

CHƯƠNG 14 : TỐI ƯU HOÁ

§1. PHƯƠNG PHÁP TỈ LỆ VÀNG

Trong chương 8 chúng ta đã xét bài toán tìm nghiệm của phương trình phi tuyến tức là tìm giá trị của x mà tại đó hàm triệt tiêu. Trong phần này chúng ta sẽ đặt vấn đề tìm giá trị của x mà tại đó hàm đạt giá trị cực trị (cực đại hay cực tiểu). Phương pháp tiết diện vàng là một phương pháp đơn giản và hiệu quả để tìm giá trị cực trị của hàm.

Giả sử ta có hàm $y = f(x)$ và cần tìm giá trị cực trị trong khoảng $[a, b]$. Khi tìm nghiệm chỉ cần biết 2 giá trị của hàm là ta khẳng định được nghiệm có nằm trong khoảng đã cho hay không bằng cách xét dấu của hàm. Khi tìm giá trị cực trị ta phải biết thêm một giá trị nữa của hàm trong khoảng $[a, b]$ thì mới khẳng định được hàm có đạt cực trị trong đoạn đã cho hay không. Sau đó ta chọn thêm một điểm thứ tư và xác định xem giá trị cực trị của hàm sẽ nằm trong đoạn nào.

Theo hình vẽ, khi chọn điểm trung gian c ta có :

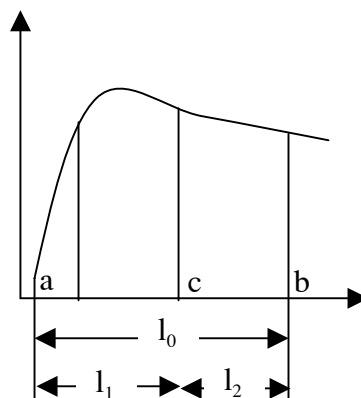
$$l_1 + l_2 = l_0 \quad (1)$$

và để tiện tính toán ta chọn :

$$\frac{l_1}{l_0} = \frac{l_2}{l_1} \quad (2)$$

Thay thế (1) vào (2) ta có :

$$\frac{l_1}{l_1 + l_2} = \frac{l_2}{l_1} \quad (3)$$



Gọi $r = \frac{l_2}{l_1}$, ta nhận được phương trình :

$$1 + r = \frac{1}{r} \quad (4)$$

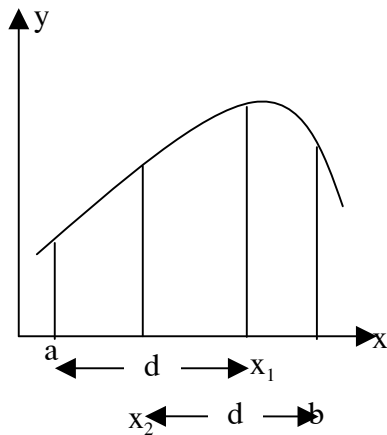
hay : $r^2 + r - 1 = 0 \quad (5)$

Nghiệm của phương trình (5) là :

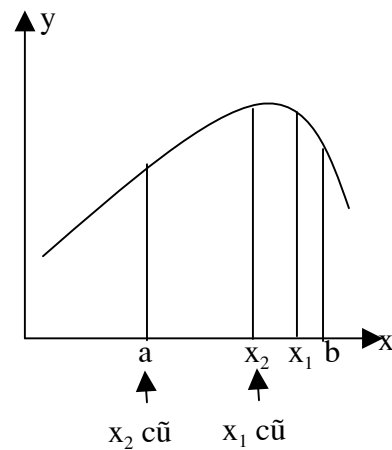
$$r = \frac{-1 + \sqrt{1 - 4(-1)}}{2} = \frac{\sqrt{5} - 1}{2} = 0.61803... \quad (6)$$

Giá trị này đã được biết từ thời cổ đại và được gọi là “tỉ lệ vàng”. Như trên đã nói, phương pháp tỉ lệ vàng được bắt đầu bằng 2 giá trị đã cho của biến x là a và b . Sau đó ta chọn 2 điểm x_1 và x_2 bên trong khoảng $[a, b]$ theo tỉ lệ vàng:

$$d = \frac{\sqrt{5} - 1}{2} = 0.61803...$$



a



b

Ta tính giá trị của hàm tại các điểm bên trong đoạn $[a, b]$. Kết quả có thể là một trong các khả năng sau :

1. Nếu, như trường hợp hình a, $f(x_1) > f(x_2)$ thì giá trị cực trị của hàm nằm trong $[x_2, b]$ và x_2 trở thành a và ta tính tiếp.
2. Nếu $f(x_1) < f(x_2)$ thì giá trị cực trị của hàm nằm trong $[a, x_1]$ và x_1 trở thành b và ta tính tiếp.

