Tiểu luận cá nhân

HÀM SINH XÁC SUẤT

(PROBABILITY GENERATING FUNCTIONS)

**Tóm tắt**

**Từ khóa**

**1. Giới thiệu**

Hàm sinh

Hàm sinh là một trong những sáng tạo thần tình, bất ngờ, nhiều ứng dụng của toán rời rạc. Nói một cách nôm na, hàm sinh chuyển những bài toán về *dãy số* thành những bài toán về *hàm số*. Điều này là rất tuyệt vời vì chúng ta đã có trong tay cả một cỗ máy lớn để làm việc với các hàm số. Nhờ vào hàm sinh, chúng ta có thể áp dụng cỗ máy này vào các bài toán dãy số. Bằng cách này, chúng ta có thể sử dụng hàm sinh trong việc giải tất cả các dạng toán về phép đếm. Có cả một ngành toán học lớn nghiên cứu về hàm sinh, vì thế, trong bài này, chúng ta chỉ tìm hiểu những vấn đề căn bản nhất về chủ đề này.

Trong bài viết này, các dãy số sẽ được để trong ngoặc < > để phân biệt với các đối tượng toán học khác.

**2. Hàm sinh**

## I Tổng quan hàm sinh

Hàm sinh được sử dụng phổ biến trong toán học, và đóng một vai trò quan trọng trong lý thuyết xác suất. Xét chuỗi {ai: i = 0; 1; 2; :::} thuộc tập số thực : là một trong những dãy số đã được phân tích trong một vài hàm sinh. Hàm sinh phổ dụng của chuỗi được định nghĩa như sau :

G(s) =

với những giá trị của các tham số s mà tổng hội tụ. Đối với một chuỗi được xét, tồn tại một bán kính hội tụ R( ≥ 0) như tổng hội tụ tuyệt đối nếu |s| < R và phân kỳ nếu |s| < R.

Với nhiều chuỗi được định nghĩa hoàn chỉnh, G(s) có thể được viết trong hình thức kín, và những số cá nhân trong chuỗi có thể được phục hồi hoặc mở rộng bằng cách dẫn xuất.

**1. Hàm sinh**

*Hàm sinh thường* của dãy số vô hạng <g0, g1, g2, g3,…> là chuỗi luỹ thừa hình thức

G(x) = g0 + g1x + g2x2 + g3x3 …

Ta gọi làm sinh là chuỗi hình thức bởi vì thông thường ta sẽ chỉ coi x là một ký hiệu thay thế thay vì một số. Chỉ trong một vài trường hợp ta sẽ cho x nhận các giá trị thực, vì thế ta gần như cũng không để ý đến sự hội tụ của các chuỗi. Có một số loại hàm sinh khác nhưng trong bài này, ta sẽ chỉ xét đến hàm sinh thường.

Trong bài này, ta sẽ ký hiệu sự tương ứng giữa một dãy số và hàm sinh bằng dấu mũi tên hai chiều như sau

<g0, g1, g2, g3, …> ↔ g0 + g1x + g2x2 + g3x3 +…

Ví dụ, dưới đây là một số dãy số và hàm sinh của chúng

<0, 0, 0, 0, …> ↔ 0 + 0.x + 0.x2 + 0.x3 + … = 0

<1, 0, 0, 0, …> ↔ 1 + 0.x + 0.x2 + 0.x3 + … = 1

<3, 2, 1, 0, …> ↔ 3 + 2x + x2 + 0.x3 + … = x2 + 2x + 3

Quy tắc ở đây rất đơn giản: Số hạng thứ i của dãy số (đánh số từ 0) là hệ số của xi trong hàm sinh.

Nhắc lại công thức tính tổng của các số nhân lùi vô hạn là



Đẳng thức này không đúng với |z| ≥ 1, nhưng một lần nữa ta không quan tâm đến vấn đề hội tụ. Công thức này cho chúng ta công thức tường minh cho hàm sinh của hàng loạt các dãy số

<1, 1, 1, 1, …> ↔ 1 + x + x2 + x3 + … = 1/(1-x)

<1, -1, 1, -1, …> ↔ 1 - x + x2 - x3 + … = 1/(1+x)

<1, a, a2, a3, …> ↔ 1 + ax + a2x2 + a3x3 + … = 1/(1-ax)

<1, 0, 1, 0, 1, 0, ...> ↔ 1 + x2 + x4 + … = 1/(1-x2)

**Các phép toán trên hàm sinh**

Phép màu của hàm sinh nằm ở chỗ ta có thể chuyển các phép toán thực hiện trên dãy số thành các phép toán thực hiện trên các hàm sinh tương ứng của chúng. Chúng ta cùng xem xét các phép toán và các tác động của chúng trong thuật ngữ dãy số.

**2.1. Nhân với hằng số**

Khi nhân hàm sinh với một hằng số thì trong dãy số tương ứng, các số hạng sẽ được nhân với hằng số đó. Ví dụ

<1, 0, 1, 0, 1, 0, ...> ↔ 1 + x2 + x4 + … = 1/(1-x2)

Nhân hàm sinh với 2, ta được

2/(1-x2) = 2 + 2x2 + 2x4 + …

là hàm sinh của dãy số

<2, 0, 2, 0, 2, 0, …>

**Quy tắc 1.** (Quy tắc nhân với hằng số)

Nếu <f0, f1, f2, f3, …> ↔ F(x) thì <cf0, cf1, cf2, cf3, …> ↔ cF(x)

Chứng minh.

<cf0, cf1, cf2, cf3, …> ↔ cf0 + (cf1)x + (cf2)x2 + (cf3)x3 + …

= c(f0 + f1x+f2x2 + f3x3 + …)

= cF(x).

**2.2. Cộng**

Cộng hai hàm sinh tương ứng với việc cộng các số hạng của dãy số theo đúng chỉ số. Ví dụ, ta cộng hai dãy số trước đó

<1, 1, 1, 1, …> ↔ 1/(1-x)

+ <1, -1, 1, -1, …> ↔ 1/(1+x)

<2, 0, 2, 0, …> ↔ 1/(1-x) + 1/(1+x)

Bây giờ ta thu được hai biểu thức khác nhau cùng sinh ra dãy (2, 0, 2, 0, …). Nhưng điều này không có gì ngạc nhiên vì thực ra chúng bằng nhau:

1/(1-x) + 1/(1+x) = [(1+x) + (1-x)]/(1-x)(1+x) = 2/(1-x2)

**Quy tắc 2.** (Quy tắc cộng)

Nếu <f0, f1, f2, …> ↔ F(x), <g0, g1, g2, …> ↔ G(x)

thì <f0+g0, f1+g1, f2+g2, …> ↔ F(x) + G(x)

Chứng minh.

<f0+g0, f1+g1, f2+g2, …> ↔ f0+g0+ (f1+g1)x + (f2+g2)x2 + …

= (f0 + f1x + f2x2 + …) + (g0 + g1x + g2x2 + …)

= F(x) + G(x)

**2.3. Dịch chuyển sang phải**

Ta bắt đầu từ một dãy số đơn giản và hàm sinh của nó

<1, 1, 1, 1, …> ↔ 1/(1-x)

Bây giờ ta *dịch chuyển* dãy số *sang phải* bằng cách thêm k số 0 vào đầu

<0, 0, …, 0, 1, 1, 1, …> ↔ xk + xk+1 + xk+2 + …

= xk(1+x+x2 + …)

= xk/(1-x)

Như vậy, thêm k số 0 vào đầu dãy số tương ứng với việc nhân hàm sinh với xk. Điều này cũng đúng trong trường hợp tổng quát.

**Quy tắc 3.** (Quy tắc dịch chuyển phải)

Nếu <f0, f1, f2, …> ↔ F(x)

thì <0, …, 0, f0, f1, f2, …> ↔ xk.F(x) (có k số 0)

Chứng minh.

