HỘI THẢO CÁC TRƯỜNG THPT CHUYÊN KHU VỰC
DUYÊN HẢI VÀ ĐỒNG BẰNG BẮC BỘ – NĂM 2020

HỘI THẢO KHOA HỌC LẦN THỨ XIII
MÔN TIN HỌC

CHUYÊN ĐỀ
ỨNG DỤNG CẤU TRÚC DỮ LIỆU BẢNG BĂM ĐỂ GIẢI CÁC BÀI TOÁN SO KHỚP XÂU

Tháng 9 năm 2020

MỤC LỤC

A. PHẦN MỞ ĐẦU

Mục đích của giảng dạy tin học ở trường THPT là cung cấp cho học sinh kiến thức về tin học, phương pháp giải bài toán trong tin học và rèn luyện tư duy, kỹ năng lập trình cho học sinh.

Khi giải quyết một bài toán trong tin học, theo tôi vấn đề mà được các em học sinh quan tâm nhất đó là lựa chọn và sử dụng thuật toán nào cho bài toán là tối ưu nhất dựa trên nền tảng kiến thức của các em đã được học. Không phải lúc nào và với lớp bài toán nào chúng ta cũng có một cách làm như nhau được mà đòi hỏi phải sử dụng các kiến thức linh hoạt. Chúng ta cần đánh giá được sự tác động của các thuật toán áp dụng vào trong bài toán của mình ở các mức độ khác nhau và từ đó định hướng tìm ra giải thuật phù hợp nhất.

Trong tin học, ngoài việc lựa chọn thuật toán tối ưu cho bài toán thì việc lựa chọn cấu trúc dữ liệu phù hợp cho bài toán rất quan trọng vì khi chọn cấu trúc dữ liệu phù hợp cũng làm tăng hiệu quả xử lý bài toán.

Trong bài viết này, tôi trình bày cấu trúc dữ liệu bảng băm cũng như vận dụng bảng băm để giải quyết một số bài toán so khớp xâu trong tin học.

B. PHẦN NỘI DUNG

I. Cơ sở lý thuyết

1. Phép băm

Phép Băm là phép biến đổi giá trị khóa của phần tử dữ liệu thành số và sử dụng số này để đánh địa chỉ cho phần tử dữ liệu trên bảng. Dựa trên bảng địa chỉ của phép băm chúng ta có thể tham chiếu trực tiếp đến các phần tử dữ liệu.

2. Bảng băm

2.1. Khái niệm

Bảng băm T là một mảng lưu trữ địa chỉ định vị phần tử dữ liệu có khóa phân biệt.

Bảng băm T có kích thước m thì các địa chỉ (chỉ số) trong T được đánh từ 0 đến m-1, mỗi địa chỉ tương ứng với 0 hoặc 1 hoặc nhiều khóa.

Ví dụ: Cho một tập dữ liệu D gồm 4 phần tử có giá trị tương ứng là 4 xâu “tra”, “cofe”, “banh”, “keo”. Hãy cho biết tập dữ liệu D có chứa xâu S hay không.

Thực hiện phép băm

- Ta qui ước giá trị của các kí tự chữ cái thường tương ứng là 'a' có giá trị là 1, 'b' có giá trị là 2, 'c' có giá trị là 3, ..., 'z' có giá trị là 26.
- Phép băm được chọn thực hiện là phép tính tổng các giá trị số tương ứng của các kí tự có trong xâu.
- Thực hiện phép băm cho các phần tử của tập D:

  “tra” => 20 + 18 + 1 = 39

  “cofe” = 3 + 15 + 6 + 5 =29

  “banh” = 2 + 1 + 14 + 8 = 25

  “keo” = 11 + 5 + 15 = 31

- Theo kết quả phép băm, ta xây dựng bảng băm T như sau: Bảng băm T là một mảng các phần tử kiểu xâu. Các phần tử có khóa là “tra”, “cofe”, “banh”, “keo” được đặt lần lượt vào các vị trí 39, 29, 25, 31 của mảng T, các vị trí còn lại của mảng T là rỗng.

T	0	...	25	...	29	...	31	...	39	...
	“”	“”	“banh”	“”	“cofe”	“”	“keo”	“”	“tra”	“”

- Để biết xâu S có trong tập dữ liệu D hay không thì ta chỉ việc tính giá trị băm của S và truy xuất trực tiếp từ bảng băm T để kiểm tra. Chẳng hạn:
  - Với S = “gao”

    Giá trị băm (“gao”) = 7 + 1 + 15 =23

    Ta có T[23] là rỗng, do đó xâu S không có trong tập D
  - Với S = “banh”

    Giá trị băm (“banh”) = 2 + 1 + 14 + 8 = 25

    Ta có T[25] khác rỗng, do đó xâu S có trong tập D.

2.2. Phân loại bảng băm

2.2.1. Bảng băm đóng

Bảng băm đóng là bảng băm mà mỗi khóa ứng với một địa chỉ, thời gian truy xuất bằng hằng số.

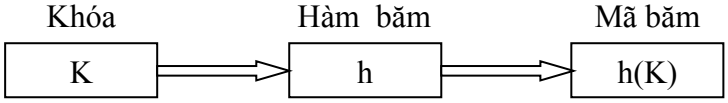
2.2.2. Bảng băm mở

Bảng băm mở là bảng băm mà một số khóa có cùng địa chỉ, lúc này mỗi mục địa chỉ sẽ là một danh sách liên kết các phần tử có cùng địa chỉ, thời gian truy xuất có thể bị chậm đôi chút.

3. Hàm băm

3.1. Khái niệm

Hàm băm là một ánh xạ h từ tập các giá trị khoá K của dữ liệu vào tập các số nguyên {0, 1,…, m -1}, m là kích thước của bảng băm, tức là h: K → {0, 1, …, m-1}. Với mỗi k ∈ K, ta có h(k) là một địa chỉ trong bảng băm.



Giá trị trả về của hàm băm gọi là giá trị băm hay mã băm.

3.2. Sự đụng độ

Sự đụng độ (xung đột) là khi các khóa khác nhau nhưng khi băm cho kết quả cùng mã băm.