Cái lợi của phương pháp tỉ lệ vàng theo hình a là giá trị x_1 cũ trở thành giá trị x_2 mới nên giá trị $f(x_2)$ mới chính là giá trị $f(x_1)$ cũ nên ta không cần tính lại nó. Chương trình mô tả thuật toán trên như sau:

Chương trình 14-1

```
//tiet_dien_vang;
#include <conio.h>
#include <stdio.h>
#include <math.h>
```

```
float eps=1e-6;
```

```
float f(float x)
{
    float a=2*sin(x)-x*x/10;
    return(a);
};
```

```
float max(float xlow, float xhigh)
{
    float xl, xu, r, d, x1, x2, f1, f2, xopt, s;
    int lap;
    xl=xlow;
    xu=xhigh;
    lap=1;
```

```

r=(sqrt(5.0)-1.0)/2.0;
d=r*(xu-xl);
x1=xl+d;
x2=xu-d;
f1=f(x1);
f2=f(x2);
if (f1>f2)
    xopt=x1;
else
    xopt=x2;
do
{
    d=r*d;
    if (f1>f2)
    {
        x1=x2;
        x2=x1;
        x1=x1+d;
        f2=f1;
        f1=f(x1);
    }
    else
    {
        xu=x1;
        x1=x2;
        x2=xu-d;
        f1=f2;
        f2=f(x2);
    }
    lap=lap+1;
    if (f1>f2)
        xopt=x1;
    else
        xopt=x2;
    if (xopt!=0)
        s=(1.0-r)*fabs((xu-xl)/xopt)*100;
}
while((s>eps)&&(lap<=20));
float k=xopt;
return(k);
}

```

```

float min(float xlow,float xhigh)
{
    float xl,xu,r,d,x1,x2,f1,f2,fx,xopt,s;
    int lap;
    xl=xlow;

```

```

    xu=xhigh;
    lap=1;
    r=(sqrt(5.0)-1.0)/2,0;
    d=r*(xu-xl);
    x1=xl+d;
    x2=xu-d;
    f1=f(x1);
    f2=f(x2);
    if (f1<f2)
        xopt=x1;
    else
        xopt=x2;
    do
    {
        d=r*d;
        if (f1<f2)
        {
            xl=x2;
            x2=x1;
            x1=xl+d;
            f2=f1;
            f1=f(x1);
        }
        else
        {
            xu=x1;
            x1=x2;
            x2=xu-d;
            f1=f2;
            f2=f(x2);
        }
        lap=lap+1;
        if (f1<f2)
            xopt=x1;
        else
            xopt=x2;
        if (xopt!=0)
            s=(1.0-r)*fabs((xu-xl)/xopt)*100;
    }
    while ((s>eps)&&(lap<=20));
    float r1=xopt;
    return(r1);
}

void main()
{
    float x,y,xlow,xhigh,eps;

```

```

clrscr();
printf("TIM CUC TRI CUA HAM BANG PHUONG PHAP TIET DIEN VANG\n");
printf("\n");
printf("Cho khoang can tim cuc tri\n");
printf("Cho can duoi a = ");
scanf("%f",&xlow);
printf("Cho can tren b = ");
scanf("%f",&xhigh);

if (f(xlow)<f(xlow+0.1))
{
    x=max(xlow,xhigh);
    y=f(x);
    printf("x cuc dai = %10.5f\n",x);
    printf("y cuc dai = %10.5f\n",y);
}
else
{
    x=min(xlow,xhigh);
    y=f(x);
    printf("x cuc tieu = %10.5f y cuc tieu = %10.5f",x,y);
}
getch();
}

```

Trong chương trình này ta cho $a = 0$; $b = 4$ và tìm được giá trị cực đại $y = 1.7757$ tại $x = 1.4276$

§2.PHƯƠNG PHÁP NEWTON

Khi tính nghiệm của phương trình $f(x) = 0$ ta dùng công thức lặp Newton-Raphson :