<0, …, 0, f0, f1, f2, …> ↔ f0xk + f1xk+1 + f2xk+2 + …

= xk(f0 + f1x + f2x2 + …)

= xkF(x)

**2.4. Đạo hàm**

Điều gì sẽ xảy ra nếu ta lấy đạo hàm của hàm sinh? Chúng ta hãy bắt đầu từ việc lấy đạo hàm của một hàm sinh đã trở nên quen thuộc của dãy số toàn 1:



Ta tìm được hàm sinh cho dãy số <1, 2, 3, 4, …> !

Tổng quát, việc lấy đạo hàm của hàm sinh có hai tác động lên dãy số tương ứng: các số hạng được nhân với chỉ số và toàn bộ dãy số được dịch chuyển trái sang 1 vị trí.

**Quy tắc 4.** (Quy tắc đạo hàm)

Nếu <f0, f1, f2, …> ↔ F(x)

thì <f1, 2f2, 3f3, ..> ↔ dF(x)/dx

Chứng minh.

<f1, 2f2, 3f3, ..> ↔ f1 + 2f2x + 3f3x2 + …

= (d/dx)(f0 + f1x + f2x2 + f3x3 + …)

= dF(x)/dx

Quy tắc đạo hàm là một quy tắc rất hữu hiệu. Trong thực tế, ta thường xuyên cần đến một trong hai tác động của phép đạo hàm, nhân số hạng với chỉ số và dịch chuyển sang trái. Một cách điển hình, ta chỉ muốn có một tác động và tìm cách “vô hiệu hoá” tác động còn lại. Ví dụ, ta thử tìm hàm sinh cho dãy số <0, 1, 4, 9, 16, …>. Nếu ta có thể bắt đầu từ dãy <1, 1, 1, 1, …> thì bằng cách nhân với chỉ số 2 lần, ta sẽ được kết quả mong muốn

<0.0, 1.1, 2.2, 3.3, …> = <0, 1, 4, 9, …>

Vấn đề là ở chỗ phép đạo hàm không chỉ nhân số hạng dãy số với chỉ số mà còn dịch chuyển sang trái 1 vị trí. Thế nhưng, quy tắc 3 dịch chuyển phải cho chúng ta cách để vô hiệu hoá tác động này: nhân hàm sinh thu được cho x.

Như vậy cách làm của chúng ta là bắt đầu từ dãy số <1, 1, 1, 1, …>, lấy đạo hàm, nhân với x, lấy đạo hàm rồi lại nhân với x.

<1, 1, 1, 1, …> ↔ 1/(1-x)

<1, 2, 3, 4, …> ↔ (d/dx)(1/(1-x)) = 1/(1-x)2

<0, 1, 2, 3, 4, …> ↔ x/(1-x)2

<1, 4, 9, 16, …> ↔ (d/dx)( x/(1-x)2) = (1+x)/(1-x)3

<0, 1, 4, 9, 16, …> ↔ x(1+x)/(1-x)3

Như vậy hàm sinh cho dãy các bình phương là x(1+x)/(1-x)3.

**6. Các hàm sinh thường gặp**

**6.1. Định lý nhị thức mở rộng.**

Với u là một số thực và k là số nguyên không âm. Lúc đó hệ số nhị thức mở rộng  được định nghĩa như sau



**Định lý 2.** Cho x là số thực với |x| < 1 và u là một số thực. Lúc đó



Định lý này có thể được chứng minh khá dễ dàng bằng cách sử dụng định lý Taylor.

Ví dụ. Tìm khai triển luỹ thừa của các hàm sinh (1+x)-n và (1-x)-n

Giải: Theo định lý nhị thức mở rộng, có thể suy ra



Theo định nghĩa



Từ đó



Thay x bằng –x, ta được



Ví dụ. Tìm khai triển luỹ thừa của (1-x)-1/2

Giải: Theo định lý nhị thức mở rộng, ta có



Theo định nghĩa



Từ đó



Thay x bằng –x, ta được



**6.2. Bảng các hàm sinh thường gặp**

|  |  |  |
| --- | --- | --- |
| **Hàm số** | **Khai triển luỹ thừa** | **ak** |
| 1/(1-x) | 1 + x + x2 + x3 + … | 1 |
| 1/(1+x) | 1 – x + x2 – x3 + … | (-1)k |
| 1/(1-ax) | 1 + ax + a2x2 + a3x3 + … | ak |
| (1-xn+1)/(1-x) | 1 + x + x2 + …+ xn | 1 nếu k ≤ n, 0 nếu k > n |
| (1+x)n |  |  |
| 1/(1-x)n |  |  |
| 1/(1-x)2 | 1 + 2x + 3x2 + 4x3 + … | k+1 |
| 1/(1-ax)2 | 1 + 2ax + 3a2x2 + 4a3x3 + … | (k+1)ak |
| 1/(1-xr) | 1 + xr + x2r + x3r + … | 1 nếu r | k và 0 trong trường hợp ngược lại |
| 1/(1+xr) | 1 - xr + x2r - x3r + … | (-1)s nếu k=sr và 0 trong trường hợp ngược lại |
| ln(1+x) | x – x2/2 + x3/3 – x4/4 + … | 0 khi k = 0 và (-1)k/k |
| ln(1-x) | - x – x2/2 – x3/3 – x4/4 – … | 0 khi k = 0 và -1/k |
| arctgx | x + x3/3 + x5/5 + … | 0 với k chẵn và  1/k với k lẻ |

**3. Hàm sinh xác suất**

## II Định nghĩa và thuộc tính

Xem xét một số X riêng biệt lấy giá trị không âm. Ta viết

pk = P(X=k), k = 0,1,2,…

(nếu X là một số hữu hạn, ta chỉ cần thêm vào những xác suất bằng không ứng với các giá trị không xảy ra). Hàm sinh xác suất (PGF) của X được định nghĩa như sau

GX(s) = = E(sX)

Chú ý rằng GX(1) = 1, vì vậy dãy số hội tụ hoàn toàn cho |s| ≤ 1. Cũng như GX(0) = p0. Đối với một số phân phối phổ biến hơn, hàm sinh xác suất như sau :

*(i) Hằng số – nếu pc  = 1, pk = 0, k ≠ c*, ta có

GX(s) = E(sX) = sc

*(ii) Dãy Bernoulli – nếu p1 = p, p0 = 1 – p = q , pk = 0, k ≠ 0 hoặc 1*, ta có

GX(s) = E(sX) = q + ps

*(iii) Geometric – nếu pk = pqk-1, k = 1,2,…; q = 1-p*, ta có

*GX(s) = nếu |s| < q-1*

*(iv) Binomial – nếu X ̴ Bin(n,p)*, ta có

*GX(s) = (q + ps)n, (q = 1 - p)*

*(v) Poisson – nếu X ̴ Poisson(λ)*, ta có

*GX(s) = = e λ(s-1)*

*(vi) Negative binomial – nếu X ̴ NegBin(n,p)*, ta có

GX(s) = = nếu |s| < q-1 và p + q = 1

***Định lý khẳng định :***

Nếu X và Y có hàm sinh xác suất tương ứng GX và GY, ta có

*GX(s) = GY(s) với mọi s (a)*

*Nếu và chỉ nếu P(X = k) = P(Y = k) với k = 0,1,… (b)*

*Chú ý :* nếu và chỉ nếu X và Y có cùng phân phối xác suất.

**Chứng minh :** Ta cần chứng minh rằng (a) ứng dụng cho (b). Bán kính hội tụ của GX và GY thì ≥ 1, do đó, chúng có thể mở rộng chuỗi về gốc

GX(s) = P(X = k)

GY(s) = P(Y = k)

Nếu GX = GY, có 2 dãy số có hệ số giống nhau.