$k_i \neq k_j$ nhưng ta có $h(k_i) = h(k_j)$

3.3. Tính chất của hàm băm

Một hàm băm phải thỏa mãn các tính chất sau:

- Tính toán nhanh.
- Các khóa được phân bố đều trong bảng.
- Ít xảy ra đụng độ.
- Xử lý được các loại khóa có kiểu khác nhau

3.4. Một số hàm băm thông dụng

3.4.1. Hàm cắt bỏ

Hàm cắt bỏ sẽ loại bỏ bớt một phần nào đó của khóa.

Ví dụ: Cho khóa là một số nguyên k = 842615.

Ta quy ước: Hàm cắt bỏ sẽ bỏ bớt các chữ số hàng lẻ (1,3,5…).

Vậy h(k) = h(842615) = 821.

Nhận xét: Hàm cắt bỏ khó có phân bố đều.

3.4.2. Hàm gấp

Hàm gấp sẽ chia số nguyên đó thành một số đoạn tùy chọn, sau đó kết hợp các phần đó lại theo một quy ước nào đó.

Ví dụ: Cho khóa là một số nguyên k = 842615.

Số các hàng lẻ: 465

Số các hàng chẵn: 821

Vậy h(k)=465+821=1286.

Nhận xét: Tính chất thứ hai có thể thỏa mãn tốt hơn.

3.4.3. Hàm phần dư

Hàm phần dư sẽ lấy phần dư của khóa k chia cho m, sử dụng phép chia lấy dư mod.

Ta có h(k) = k mod m, việc chọn giá trị m sẽ ảnh hưởng đến giá trị h(k), thông thường chọn m là một số nguyên tố.

Nhận xét: Cách lấy phần dư cho khả năng tránh hiện tượng xung đột.

II. Ứng dụng cấu trúc dữ liệu bảng băm để giải bài toán so khớp xâu

1. Lựa chọn hàm băm

Viêc lựa chọn hàm băm có ảng hưởng rất lớn đến kết quả của bài toán. Lựa chọn hàm băm phải phù hợp, chính xác cao. Khi lựa chọn một hàm băm thì một tiêu chí được quan tâm nhất đó là hàm băm ít xảy ra đụng độ.

Sử dụng hàm băm là hàm lấy phần dư: h(k) = k mod m

Trong đó, k là khóa**,** m là một số nguyên tố phù hợp

Nhận xét:

- Nếu $k_i = k_j$ thì $h(k_i) = h(k_j)$, lập luận đúng.
- Nếu $h(k_i) = h(k_j)$ thì $k_i = k_j$, lập luận không đúng(vì có thể xảy ra đụng độ).
- Để sử dụng hàm lấy phần dư làm hàm băm thì chúng ta phải chấp nhận lập luận "$h(k_i) = h(k_j)$ thì $k_i = k_j$" là đúng.

Vậy ta có: $k_i = k_j \Leftrightarrow h(k_i) = h(k_j)$

2. Bài toán so khớp xâu

2.1. Phát biểu bài toán

Cho hai xâu A và B chỉ gồm các chữ cái Latin thường. Xâu A có x kí tự và xâu B có y kí tự.

Yêu cầu: Hãy tìm tất cả các vị trí mà xâu B xuất hiện trong xâu A.

2.2. Thuật toán

Tóm tắt bài toán

- Xâu A gồm x kí tự: A[1..x]
- Xâu B gồm y kí tự: B[1..y]
- Xâu con từ i đến j của xâu A là: A[i..j]
- Chúng ta cần tìm tất cả các vị trí i thỏa mãn ($1 \le i \le x-y+1$) mà A[i..(i+y-1)] = B[1..y]

Xây dựng hàm băm, bảng băm

- Tập các chữ cái Latin thường gồm có 26 chữ cái {a, b,…, z}.
- Ta nhìn nhận một xâu chữ cái là một số biểu diễn trong hệ đếm cơ số 26. Hệ đếm cơ số 26 sử dụng 26 ký hiệu, gồm a, b, …, z có giá trị tương ứng là 0, 1, …, 25.
- Chọn m là số nguyên tố, chẳng hạn $10^9 + 7$ hoặc $2.10^9 + 11$, …
- Tính giá trị mã băm h(s) cho xâu s.

Ví dụ: Cho xâu s = "abcd". Tính giá trị h(s)

- Tính giá trị mã băm h(s) cho xâu s chính là tính giá trị của biểu diễn xâu s trong hệ đếm cơ số 26.
- Ta có "abcd"$_{26}$ = $0*26^3 + 1*26^2 + 2*26^1 + 3*26^0$

= (ord(‘a’)-97)*26^3 + (ord(‘b’)-97)*26^2 + (ord(‘c’)-97)*26^1 + (ord(‘d’)-97)*26^0

= 731.

- Vậy h(s) = h(“abcd”) mod m = 731 mod m

- Tính giá trị mã băm của đoạn xâu con s[i..j] ($1 \le i \le j \le n$), n là số kí tự trong xâu s.

Ví dụ: Cho xâu s= “abcdef”. Tính giá trị mã băm của đoạn s[3..6]

Ta có:

- h(s[3..6]) = h(“cdef”)
  = 2*26^3 + 3*26^2 + 4*26 + 5*26^0
  = (ord(‘c’)-97)*26^3 + (ord(‘d’)-97)*26^2 + (ord(‘e’)-97)*26^1 + (ord(‘f’)-97)*26^0
- h(s[1..6]) = h(“abcdef”)
  = 0*26^5 + 1*26^4 + 2*26^3 + 3*26^2 + 4*26 + 5*26^0
  = (ord(‘a’) - 97)*26^5 + (ord(‘b’) - 97)*26^4 + (ord(‘c’) - 97)*26^3 + (ord(‘d’) - 97)*26^2 + (ord(‘e’) - 97)*26^1 + (ord(‘f’) - 97)*26^0
- h(s[1..2]) = h(“ab”)
  = 0*26^1 + 1*26^0
  = (ord(‘a’)-97)*26^1 + (ord(‘b’)-97)*26^0
- h(s[3..6])= h(s[1..6]) – h(s[1..2]) * 26^4.

Vậy ta có: h(s[i..j]) = (h(s[1..j]) – h(s[1..(i-1)])*26^{j-i+1}) mod m.

