Manderbrot Set 的生成和探索

邵盛栋
信息与计算科学 3200103951

摘要　对于 Mandelbrot 集，它展示了一个精细且无限复杂的边界，随着放大倍数的增加，它会显示出越来越精细的递归细节，本文将在研究 Mandelbrot 集的基础上，首先探究 Mandelbrot 集的理论知识，寻找适当的算法并利用 C++ 语言编写程序，最终实现 Mandelbrot 集可视化。

1 引言

Mandelbrot 集是复平面中 c 值的集合，其临界点的轨道为 $z = 0$ 在二次映射的迭代下

$$z_{n+1} = z_n^2 + c$$

仍然有界。因此，复数 c 是 Mandelbrot 集中的元素，不妨假设开始时 $z_0 = 0$，在一次次迭代后 $|z_n|$ 对于任意 $n > 0$ 有界，则对应的 c 即属于 Mandelbrot 集。

例如，对于 $c = 1$，序列是 $0, 1, 2, 5, 26, \ldots$，趋于无穷，则说明 1 不是 Mandelbrot 集中的元素；另一方面，对于 $c = -1$，序列是 $0, -1, 0, -1, 0, \ldots$，是有界的，则说明-1 是 Mandelbrot 集中的元素 [?]。

2 问题背景

Mandelbrot 集起源于复动力学，这是 20 世纪初法国数学 Pierre Fatou 和 Gaston Julia 首次研究的领域。1978 年，罗伯特 · W · 布鲁克斯和彼得 · 马特尔斯基首次定义并绘制了这种分形，作为 Kleinian groups 研究的一部分。1980 年 3 月 1 日，Benoit Mandelbrot 在纽约约克镇高地的 IBM 托马斯 J. 沃森研究中心首次看到了该集合的可视化。

Mandelbrot 在 1980 年发表的一篇文章中研究了二次多项式的参数空间。Mandelbrot 集的数学研究真正始于数学家 Adrien Douady 和 John H. Hubbard (1985) 的工作，他们建立了许多其基本属性, 并命名该集合以纪念 Mandelbrot 在分形几何方面的有影响的工作。

1985 年 8 月的《科学美国人》的封面文章向广大读者介绍了计算 Mandelbrot 集的算法。封面由不来梅大学的 Peitgen 、Richter 和 Saupe 创作。Mandelbrot 集在 1980 年代中期作为计算机图形演示变得突出，当时个人计算机变得足够强大，可以以高分辨率绘制和显示该集。

3 数学理论

Mandelbrot 集是一个紧集，由于它是闭合的并且包含在以原点为圆心的半径为 2 的闭合圆中，即点 c 属于 Mandelbrot 集当且仅当 $|z_n| \leq 2$ 对于所有 $n \geq 0$ 有界 [?]，则有如下定理：

定理 1. *0 的轨道趋向于无穷大当且仅当在某个点它的模 > 2。*

该定理可分为以下两个定理：

定理 2. *若 $c \in M$，则 $|c| \leq 2$。*

定理 3. *若 $c \in M$，则 $|z_n| \leq 2, (n = 1, 2, \ldots)$。*

下面是对定理三的证明：

Proof. 要证明若 $|z_n| > 2, (n = 1, 2, \ldots)$，则 $c \notin M$
首先分别讨论 $|c| > 2$ 与 $|c| \leq 2$ 两种情形
由定理 2，$z_n > 2, (n = 1, 2, \ldots)$ 且 $|c| > 2$ 时，$c \notin M$.
接着要证明 $|c| \leq 2$ 时的情况：