Ví dụ : Nếu X có hàm sinh xác suất với q = 1 – p, ta có thể kết luận rằng :

X ̴ Geometric(p)

Cho hàm A(s) với một hàm sinh xác suất của một số X, ta có thể được pk = P(X = k)

bằng cách triển khai A(s) trong chuỗi số trong s và đặt

pk = hệ số của sk;

hoặc cách khác A(s) k lần có liên quan đến s và đặt s = 0

Chúng ta có thể mở rộng định nghĩa của hàm sinh xác suất với hàm của X. Hàm sinh xác suất của Y = H(X) là

GY(s) = GH(X)(s) = E(sH(X)) =

Nếu H khá đơn giản, nó thể hiện GY(s) trong điều kiện của GX(s).

Ví dụ : Đặt Y = a + bX thì

GY(s) = E(sY) = E(sa+bX)

= saE[(sb)X] = saGX(sb)

## III Tóm lược

**Lý thuyết**

Lấy X là số đếm và số thứ r phát sinh của chính hàm sinh xác xuất GX(s) khi s = 1, thì

= E[X(X – 1)…(X – r + 1)]

***Chứng minh***

(s) = [GX(s)]

= []

=

(giả sử không mâu thuẫn khi đổi chỗ và ). Chuỗi hội tụ với |s| ≤ 1, vì thế :

(1) = E[X(X - 1)…(X – r + 1)], r ≥ 1

Ngoài ra :

(1) (hoặc (1)) = E(X)

Và

(1) (hoặc (1)) = E[X(X - 1)]

= E(X2) – E(X)

= Var(X) + [E(X)]2 – E(X)

Vì thế

Var(X) = (1) – [(1)2] + (1).

Ví dụ : Nếu X ̴ Poiss(λ), thì

GX(s) = e λ(s-1);

(s) = λeλ(s-1)

* E(X) = (1) = λe0 = λ.

(s) = λ2eλ(s-1)

* Var(X) = λ2 – λ2 + λ = λ

## IV Tổng của các biến ngẫu nhiên độc lập

**Lý thuyết**

Đặt X và Y là biến đếm độc lập, hàm sinh xác suất tương ứng với GX(s) và GY(s) và cho Z = X + Y.

GZ(s) = GX+Y(s) = GX(s)GY(s)

***Chứng minh***

GZ(s) = E(sZ) = E(sX+Y)

= E(sX)E(sY) (độc lập)

= GX(s)GY(s)

**Hệ quả**

Nếu X1, … ,Xn là các biến đếm độc lập, tương ứng với các hàm sinh xác suất GX1(s), … ,GXn(s) (và n là số nguyên cho trước),

GX1+ … + Xn(s) = GX1(s) … GXn(s)

**Ví dụ 1:**

Tìm phân phối của n số độc lập Xi, i = 1, … n, khi Xi ̴ Poisson(λi)

**Giải**

GXi(s) = e λi(s-1).

* GX1 + X2 + … + Xn(s) = λi(s-1)

= e(λ1+ … + λn)(s-1).

Đây là hàm sinh xác suất Poisson

**Ví dụ 2:**

Trong chuỗi số độc lập n Bernoulli

Đặt X = = số lượng đúng trong n lần. Tìm phân phối xác suất của X ?

**Giải:** Từ những lần thử độc lập nhau, I1, … In độc lập.

GX(s) = GI1, GI2, … GIn(s).

Nhưng GIi(s) = q + ps, I = 1, … n.

GX(s) = (q + ps)n =

P(X=x) = hệ số của sx trong GX(s)

= p+qn-x, x = 0, … , n

X ̴ Bin(n, p).

## V Tổng của số ngẫu nhiên độc lập

**Lý thuyết**

Cho N, X1, X2, … là các số đếm độc lập. Nếu tập {Xi} có phân phối giống nhau, với mỗi hàm sinh xác suất GX.

Có hàm sinh xác suất

SN = X1 + … + XN

GSN = GN(GX(s))

***Chứng minh :*** Ta có

(s) = E()

= (điều kiện với N)

=

=

= (hệ quả đã chứng minh trước đó)

= GN(GX(s)) (theo định nghĩa của GN)

**Hệ quả**

1) E(SN) = E(N) . E(X)

Chứng minh

[(s)] = [GN(GX(s))]

= . khi u = GX(s)

Đặt s = 1, (vì vậy u = GX(1) = 1), ta có

E(SN) = [] . [] = E(N) . E(X)

Tương tự ta có thể suy luận rằng

Var(SN) = E(N)Var(X) + Var(N) [E(X)]2 (3.19)

## VII Quá trình phân nhánh

### 1. Định nghĩa

Xét giả thuyết một cá thể sinh ra trong một điểm thời gian và chết đi trong quá trình sinh ra các cá thể mới. Ta cho rằng

*i) Kích thước của quần thể là độc lập, mỗi cá thể mang giá trị 0,1,2…*

*ii) Kích thước của quần thể được mô tả, các cá thể con trong quần thể, C, được xây dựng*

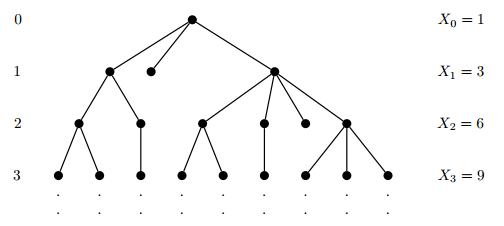
P(C = k) = pk, k = 0,1,2…

Sự phát triển cộng đồng quần thể trong quá trình thời gian, được gọi là quá trình phân nhánh : cung cấp một mô hình phát triển đơn giản, tạo ra mối quan hệ của các quần thể.

Đặt Xn = số cá thể sinh ra trong thời gian n

Sự phát triển của quần thể được mô tả bằng chuỗi X0, X1, X2,… Ta cho X0 = 1, bắt đầu với một cá thể xác định

Cây quần thể được thể hiện như sau :



Ta có thể dùng hàm sinh xác suất để duyệt quá trình này

### 2. Sự phát triển của quần thể

Đặt G(s) là hàm sinh xác suất của C

E:\q5.png

Và Gn(s) là hàm sinh xác suất của Xn

E:\q5.png

G0(s) = s, P(X0 = 1) = 1; P(X0 = x) = 0 với x ≠ 1 và G1(s) = G(s)

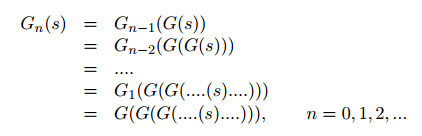
Xn = C1 + C2 + … + Cxn-1

Những điểm Ci là kích thước của quần thể được sản sinh bởi thế hệ thứ i thành viên của thế hệ thứ (n – 1)

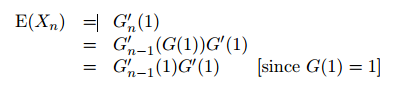
Vì thế, Xn là tổng của số ngẫu nhiên không phụ thuộc và được xây dựng giống nhau, ta có

Gn(s) = Gn-1(G(s)) với n = 2,3…

Và điều này cũng đúng cho n = 1. Lặp lại kết quả này, ta có

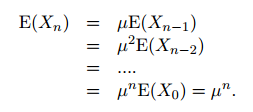


Gn là quần thể thứ n của G

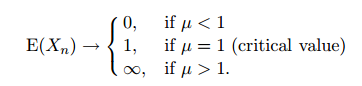


E(Xn) = E(Xn-1)ὴ

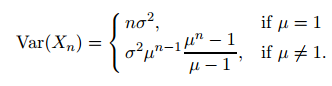
ὴ = E(C ) là ý nghĩa kích thước quần thể. Do đó