- Chọn số chia cho hàm phần dư m = 1000000000 + 7.
- Gọi HashB là mã băm của xâu B.
- Tính mã băm của xâu B.

```
HashB := 0;
for i:=1 to y do HashB:=(HashB*26 + (ord(B[i])-97)) mod m;
```

- Gọi HashA[i] là mã băm của đoạn xâu con A[1..i].
- Tính mã Hash tất cả các tiền tố của Xâu A.

```
HashA[0]:=0;
for i:=1 to x do HashA[i]:=(HashA[i-1]*26 + (ord(A[i])-97)) mod m;
```

- Gọi Pow[i] là giá trị của 26^i.
- Tính giá trị Pow[i], ($0 \le i \le x$).

```
Pow[0]:=1;
for i:=1 to x do Pow[i]:=Pow[i-1]*26 mod m;
```

- Tính giá trị mã băm của đoạn xâu con A[i..j], ($1 \le i \le j \le x$), x là số kí tự trong xâu A.

- h(A[i..j]) = (HashA[j] - HashA[i-1]*Pow[j-i+1]) mod m (*)
- Xét công thức (*), ta có:
  $1 \le Pow[t] \le m-1$ với $1 \le t \le x$
  $0 \le HashA[i] \le m-1$ với $1 \le i \le x$
  $0 \le HashA[j] \le m-1$ với $1 \le i \le x$

=> $0 \le HashA[i]*Pow[t] \le (m-1)^2$ với $(1 \le i, j \le x)$

=> $-(m-1)^2 \le HashA[j] - HashA[i]*Pow[t] \le (m-1)$ với $(1 \le i, j, t \le x)$

=> h(A[i..j]) = (HashA[j] - HashA[i-1]*Pow[j-i+1]) mod m có thể nhận giá trị âm (không phù hợp với tiêu chí giá trị mã băm là một số tự nhiên).

Sử dụng tính chất trong quan hệ đồng dư ($a \equiv a + k.m \bmod m$) để giải quyết sự bất hợp lý nêu trên.

Vậy ta có: h(A[i..j]) = (HashA[j] - HashA[i-1]*Pow[j-i+1] + m*m) mod m

- Gọi GetHash(i,j) là hàm lấy giá trị mã băm của đoạn xâu con A[i..j].

```
Function GetHash(i,j:longint):Int64;
Begin
    Exit((HashA[j]-HashA[i-1]*Pow[j-i+1] + m*m) mod m);
End;
```

Thuật toán giải quyết bài toán

Để chỉ ra B xuất hiện ở những vị trí nào trong A, chúng ta cần duyệt qua mọi vị trí xuất phát có thể của B trong A.

Giả sử vị trí đó là i, chúng ta sẽ kiểm tra A[i. . i + y − 1] có bằng với B hay không việc này đông nghĩa với việc chúng ta cần tính và kiểm tra mã băm của đoạn xâu con A[i. . i + y − 1] có bằng mã băm của xâu B hay không.

```
for i:=1 to x-y+1 do
    if (GetHash(i,i+y-1) = HashB) then writeln(i);
```

Chương trình tham khảo

```
Uses Math;
Const fi=''; fo='';
Var f:Text;
  A,B:Ansistring;
  Hash,Pow:array[0..1000000] of Int64;
  m,x,y,HashB:Int64;
Procedure Build;
Var i:longint;
Begin
   Assign(f,fi); Reset(f);
   Readln(f,A); x:=length(A);
   Readln(f,B); y:=length(B);
   Close(f);
   m:=1000000007;
   For i:=1 to x do
      Hash[i]:=(Hash[i-1]*26+ord(A[i])-97) mod m;
   For i:=1 to y do HashB:=(HashB*26+ord(B[i])-97)mod m;
   Pow[0]:=1;
   For i:=1 to x do Pow[i]:=Pow[i-1]*26 mod m;
End;
```

```
Function Get(i,j:longint):Int64;
Begin
    Exit((Hash[j]-Hash[i-1]*Pow[j-i+1] + m*m)mod m);
End;
Procedure Output;
Var i:longint;
Begin
    Assign(f,fo); ReWrite(f);
    For i:=1 to x-y+1 do
     If Get(i,i+y-1)=HashB then Write(f,i,' ');
    Close(f);
End;
BEGIN
  Build;
  Output;
END.
```

III. BÀI TẬP VẬN DỤNG

1. Bài 1: Tiền tố và hậu tố

Nguồn: https://vn.spoj.com/problems/C11STR2/

1.1 Đề bài

Xâu a được gọi là tiền tố của xâu b nếu xâu a trùng với phần đầu của xâu b.

Ví dụ: pre là tiền tố của prefix.

Xâu a được gọi là hậu tố của xâu b nếu xâu a trùng với phần cuối của xâu b.

Ví dụ: fix là hậu tố của suffix.

Cho hai xâu a, b gồm các kí tự Latin thường. Hai xâu a và b không nhất thiết phải khác nhau và có độ dài không quá 10^5 kí tự .

Yêu cầu: Tìm một xâu kí tự c thỏa mãn

- Xâu a là tiền tố của xâu c;
- Xâu b là hậu tố của xâu c;
- Độ xài xâu c là ngắn nhất.

Dữ liệu: Vào từ file văn bản BAI1.INP

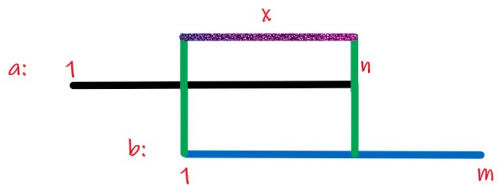
- Dòng 1 chứa xâu kí tự a;
- Dòng 2 chứa xâu kí tự b.

Kết quả: Ghi vào file văn bản BAI1.OUT gồm một dòng duy nhất là xâu kết quả c.

