Contents

[1.Giới thiệu 2](#_Toc375694219)

[1.1 Bài toán DLA: 2](#_Toc375694220)

[1.2 Các ứng dụng thực tế: 2](#_Toc375694221)

[1.3 Bài toán đặt ra : 2](#_Toc375694222)

[2.Lý thuyết 3](#_Toc375694223)

[2.1 Phát biểu định lý: 3](#_Toc375694224)

[2.2 Phương pháp SOR: 3](#_Toc375694225)

[2.3 Ứng dụng vào bài toán DLA: 4](#_Toc375694226)

[3.Cài đặt 5](#_Toc375694227)

[3.1 Cấu trúc dữ liệu: 5](#_Toc375694228)

[3.2 Cấu trúc chương trình: 5](#_Toc375694229)

[3.3 Vẽ đồ thị: 6](#_Toc375694230)

[4. Kết quả chạy thử: 7](#_Toc375694231)

[5.Kết luận : 8](#_Toc375694232)

# 1.Giới thiệu

## 1.1 Nhóm thực hành :

**Phạm Lê Hải – 2011.1486.**

**Lê Xuân Bộ - 2011.1182.**

**Vũ Văn Cương – 2011.1208.**

**Đề Tài : Mô Hình DLA.**

## 1.2 Bài toán DLA:

Diffusion-Limited Aggregation (DLA) là một quá trình, chuyên dùng để mô tả sự phát triển vô hướng một cách ngẫu nhiên của các cá thể.

## 1.3 Các ứng dụng thực tế:

DLA không chỉ được dùng để mô tả sự phát triển của các sinh vật sống (như sự phát triển của các loại virus, các loại tảo, các tế bào) mà còn có thể mô tả các quá trình vật lý khác ( như quá trình hòa tan, quá trình thẩm thấu v..v).

## 1.4 Bài toán đặt ra :

Trong rất nhiều ứng dụng của DLA, báo cáo này sẽ trình bày đến quá trình mô phỏng sự phát triển của cá thể sinh virus, trong một môi trường hữu hạn có chứa thức ăn, virus sẽ không ngừng ăn thức ăn và phát triển.

# 2.Lý thuyết

## 2.1 Phát biểu định lý:

Trong quá trình mô tả, vi khuẩn sẽ phát triển từ 1 tế bào ban đầu. Sự phát triển của vi khuẩn trong môi trường thức ăn có tính xác xuất phụ thuộc vào nồng độ thức ăn của những ô xung quanh nó.

Nồng độ thức ăn của môi trường cũng liên tục biến đổi, những ô có vi khuẩn sẽ bị ăn hết thức ăn, sự chênh lệch thức ăn sẽ bị thay đổi bởi phương trình lan truyền.

Phương trình lan truyền mô phỏng sự thay đổi của nồng độ thức ăn, vì sự thay đổi này rất nhanh so với tốc độ phát triển của virus nên, thời gian lan truyền có thể coi là rất nhỏ, ta sẽ sử dụng **phương trình lan truyền không phụ thuộc thời gian** để xác định nồng độ thức ăn.



D : Hệ số lan truyền.

C : Nồng độ.

Vì là phương trình không phụ thuộc vào thời gian nên, đạo hàm của nồng độ theo thời gian coi như bằng 0.



Phương trình trên có thể giải trực tiếp hoặc dùng phương pháp lặp. Phương pháp lặp Successive Over Relaxation sẽ được trình bày trong phần sau của báo cáo.

## 2.2 Phương pháp SOR:

Phương pháp lặp SOR sử dụng công thức sau để tính toán Ci,j:



Trong đó w là tham số hiệu chỉnh. Tham số w sẽ biểu hiện nồng độ tại 1 vị trí ở bước trước ảnh hưởng đến bước sau nhiều đến mức nào. Trong phương pháp SOR, ta sử dụng w trong khoảng (1,2).

Cụ thể, trong báo cáo, chúng tôi sẽ khảo sát với w=1.5

## 2.3 Ứng dụng vào bài toán DLA:

Thuật toán cơ bản để mô phỏng quá trình DLA như sau :

1.Giải phương trình lan truyền để thu được ma trận nồng độ thức ăn.

2.Để virus sinh sản( theo xác xuất).

3.Quay lại bước 1.