Có các suy luận



Tương tự



**Ví dụ 1 :**

Kiểm tra một phân nhánh C được định nghĩa phân phối hình học

pk = pqk, k = 0,1,2,…; 0 < p – 1 < 1, với p ≠ q

**Giải:** hàm sinh xác suất của C là

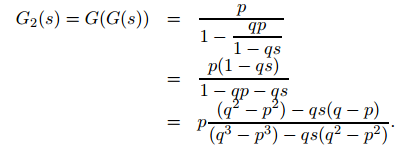
E:\q2.png

Ta cần giải quyết hàm Gn(s) = Gn-1(G(s))

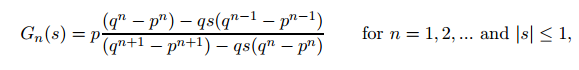
Đầu tiên, nếu |s| ≤ 1,

E:\q2.png

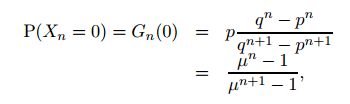
Sau đó



Ta phỏng đoán rằng



Và đây là kết quả phát triển từ n



### 3. Xác suất tuyệt chủng

Xác suất tiến trình tuyệt chủng thứ n được mô tả

En = P(Xn = 0)

en ≤ 1 và en ≤ en+1. {en} là một dãy đơn điệu

E:\q2.png

Gọi là xác suất tuyệt chủng cuối cùng

**Định lý 1**

e là gốc không âm nhỏ nhất của công thức x = G(x)

**Định lý 2**

e = 1 nếu và chỉ nếu ὴ ≤ 1

**4. Ứng dụng**

## VI Dùng hàm sinh để giải quyết quan hệ tái phát sinh

Thay vì sử dụng lý thuyết của quan hệ từ khi giải quyết các mối quan hệ tái phát sinh trong quá trình giải quyết bằng các điều kiện, người ta thường chuyển đổi các mối quan hệ trong phương trình cho hàm sinh, để được giải quyết mục tiêu theo các điều kiện ràng buộc thích hợp.

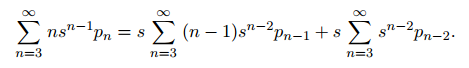
**Ví dụ 1:**

E:\q1.png

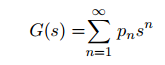
Bắt đầu với xác suất pn. Giải quyết vấn đề này bằng hàm sinh xác suất

**Giải**

Bằng nsn-1 và cộng tất cả các giá trị của n ta có được



Hàm sinh



(đây không là hàm sinh xác suất, khi {pn : n ≥ 1} không tạo ra một xác suất) đó là

E:\q4.png

Bây giờ phương trình này được giải quyết theo các điều kiện ràng buộc

G(0) = 0

Kết quả là

E:\q5.png

Giải G(s) theo chuỗi số trong s, và phân rã của sn

E:\q5.png

**4. Đếm bằng hàm sinh**

Hàm sinh có thể được áp dụng trong các bài toán đếm. Nói riêng, các bài toán chọn các phần tử từ một tập hợp thông thường sẽ dẫn đến các hàm sinh. Khi hàm sinh được áp dụng theo cách này, hệ số của xn chính là số cách chọn n phần tử.

**4.1. Chọn các phần tử khác nhau**

Hàm sinh cho dãy các hệ số nhị thức được suy ra trực tiếp từ định lý nhị thức



Như vậy hệ số của xn trong (1+x)k bằng số cách chọn n phần tử phân biệt từ một tập hợp gồm k phần tử. Ví dụ, hệ số của x2 là C2k, số cách chọn 2 phần tử từ tập hợp k phần tử. Tương tự, hệ số của xk+1 là số cách chọn k+1 phần tử từ tập hợp k phần tử và như thế, bằng 0.

**4.2. Xây dựng các hàm sinh để đếm**

Thông thường ta có thể dịch mô tả của bài toán đếm thẳng sang ngôn ngữ hàm sinh để giải. Ví dụ, ta có thể chứng tỏ rằng (1+x)k sẽ sinh ra số các cách chọn n phần tử phân biệt từ tập hợp k phần tử mà không cần dùng đến định lý nhị thức hay các hệ số nhị thức!

Ta làm như sau. Đầu tiên, ta hãy xét tập hợp có một phần tử {a1}. Hàm sinh cho số cách chọn n phần tử từ tập hợp này đơn giản là 1 + x. Ta có 1 cách chọn không phần tử nào, 1 cách chọn 1 phần tử và 0 cách chọn hai phần tử trở lên. Tương tự, số cách chọn n phần tử từ tập hợp {a2} cũng cho bởi hàm sinh 1 + x. Sự khác biệt của các phần tử trong hai trường hợp trên là không quan trọng.

Và bây giờ là ý tưởng chính: *hàm sinh cho số cách chọn các phần tử từ hợp của hai tập hợp bằng tích các hàm sinh cho số cách chọn các phần tử từ mỗi tập hợp.* Chúng ta sẽ giải thích chặt chẽ điều này, nhưng trước hết, hãy xem xét một ví dụ. Theo nguyên lý này, hàm sinh cho số cách chọn các phần tử từ tập hợp {a1, a2} là

(1+x). (1+x) = (1+x)2 = 1 + 2x + x2

Có thể kiểm chứng rằng đối với tập hợp {a1, a2} ta có 1 cách chọn 0 phần tử, 2 cách chọn 1 phần tử, 1 cách chọn 2 phần tử và 0 cách chọn 3 phần tử trở lên.

Tiếp tục áp dụng quy tắc này, ta sẽ được hàm sinh cho số cách chọn n phần tử từ tập hợp k phần tử

(1+x).(1+x)…(1+x) = (1+x)k

Đây chính là công thức hàm sinh mà ta đã nhận được bằng cách sử dụng định lý nhị thức. Nhưng lần này, chúng ta đã đi thẳng từ bài toán đếm đến hàm sinh.

Chúng ta có thể mở rộng điều này thành một nguyên lý tổng quát.

**Quy tắc 5** (Quy tắc xoắn). Gọi A(x) là hàm sinh cho cách chọn các phần tử từ tập hợp A và B(x) là hàm sinh cho cách chọn các phần tử từ tập hợp B. Nếu A và B là rời nhau thì hàm sinh cho cách chọn các phần tử từ A ∪ B là A(x).B(x).

Quy tắc này là khá đa nghĩa, vì cần hiểu chính xác cách chọn các phần tử từ một tập hợp có nghĩa là thế nào? Rất đáng chú ý là Quy tắc xoắn vẫn đúng cho nhiều cách hiểu khác nhau của từ *cách chọn*. Ví dụ, ta có thể đòi hỏi chọn các phần tử phân biệt, cũng có thể cho phép được chọn một số lần có giới hạn nào đó, hoặc cho chọn lặp lại tuỳ ý. Một cách nôm na, giới hạn duy nhất là (1) thứ tự chọn các phần tử không quan trọng (2) những giới hạn áp dụng cho việc chọn các phần tử của A và B cũng áp dụng cho việc chọn các phần tử của A ∪ B (Chặt chẽ hơn, cần có một song ánh giữa các cách chọn n phần tử từ A ∪ B với bộ sắp thứ tự các cách chọn từ A và B chứa tổng thể n phần tử)

Chứng minh. Định nghĩa



Đầu tiên ta hãy tính tích A(x).B(x) và biểu diễn hệ số cnthông qua các hệ số a và b. Ta có thể sắp xếp các số hạng này thành dạng bảng

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | b0 | b1x | b2x2 | b3x3 | … |  |
| a0 | a0b0 | a0b1x | a0b2x2 | a0b3x3 | … |  |
| a1x | a1b0x | a1b1x2 | a1b2x3 |  |  |  |
| a2x2 | a2b0x2 | a2b1x3 |  |  |  |  |
| a3x3 | a3b0x3 |  |  |  |  |  |
| … |  |  |  |  |  |  |

Chú ý rằng các số hạng có cùng luỹ thừa của x xếp trên các đường chéo /. Nhóm tất cả các số hạng này lại, ta thấy rằng hệ số của xn trong tích bằng

cn = a0bn + a1bn-1 + … + anb0

Bây giờ ta chứng minh rằng đây cũng chính là số cách chọn n phần tử từ A ∪ B. Một cách tổng quát, ta có thể chọn n phần tử từ A ∪ B bằng cách chọn j phần tử từ A và n-j phần tử từ B, trong đó j là một số từ 0 đến n. Điều này có thể được thực hiện bằng ajbn-j cách. Lấy tổng từ 0 đến n, ta có

a0bn + a1bn-1 + … + anb0

cách chọn n phần tử từ A ∪ B. Đó chính xác là giá trị cn đã được tính ở trên.