Ví dụ:

BAI1.INP	BAI1.OUT
abca cab	abcab
abc abc	abc

1.2. Hướng dẫn giải thuật



- Xâu a có n kí tự, a[1, ..., n]
- Xâu b có m kí tự, b[1, ..., m]
- Tìm phần chung x dài nhất là hậu tố của xâu a, là tiền tố của xâu b
    + Gọi Ha[i] là mã hash của đoạn xâu con a[1..i], xét xâu từ trái qua phải
    + Gọi Hb[i] là mã hash của đoạn xâu con b[1..i] , xét xâu từ trái qua phải
    + Đoạn xâu con có độ dài x vừa là tiền tố của xâu b, vừa là hậu tố của xâu a nêu:

$$Hb[x] = Ha[n]-Ha[n-x]*pow[x]$$

- Xóa x kí tự đầu tiên của xâu b
- Kết quả của bài toán là xâu ghép: a+b[x+1, x+2, ..., m]

1.3. Chương trình tham khảo

```
CONST
FI='BAI1.INP'; FO='BAI1.OUT';
MAXN=100005;
vc=round(1e9);
VAR A,B:ANSISTRING;
   ha,hb:array[0..maxn] of int64;
   Pow:array[0..maxn] of int64;
   n,m:longint;
   f:text;
function min(x,y:longint):longint;
begin
        if(x<y) then exit(x)else exit(y);
end;
function max(x,y:longint):longint;
begin
        if(x>y) then exit(x)else exit(y);
end;
procedure doc;
begin
        assign(f,fi); reset(f);
        readln(f,a);
        readln(f,b);
        close(f);
end;
procedure Build;
VAR i:longint;
begin
```

```
        Ha[0]:=0;
        Hb[0]:=0;
        pow[0]:=1;
        n:=length(a); m:=length(b);
        for i:=1 to max(n,m) do pow[i]:=(pow[i-1]*26) mod vc;
        for i:=1 to n do Ha[i]:=(Ha[i-1]*26+ord(A[i])-97) mod vc;
        for i:=1 to m do Hb[i]:=(Hb[i-1]*26+ord(B[i])-97) mod vc;
end;
function KT(x:longint):boolean;
var   v1,v2:int64;
begin
        v1:=Hb[x];
        v2:=(Ha[n]-Ha[n-x]*pow[x]+vc*vc)mod vc;
        if v1=v2 then exit(true);
        exit(false);
end;
procedure Process;
var  x,i:longint; kq:ansistring;
begin
assign(f,fo); rewrite(f);
        x:=0;
        for i:=1 to min(n,m) do  if KT(i) then x:=i;
        delete(b,1,x);
        writeln(f,a+b);
        close(f);
end;
Begin
        doc;
        Build;
        Process;
End.
```

Độ phức tạp của thuật toán: O(n, m)

1.4. Test và code

https://drive.google.com/drive/folders/1D567hriA-itxBcZ2EVQ2fQwifoxQ-h4q?usp=sharing

2. Bài 2: GENOME

Nguồn: http://www.olp.vn/luu-tru/2010/de-thi/DethiKhoiChuyenTin_OLP2010.pdf?attredirects=0&d=1

2.1. Đề bài

DNA là thành phần cơ bản cấu tạo thành bộ genome của sinh vật. DNA bao gồm 4 loại khác nhau là {A,X,T,G}. Để nghiên cứu các sinh vật ở mức độ phân tử, người ta tiến hành giải mã bộ genome của chúng.

Để giải mã bộ genome của một sinh vật, máy giải mã thế hệ mới sẽ sinh ra N đoạn cơ sở, mỗi đoạn cơ sở là một dãy bao gồm 30 DNA. Các đoạn cơ sở sẽ được ghép nối với nhau để tạo thành một bộ genome hoàn chỉnh.

Ta nói một đoạn DNA X được bao phủ bởi một đoạn cơ sở Y nếu tồn tại một đoạn của Y trùng với X. Giả sử k là một số nguyên dương, một đoạn DNA X được gọi là đoạn tin tưởng cấp k nếu X được bao phủ bởi ít nhất kđoạn cơ sở.

Yêu cầu: Cho n đoạn cơ sở và số nguyên dương k, hãy tìm đoạn tin tưởng cấp k có độ dài lớn nhất.

Dữ liệu: Vào từ file văn bản BAI2.INP

- Dòng đầu chứa hai số nguyên dương n và k ($0< k \leq n \leq 30.000$);
- Trong n dòng tiếp theo, mỗi dòng chứa một đoạn cơ sở.

Kết quả: Ghi vào file văn bản BAI2.OUT một số nguyên xác định độ dài của đoạn tin tưởng tìm được (ghi -1 nếu không tồn tại đoạn tin tưởng cấp k).

Ví dụ:

BAI2.INP	BAI2.OUT
4 3 AAAAAAAAATAAAATAAAAAAAAAAAAATG AAAAAAAAAAAAAAAAAAAATAAATGAAAA AAAAAAAAAAAAAAAAAATGAAAAAAAA A AAAAAAAAAAAAATGAAAAAAAGGGGAAA A	15

2.2. Hướng dẫn giải thuật

Chặt nhị phân theo độ dài của đoạn cơ sở:

- Với độ dài cần xét là d:
  + Gọi C[i] là mã hash của đoạn DNA thứ i có độ dài là d
  + Gọi D[i] là chỉ số của đoạn cơ sở chứa đoạn DNA thứ i
  + Ta cần kiểm tra xem có đủ k đoạn DNA khác nhau (không có hai đoạn DNA cùng thuộc một đoạn cơ sở) có cùng mã Hash hay không.

```
QS(1,M); //sort mảng C
C[0]:=-1;
For i:=1 to M do
        If C[i]<>C[i-1] then
                Begin
                If Res>=k then Exit(True);
                Res:=1;
                End
        Else
        If D[i]<>D[i-1] then Inc(res);
  Exit(False);
```