Cụ thể:

**Bước 1:** Ma trận nồng độ sẽ được sinh ra bởi phương pháp SOR.

**Bước 2:** Mỗi ô ở bên cạnh vi khuẩn sẽ có một xác xuất bị lây nhiễm, xác xuất đó tỉ lệ với nồng độ thức ăn ở đó.

Cụ thể :

Ảnh trên mô tả một giai đoạn trong quá trình phát triển virus, những ô đen là những ô đã lây nhiễm, những ô trắng là những ứng cử viên cho virus phát triển thêm.

Xác xuất virus phát triển ở mỗi ô trắng tính bằng:



Số mũ *η* định xác định hình dạng của khu chứa virus, thường được lấy từ 0.5 đến 2.

Trong báo cáo, *η* sẽ được khảo sát với vài giá trị khác nhau.

# 3.Cài đặt

Chương trình được cài đặt trên ngôn ngữ C, trình biên dịch GNU, qua công cụ Code::block trên môi trường windows.

## 3.1 Cấu trúc dữ liệu:

Virus được mô tả bởi kiểu **VirusPlace** trên C:

typedef struct **VirusPlace**{

long x;

long y;

} **VirusPlace**;

Trong đó, x và y lần lượt là tọa độ hàng, tọa độ cột của Virus.

Mảng **virus**[] chứa danh sách các virus đã xuất hiện, được khởi tạo với 1 virus duy nhất. Biến **nVirus** chứa số virus hiện có.

Mảng **candidate**[] chứa danh sách các ứng cử viên có thể lây nhiễm virus trong bước hiện tại, biến **nCandidate** chứa số lượng ứng cử viên.

Mảng **c**[][] biểu diễn thức ăn còn lại trong khu vực, được khởi tạo bằng 1, trừ 1 ô duy nhất chứa virus đầu tiên.

Mảng **chance**[] chứa danh sách các xác xuất của các ứng cử viên, số lượng cũng là **nCandidate**.

Mảng **grow**[][] dùng để đánh dấu vị trí của các virus.

## 3.2 Cấu trúc chương trình:

Chương trình biểu diễn 3 bước của thuật toán DLA gồm:

**Vào ra:**

* Hàm **readFile**(): Đọc dữ liệu đầu vào.
* Hàm **writeFile**(): Ghi ma trận C[][] ra file ‘output.txt’

**Tính toán:**

* Hàm **addVirus**(u,v,x): Đánh dấu có virus ở ô (u,v), đó là virus sinh ra từ ứng cử viên candidate[x]. Đồng thời thêm các ô xung quanh ô (u,v) vào mảng ứng cử viên.
* **Hàm init():** Khởi tạo mảng **Grow**[][]=false. Khởi tạo Ci,j=1. Khởi tạo virus đầu tiên bằng **addVirus**().
* **Hàm sor():** Thực hiện phương pháp lặp SOR để tính cả ma trận C[][].
* **Hàm eat():** Đưa các ô Ci,j có virus về 0.
* **Hàm computeProbality():** Tính xác suất của các ứng cử viên.
* **Hàm growth():** Sinh ngẫu nhiên các virus.
* **Hàm solve():** Chừng nào chưa đủ số virus còn lặp : **sor**() 🡪**eat**() 🡪 **computeProbality**() 🡪 **growth**().
* **Hàm main():** Đọc dữ liệu, gọi hàm **solve**() rồi in dữ liệu.

## 3.3 Vẽ đồ thị:

Ma trận kết quả trong file ‘output.txt’.

Trong matlab ta đọc ma trận vào biến x bằng lệnh:

x=dlmread(‘output.txt’).

Đồ thị vẽ bằng lệnh

Imagesc(x).