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

Một cách tương tự, để tìm giá trị cực trị của hàm $f(x)$ ta đặt $g(x)=f'(x)$. Như vậy ta cần tìm giá trị của x để $g(x) = 0$. Như vậy công thức lặp Newton-Raphson sẽ là :

$$x_{i+1} = x_i - \frac{g(x_i)}{g'(x_i)} = x_i - \frac{f'(x_i)}{f''(x_i)}$$

Các đạo hàm $f'(x_i)$ và $f''(x_i)$ được xác định theo các công thức :

$$f'(x_i) = \frac{f(x_i + h) - f(x_i - h)}{2h}$$

$$f''(x_i) = \frac{f(x_i + h) - 2f(x_i) + f(x_i - h)}{h^2}$$

Tại giá trị $f'(x) = 0$ hàm đạt giá trị cực đại nếu $f''(x) < 0$ và cực tiểu nếu $f''(x) > 0$. Chương trình sau mô tả thuật toán trên.

Chương trình 14-2

```
//Phuong phap New_ton;
#include <conio.h>
#include <stdio.h>
#include <math.h>
#include <stdlib.h>

float f(float x)
{
    float a=2*sin(x)-x*x/10;
    return(a);
}

float f1(float x)
{
    float a=2*cos(x)-x/5.0;
    return(a);
}

float f2(float x)
{
    float a=-2*sin(x)-1.0/5.0;
    return(a);
}

void main()
{
    float a,eps,x[50],y1,t;
    clrscr();
    printf("TINH CUC TRI BANG PHUONG PHAP NEWTON\n");
    printf("\n");
    printf("Cho diem bat dau tinh a = ");
    scanf("%f",&a);
    eps=1e-6;
    int i=1;
    x[i]=a;
    do
    {
        x[i+1]=x[i]-f1(x[i])/f2(x[i]);
        t=fabs(x[i+1]-x[i]);
        x[i]=x[i+1];
        i++;
        if (i>1000)
        {
            printf("Khong hoi tu sau 1000 lan lap");
            getch();
        }
    }
}
```

```

        exit(1);
    }
}
while (t>=eps);
printf("\n");
y1=f2(x[i]);
if (y1>0)
    printf("x cuc tieu = %10.5f y cuc tieu = %10.5f",x[i],f(x[i]));
else
    printf("x cuc dai = %10.5f y cuc dai = %10.5f",x[i],f(x[i]));
    getch();
}

```

Ta có kết quả $x = 1.42755, y = 1.77573$

§3. PHƯƠNG PHÁP PARABOL

Nội dung của phương pháp parabol là ta thay đường cong $y = f(x)$ bằng một đường cong parabol mà ta dễ dàng tìm được giá trị cực trị của nó. Như vậy trong khoảng $[a, b]$ ta chọn thêm một điểm x bất kỳ và xấp xỉ hàm $f(x)$ bằng parabol qua 3 điểm a, x , và b . Sau đó ta đạo hàm và cho nó bằng 0 để tìm ra điểm cực trị của parabol này. Giá trị đó được tính bằng công thức:

$$x_1 = \frac{f(a)(x^2 - b^2) + f(x)(b^2 - a^2) + f(b)(a^2 - x^2)}{2f(a)(x - b) + 2f(x)(b - a) + 2f(b)(a - x)}$$

Sau đó tương tự phương pháp tỉ lệ vàng ta loại trừ vùng không chứa giá trị cực trị và tiếp tục quá trình trên cho đến khi đạt độ chính xác mong muốn. Chương trình được viết như sau:

Chương trình 14-3

```

//phuong phap parabol
#include <conio.h>
#include <stdio.h>
#include <math.h>

float f(float x)
{
    float f1=2*sin(x)-x*x/10;
    return(f1);
}

void main()
{
    float a,b,x0,x1,x2,x3,f3;

    clrscr();
    printf("TIM CUC TRI BANG PHUONG PHAP PARABOL\n");
    printf("\n");

```

```

printf("Cho doan can tim cuc tri [a,b]\n");
printf("Cho diem dau a = ");
scanf("%f",&a);
printf("Cho diem cuoi b = ");
scanf("%f",&b);
x0=a;
x2=b;
x1=(x0+x2)/4;
do
{
    x3=(f(x0)*(x1*x1-x2*x2)+f(x1)*(x2*x2-x0*x0)+f(x2)*(x0*x0-x1*x1))
/(2*f(x0)*(x1-x2)+2*f(x1)*(x2-x0)+2*f(x2)*(x0-x1));
    f3=f(x3);
    if (x3>x1)
        x0=x1;
    else
        x2=x1;
    x1=x3;
}
while (fabs(x2-x0)>1e-5);
printf("\n");
f3=(f(x2+0.01)-2*f(x2)+f(x2-0.01))/(0.01*0.01);
if (f3<0)
    printf("x cuc dai = %10.5f  y cuc dai = %10.5f",x2,f(x2));
else
    printf("x cuc tieu = %10.5f  y cuc tieu = %10.5f",x2,f(x2));
getch();
}

```

Chạy chương trình này với $a = 0$ và $b = 4$ ta có x cực đại là 1.42755 và y cực đại là 1.77573.