Biểu thức cn = a0bn + a1bn-1 + … + anb0 đã được biết đến trong môn xử lý tín hiệu số; dãy <c0, c1, c2, c3, …> là *xoắn* (convolution) của hai dãy <a0, a1, a2, a3, …> và <b0, b1, b2, b3, …>.

**4.3. Chọn các phần tử có lặp**

Xét bài toán: Có bao nhiêu cách chọn 12 cây kẹo từ 5 loại kẹo? Bài toán này có thể tổng quát hoá như sau: Có bao nhiêu cách chọn ra k phần tử từ tập hợp có n phần tử, trong đó ta cho phép một phần tử có thể được chọn nhiều lần? Trong thuật ngữ này, bài toán chọn kẹo có thể phát biểu có bao nhiêu cách chọn 12 cây kẹo từ tập hợp

{kẹo sữa, kẹo sô-cô-la, kẹo chanh, kẹo dâu, kẹo cà-phê}

nếu ta cho phép lấy nhiều viên kẹo cùng loại. Ta sẽ tiếp cận lời giải bài toán này từ góc nhìn của hàm sinh.

Giả sử ta chọn n phần tử (có lặp) từ tập hợp chỉ có duy nhất một phần tử. Khi đó có 1 cách chọn 0 phần tử, 1 cách chọn 1 phần tử, 1 cách chọn 2 phần tử … Như thế, hàm sinh của cách chọn có lặp từ tập hợp có 1 phần tử bằng

<1, 1, 1, 1, …> ↔ 1 + x + x2 + x3 + … = 1/(1-x)

Quy tắc xoắn nói rằng hàm sinh của cách chọn các phần tử từ hợp của các tập hợp rời nhau bằng tích của các hàm sinh của cách chọn các phần tử từ mỗi tập hợp:



Như thế, hàm sinh của cách chọn các phần tử từ tập hợp n phần tử có lặp là 1/(1-x)n.

Bây giờ ta cần tính các hệ số của hàm sinh này. Để làm điều này, ta sử dụng công thức Taylor:

**Định lý 1** **(Định lý Taylor)**

****

Định lý này nói rằng hệ số của xk trong 1/(1-x)n bằng đạo hàm bậc k của nó tại điểm 0 chia cho k!. Tính đạo hàm bậc k của hàm số này không khó. Đặt

g(x) = 1/(1-x)n = (1-x)-n

Ta có

g’(x) = n(1-x)-n-1

g’’(x) = n(n+1)(1-x)-n-2

g’’’(x) = n(n+1)(n+2)(1-x)-n-3

…

g(k)(x) = n(n+1)…(n+k-1)(1-x)-n-k

Từ đó, hệ số của xk trong hàm sinh bằng



Như vậy số cách chọn k phần tử có lặp từ n phần tử bằng 

**5. Một bài toán đếm “bất khả thi”**

Từ đầu bài đến giờ ta đã thấy những ứng dụng của hàm sinh. Tuy nhiên, những điều này ta cũng có thể làm được bằng những cách khác. Bây giờ ta xét một bài toán đếm rất khó chịu. Có bao nhiêu nhiêu cách sắp một giỏ n trái cây thoả mãn các điều kiện ràng buộc sau:

* Số táo phải chẵn
* Số chuối phải chia hết cho 5
* Chỉ có thể có nhiều nhất 4 quả cam
* Chỉ có thể có nhiều nhất 1 quả đào

Ví dụ, ta có 7 cách sắp giỏ trái cây có 6 trái:

Táo 6 4 4 2 2 0 0

Chuối 0 0 0 0 0 5 5

Cam 0 2 1 4 3 1 0

Đào 0 0 1 0 1 0 1

Các điều kiện ràng buộc này quá phức tạp và có cảm giác như việc đi tìm lời giải là vô vọng. Nhưng ta hãy xem hàm sinh sẽ xử lý bài toán này thế nào.

Trước hết, ta đi tìm hàm sinh cho số cách chọn táo. Có 1 cách chọn 0 quả táo, có 0 cách chọn 1 quả táo (vì số táo phải chẵn), có 1 cách chọn 2 quả táo, có 0 cách chọn 3 quả táo …Như thế ta có

A(x) = 1 + x2 + x4 + … = 1/(1-x2)

Tương tự, hàm sinh cho số cách chọn chuối là

B(x) = 1 + x5 + x10 + … = 1/(1-x5)

Bây giờ, ta có thể chọn 0 quả cam bằng 1 cách, 1 quả cam bằng 1 cách, … Nhưng ta không thể chọn hơn 4 quả cam, vì thế ta có

C(x) = 1 + x + x2 + x3 + x4 = (1-x5)/(1-x)

Và tương tự, hàm sinh cho số cách chọn đào là

D(x) = 1 + x = (1-x2)/(1-x)

Theo quy tắc xoắn, hàm sinh cho cách chọn từ cả 4 loại trái cây bằng



Gần như tất cả được giản ước với nhau! Chỉ còn lại 1/(1-x)2 mà ta đã tìm được chuỗi luỹ thừa từ trước đó. Như thế số cách sắp giỏ trái cây gồm n trái cây đơn giản bằng n+1. Điều này phù hợp với kết quả mà ta đã tìm ra trước đó, vì có 7 cách sắp cho giỏ 6 trái cây. *Thật là thú vị!*

# B Phân tích một số thuật toán sắp xếp

## I Định nghĩa độ phức tạp

Thời gian mà [máy tính](http://vi.wikipedia.org/wiki/M%C3%A1y_t%C3%ADnh) khi thực hiện một [thuật toán](http://vi.wikipedia.org/wiki/Thu%E1%BA%ADt_to%C3%A1n) không chỉ phụ thuộc vào bản thân thuật toán đó, ngoài ra còn tùy thuộc từng máy tính. Để đánh giá hiệu quả của một thuật toán, có thể xét số các phép tính phải thực hiện khi thực hiện thuật toán này. Thông thường số các phép tính được thực hiện phụ thuộc vào cỡ của bài toán, tức là độ lớn của đầu vào. Vì thế **độ phức tạp thuật toán** là một hàm phụ thuộc đầu vào. Tuy nhiên trong những ứng dụng thực tiễn, chúng ta không cần biết chính xác hàm này mà chỉ cần biết một ước lượng đủ tốt của chúng.

Để ước lượng độ phức tạp của một thuật toán ta thường dùng khái niệm bậc O-lớn và bậc Θ (bậc [Theta](http://vi.wikipedia.org/wiki/Theta)).