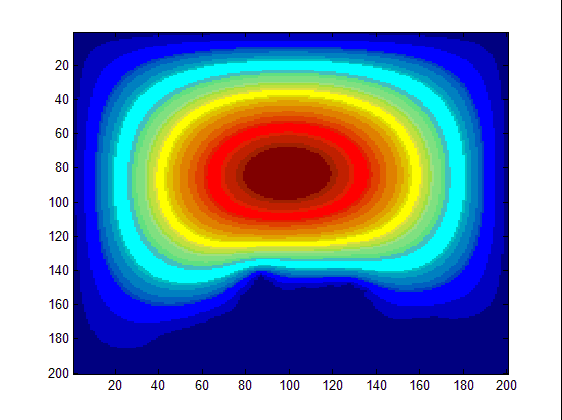
# 4. Kết quả chạy thử:

Ta thả virus đầu tiên ở đáy dưới của màn hình. Chạy với các kích thước khác nhau:

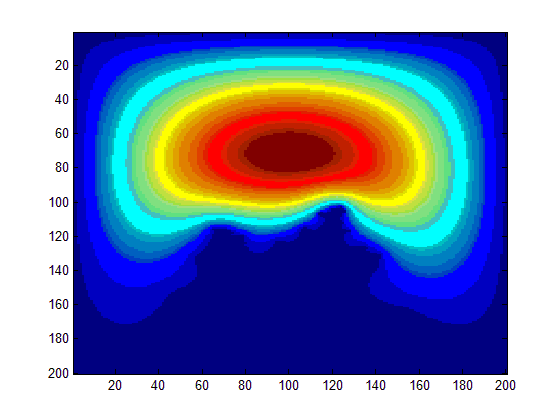
Vị trí màu càng đỏ là còn càng nhiều thức ăn, vị trí màu đỏ nhạt hoặc vàng là vùng có ít thức ăn , vị trí xanh thẫm nhất là khu vực virus đã xuất hiện.

Chạy với kích thước 200x200 sinh sản ra 10000 virus.

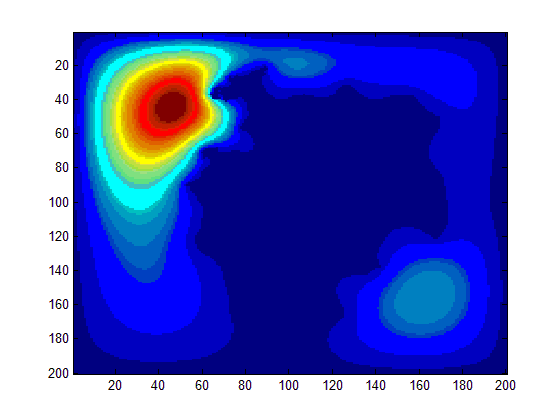
* Với số mũ *η =* 0 :



* Với số mũ *η =* 1 :

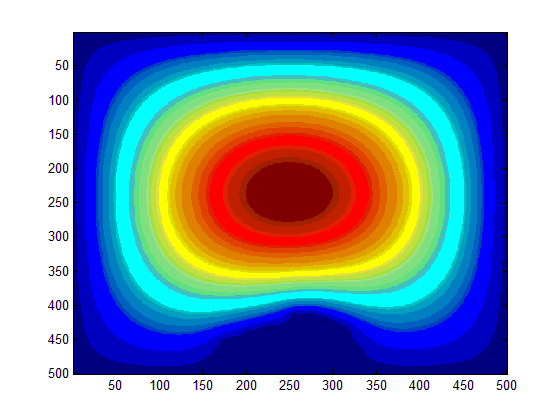


* Với số mũ *η =* 2 :

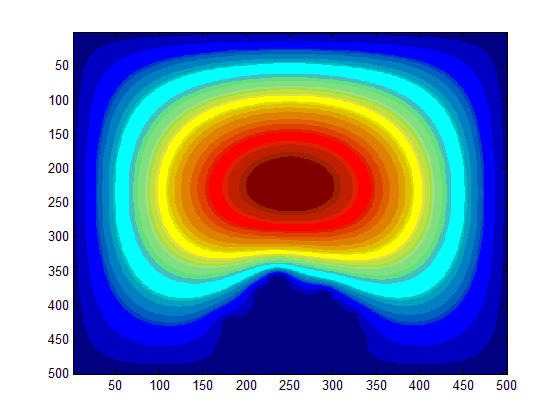


Chạy với kích thước 500x500:

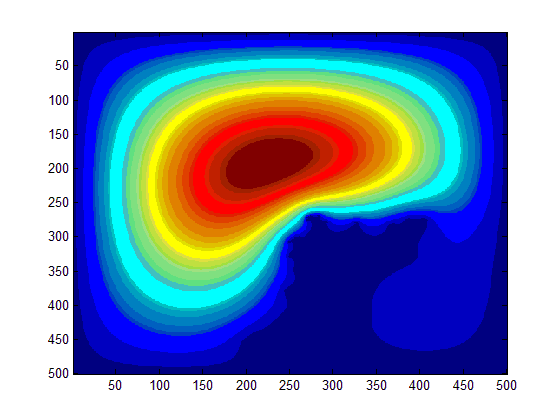
* Với số mũ *η =* 0 :



* Với số mũ *η =* 1 :

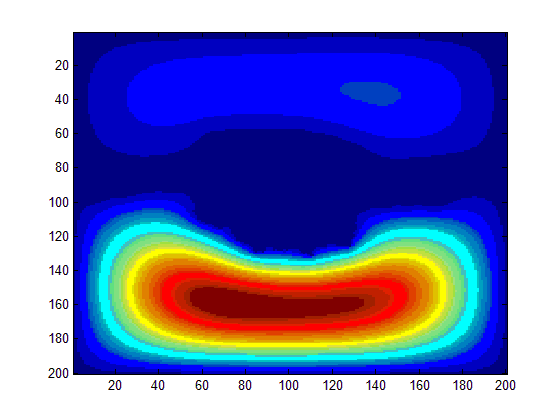


* Với số mũ *η =* 2 :

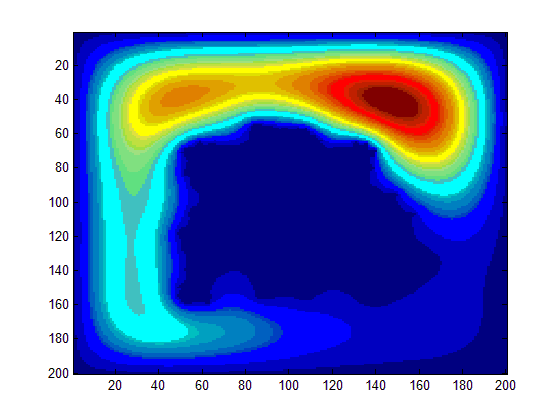


Tương tự nhưng cho virus phát triển từ giữa màn hình:

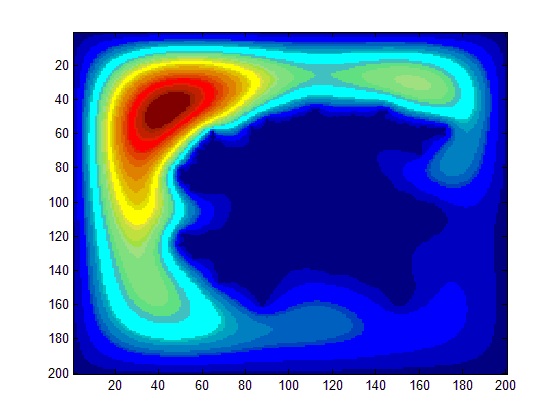
* Với số mũ *η =* 0 :



* Với số mũ *η =* 1 :



* Với số mũ *η =* 2 :



**Nhận xét:**

Số mũ *η* có ảnh hưởng đến sự phát triển của virus.

Với *η=0* , xác suất xuất hiện virus mới sẽ giống nhau với mọi vị trí, nên hình vẽ có tính cân xứng.

Với *η=1* , xác suất xuất hiện virus mới tỉ lệ thuận với nồng độ thức ăn có sẵn tại ô đó, điều này tương đối hợp lý trong tự nhiên. Hình vẽ thường chỉ xuất hiện 1 số nhánh phát triển.

Với *η=2 ,* xác suất tỉ lệ với bình phương nồng độ thức ăn có sẵn, vì thế, nơi nào có nồng độ thức ăn cao hơn sẽ có xác xuất virus mới nhiều hơn rất nhiều. Do đó, virus phát triển với rất ít nhánh rẽ.

# 5.Kết luận :

Báo cáo đã trình bày về nguyên tắc và thuật toán của phương pháp mô phỏng DLA, đồng thời cũng trình bày về các kết quả chạy thử trực tiếp cũng như một số đánh giá, kết luận của nhóm.

Trong báo cáo có thể còn sơ xuất, mong thầy giáo nhắc nhở, góp ý thêm.

Chúc thầy sức khỏe.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*