§4. PHƯƠNG PHÁP ĐƠN HÌNH(SIMPLEX METHOD)

Trong thực tế nhiều bài toán kinh tế, vận tải có thể được giải quyết nhờ phương pháp quy hoạch tuyến tính. Trước hết ta xét bài toán lập kế hoạch sản xuất sau:

Một công ty muốn sản xuất 2 loại sản phẩm mới là A và B bằng các nguyên liệu 1, 2, và 3. Suất tiêu hao nguyên liệu để sản xuất các sản phẩm cho ở bảng sau:

	Sản phẩm A	Sản phẩm B
Nguyên liệu 1	2	1
Nguyên liệu 2	1	2
Nguyên liệu 3	0	1

Số liệu này cho thấy để sản xuất một đơn vị sản phẩm A cần dùng 2 đơn vị nguyên liệu 1, một đơn vị nguyên liệu 2 và để sản xuất một đơn vị sản phẩm B cần dùng 1 đơn vị

nguyên liệu 1, hai đơn vị nguyên liệu 2, 1 đơn vị nguyên liệu 3. Trong kho của nhà máy hiện có dự trữ 8 đơn vị nguyên liệu 1, 7 đơn vị nguyên liệu 2 và 3 đơn vị nguyên liệu 3. Tiền lãi một đơn vị sản phẩm A là 4.000.000 đ, một đơn vị sản phẩm B là 5.000.000 đ. Lập kế hoạch sản xuất sao cho công ty thu được tiền lãi lớn nhất.

Bài toán này là bài toán tìm cực trị có điều kiện. Gọi x_1 là lượng sản phẩm A và x_2 là lượng sản phẩm B ta đi đến mô hình toán học:

$$f(x) = 4x_1 + 5x_2 \rightarrow \max$$

với các ràng buộc : $2x_1 + x_2 \leq 8$ (ràng buộc về nguyên liệu 1)

$$x_1 + 2x_2 \leq 7 \quad (\text{ràng buộc về nguyên liệu 2})$$

$$x_2 \leq 3 \quad (\text{ràng buộc về nguyên liệu 3})$$

$$x_1 \geq 0, x_2 \geq 0$$

Một cách tổng quát ta có bài toán được phát biểu như sau : Cho hàm mục tiêu $C^T X \rightarrow \max$ với điều kiện ràng buộc $AX \leq B$ và $X \geq 0$. Thuật toán để giải bài toán gồm hai giai đoạn

- tìm một phương án cực biên một đỉnh
- kiểm tra điều kiện tối ưu đối với phương án tìm được ở giai đoạn 1. Nếu điều kiện tối ưu được thỏa mãn thì phương án đó là tối ưu. Nếu không ta chuyển sang phương án mới.

Chương trình giải bài toán được viết như sau :

Chương trình 14-4

```
//simplex;
```

```
#include <conio.h>
```

```
#include <stdio.h>
```

```
int m,n,n1,it,i,j,h1,h2,hi,m1,ps,pz,v,p;
```

```
float bv[20];
```

```
float a[20][20];
```

```
float h,mi,x,z;
```

```
void don_hinh()
```

```
{
    int t;
    float hi;
    if (p!=2)
        for (i=1;i<=m;i++)
            bv[i]=n+i;
    if (p==2)
    {
        h1=n;
        h2=m;
    }
    else
    {
        h1=m;
        h2=n;
    }
}
```

```

    }
    for (i=1;i<=m1;i++)
        for (j=1;j<=h1;j++)
            {
                a[i][h2+j]=0.0;
                if (i==j)
                    a[i][h2+j]=1.0;
            }
    it=0;
    t=1;
    while (t)
    {
        it=it+1;
        if (it<(m*n*5))
        {
            mi=a[m1][1];
            ps=1;
            for (j=2;j<=n1-1;j++)
                if (a[m1][j]<mi)
                {
                    mi=a[m1][j];
                    ps=j;
                }
            if (mi>-0.00001)
            {
                z=a[m1][n1];
                t=0;
            }
            mi=1e+20;
            pz=0;
            for (i=1;i<=m1-1;i++)
            {
                if (a[i][ps]<=0.0)
                    continue;
                h=a[i][n1]/a[i][ps];
                if (h<mi)
                {
                    mi=h;
                    pz=i;
                }
            }
            if (pz==0)
            {
                if (p==2)
                {
                    printf("Khong ton tai nghiem\n");
                    t=0;
                }
            }
        }
    }
}