O(n)

Gọi n là độ lớn đầu vào. Tùy thuộc từng bài toán mà n có thể nhận những giá trị khác nhau. Chẳng hạn, bài toán tính [giai thừa](http://vi.wikipedia.org/wiki/Giai_th%E1%BB%ABa) thì n chính là số cần tính giai thừa. Nhiều bài toán [số trị](http://vi.wikipedia.org/w/index.php?title=Ph%C6%B0%C6%A1ng_ph%C3%A1p_s%E1%BB%91&action=edit&redlink=1), chẳng hạn tính [sai phân](http://vi.wikipedia.org/w/index.php?title=Sai_ph%C3%A2n&action=edit&redlink=1) thì n là số [chữ số có nghĩa](http://vi.wikipedia.org/w/index.php?title=Ch%E1%BB%AF_s%E1%BB%91_c%C3%B3_ngh%C4%A9a&action=edit&redlink=1) cần đạt được. Trong các phép tính đối với [ma trận](http://vi.wikipedia.org/wiki/Ma_tr%E1%BA%ADn) thì n là số hàng hoặc cột của ma trận.

Độ phức tạp của bài toán phụ thuộc vào n. Ở đây ta không chỉ đặc trưng độ phức tạp bởi số lượng phép tính, mà dùng một đại lượng tổng quát là *tài nguyên cần dùng* R(n). Đó có thể là số lượng phép tính (có thể tính cả số lần truy nhập bộ nhớ, hoặc ghi vào bộ nhớ); nhưng cũng có thể là thời gian thực hiện chương trình (*độ phức tạp về thời gian*) hoặc [dung lượng bộ nhớ](http://vi.wikipedia.org/w/index.php?title=Dung_l%C6%B0%E1%BB%A3ng_b%E1%BB%99_nh%E1%BB%9B&action=edit&redlink=1) cần phải cấp để chạy chương trình (*độ phức tạp về không gian*).

Xét quan hệ giữa tài nguyên và độ lớn đầu vào, nếu như tìm được [hằng số](http://vi.wikipedia.org/wiki/H%E1%BA%B1ng_s%E1%BB%91) C>0, C không phụ thuộc vào n, sao cho với n đủ lớn, các hàm  R(n), g(n) đều dương và

R(n)\leq C.g(n)

thì ta nói thuật toán có độ phức tạp cỡ O(g(n)).

Các độ phức tạp thường gặp đối với các thuật toán thông thường gồm có:

Độ phức tạp hằng số, O(1). Số phép tính/thời gian chạy/dung lượng bộ nhớ không phụ thuộc vào độ lớn đầu vào. Chẳng hạn như các thao tác hệ thống: đóng, mở file.

Độ phức tạp tuyến tính, O(n). Số phép tính/thời gian chạy/dung lượng bộ nhớ có xu hướng tỉ lệ thuận với độ lớn đầu vào. Chẳng hạn như tính tổng các phần tử của một mảng một chiều.

Độ phức tạp đa thức, O(P(n)), với P là đa thức bậc cao (từ 2 trở lên). Chẳng hạn như các thao tác tính toán với mảng nhiều chiều (tính định thức ma trận).

Độ phức tạp [logarit](http://vi.wikipedia.org/wiki/Logarit), O(\log n) (chú ý: bậc của nó thấp hơn so với O(n)). Chẳng hạn thuật toán Euclid để tìm [ước số chung lớn nhất](http://vi.wikipedia.org/wiki/%C6%AF%E1%BB%9Bc_s%E1%BB%91_chung_l%E1%BB%9Bn_nh%E1%BA%A5t).

Độ phức tạp [hàm mũ](http://vi.wikipedia.org/wiki/H%C3%A0m_m%C5%A9), O(2^n). Trường hợp này bất lợi nhất và sẽ rất phi thực tế nếu thực hiện thuật toán với độ phức tạp này.

\Omega(f(n)),  \Theta(h(n)).

Tương tự như với bậc O-lớn, nếu như tìm được các hằng số C, k_1, k_2 đều dương và không phụ thuộc vào n, sao cho với n đủ lớn, các hàm  R(n), f(n)  và h(n)  đều dương và

R(n)\geq C \cdot f(n)

k_1\cdot h(n) \leq R(n) \leq k_2\cdot h(n)

thì ta nói thuật toán có độ phức tạp cỡ lớn hơn \Omega(f(n)), và đúng bằng cỡ \Theta(h(n)).

Như vậy nếu xét một cách chặt chẽ, kí hiệu Θ mới biểu thị độ phức tạp của thuật toán một cách chặt chẽ. Tuy nhiên qua một thời gian dài kí hiệu O(n) cũng đã được dùng phổ biến.

**7. Các ví dụ có lời giải**

**7.1. Cấp số nhân cộng**

Ta thử tìm lại công thức tính số hạng tổng quát cho cấp số nhân cộng, tức là dãy số xác định bởi a0 = a, an = axn-1 + d với mọi n = 1, 2, 3, …

Đặt F(x) = a0 + a1x + a2x2 + a3x3 + …

Ta có F(x) = a0 + a1x + a2x2 + a3x3 + …

= a0 + (qa0 + d)x + (qa1+d)x2 + (qa2+d)x3 + …

**=** a0 + qx(a0+a1x+a2x2+…) + dx(1+x+x2+…)

= a + qxF(x) + dx/(1-x)

Từ đó suy ra



Ta tìm phân tích dạng



Quy đồng mẫu số chung, ta được a + (d-a)x = A + B – (B+qA)x. Từ đó

A + B = a, B + qA = a – d

Suy ra A = d/(1-q) và B = a – d/(1-q).

Cuối cùng, áp dụng các công thức khai triển quen thuộc, ta được

.

**7.2. Phương trình sai phân không thuần nhất**

Tiếp theo, ta xem hàm sinh “làm việc” thế nào đối với các phương trình sai phân không thuần nhất.

Xét bài toán: Tìm công thức tổng quát của dãy số cho bởi a0 = 1, an = 2an-1 + 3n với n = 1, 2, 3, ..

Theo đúng sơ đồ trên, ta xét F(x) = a0 + a1x + a2x2 + a3x3 + …

Và thực hiện việc khai triển vế phải:

F(x) = a0 + a1x + a2x2 + a3x3 + …

= a0 + (2a0+3)x + (2a1+32)x2 + (2a2+33)x3 + …

= (1 + 3x + (3x)2 + (3x)3 + …) + 2x(a0+a1x+a2x2+…)

= 1/(1-3x) + 2xF(x)

Từ đó suy ra



Áp dụng công thức khai triển luỹ thừa cho các hàm số thường gặp, ta tìm được

an = 3n+1 – 2n+1.

Ta xem xét một ví dụ khác: Tìm công thức tổng quát của dãy số cho bởi a0 = 1, an = 2an-1 + n.3n.

Đặt F(x) = a0 + a1x + a2x2 + a3x3 + … Xét

F(x) = a0 + a1x + a2x2 + a3x3 + …

= a0 + (2a0+1.3)x + (2a1+2.32)x2 + (2a2+3.33)x3 + …

= 1 + 3x(1 + 2.(3x) + 3(3x)2 + …) + 2x(a0+a1x+a2x2+…)

= 1 + 3x/(1-3x)2 + 2xF(x)

Từ đó suy ra



Dùng phương pháp hệ số bất định, ta tìm được phân tích



Và từ đó

an = 3(n+1)3n – 9.3n + 7.2n = (n-2)3n+1 + 7.2n.

**7.3. Trường hợp phương trình đặc trưng có nghiệm kép**

Tiếp theo, ta xem xét hàm sinh “xử lý” trường hợp phương trình đặc trưng có nghiệm kép như thến nào.

Xét bài toán: Tìm công thức tổng quát của dãy số xác định bởi a0 = 1, a1 = 4, an = 4(an-1-an-2) với mọi n = 2, 3, 4, …

Theo sơ đồ chung, ta xét F(x) = a0 + a1x + a2x2 + a3x3 + a4x4 + …

Bỏ qua hai số hạng đầu, các số hạng từ a2 trở đi được tính theo các số hạng trước đó

F(x) = a0 + a1x + (4a1-4a0)x2 + (4a2-4a1)x3 + (4a3-4a2)x4 + …

= 1 + 4x + 4x(a1x+a2x2+a3x3+…) – 4x2(a0+a1x+a2x2+…)

= 1 + 4x + 4x(F(x)-1) – 4x2F(x)

Từ đó

F(x) = 1/(1-2x)2 = 1 + 2.2x + 3.22x2 + …

Suy ra an = (n+1)2n.