- Tìm d lớn nhất thỏa mãn điều kiện trên

2.3. Chương trình tham khảo

```
Const fi='BAI2.INP'; fo='BAI2.OUT';
Var f:Text;
   G:array['A'..'Z'] of Byte;
   N,K,VC:Int64;
   H:array[0..30000,0..30] of Int64;
   P:array[0..30] of Int64;
   D,C:array[0..1000000] of Longint;
Procedure Build;
Var
        i,j:Longint;
        X:String;
Begin
        G['A']:=0;
        G['T']:=1;
        G['G']:=2;
        G['X']:=3;
        Assign(f,fi); Reset(f);
        Readln(f,N,K);
        P[0]:=1;
        VC:=1000000007;
        For i:=1 to N do
        Begin
        Readln(f,X);
        For j:=1 to 30 do H[i,j]:=(H[i,j-1]*4+G[X[j]]) mod VC;
        End;
        For i:=1 to 30 do P[i]:=P[i-1]*4 mod VC;
        Close(f);
End;
Procedure QS(L,R:Longint);
Var
        tg,x,y:Int64;
        i,j:Longint;
Begin
        i:=L; j:=R;
        x:=C[(i+j) div 2];
        y:=D[(i+j) div 2];
        Repeat
                While (C[i]<x) or ((C[i]=x) and (D[i]<y)) do Inc(i);
                While (C[j]>x) or ((C[j]=x) and (D[j]>y)) do Dec(j);
                If i<=j then
                        Begin
```

```
                        tg:=D[i]; D[i]:=D[j]; D[j]:=tg;
                        tg:=C[i]; C[i]:=C[j]; C[j]:=tg;
                        Inc(i); Dec(j);
                    End;
        Until i>j;
        If i<R then QS(i,R);
        If j>L then QS(L,j);
End;
Function kt(x:Longint):Boolean;
Var
        i,j,M,res:Longint;
        Y:Int64;
Begin
        M:=0;
        res:=0;
        For i:=1 to N do
        For j:=1 to 30 do
                If j+x-1<=30 then
                    Begin
                       Y:=(H[i,j+x-1]-H[i,j-1]*P[x]+VC*VC) mod VC;
                        Inc(M);
                        C[M]:=Y;
                        D[M]:=i;
                    End;
        QS(1,M);
        C[0]:=-1;
        For i:=1 to M do
                If C[i]<>C[i-1] then
                     Begin
                        If Res>=K then Exit(True);
                        Res:=1
                     End
                Else
                        If D[i]<>D[i-1] then Inc(res);
        Exit(False);
End;
Procedure Process;
Var l,r,tg:Longint;
Begin
        Assign(f,fo); Rewrite(f);
        l:=1; r:=30;
```

```
        While l<=r do
        Begin
                tg:=(l+r) div 2;
                If kt(tg) then l:=tg+1
                Else   r:=tg-1;
                Fillchar(C,sizeof(C),0);
                Fillchar(D,sizeof(D),0);
        End;
        If r=0 then r:=-1;
        Writeln(f,r);
        Close(f);
End;
BEGIN
        Build;
        Process;
END.
```

Độ phức tạp của thuật toán: $O(30*n*\log_2 30)$

2.4. Test và code

https://drive.google.com/drive/folders/1D567hriA-itxBcZ2EVQ2fQwifoxQ-h4q?usp=sharing

3. Bài 3. MORSE

Nguồn: https://vn.spoj.com/problems/MORSE/

3.1. Đề bài

Hiện nay, khi công nghệ thông tin phát triển, con người thường trao đổi với nhau bằng điện thoại, fax hay email. Hãy quay ngược thời gian lại 100 năm, khi đó con người không có điện thoại hay fax, lại càng chẳng có email, người ta phải liện lạc với nhau bằng mật mã MORSE. Đây là loại mật mã mà các kí tự chỉ được mã hóa bằng 2 kí hiệu “.” (tích) và “-“ (tè). Để chuyển đi một văn bản, người ta mã hóa từng chữ cái trong văn bản đó thành những dãy kí tự tích tè. Trên thế giới đã quy định một quy tắc mã hóa chuẩn như sau:

A	.-	B	-...	C	-.-.
D	-..	E	.	F	..-.
G	--.	H	....	I	..
J	.---	K	-.-	L	.-..
M	--	N	-.	O	---
P	.--.	Q	--.-	R	.-.
S	...	T	-	U	..-
V	...-	W	.--	X	-..-
Y	-.--	Z	--..		

Hạn chế của cách mã hóa này là một dãy mã hóa có thể có nhiều cách giải mã. Ví dụ dãy -.-..-- có thể được hiểu là CAT hay NXT đều đúng. Rõ ràng là trong trường hợp bình thường, chúng ta sẽ phải hiểu là CAT vì NXT không có nghĩa. Tuy vậy một dãy mã hóa vẫn có thể có nhiều cách giải mã có nghĩa.

Yêu cầu: Bạn có trong tay một dãy đã mã hóa và một danh sách các từ có nghĩa, bạn hãy tính xem có bao nhiêu cách giải mã có nghĩa. (Một cách giải mã có nghĩa là một cách chia đoạn MORSE ban đầu thành các đoạn con liên tiếp sao cho mỗi đoạn được giải mã thành một từ có nghĩa)

Dữ liệu: Vào từ file văn bản BAI3.INP

- Dòng thứ nhất ghi xâu đã mã hóa gồm không quá 10000 kí tự tích tè;
- Dòng thứ hai ghi số n là số các từ có nghĩa ($n \leq 1000$);
- Trong n dòng tiếp theo, mỗi dòng ghi một từ có nghĩa. Các từ là các xâu không rỗng gồm không quá 10 kí tự trong số các kí tự 'A' .. 'Z'.

Kết quả: Ghi vào file văn bản BAI3.OUT số P là phần dư số cách giải mã có nghĩa chia cho 10000007.

Ví dụ:

BAI3.INP	BAI3.OUT
.---.--.-.-.-.---...-.---. 6 AT TACK TICK ATTACK DAWN DUSK	2

3.2 Hướng dẫn giải thuật

- Gọi S là xâu mã hóa đề cho
- Gọi C[i] là độ dài mã MORSE tương ứng của từ thứ i
- Gọi D[i] là mã hash tương ứng mã MORSE của từ thứ i
- Gọi H[i] là mã hash của đoạn xâu mã hóa S[1..i]
- Gọi L[i] là số cách giải mã có nghĩa của đoạn xâu mã hóa S[1..i]
- Tính mảng L:

```
L[0]:=1;
For i:=1 to M do
For j:=1 to N do
    If i+C[j]-1>M then Break
    Else
    Begin
        X:=(H[i+C[j]-1]-H[i-1]*P[C[j]]+VC*VC) mod VC;
        If X=D[j] then
        L[i+C[j]-1]:=(L[i-1]+L[i+C[j]-1]) mod 10000007;
```

End;

- Kết quả bài toán: L[M]

