计算物理作业 11

刘畅, PB09203226

2012 年 11 月 5 日

[**作业 11**]: 模拟二维 DLA 的生长过程, 如对原模型作变动则更佳.

1 算法

二维 DLA 的算法是这样的: 初始时刻, 一个粒子放在 lattice 的中心, 作为初始的 cluster. 算法的每一步都在一个半径为 r 的圆的随机位置释放一个粒子, 让它做随机游走, 如果碰到 cluster, 就把这个粒子加入这个 cluster. 如果跑出了半径为 $2r$ 的圆外 (或整个 lattice 的外面), 这个粒子就作废. 算法在成功地在 cluster 中增加了给定数量的粒子后结束.

2 程序

这个问题中, 和前面一样, 我们用一个二维 `bool` 型数组来表示整个 lattice. `true` 表示这个位置有粒子在 cluster 中, `false` 表示这个位置还没有被占用. 这个数组在堆上分配: (`main()`)

```
bool (*lattice)[CLUSTER_DIM] = (bool (*)[CLUSTER_DIM])
    malloc(CLUSTER_DIM * CLUSTER_DIM * sizeof(bool));
```

然后要将这个数组初始化成 `false`, 中心设成 `true` (`dla_simulation()`):

```
    /* initialize the cluster */
    for (i = 0; i < dim; i++)
        for (j = 0; j < dim; j++)
            lattice[i][j] = false;
    lattice[dim/2][dim/2] = true;
```

按照前面的算法, 接下来要在半径为 r 的圆上随机释放一个粒子. (`dla_simulation()`):

```
theta = rand_norm() * 2.0 * CONST_PI;
x = dim/2 + radius * cos(theta);
y = dim/2 + radius * sin(theta);
```

再进行随机游走 (`random_walk()`):

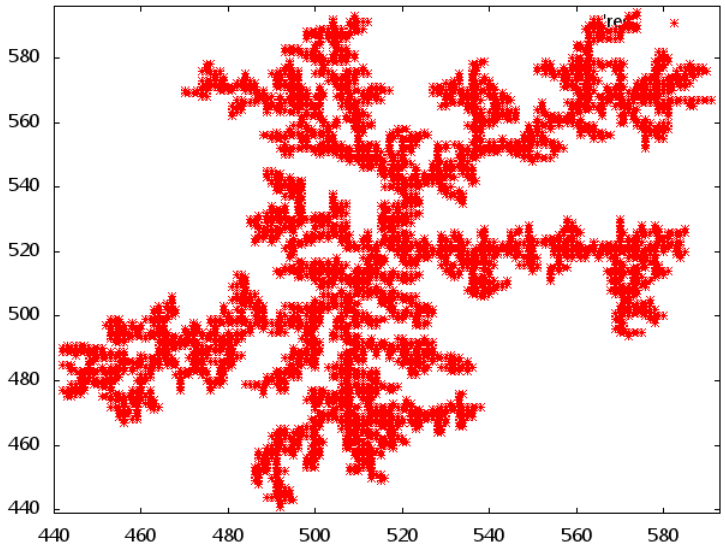
```
do {
    i = sample_unif_dir();
    x += delta_x[i];
    y += delta_y[i];
    if (lattice[x][y]) {
        *px = x - delta_x[i];
        *py = y - delta_y[i];
        return true;
    }
} while ( (0 <= x) && (x < dim) && (0 <= y) && (y < dim) &&
     (pow(x-dim/2,2) + pow(y-dim/2,2) < pow(max_radius,2)) );
return false;
```

上面的代码中, `sample_unif_dir()` 从 4 个方向中随机挑选一个, 然后将相应的 $\Delta\vec{r}$ 增加到 $\vec{r} = (x, y)$ 上. 如果新的位置被占据了, 旧的位置就被记录下来, 并且 (在 `dla_simulation()` 中) 把相应位置的 `lattice` 设置成 `true`. 否则, 就继续行走, 直到粒子离开半径为 `max_radius` 的圆, 或离开整个 lattice.

这个过程被反复执行 `nparticles` 次, 表明一共生长了 `nparticles` 个粒子. 代码在 `dla_simulation()`, `for` 循环那一行.

3 结果

我们将格点设成 1024×1024, 行走 4096 步 (见 `main()`). 将结果做成图:



可以看到和书上的结果是一致的.