|  |
| --- |
| **HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**  **KHOA CÔNG NGHỆ THÔNG TIN 2**  🙞🙞🙞🕮🙜🙜🙜    **BÁO CÁO CHUYÊN ĐỀ KHOA HỌC**  **CẤP HỌC VIỆN**  **Các phương pháp tối ưu thuật toán**  **Báo cáo viên: ThS. Lưu Nguyễn Kỳ Thư**  **Năm học: 2015-2016**  *Thành phố Hồ Chí Minh, tháng 10/2016* |

**Phần 1: Giới thiệu mở đầu**

Mục tiêu của chuyên đề:

1. Nghiên cứu các phương pháp tối ưu thuật toán theo 2 tiêu chí: tiết kiệm bộ nhớ và giảm thời gian thực thi .
2. Nội dung:   
   - Tìm hiểu cách thức lưu trữ dữ liệu sao cho tiết kiệm bộ nhớ;

* Tìm hiểu các phương pháp để tối ưu thời gian thực thi của thuật toán

c. Kết quả: Các chương trình minh họa các phương pháp tối ưu.

**Phần 2: Nội dung chuyên đề khoa học**

1. **Giới thiệu :** Khi viết chương trình để giải quyết 1 bài toán với lượng dữ liệu lớn, vấn đề cần giải quyết là thời gian thực thi ; và bởi vì lượng dữ liệu lớn nên ta lại phải nghĩ cách để lưu dữ liệu vào bộ nhớ trong sao cho tiết kiệm nhất.
2. **TỐI ƯU HÓA CHƯƠNG TRÌNH**

***1. Tinh chế cách viết*:**

**Bài toán 1** : Tính biểu thức E = 1/1! + 3/2! + 5/3! + … + (2n-1)/n!

|  |  |
| --- | --- |
| Giải thuật A | Giải thuật B |
|  | Tối ưu chương trình bằng cách giảm các vòng lặp lồng nhau. |
| E:=0;  For i:=1 to n do  Begin  Gt:=1;  For j:=1 to i do Gt:=Gt\*j;  E:= E +(2\*i-1)/Gt;  End; | E:=0;  Gt:=1;  For i:=1 to n do  Begin  Gt:=Gt\*i;  E:= E +(2\*i-1)/Gt;  End; |

Độ phức tạp của giải thuật A là n+(1+n)n/2 +1 có cấp O(n2)

Độ phức tạp của giải thuật B là 2\*n +2 có cấp là O(n)

**Bài toán 2**: Nhân 2 ma trận vuông

|  |  |
| --- | --- |
| **Thuật toán A** | **Thuật toán B** |
|  | Tối ưu chương trình bằng cách tăng số lệnh thực hiện trong một bước vòng lặp để giảm số bước vòng lặp. |
| void gt1(Day a,Day b, Day c)  { clock\_t t1, t2;  t1= clock();  int i,j,k;  for ( i =1 ; i <=MAX ;i++)  for ( j =1 ; j <=MAX ; j++)  { c[i][j]=0;  for ( k =1 ; k<=MAX ; k++)  c[i][j]+= a[i][k]\*b[k][j];  }  } | void gt2(Day a,Day b, Day c)  {  clock\_t t1, t2;  t1= clock();  int i,ii,jj,j,k,kk ;  int n=MAX;  if (n % 2 == 1) n++;  for (i=1; i<n ;i+=2)  {  ii=i+1;  for (j=1; j<n ;j+=2)  {  jj=j+1;  c[i][j]=0; c[i][jj]=0;  c[ii][j]=0; c[ii][jj]=0;  for (k=1; k<n ;k+=2)  { kk=k+1;  c[i][j]+= a[i][k]\*b[k][j] + a[i][kk]\*b[kk][j];  c[i][jj]+= a[i][k]\*b[k][jj] + a[i][kk]\*b[kk][jj];  c[ii][j]+= a[ii][k]\*b[k][j] + a[ii][kk]\*b[kk][j];  c[ii][jj]+= a[ii][k]\*b[k][jj] + a[ii][kk]\*b[kk][jj];  }  }// j  } // I  } |

**3. TINH CHẾ THUẬT TOÁN**

**Bài toán 1** : Chương trình tính giá trị của đa thức:

Y = an.xn + an-1. xn-1 + … + a1.x + a0=

= (an(((x+an-1) + ……. )x+ a0

|  |  |
| --- | --- |
| **Giải thuật A** | **Giải thuật B** |
| Y:= a[0];  Xtam:=1;  For i:=1 to n do  Begin Xtam:=X \* Xtam;  Ytam:= a[i ]\*Xtam;  Y:= y + Ytam;  End; | Y:= a[n];  For i:=n-1 downto 0 do  Begin  Ytam:=X\*Y;  Y:= Ytam + a[i];  End; |

Giải thuật A có độ phức tạp : 3\*n +2;

Phân tích : Y= ((((an.X+ an-1).X + an-2)X…)+ á1).X + a0 ta được thuật toán B

Giải thuật B có độ phức tạp là : 2n+1 tức nhanh hơn giải thuật A

Giải thuật B tinh giảm được số phép gán trong vòng lặp. Ngoài ra cả hai giải thuật đều áp dụng phương pháp gán đơn giản.

