大,效率低。本文基于演化的思想剖析了曲线的生成过程,提出了一种非递归算法。

2 Peano 曲线特征

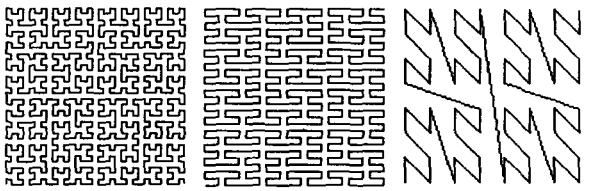
经典曲线是 1 维的,相对于平面,曲线的空隙很大。但 Peano 曲线的 Hausdoff 维数为 2。在曲线理论建立以前,人们对"什么是曲线"认识不尽相同,Peano 曲线的提出是为了作为曲线的一个特例而构造的,在分形理论出现以后,就很容易发现它是一个典型的分形图形。Peano 曲线有多种形式,图 1 就是几个例子。

Peano 曲线主要有如下几个特点:

· 非自交性 曲线有两个出口点:始点和终点。从始点到终点是"一笔画"的。

· 空间无关性 不管矩形面积多么大,当迭代次数足够大,都可以填满整个区域。

· 分形特征 比如,精细的结构、自相似性、标度不变性等。



(1) Peano1 (2) Peano2 (3) Peano3

图 1 Peano 曲线

· 初始图形简单 这些曲线都是从一个简单的初始图出发,按照一定的规则迭代产生的。

· 迭代规则简单 在分形理论中,迭代规则也叫做生成元,生成元往往由简单的步骤组成。

· 条件敏感性 条件包括初始图形和迭代规则,两者中任何一个发生微小的变化,结果图形变化较大。

3 算法思想

3.1 L 系统算法

Lindenmayer 于 1980 年提出了一种空间填充曲线的构造模式,即著名的 Lindenmayer System(L-system),或称 Parallel String-Rewrite Systems,它由一个原子与一个产生式规则集组成,而这种产生式规则决定了原子的复写特征[6]。

L 系统进行曲线绘制即是产生式规则字符串替换原子的

*)基金项目:国家自然科学基金资助项目(60473015)。王能超 博导,研究方向为演化算法,快速算法,并行计算。陈宁涛 博士生,研究方向为分形与混沌理论及可视化、并行计算。施宝昌 教授,博导,研究方向为并行算法,快速算法,流体计算,非线性最优化方法,多目标决策。

过程。比如，原子是 F，产生式规则是 $F\rightarrow F+F-F+F$。指定迭代次数，用字符串 $F+F-F+F$ 替换每一个 F。迭代一次，当前字符串就是 $F+F-F+F$。再迭代一次，当前字符串变为：$F+F-F+F+F+F-F+F-F+F-F+F+F+F-F+F$。迭代指定次数后，用类似于乌龟爬行的模型解释字符串。F 表示按照当前方向画一条指定长度的线段，+表示在当前方向的基础上顺时针旋转指定角度，-表示在当前方向的基础上逆时针旋转指定角度。

比如[6]，Koch 曲线的 L 系统字符串表示是：F，$F\rightarrow F+F-F+F$，60°；龙曲线的 L 系统字符串表示是：X，$X\rightarrow X+YF+$，$Y\rightarrow -FX-Y$，45°；Peano1 的 L 系统字符串表示是：L，$L\rightarrow +RF-LFL-FR+$，$R\rightarrow -LF+RFR+FL-$，45°；Peano2 的 L 系统字符串表示是[7]：X，$X\rightarrow XFYFX+F+YFXFY-F-XFYFX$，$Y\rightarrow YFXFY-F-XFYFX+F+YFXFY$，90°；Peano3 的 L 系统算法我们没有找到。

用 L 系统绘制曲线是两个过程：设计 L 系统和利用 L 系统进行绘制。设计 L 系统（给出公理，定义产生式）的过程是依照自相似结构形成信息压缩的过程。利用 L 系统进行绘制，则是它的逆过程，即信息膨胀过程，或者说复原过程。如何从被描述的对象提取小量的信息，从而完成 L 系统的设计，这是困难的也是十分有意义的研究课题。与此相比，目前研究较多的是绘制，即信息复原工作[1]。