3.3 Chương trình tham khảo

```
Const
fi='bai3.inp'; fo='bai3.out';
T:array['A'..'Z'] of string=('.-', '-...', '-.-.', '-..', '.', '..-.', '--.', '....', '..', '.---', '-.-', '.-..', '--',
'-.', '---', '.--.', '--.-', '.-.', '...', '-', '..-', '...-', '.--', '-..-', '-.--', '--..');
Var
   f,g:Text;
   S:ansistring;
   D,P,H,L,C:array[0..10005] of Int64;
   N,VC,M:Int64;
Function Ox(x:char):byte;
Begin
    If x='.' then Exit(0);
    Exit(1);
End;
Procedure QS(L,R:Longint);
Var tg,x:Int64;
   i,j:Longint;
Begin
    i:=L; j:=R; x:=C[(i+j) div 2];
    Repeat
        While C[i]<x do Inc(i);
        While C[j]>x do Dec(j);
        If i<=j then
         Begin
            tg:=D[i]; D[i]:=D[j]; D[j]:=tg;
            tg:=C[i]; C[i]:=C[j]; C[j]:=tg;
            Inc(i); Dec(j);
         End;
    Until i>j;
    If i<R then QS(i,R);
    If j>L then QS(L,j);
End;
Procedure Build;
Var
        X,Y:String;
        i,j:Longint;
Begin
```

```
        Assign(f,fi); Reset(f);
        Readln(f,S);
        VC:=1000000007;
        M:=length(S);
        Readln(f,N);
        P[0]:=1;
        For i:=1 to N do
                Begin
                Readln(f,X); Y:='';
                For j:=1 to length(X) do Y:=Y+T[X[j]]; C[i]:=Length(Y);
                For j:=1 to length(Y) do
                D[i]:=(D[i]*2+Ox(Y[j])) mod VC;
                End;
        For i:=1 to M do H[i]:=(H[i-1]*2+Ox(S[i])) mod VC;
        For i:=1 to M do P[i]:=P[i-1]*2 mod VC;
        QS(1,N);
        Close(f);
End;
Procedure Process;
Var
        i,j:Longint;
        X:Int64;
Begin
        Assign(f,fo); Rewrite(f);
        L[0]:=1;
        For i:=1 to M do
        For j:=1 to N do
                If i+C[j]-1>M then Break
                Else
                Begin
                        X:=(H[i+C[j]-1]-H[i-1]*P[C[j]]+VC*VC) mod VC;
                        If X=D[j] then
                        L[i+C[j]-1]:=(L[i-1]+L[i+C[j]-1]) mod 10000007;
                End;
        Writeln(f,L[M]);
        Close(f);
End;
BEGIN
   Build;
   Process;
END.
```

Độ phức tạp của thuật toán: O(n*m), với n là số từ có nghĩa, m là số kí tự trong xâu mã hóa

3.4. Test và code

https://drive.google.com/drive/folders/1D567hriA-itxBcZ2EVQ2fQwifoxQ-h4q?usp=sharing

4. Bài 4. Chuỗi con xuất hiện k lần

Nguồn: https://vn.spoj.com/problems/DTKSUB/

4.1. Đề bài

Một xâu A[1..m] được gọi là xuất hiện trong xâu B[1..n] đúng k lần khi và chỉ khi tồn tại k vị trí i phân biệt sao cho A[1..m] = B[i..i+m-1].

Yêu cầu: Cho xâu S có độ dài n và một số nguyên k. Tìm xâu dài nhất xuất hiện ít nhất k lần trong xâu S.

Dữ liệu: Vào từ file văn bản BAI4.INP

- Dòng 1 chứa hai số nguyên n và k ($1 \leq n \leq 50000$; $1 \leq k \leq 200$);
- Dòng 2 chứa xâu S có độ dài n (gồm các chữ cái in thương viết liên tiếp nhau).

Kết quả: Ghi vào file văn bản BAI4.OUT một số duy nhất là độ dài của xâu dài nhất xuất hiện ít nhất k lần trong xâu S.

Ví dụ:

BAI4.INP	BAI4.OUT
5 2 xxxxx	4

4.2. Hướng dẫn giải thuật

- Gọi Hash[i] là mã hash của đoạn xâu mã hóa S[1..i]
- Gọi T[z] là số đoạn xâu khác nhau có mã hash là z
- Get(i,i+d-1) lấy mã hash của đoạn xâu có độ dài d bắt đầu từ vị trí i
- Chặt nhị phân theo độ dài của xâu S:
    - Với độ dài cần xét là d: Ta cần kiểm tra xem có đủ k đoạn xâu khác nhau có độ dài d trên xâu S có cùng mã Hash hay không.

```
For i:=1 to N-d+1 do
        Begin
                z:=Get(i,i+d-1);
                inc(T[z]);
                if T[z]>=k then Exit(True);
        End;
Exit(False);
```

    - Tìm d lớn nhất thỏa mãn điều kiện trên.

4.3. Chương trình tham khảo

```
Const fi='bai4.inp'; fo='bai4.out';
VC=1000007;
Var f:text;
   N,K:longint;
   X:Ansistring;
```

```
   Hash,P:array[0..100000] of int64;
   T:array[0..1000007] of longint;
Procedure Build;
Var i:longint;
Begin
Assign(f,fi); Reset(f);
Readln(f,N,K);
Readln(f,X);
For i:=1 to N do Hash[i]:=(Hash[i-1]*26+ord(X[i])-97) mod VC;
P[0]:=1;
For i:=1 to N do P[i]:=(P[i-1]*26) mod VC;
Close(f);
End;
Function Get(i,j:longint):int64;
Begin
    Exit((Hash[j]-Hash[i-1]*P[j-i+1]+VC*VC) mod VC);
End;
Function kt(d:longint):Boolean;
Var i,l:longint;
   ha:int64;
Begin
    For i:=1 to N-d+1 do
     Begin
        ha:=Get(i,i+d-1);
        Inc(T[ha]);
        If T[ha]>=k then Exit(True);
     End;
    Exit(False);
End;
Procedure Access;
Var d,c,tg:longint;
Begin
    Assign(f,fo); Rewrite(f);
    d:=1; c:=N-K+1;
    While d<=c do
     Begin
        tg:=(d+c) div 2;
        If kt(tg) then d:=tg+1
        Else          c:=tg-1;
        Fillchar(T,sizeof(T),0);
     End;
```

```
    Writeln(f,c);
    Close(f);
End;
BEGIN
   Build;
   Access;
END.
```

Độ phức tạp của thuật toán: $O(n\log_2 n)$

4.4. Test và code

https://drive.google.com/drive/folders/1D567hriA-itxBcZ2EVQ2fQwifoxQ-h4q?usp=sharing

5. Bài 5. Xâu đối xứng dài nhất

Nguồn: https://vn.spoj.com/problems/PALINY/

5.1. Đề bài

Cho xâu S chỉ gồm các kí tự chữ cái Latin thường.

Yêu cầu: Tìm xâu đối xứng dài nhất gồm các kí tự liên tiếp trong xâu S.