**7.4. Một ứng dụng của quy tắc xoắn**

Quy tắc xoắn còn có một cách phát biểu khác: Nếu <a0, a1, a2, a3, …> có hàm sinh là F(x), <b0, b1, b2, b3, …> có hàm sinh là G(x) thì xoắn của chúng, dãy <c0, c1, c2, c3, …> với cn = a0bn + a1bn-1 + … + anb0 có hàm sinh là F(x).G(x).

Ta sẽ dùng quy tắc này để giải một dãy số đệ quy khá đặc biệt: Cho dãy số {an} xác định bởi a0 = 1, a0an + a1an-1 + … + ana0 = 1 với mọi n = 1, 2, 3 … Hãy tìm công thức tổng quát tính an.

Ta tính thử các số hạng đầu tiên của dãy số: Với n =1, ta có a0a1+a1a0 = 1 suy ra a1 = 1/2, với n = 2, a0a2 + a1a1 + a2a0 = 1 suy ra a2 = 3/8. Tương tự, a3 = 5/16, a4 = 35/128 …Quy luật khá là phức tạp!

Ta thử dùng phương pháp hàm sinh. Mọi việc vẫn bắt đầu từ cách đặt F(x) = a0 + a1x + a2x2 + a3x3 + …

Tuy nhiên, nếu dùng phép thế thông thường an = (1 – (a1an-1+a2an-2+…+an-1a1))/a0 thì chúng ta sẽ không thu được phương trình để tìm F(x) như mong muốn. Tuy nhiên, hệ thức a0an + a1an-1 + … + ana0 = 1 gợi cho chúng ta ý nghĩ sử dụng quy tắc xoắn. Cụ thể nếu <a0, a1, a2, a3 …> có hàm sinh là F(x) thì theo quy tắc xoắn, F2(x) sẽ là hàm sinh của dãy “bình phương xoắn” của <a0, a1, a2, a3 …> tức là dãy <c0, c1, c2, c3, …> với cn = a0an + a1an-1 + … + ana0. Nhưng, theo như điều kiện đề bài thì cn = 1 với mọi n = 0, 1, 2, 3, … có nghĩa là

F2(x) ↔ (1, 1, 1, 1, …) ↔ 1 + x + x2 + x3 + … = 1/(1-x)

Từ đó F2(x) = (1-x)-1, suy ra F(x) = (1-x)-1/2.

Áp dụng công thức đã tìm được trong phần 6.1, ta tìm được .

## II Các thuật toán sắp xếp

### 1. Thuật toán sắp xếp bằng phương pháp Quick Sort

***Cài đặt :***

void QuickSort(float a[],int l,int r)

{

int i,j;

int x;

i=l;

j=r;

x=a[(l+r)/2];

do

{

while (a[i]<x)i++;

while(a[j]>x)j--;

if(j<=j)

{

if(i<j)

{

int temp=a[i];

a[i]=a[j];

a[j]=temp;

}

i++;

j--;

}

}

while(i<j);

if(l<j)QuickSort(a,l,j);

if(i<r)QuickSort(a,i,r);

}

***Đánh giá thuật toán***

*Ta nhận thấy hiệu quả của thuật toán phụ thuộc vào việc chọn giá trị mốc (hay phần tử chốt).*

o   Trường hợp tốt nhất: mỗi lần phân hoạch ta đều chọn được phần tử median (phần tử lớn hơn hay bằng nửa số phần tử và nhỏ hơn hay bằng nửa số phần tử còn lại) làm mốc. Khi đó dãy được phân hoạch thành hai phần bằng nhau, và ta cần log2(n) lần phân hoạch thì sắp xếp xong. Ta cũng dễ nhận thấy trong mỗi lần phân hoạch ta cần duyệt qua n phần tử. Vậy độ phức tạp trong trường hợp tốt nhất thuộc O(nlog2(n)).

o   Trường hợp xấu nhất: mỗi lần phần hoạch ta chọn phải phần tử có giá trị cực đại hoặc cực tiểu làm mốc. Khi đó dãy bị phân hoạch thành hai phần không đều: một phần chỉ có một phần tử, phần còn lại có n-1 phần tử. Do đó, ta cần tới n lần phân hoạch mới sắp xếp xong. Vậy độ phức tạp trong trường hợp xấu nhất thuộc O(n2).

*Tổng kết lại, ta có độ phức tạp của Quick Sort như sau:*

·        *Trường hợp tốt nhất: O(nlog2n)*

·        *Trường hợp xấu nhất: O(n2)*

·        *Trường hợp trung bình: O(nlog2n)*

### 2. Thuật toán sắp xếp bằng phương pháp ShellSort

***Cài đặt***

void ShellSort(float a[],int n,int k)

{

int step,i,j;

int x;

for(step=1;step<=k;step++)

{

for(i=1;i<=n;i++)

{

x=a[i];

j=i-1;

while((x<a[j])&&(j>=1))

{

a[j+1]=a[j];

j=j-1;

}

a[j+1]=x;

}

}

}

***Đánh giá thuật toán***

o   Yếu tố quyết định chính của thuật toán chính là cách chọn khoảng cách h trong từng bước sắp xếp và số bước sắp xếp k. Nhưng phải thỏa 2 điều kiện sau: hi > hi + 1 và hk= 1.

o   Các phần tử h không được là bội số của nhau nhằm tránh hiện tượng mỗi bước sắp thứ tự phải tổ hợp 2 nhóm mà bước trước chúng không hề có ảnh hưởng lẫn nhau. Điều mong muốn là ảnh hưởng giữa các nhóm khác nhau càng nhiều càng tốt. **???**

o   Việc đánh giá giải thuật Shell sort hiện nay rất phức tạp, thậm chí 1 số chưa được chứng minh. Nhưng có 1 điều chắc chắn là hiệu quả của thuật toán phụ thuộc vào dãy các độ dài được chọn. Trong trường hợp chọn dãy độ dài theo công thức hi= (hi – 1- 1)/2 và hk= 1, k = log2- 1 thì giải thuật có độ phức tạp tương đương n1,2 << n2

### 3. Thuật toán sắp xếp bằng phương pháp nổi bọt

***Cài đặt :***

void BubbleSort(float a[],int n)

{

int temp;

for(int i=1;i<n;i++)

for(int j=n;j>i;j--)

if(a[j]<a[j-1])

{

temp=a[j];

a[j]=a[j-1];

a[j-1]=temp;

}

}

***Phân tích thuật toán***

o   Thấy ngay số phép so sánh là luôn không đổi, tức không phụ thuộc vào tình trạng ban đầu của dãy. Với i bất kỳ, ta luôn phải so sánh V[j] với V[j-1], mà j chạy từ n đến i+1, tức ta tốn n-i phép so sánh. Thêm nữa, i chạy từ 1 đến n-1. Vậy ta tính được số phép so sánh tổng cộng: ∑(n-i) với i chạy từ 1 đến n-1 = (n-1) + (n-2) + … + 1 = n(n-1)/2.

o   Số phép hoán vị (tương đương 3 phép gán) lại phụ thuộc vào tình trạng ban đầu của dãy. Cụ thể như sau:

§  Trường hợp tốt nhất: Dãy ban đầu đã có thứ tự. Ta thấy ngay ta không tốn một phép hoán vị nào.