3.2 演化算法

L 系统算法显然是一种递归算法，而将要介绍的演化算法属于非递归算法。

现实问题往往比较复杂，人们一直在追求用简单的方法去解决它。从简单到复杂是个简单的问题，而反问题相对比较复杂。但是如何从一个复杂问题寻找隐含的简单规律一直是一个研究重点。幸运的是，人类在这方面的努力一直没有停息。比如数学史上的三件事就证明了一点：1715 年，Taylor 公式的提出使得人们可以用简单的多项式集合 $\{x^k\}$ 经过平移、压缩和组合来产生任意的函数 $f(x)$；大约百年之后的 1822 年，Fourier 提出了用三角函数集合 $\{\cos(x),\sin(x)\}$ 来达到上述目的；又是百年之后的 1923 年，Walsh 提出了用方波函数 $R(x)$ 产生的 Walsh 函数系[8] $\{W_i(x)\}$ 经过平移、压缩和组合来产生任意的函数 $f(x)$。这三位伟大的数学家的工作都是试图通过简单来模拟复杂，并且他们的工作在科学发展的道路上都获得了巨大成功。这种思想很值得我们在面临复杂问题时所采用。他们的方法的共同点都是通过基本元素 $\{x^k\}$、$\{\cos(x),\sin(x)\}$ 和 $\{W_i(x)\}$ 经过变换和组合形成复杂函数 $f(x)$。这其中隐含着三个步骤：确定基本元素；基本元素的变换；基本元素的组合。本文的演化思想体现在第二步上。

由于 Peano 曲线的矩形特征，很容易把它看成一个几何矩阵，矩阵元素是由曲线的基本元素组成。若基本元素的个数确定，通过对基本元素进行编码就很容易把几何矩阵转化成代数矩阵，相应的基本元素的变换就转化为代数矩阵的运算。而通过分析 Peano 曲线的迭代规则，发现其基本元素的个数是确定的。

4 演化算法实现及其推广

4.1 Peano1

Peano1 就是经典的 Hilbert 曲线。先分析其演化特征。它是 Peano 曲线的典型代表，也是该曲线族中最难的曲线，研究得比较多[4,5]。其经典描述如下：首先将正方形 4 等分，求出各个小正方形中心，并将它们连接；其次，将各个小正方形再细分为 4 个相同的小正方形，并连接各小正方形中心；……，按照这种方法不断细分下去，并按一定规则一一连接，就得到 Hilbert 曲线（图 2）。其演化规则如下：设当前状态是 F_{k+1}，前一状态是 F_k，F_{k+1} 由 4 部分组成，F_{k+1} 与 F_k 的关系见图 3。其中 $\overrightarrow{F_k}$ 表示顺时针旋转 90 度，$\overleftarrow{F_k}$ 表示逆时针旋转 90 度。曲线的基本元素有 4 个：开口方向不同的“U”形。基本元素的组合需要连线，从曲线的特点可以看出，每层连接 3 条线：1 条横线，2 条竖线。线段的长度逐级减半。连线格局见图 4，其中 B 表示块，其宽度与迭代的层次有关，以下雷同。

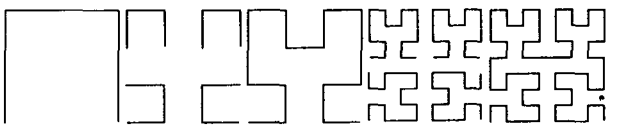


图 2 Peanol 的演化

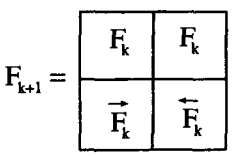


图 3 Peanol 的迭代

图 4 连线格局

4.2 Peano2

该曲线初始图是个“S”形，然后将正方形 9 等分，按照一定规则复制前一个图形，并将它们顺次连接；其次，将各个小正方形再均分成 9 个，按照一定规则复制前一个图形，并顺次连接；……，按照这种方法不断细分下去，并按一定规则一一连接，就得到 Peano2（图 5）。其演化规则如下：F_{k+1} 由 9 部分组成，F_{k+1} 与 F_k 的关系见图 6。其中 $\ddot{F}_k$ 表示与 F_k 左右偶对称。曲线的基本元素有 2 个：“S”形和反“S”形。每层连接 8 条线，线段长度递减为 1/3。连线格局有两种对偶形式（图 7）。

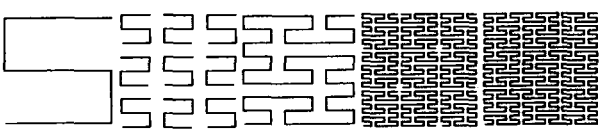


图 5 Peano2 的演化

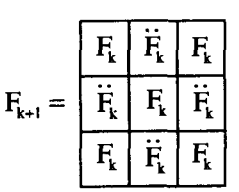


图 6 Peano2 的迭代

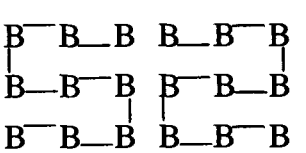


图 7 两种连线格局

4.3 Peano3

其初始图是个“N”形，然后将正方形 4 等分，按照一定规则复制前一个图形，并将它们依次连接：其次，将各个小正方形再均分为 4 个，按照一定规则复制前一个图形，并依次连接；……，按照这种方法不断细分下去，并按一定规则一一连接，就得到 Peano3（图 8）。其演化规则如图 9。而曲线的基本元素只有 1 个：“N”形。每层连接 3 条线，线段的长度逐级减