Dữ liệu: Vào từ file văn bản BAI5.INP

- Dòng 1 chứa số nguyên n là số kí tự của xâu S ($1 \le n \le 50000$);
- Dòng 2 chứa xâu ký tự S.

Kết quả: Ghi vào file văn bản BAI5.OUT gồm một số duy nhất là độ dài của xâu đối xứng dài nhất.

Ví dụ:

BAI5.INP	BAI5.OUT
5 abacd	3

5.2. Hướng dẫn giải thuật

- Gọi Ha[i] là mã hash của đoạn xâu con S[1..i], xét xâu từ trái qua phải
- Gọi Hb[i] là mã hash của đoạn xâu con S[n..i], xét xâu từ phải qua trái
- Một đoạn xâu con có độ dài là d, bắt đầu từ vị trí i trên xâu S là đối xứng nếu:

Mã hash(S[i..i+d-1]) = Mã hash(S[i+d-1..i])

Ha[i+length-1]-Ha[i-1]*P[length] = Hb[i]-Hb[i+length]*P[length]

- Chặt nhị phân theo độ dài của xâu S:
  - Với độ dài cần xét là d, ta cần kiểm tra có đoạn xâu con độ dài d trên xâu S là đối xứng.

```
For i:=1 to N-d+1 do
    Begin
      A:=(Ha[i+d-1]-Ha[i-1]*P[d] +VC*VC) mod VC;
      B:=(Hb[i]-Hb[i+d]*P[d] +VC*VC) mod VC;
      If A=B then Exit(True);
    End;
Exit(False);
```

  - Tìm d lớn nhất thỏa mãn điều kiện trên

5.3. Chương trình tham khảo

```
Const fi='bai5.inp'; fo='bai5.out';
Var f:Text;
   N:longint;
   X:ansistring;
   VC:Int64;
   P,Ha,Hb:Array[0..1000000] of Int64;
Procedure Build;
Var i:longint;
Begin
     Assign(f,fi); Reset(f);
     Readln(f,N); VC:=1000000007;
     Readln(f,X); P[0]:=1;
     For i:=1 to N do
      Begin
         Ha[i]:=(Ha[i-1]*26+Ord(X[i])-97) mod VC;
         Hb[N-i+1]:=(Hb[N-i+2]*26+Ord(X[N-i+1])-97) mod VC;
         P[i]:=P[i-1]*26 mod VC;
      End;
     Close(f);
End;
Function kt(length:longint):Boolean;
Var i:longint;
    A,B:Int64;
Begin
     For i:=1 to N-length+1 do
      Begin
         A:=(Ha[i+length-1]-Ha[i-1]*P[length] +VC*VC) mod VC;
         B:=(Hb[i]-Hb[i+length]*P[length] +VC*VC) mod VC;
         If A=B then Exit(True);
      End;
     Exit(False);
End;
Procedure C2;
Var d,c,tg:longint;
Begin
     Assign(f,fo); Rewrite(f);
     d:=1; c:=N;
     While d<=c do
      Begin
         tg:=(d+c) div 2;
```

```
        If kt(tg) or kt(tg+1) then d:=tg+1
        Else                  c:=tg-1;
    End;
    Writeln(f,c);
    Close(f);
End;
BEGIN
   Build;
   C2;
END.
```

5.4. Test và code

https://drive.google.com/drive/folders/1D567hriA-itxBcZ2EVQ2fQwifoxQ-h4q?usp=sharing

6. Bài 6. Xử lý xâu

Nguồn: https://codeforces.com/group/FLVn1Sc504/contest/271720/problem/D

6.1. Đề bài

Bờm là một học sinh chuyên tin. Hôm nay Bờm được thầy dạy về thứ tự từ điển và các bài toán liên quan. Sau một hồi giảng giải và định nghĩa thứ tự từ điển là gì, thầy lấy ngay một ví dụ cho lớp. Thầy viết lên bảng 2 chuỗi kí tự dài ơi là dài, và hỏi cả lớp "Chuỗi thứ nhất có thứ tự từ điển như thế nào đối với chuỗi thứ hai: đứng trước ('<'), đứng sau ('>') hay bằng nhau ('=') ???".

Cả lớp thì đang hoang mang, vì cũng chẳng có ai hiểu được định nghĩa "Thứ tự từ điển là gì?" của thầy, nói gì đến việc giải bài tập. Nhưng Bờm thì ngược lại, do đã chuẩn bị và xem bài trước ở nhà nên đã trả lời ngay được câu hỏi của thầy sau khi thấy vừa dứt lời. Bờm ngồi chơi trong lúc mọi người đang thảo luận xôn xao, nên đã tạo thêm một số ví dụ nữa về thứ tự từ điển để có thể hiểu sâu thêm về bài học. Nhìn ngay lên bảng, Bờm phát hiện từ 2 xâu trong ví dụ của thầy, Bờm có thể tự sinh ra rất nhiều ví dụ khác. Cụ thể hơn, Bờm chọn một xâu con trong xâu thứ nhất và một xâu con trong xâu thứ hai, thế là có ngay một cặp xâu để mà so sánh. Xâu con ở đây được hiểu là một dãy các ký tự liên tiếp.

Thế là Bờm liên tục sinh ra các ví dụ và trả lời chúng. Bờm càng làm càng nhạy, và trả lời các câu hỏi về thứ tự từ điển càng nhanh. Đến nỗi trong 1 giây Bờm đã có thể trả lời đến tất cả là 10^6 câu hỏi!

Yêu cầu:

Cho 2 xâu A và B chỉ chứa các kí tự chữ cái Latin thường và một danh sách gồm q câu hỏi có dạng (x, y, u, v), với ý nghĩa cần so sánh thứ tự từ điển của xâu con A[x..y] và B[u..v]. Các kí tự của một xâu được đánh số từ trái qua phải, bắt đầu bằng 1; và ký hiệu A[x..y] thể hiện xâu con từ kí tự thứ x đến y của xâu A.

Bạn hãy viết một chương trình mô tả lại hoạt động trả lời các câu hỏi của Bờm.

Lưu ý:

Xâu $a_1, a_2, ..., a_n$ (a_i là kí tự thứ i của xâu A) có thứ tự từ điển nhỏ hơn xâu $b_1, b_2, ..., b_m$ nếu:

- $n < m$ và $a_i = b_i$ với mọi i ($1 \le i \le n$) hoặc
- với k ($1 \le k \le \min(m, n)$) là giá trị nhỏ nhất thỏa $a_k \ne b_k$ thì $a_k < b_k$.
- Hai xâu có thứ tự từ điển bằng nhau nếu không thể xác định được xâu nào có thứ tự từ điển nhỏ hơn.