```

```

        }
        else
        {
            printf("Nghiem khong bi gioi han\n");
            t=0;
        }
    }
    if (p==1)
        bv[pz]=ps;
    hi=a[pz][ps];
    for (j=1;j<=n1;j++)
        a[pz][j]=a[pz][j]/hi;
    if (pz!=1)
        for (i=1;i<=pz-1;i++)
        {
            hi=a[i][ps];
            for (j=1;j<=n1;j++)
                a[i][j]=a[i][j]-hi*a[pz][j];
        }
    for (i=pz+1;i<=m1;i++)
    {
        hi=a[i][ps];
        for (j=1;j<=n1;j++)
            a[i][j]=a[i][j]-hi*a[pz][j];
    }
}
else
    printf("Nghiem bat thuong");
}

}

void main()
{
    clrscr();
    printf("PHUONG PHAP DON HINH\n");
    printf("\n");
    fflush();
    printf("Cho bai toan tim max(1) hay min(2)(1/2)? : ");
    scanf("%d",&p);
    printf("Cho so bien n = ");
    scanf("%d",&n);
    printf("Cho so dieu kien bien m = ");
    scanf("%d",&m);
    n1=n+m+1;
    if (p==2)
        m1=n+1;
    else

```

```

    m1=m+1;
printf("Cho ma tran cac dieu kien bien\n");
for (i=1;i<=m;i++)
    for (j=1;j<=n;j++)
        if (p==2)
        {
            printf("a[%d][%d] = ",i,j);
            scanf("%f",&a[j][i]);
        }
        else
        {
            printf("a[%d][%d] = ",i,j);
            scanf("%f",&a[i][j]);
        }
printf("\n");
printf("Cho ma tran ve phai\n");
for (i=1;i<=m;i++)
    if (p==2)
    {
        printf("b[%d] = ",i);
        scanf("%f",&a[m1][i]);
    }
    else
    {
        printf("b[%d] = ",i);
        scanf("%f",&a[i][n1]);
    }
printf("\n");
printf("Cho ham muc tieu\n");
for (j=1;j<=n;j++)
    if (p==2)
    {
        printf("z[%d] = ",j);
        scanf("%f",&a[j][n1]);
    }
    else
    {
        printf("z[%d] = ",j);
        scanf("%f",&a[m1][j]);
    }
if (p==2)
    hi=m;
else
    hi=n;
for (j=1;j<=hi;j++)
    a[m1][j]=-a[m1][j];
a[m1][n1]=0.0;

```

```

don_hinh();
printf("\n");
printf("NGHIEM TOI UU HOA\n");
if (p==2)
    printf("Bai toan cuc tieu tieu chuan\n");
else
    printf("Bai toan cuc dai tieu chuan\n");
printf("sau %d buoc tinh",it);
printf("\n");
for (j=1;j<=n;j++)
{
    if (p==2)
        x=a[m1][m+j];
    else
    {
        v=0;
        for (i=1;i<=m;i++)
            if (bv[i]==j)
            {
                v=i;
                i=m;
            }
        if (v==0)
            x=0.0;
        else
            x=a[v][n1];
    }
    printf("x[%d] = %10.5f\n",j,x);
}
printf("\n");
printf("Gia tri toi uu cua ham muc tieu = %10.5f\n",z);
getch();
}

```

Dùng chương trình này giải bài toán có hàm mục tiêu :

$$z = 80x_1 + 56x_2 + 48x_3 \rightarrow \min$$

với ràng buộc :

$$3x_1 + 4x_2 + 2x_3 \geq 15$$

$$2x_1 + 3x_2 + x_3 \geq 9$$

$$x_1 + 2x_2 + 6x_3 \geq 18$$

$$x_2 + x_3 \geq 5$$

$$x_1, x_2, x_3 \geq 0$$

Ta cần nhập vào chương trình là tìm min, với số biến $n=3$, số điều kiện biên $m=4$, các hệ số $a[1,1] = 3$; $a[1,2] = 4$; $a[1,3] = 2$; $a[2,1] = 2$; $a[2,2] = 3$; $a[2,3] = 1$; $a[3,1] = 1$; $a[3,2] = 2$; $a[3,3] = 6$; $a[4,1] = 0$; $a[4,2] = 1$; $a[4,3] = 1$; $b[1] = 15$; $b[2] = 9$; $b[3] = 18$; $b[4] = 5$; $z[1] = 80$; $z[2] = 56$; $z[3] = 48$ và nhận được kết quả :