半。连线格局见图 10。

图 8　Peano3 的演化

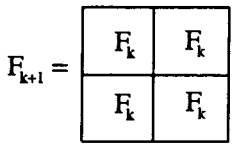


图 9　Peano3 的迭代

图 10　连线格局

4.4　实验结果

在主频为 2.4GHz 微机(256M 内存)上比较了 L 系统算法和本文算法(表 1 和表 2)。可以看出,Peano1 算法比 L 系统算法速度提高了将近 1 倍,Peano2 算法也有明显提高。Peano2 耗时的原因是随着迭代次数的增大,基本元素的个数增长比较快。Peano3 的计算时间与 Peano1 相当,由于没有找到相应的 L 系统算法,故没有比较。其中 LSA (L Systems Algorithm)表示 L 系统算法,EA (Evolutionary Algorithm)表示本文算法,Times 表示迭代次数。

表 1　Peano1 的两种算法的计算时间比较　单位:秒

Times	6	7	8	9	10	11	12
LSA	0.01	0.03	0.17	0.44	1.74	5.64	21.29
EA	0.00	0.01	0.05	0.20	0.83	3.38	13.56

表 2　Peano2 的两种算法的计算时间比较　单位:秒

Times	4	5	6	7	8
LSA	0.04	0.20	1.43	12.67	121.64
EA	0.03	0.18	1.22	10.57	95.00

4.5　算法分析

算法是基于 Visual C^{++} 实现的,CPeanoCurve 类定义如下:

```
class CPeanoCurve
{
private:// Attributes
  int m_xOrig; int m_yOrig;//曲线的左上角坐标
  int m_segLength;//基本元素中线段的长度
  int m_depth;//迭代深度
public:// Operations
  CPeanoCurve(int xOrig=0,int yOrig=0,int segLength=10,int
  depth=3);//构造函数
 void Draw(CDC* pDC);
 void Link(CDC* pDC,int B);
};
```

其中成员函数 Draw()完成矩阵变换和基本元素的绘制,函数 Link()根据设定的连线格局完成基本元素的连接。矩阵变换的作用就是确定最终绘图区域中的网格点上的基本元素的形态,而具体每个网格点上的元素形态根据迭代公式很容易求出。在绘制图形时,采用块复制[8]的方式进行,这大大加快了运算时间。而连线格局对于每一种 Peano 曲线来说是固定的,连线时只需要根据 B 块的大小求出连线坐标即可。这种算法相对于基于递归嵌套运算的算法来说,极大地提高了效率。

4.6　算法推广

通过分析可以发现,Peano1 的初始图形的两个出口在一条水平线上,而 Peano2 和 Peano3 的初始图形不在一条水平线上。其实很容易证明如果初始图形在一条水平线上或者在一条垂直线上,就需要通过类似于 Peano1 的旋转才能得到"一笔画"的空间填充曲线;否则就不需要旋转,就可以采用类似于 Peano2 和 Peano3 的算法生成曲线。

因此,通过修改基本元素的形态还可以推广该算法,比如将 Peano1 中的"U"修改成"M";将 Peano2 中的"S"和反"S"改成"Z"和反"Z",或者改成"L"和反"L";将 Peano3 中的"N"改成"S"、"Z"或者"L"。而且不限于这些,可以生成形形色色的其他漂亮的图形。如果改变迭代规则会怎样呢?这是我们下一步的研究计划。我们已开发了空间填充曲线软件包,限于篇幅,不再赘述。

结语　这种非递归生成算法既能保持递归算法简单、清晰、易实现的优点,又能避免递归算法极耗内存的缺陷,对于类似 Peano 曲线的空间填充曲线的生成具有借鉴意义。

参 考 文 献

1　齐东旭.分形及其计算机生成.北京:科学出版社,1994

2　Song Zhexuan, Roussopoulos Nick. Using Hilbert curve in image storing and retrieving. In: Proc. of the 2000 ACM workshops on Multimedia. Los Angeles, California, United States, 2000

3　Böhm C, Berchtold S, Keim D A. Searching in high-dimensional spaces Index structures for improving the performance of multimedia databases. ACM Computing Surveys (CSUR), 2001, 33(3): 322～373

4　Sagan H. On the geometrization of the Peano curve and the arithmetization of the Hilbert curve. J. Math. Educ. Sci. Tech., 1992, 23(3): 403～411

5　McClure M. Self-similar structure in Hilbert's space-filling curve, Mathematics Magazine. Academic Research Library, 2003, 76(1): 40～47

6　Dickau RM. Two-dimensional L-systems, URL: http://forum.swarthmore.edu/advanced/robertd/lsys2d.html. 1996

7　Paul Bourke. URL: http://astronomy.swin.edu.au/～pbourke/fractals/peano/

8　王能超. Walsh 函数的演化生成. 中国图象图形学报, 1996, 1(3): 225～231