**Bài toán 2**: Phân loại ký tự

Cho một chuỗi gồm một triệu ký tự, hãy phân loại mỗi ký tự theo 4 kiểu sau: Kiểu chữ thường, kiểu chữ hoa, chữ số và kiểu khác.

Một cách giải thông thường là xác định một ký tự nào đó trong bảng mã ASCII thuộc kiểu "khac" bằng cách sử dụng 6 lần so sánh như sau:

If (kytu>='a') and (kytu<='z') then ký tự là ký tự thường

Else If (kytu>='A') and (kytu<='Z') then ký tự là ký tự hoa

Else If (kytu>='0') and (kytu<='9') then ký tự là ký tự số

Else Ký tự là ký tự khác.

Để tối ưu hóa chương trình, ta thiết lập một bảng truy cập như sau:

for c:=chr(0) to chr(255) do

if c IN ['a'..'z'] then TypeTable[c]:=Kieu\_chu\_thuong

else if c IN ['A'..'Z'] then TypeTable[c]:=Kieu\_chu\_hoa

else if c IN ['0'..'9'] then TypeTable[c]:=Kieu\_chu\_so

else TypeTable[c]:=Kieu\_khac

Như vậy kiểu của ký tự c là TypeTable[c] tránh được việc so sánh.

*Văn bản chương trình:*

Program Phan\_loai\_kieu\_ky\_tu;

Uses crt,dos;

Const Kieu\_khac='Khac';

Kieu\_chu\_thuong='chu thuong';

Kieu\_chu\_hoa='chu hoa';

Kieu\_chu\_so='chu so';

Var TypeTable : array[char] of string[10];

Filename: PathStr;

**Procedure Tao\_bang;**

Var c: char;

begin

for c:=chr(0) to chr(255) do

if c IN ['a'..'z'] then TypeTable[c]:=Kieu\_chu\_thuong

else if c IN ['A'..'Z'] then TypeTable[c]:=Kieu\_chu\_hoa

else if c IN ['0'..'9'] then TypeTable[c]:=Kieu\_chu\_so

else TypeTable[c]:=Kieu\_khac

end;

**Procedure Phan\_loai\_ky\_tu(Filename:PathStr);**

Var S: String; f: text; i: byte;

Begin

assign(f,filename);

{$I-}

reset(f);

{$I+}

If IoResult<> 0 then begin

writeln(#13#10'File ',filename,' khong co');

readln;

halt(0);

end;

While not eof(f) do

begin

readln(f,s);

For i:=1 to length(S) do

writeln(S[i], ' : ', TypeTable[s[i]]);

end;

close(f);

End;

Begin

clrscr;

Tao\_bang;

write('Ten file de thu phan loai : '); readln(filename);

If filename <> '' then

Phan\_loai\_ky\_tu(Filename);

readln;

End.

**Bài toán 3**: Tính số bit

Cho một file gồm một triệu từ 32 bit, hãy đếm tong số bit 1 trong file.

Một lời giải thông thường là sử dụng vòng lặp thực hiện 32 phép toán shift và phép toán logic and.

Để tối ưu hóa, ta thiết lập một bảng truy cập Counttable[0..255] chứa số bit 1 như sau:

Counttable[0]:=0;

Counttable[1]:=1;

Counttable[2]:=1;

…

Counttable[255]:=8;

Như vậy để tính số bit 1 trong từ W (4 bytes), ta dùng lệnh:

WordCount:= CountTable[W and mask]

+ CountTable[(W Rshift 8) and mask]

+ CountTable[(W Rshift 16) and mask]

+ CountTable[(W Rshift 24) and mask]

*Văn bản chương trình:*

Program So\_bit\_1\_cua\_mot\_trieu\_tu;

Uses Crt;

const mask:byte=$FF;

sotu=16000;

Type Bang\_dem = array[0..255] of byte;

Danh\_sach\_tu=array[1..sotu] of LongInt ;

Type str32= String[32];

Var i,sobit1, n:LongInt;

Count: Bang\_dem;

DStu: Danh\_sach\_tu;

**Function Dec\_Bin(n:LongInt):Str32;**

Var kt:String[1]; S: Str32;

Begin

S:='';

repeat

Str(n mod 2,kt);

S:=kt+S;

n:=n div 2;

until n =0;

Dec\_Bin:=S;

end;

**Procedure Xay\_dung\_Bang\_Count(Var Count:Bang\_dem);**

Var i,j:byte; S:Str32;

Begin

For i:=0 to 255 do

Begin

Count[i]:=0;

S:=Dec\_Bin(i);

For j:=1 to Length(S) do

If S[j]='1' then Inc (Count[i]);

End;

End;

**Procedure Phat\_sinh\_tu(Var DStu:Danh\_sach\_tu);**

Var i:Longint;

Begin

Randomize;

i:=1;

Repeat

DStu[i]:=Random($FFFF)\*2;

if dstu[i] < 0 then halt(0);

inc(i);

until i>sotu;

End;

**Procedure In\_ds\_tu(Var dstu:danh\_sach\_tu);**

Var i: LongInt;

Begin

for i:=1 to sotu do

writeln(Dec\_Bin(DStu[i]));

End;

Begin

clrscr; sobit1:=0;

Phat\_sinh\_tu(DStu);

Xay\_dung\_Bang\_Count(Count);

(\* In\_ds\_tu(dstu); \*)

For i:=1 to sotu do

begin

n:=Dstu[i];

sobit1:=sobit1+ Count[n and mask] + Count[(n shr 8) and mask] +

Count[(n shr 16) and mask] +Count[(n shr 24) and mask];

end;

writeln(#13#10,'So bit 1 trong ',sotu,' tu = ',sobit1);

readln;

end.