Dữ liệu: Vào từ file văn bản BAI6.INP

- Dòng đầu chứa hai số nguyên dương n, m là độ dài tương ứng của hai xâu A và B;
- Dòng thứ hai chứa xâu A;
- Dòng thứ ba chứa xâu B;
- Dòng thứ tư chứa số nguyên dương q là số câu hỏi trong danh sách;
- Trong q dòng tiếp theo, mỗi dòng gồm 4 số nguyên dương x, y, u, v ($1 \le x \le y \le n$; $1 \le u \le v \le m$) mô tả một câu hỏi cần trả lời.

Kết quả: Ghi vào file văn bản BAI6.OUT một dòng gồm q kí tự hoặc là '=' hoặc là '>' hoặc là '<' liên tiếp nhau tương ứng với q câu trả lời.

Ví dụ:

BAI6.INP	BAI6.OUT
13 14 bomthichdacau bomthichdabanh 3 1 10 1 10 1 10 1 11 1 11 1 11	=<>

Giới hạn: n, m, q $\le 10^6$

6.2. Hướng dẫn giải thuật

- Gọi hasha[i] là mã hash của đoạn xâu con A[1..i], xét xâu từ trái qua phải
- Gọi hashb[i] là mã hash của đoạn xâu con B[1..i], xét xâu từ trái qua phải
- gethasha(x,x+k-1) lấy giá trị mã hash của đoạn xâu con có độ dài k từ vị trí x trên xâu A.
- gethashb(u,u+k-1) lấy giá trị mã hash của đoạn xâu con có độ dài k từ vị trí u trên xâu B.

Với mỗi truy vấn (x, y, u, v):

- Đặt n=y-x+1; m=v-u+1
- Tìm số k lớn nhất mà gethasha(x,x+k-1) = gethashb(u,u+k-1), với $0 \le k \le \min(n,m)$
- Nếu k= min(n,m):
  - ✓ Nếu n=m thì kết quả là dấu '='
  - ✓ Nếu n<m thì kết quả là dấu '<'
  - ✓ Nếu n > m thì kết quả là dấu '>'
- Nếu k < min(n,m):
  - ✓ Nếu A[x+k]<B[u+k] thì kết quả là dấu '<'

✓ Nếu A[x+k]>B[u+k] thì kết quả là dấu '>'

6.3. Chương trình tham khảo

```
const m=round(1e9)+7;
var a,b:ansistring; x,y,la,lb,ra,rb:longint;
   hasha,hashb,pow:array[0..1000005] of int64;
procedure doc;
 begin
        readln(x,y);
        readln(a);
        readln(b);
 end;
procedure khoitao;
 var i:longint;
 begin
        hashb[0]:=0;
        for i:=1 to y do hashb[i]:=(26*hashb[i-1]+ord(b[i])-97) mod m;
        hasha[0]:=0;
        for i:=1 to x do hasha[i]:=(26*hasha[i-1]+ord(a[i])-97) mod m;
        pow[0]:=1;
        for i:=1 to x do pow[i]:=pow[i-1]*26 mod m;
 end;
function gethasha(i,j:longint):int64;
 begin
        exit((hasha[j]-hasha[i-1]*pow[j-i+1]+m*m) mod m);
 end;
function gethashb(i,j:longint):int64;
 begin
        exit((hashb[j]-hashb[i-1]*pow[j-i+1]+m*m) mod m);
 end;
function minxy(x,y:longint):longint;
 begin
        if x<y then exit(x) else exit(y);
 end;
function cmp:char;
 var d,c,g,l,k:longint;
 begin
        k:=0;
        l:=minxy(ra-la+1,rb-lb+1);
        d:=1; c:=l;
        while d<=c do
         begin
```

```
                g:=(d+c) div 2;
                if gethasha(la,la+g-1)=gethashb(lb,lb+g-1) then
                        begin
                                k:=g;
                                d:=g+1;
                        end
                        else c:=g-1;
        end;
        if k=l then
                if ra-la=rb-lb then exit('=')
                 else
                        if ra-la<rb-lb then exit('<')
                         else exit('>')
        else
                if a[la+k]<b[lb+k] then exit('<')
                 else exit('>');
end;
procedure xuli;
 var i,n:longint;
 begin
        readln(n);
        for i:=1 to n do
                begin
                        readln(la,ra,lb,rb);
                        write(cmp);
                end;
 end;
begin
        assign(input,'bai6.inp'); reset(input);
        assign(output,'bai6.out'); rewrite(output);
        doc;
        khoitao;
        xuli;
end.
```

Độ phức tạp của thuật toán: $O(q*\log_2(m,n))$

6.4. Test và code

https://drive.google.com/drive/folders/1D567hriA-itxBcZ2EVQ2fQwifoxQ-h4q?usp=sharing

C. PHẦN KẾT LUẬN

Chuyên đề này đã giúp bản thân tôi tìm hiểu sâu hơn kiến thức về xử lý xâu, vận dụng cấu trúc dữ liệu bảng băm trong giải quyết các bài toán so khớp xâu và giúp học sinh tìm hiểu thêm một luồng kiến thức mới hữu ích cho bản thân.

Mặc dù đã có nhiều thời gian tìm hiểu, nghiên cứu nhưng do trình độ của bản thân có hạn nên việc thực hiện nghiên cứu còn nhiều hạn chế rất mong được sự góp ý chân thành của đồng nghiệp, thầy cô giáo để nghiên cứu của tôi được hoàn thiện hơn.

Nhân đây, tôi cũng xin gửi lời cảm ơn tới qúy thầy cô, qúy bạn đồng nghiệp đã giúp đỡ, trao đổi tài liệu và góp ý để tôi hoàn thiện chuyên môn của bản thân hơn.

Tôi chân thành cảm ơn!

D. TÀI LIỆU THAM KHẢO

1. Tài liệu giáo khoa chuyên Tin tập 1, 2, 3.

2. Website: http://vn.spoj.com/

3. Website: http://codeforces.com

4. https://vi.wikipedia.org/wiki/B%E1%BA%A3ng_b%C4%83m

5. https://vnoi.info/wiki/algo/data-structures/hash-table.md