假设 $|z_n|>2$，因为 $|c|\le 2$，所以 $|z_n|>|c|$，而

$$|z_{n+1}|=|z_n^2+c|\ge|z_n^2|-|c|=|z_n|^2-|c|$$

因为 $|z_n|>|c|$

$$|z_n|^2-|c|>|z_n|^2-|z_n|$$

由 $|z_n|>2$，左右同乘 $|z_n|$ 再减去 $|z_n|$ 可得到下式

$$|z_n|^2-|z_n|>2|z_n|-|z_n|=|z_n|$$

由以上可知 $|z_{n+1}|>|z_n|$
由数学归纳法可知 $2<|z_n|<|z_{n+1}|<|z_{n+2}|<\dots$，可以看出随着迭代次数的增加 $|z_n|$ 递增并且发散。所以在 $|z_n|>2,(n=1,2,\dots)$ 且 $|c|\le 2$ 的情况下也是则 $c\notin M$。
综上所述，若存在 $|z_n|>2$，则 $c\notin M$ □

该定理只适用于 $z\mapsto z^2+c$，但通过修改边界值 2，可以适用于其他多项式族。这里的边界值不取决于 c，但在其他多项式族中可能取决于 c[?]。

4 算法

算法伪代码如下：

```
Choose a maximal iteration number N
For each pixel p of the image:
  Let c be the complex number represented
  by p
  Let z be a complex variable
  Set z to 0
  Do the following N times:
     If |z|>2 then color the pixel white,
     end this loop prematurely, go to the
     next pixel Otherwise replace z by
     z*z+c
  If the loop above reached its natural
  end:
  color the pixel p in black
  Go to the next pixel
```

利用 C++ 实现算法的具体流程为：

(1) 创建 bmp 图像文件（通过 RGB 三种颜色显现的图像），图像中的像素可作为坐标轴当中的点，并给定原点坐标；

(2) 定义一种迭代形式，来检验对应的点 c 是发散还是收敛；

(3) 检验图像中的每一个点（像素），并对收敛和发散的点进行不同的着色处理，绘制形成目标图像。

5 数值算例

设置 ox，oy，dimension 与迭代次数 N 分别为 0,0,4,100，在 run 文件中输入如下命令

```
./test picture1.bmp 0 0 4 100
```

可得到 picture1，如图??所示。

我们可以调节 N 的大小来得到更加精细的 Mandelbrot 集的图案，不妨设置 N 的值为 200，即在 run 文件中输入命令：

```
./test picture2.bmp 0 0 4 200
```

可得到 picture2，如图??所示。

我们还可以通过调节 ox，oy，dimension 的值来观察 Mandelbrot 集的细节部分，在 run 中输入如下命令：

```
./test picture3.bmp 0 0.5 1 200
```

从而得到想要观察的位置的放大图，如图??所示。

最后，我们还可以改变图案的颜色，使其更加美观，可视化效果更好，如图??所示。

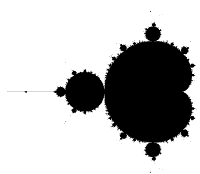

图 1: picture1

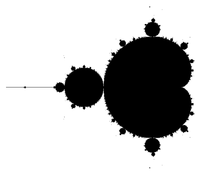

图 2: picture2

图 3: picture3

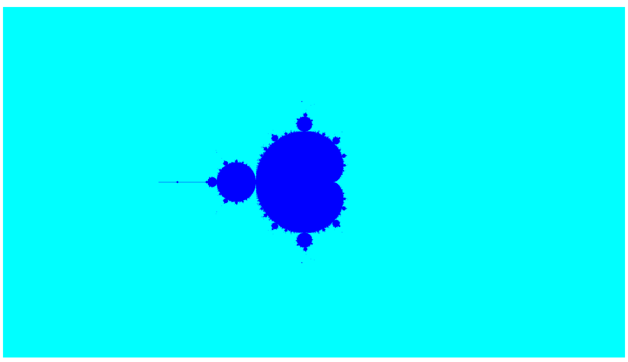

图 4: picture4

6 结论

Mandelbrot 集的迭代次数可间接反映发散程度，本文中的代码将收敛与发散两种不同情况进行区分，并赋予黑色与白色两种颜色的值，由此得到可视化效果效果。在此基础上，我们仍可以通过改变维数以及迭代次数，得到一些新的研究思路与方法。