**Bài toán 4**:

Sắp xếp một dãy số nguyên 20000 số phân biệt khác nhau (các số nguyên đó có trị từ 1..30000) mà bộ nhớ trong chỉ có 1200 từ (4 bytes/từ).

Với bộ nhớ trong hạn chế theo như đề bài, ta không thể lưu trữ hết 20000 số nguyên vào bộ nhớ trong để áp dụng các giải thuật sort nội. Trong trường hợp này, một giải pháp có thể được là lưu 20000 số vào bộ nhớ ngoài và áp dụng giải thuật sort ngoại để sắp xếp chúng. Với cách thức lưu trữ và sắp xếp này, ta sẽ tốn thời gian để truy xuất dữ liệu và sắp xếp.

Do tính chất của bài toán là 20000 số nguyên từ 1..30000 và phân biệt khác nhau, nên một giải thuật tối ưu về cả thời gian và bộ nhớ được đưa ra ở đây là:

- Ta dùng một mảng A có 3750 bytes ≅ 938 từ (4 bytes) chứa 30000 số từ 1..30000.

- Cho vòng lặp chạy từ 1 đến 20000

+ Nhập vào số n ∈ [1..30000]

+ Ta bật bit tương ứng với số n trong A lên 1

- Sau khi nhập xong 20000 số , dãy số của ta cũng đã được sắp xong.

*Văn bản chương trình:*

Program Sap\_xep\_20000\_so\_trong\_3750\_byte; (\* moi so chi xuat hien 1 lan \*)

uses crt;

Const Sobyte=3750; (\* = 30000/8\*)

So\_Max = 30000;

Var A: array [1..Sobyte] of byte; (\* chứa 20000 so tu 1 den So\_Max\*)

X: array[1..30000] of Integer; (\*Mảng chứa các số ngẫu nhiên thế cho file\*)

Procedure Init; (\* Tao cac so ngau nhien cho X \*)

Var cs,t,i : integer;

Function RandInt(A,B:Integer):Integer;

(\* Tao so ngau nhien tren doan [A,B] \*)

Begin

RandInt:= A+ Random(B-A+1);

End;

begin

Randomize;

For i:=1 to So\_Max do X[i]:=i;

For i:=1 to 20000 do

begin cs:= RandInt(i,So\_Max);

t:=X[i];

X[i] := X[cs];

X[cs] :=t;

end;

end;

Procedure Sort;

Var i,csm, vt\_bit : integer;

Begin

For i:=1 to Sobyte do A[i]:=0;

For i:=1 to 20000 do

begin

csm:=( X[i] div 8) +1 ;

vt\_bit:= X[i] mod 8;

Case vt\_bit of

1: A[csm] := A[csm] or $80;

2: A[csm] := A[csm] or $40;

3: A[csm] := A[csm] or $20;

4: A[csm] := A[csm] or $10;

5: A[csm] := A[csm] or $08;

6: A[csm] := A[csm] or $04;

7: A[csm] := A[csm] or $02;

0: A[csm-1] := A[csm-1] or $01;

End;

end;

End;

Procedure Lietke;

Var i, j: Integer; mask,page: byte; c: char;

Begin page:=1;

gotoxy(1,1) ;writeln('Page ',page:3);

For i:=1 to Sobyte do

begin

mask:=$80;

For j:=1 to 8 do

begin

if (A[i] and mask) = mask then

write ((i-1)\*8+j:8);

if wherey = 24 then begin c:=readkey; clrscr;

inc(page);

gotoxy(1,1) ;writeln('Page ',page:3);

end;

mask := mask shr 1

end;

End;

End;

BEGIN

clrscr;

Init;

Sort; (\* Lấy dữ liệu từ X và bật bit tương ứng trong A lên 1 \*)

Lietke; (\* In dãy số trong A ra màn hình. \*)

readln;

END.

**Phần 3: Kết luận, đề xuất**

* Chuyên đề đã hệ thống các phương pháp để cải tiến thuật toán tối ưu thời gian thực thi
* Chỉ tối ưu theo thuật toán đã có, và đưa phương pháp tối ưu vào
* Không đề cập đến tối ưu bằng cách giải bài toán theo thuật toán khác.

Đề xuất (khuyến nghị) hướng vận dụng kết quả của báo cáo chuyên đề: kết quả nghiên cứu có thể áp dụng trong việc đào tạo, giới thiệu cho sinh viên biết trong phần lập trình.