§  Trường hợp xấu nhất: Dãy ban đầu có thứ tự ngược. Xét i bất kỳ, ta thấy rằng mỗi lần so sánh a[j] với a[j-1], ta đều phải thực hiện hoán vị. Điều này có nghĩa là số phép hoán vị bằng n(n-1)/2.

o   *Tổng kết lại, ta có độ phức tạp của Bubble Sort thuộc O(n2) trong mọi trường hợp*

**5. Kết luận và hướng mở rộng**

# Đánh giá

Bài phân tích trên cho ta sự hiểu biết tốt hơn về hàm sinh xác suất cũng như cách ứng dụng vào giải một số bài toán phổ biến như các ví dụ ở trên. Cùng với một số thuật toán tìm kiếm, cách phân tích đánh giá độ phức tạp cho ta sự lựa chọn giải pháp tốt nhất cho bài toán ta cần giải quyết.

Do vấn đề thời gian còn hạn chế và trong quá trình trao dồi kiến thức nên bài viết vẫn còn một số hạn chế nhất định.

**Tài liệu tham khảo**

\_ Giáo trình bài giảng Probability Generating Functions (Chương 3)

\_ Generating Functions

\_ Giáo trình lý thuyết hàm sinh và ứng dụng

\_ Các bài phân tích thiết kế thuật toán

\_ The textbook *Algorithms, 4th Edition* by Robert Sedgewick and Kevin Wayne

\_ Introduction to Algorithms - Third Edition - Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein

1. Kenneth H. Rosen. *Discrete Mathematics and Its Applications*, Mc Graw-Hill, 2000.

2. Kenneth H.Rosen. *Toán học rời rạc và Ứng dụng trong tin học*, Nhà xuất bản thống kê 2002.

3. Wikipedia. Các bài: *Gererating Functions, Probability Generating Functions, Catalan numbers*.

4. Srini Devadas and Eric Lehman, *Generating Functions*, Lectures Notes, April 2005.

**=============**

**Chương 7: Giới thiệu về**

**phương pháp xác suất**

**I. Giới thiệu chung**

**II. Phân loại các thuật toán xác suất**

**III. Thuật toán xác suất số**

**IV. Thuật toán Sherwood**

**V. Thuật toán Las Vegas**

**VI. Thuật toán Monte Carlo**

**Giới thiệu chung**

** Khái niệm: Thuật toán xác xuất là thuật toán sử dụng đầu vào là các số ngẫu nhiên.**

** Nhận xét:**

**- Đầu vào này phải hữu ích**

**- Các số này là “giả ngẫu nhiên”**

**Ví dụ**

**Thuật toán sắp xếp Quicksort:**

**- Phần tử so sánh được chọn ở đầu bảng:**

**- Độ phức tạp xấu nhất là 0(n^2)**

**- Độ phức tạp trung bình là 0(n lg n)**

**Vấn đề: nếu bảng đã gần xếp đúng, không hiệu**

**quả.**

**Giải quyết: Chọn phần tử so sánh một cách ngẫu**

**nhiên trong bảng.**

**Phân loại các thuật toán xác suất**

**Thuật toán xác suất số:**

**- Dùng trong việc tính toán xấp xỉ các bài toán**

**số.**

**- Độ chính xác trung bình tỷ lệ với thời gian**

**chạy**

**- Dùng trong các ví dụ: tính toán thời gian**

**chay TB của một hệ phức tạp, cách tính chính**

**xác là rất phức tạp (hoặc không thể)**

**Thuật toán Monte Carlo:**

**- Có thể dùng trong các bài toán quyết định: đúng/sai, phân tích thừa số ng. tố,…**

**- Luôn cho câu trả lời rõ ràng nhưng**

**- Kết quả có thể không đúng**

**- Xác xuất thành công tỷ lệ với thời gian chạy**

**Thuật toán Monte Carlo:**

**- Có thể dùng trong các bài toán quyết**

**định: đúng/sai, phân tích thừa số ng. tố,…**

**- Luôn cho câu trả lời rõ ràng nhưng**

**- Kết quả có thể không đúng**

**- Xác xuất thành công tỷ lệ với thời gian**

**chạy**

**Thuật toán Las Vegas:**

**- Không bao giờ cho câu trả lời sai**

**- Có thể không cho câu trả lời**

**- Xác xuất thành công tỷ lệ với thời gian chạy**

**- Dù vấn đề nào, xác xuất hỏng cũng có thể nhỏ**

**tùy ý**

**\* Cho trả lời chính xác với độ phức tạp trung bình**

**có giới hạn**

**Thuật toán Sherwood:**

**- Luôn cho câu trả lời, trả lời luôn đúng**

**- Dùng trong các bài toán đã có thuật toán, với ĐPTTB nhỏ và ĐPT xấu nhất lớn**

**- Biễn ngẫu nhiên là để giảm sự khác nhau giữa hai trường hợp này**

**Thuật toán Monte Carlo**

**Bài toán ví dụ: Thử xem một số có nguyên**

**tố hay không.**

**Ý nghĩa: trong mật mã cần tìm các số**

**nguyên tố lớn.**

**Thuật toán đơn giản**

**Tính nguyên tố (n) {**

**while (i< n) do {**

**if (n chia hết cho i) return sai;**

**i = i+1;**

**}re**

**turn đúng;**

**} N**

**Nhận xét: cho I chạy đến sqrt(n)**

**- Để thử tính chia hết: thời gian nhiều**

**Phép thử Miller Rabin**

**Nguyên tắc:**

**a) n nguyên tố, a < n**

**=> a^{n-1} = 1 (mod n)**

**b) n nguyên tố, a <n :**

**a^2 = 1 (mod n) =>a = 1, -1 (mod n)**

**Câu hỏi: viết thuật toán test (n,a) thử hai điều kiên**

**trên. Độ p.t.t.t = ?**

**Thuật toán Miller-Rabin**

**M-R(n) {**

**i=1; chọn a ngẫu nhiên < n;**

**while (i<k and test (n,a)=false) do {**

**chọn a ngẫu nhiên < n;**

**i = i+1;}**

**return test(n,a);**

**} -**

**Số lần chọn các số ngẫu nhiên a là k lần.**

**- Tính độ p.t.t.t**

**Xác xuất của thuật toán**

**Tính chất: Nếu n lẻ, thì có ít nhất ¾ số a <**

**n sẽ cho kết quả test(n,a)= true nếu n**

**không nguyên tố.**

**Như vậy sau k lần thì xác xuất không tìm**

**thấy a cho test(n,a)=true là (1/4)^k. Và**

**ta có thể cho xac suất này nhỏ tùy ý.**

**Thuật toán Las-Vegas**

**Bài toán: Bầu một người chủ.**

**Có n người, cần chọn ra một người.**

**Đòi hỏi: các lần chạy t.t. khác nhau, chọn ra**

**những người khác nhau.**

**Ý nghĩa: Trong bài toán mạng, tránh tình trạng tắc**

**nghẽn khi các máy cùng đến một lúc.**

**Thuật toán**

**1. Mỗi người chọn một số ngẫu nhiên từ 1 đến m**

**(số người chọn)**

**2. Nếu không có ai chọn số 1, quay về bước 2.**

**3. Nếu có ít nhất 2 người chọn số 1, thì những**

**người này tiếp tục, quay về bước 1.**

**4. Nếu chỉ có 1 người chọn số 1 thì người này là**

**chủ.**

**Tính chất của thuật toán**

**- Thuật toán không chắc chắn là sẽ dừng**

**- Nhưng nếu dừng nó luôn cho kết quả**

**Đúng**

**Phân tích thuật toán**

**Xác xuất để có k người vào vòng 2 là:**

**p(n,k) = C^k\_n (1/n)^k (1-1/n)^(n-k)**

**P(n,0) = (1-1/n)^n.**

**Xác xuất để vòng 1 lặp l lần là (1-1/n)^{nl}**

**Xác xuất để t.t. không dừng là = 0**