$x[1] = 0$; $x[2] = 2.5$; $x[3] = 2.5$ và trị của hàm mục tiêu là 260

§5. PHƯƠNG PHÁP THỂ VỊ

Trong vận tải ta thường gặp bài toán vận tải phát biểu như sau : có n thùng hàng của một hãng xây dựng cần chuyển tới n địa điểm khác nhau. Giá vận tới tới mỗi địa điểm đã cho. Tìm phương án vận chuyển để giá thành là cực tiểu.

Một cách tổng quát bài toán được phát biểu :

$$\sum a_i p_i \rightarrow \min$$

Ví dụ : Cần vận chuyển 6 thùng hàng tới 6 địa điểm với giá thành cho ở bảng sau :

Thùng	1	2	3	4	5	6 → địa điểm
1	60	35	28	53	29	26
2	81	43	37	23	36	45
3	42	42	33	47	43	51
4	29	70	42	53	48	37
5	81	69	40	66	69	60
6	10	21	32	31	24	27

Để giải bài toán ta dùng thuật toán Hungary như sau :

- trừ mỗi dòng cho số min của dòng đó ta có :

$$\begin{pmatrix} 34 & 9 & 2 & 27 & 3 & 0 \\ 58 & 20 & 14 & 0 & 13 & 22 \\ 9 & 9 & 0 & 14 & 10 & 18 \\ 0 & 41 & 13 & 24 & 19 & 8 \\ 41 & 29 & 0 & 26 & 29 & 20 \\ 0 & 11 & 22 & 21 & 14 & 17 \end{pmatrix}$$

- trừ mỗi cột cho số min của cột đó

$$\begin{pmatrix} 34 & 0 & 2 & 27 & 0 & 0 \\ 58 & 11 & 14 & 0 & 10 & 22 \\ 9 & 0 & 0 & 14 & 7 & 12 \\ 0 & 32 & 13 & 24 & 16 & 8 \\ 41 & 20 & 0 & 26 & 26 & 20 \\ 0 & 2 & 22 & 21 & 11 & 17 \end{pmatrix}$$

Mục tiêu của thuật toán Hungary là biến đổi ma trận giá thành sao cho có thể đọc giá trị tối ưu từ ma trận. Điều này được thực hiện khi mỗi hàng và cột chứa ít nhất một số 0. Nếu ta vẽ một đoạn thẳng qua mỗi hàng và cột chứa số 0 thì khi đó số đoạn thẳng tối thiểu qua tất cả các số 0 phải là 6. Trong ma trận trên ta chỉ mới dùng 5 đoạn thẳng nghĩa là chưa có giá trị tối ưu. Để biến đổi tiếp tục ta tìm trị min của các phần tử chưa nằm trên bất kì đoạn thẳng nào. Trị số đó là 7. Lấy các phần tử không nằm trên đoạn thẳng nào trừ đi 7 và cộng các phần tử nằm trên hai đoạn thẳng với 7 ta có ma trận :

$$\begin{pmatrix} 41 & 7 & 9 & 27 & 0 & 0 \\ 65 & 18 & 21 & 0 & 10 & 22 \\ 9 & 0 & 0 & 7 & 0 & 5 \\ 0 & 32 & 13 & 17 & 9 & 1 \\ 41 & 20 & 0 & 19 & 19 & 13 \\ 0 & 2 & 22 & 14 & 4 & 10 \end{pmatrix}$$

Do số đoạn thẳng tối thiểu còn là 5 nên ta lặp lại bước trên và nhận được ma trận mới :

42	7	10	28	0	0
65	17	21	0	9	21
10	0	1	8	0	5
0	31	13	17	8	0
41	19	0	19	18	12
0	1	22	14	3	9

Số đoạn thẳng cần để qua hết các số 0 là 6 nghĩa là ta đã tìm được trị tối ưu. Ta đánh dấu 6 số 0 sao cho mỗi hàng và mỗi cột chỉ có 1 số được đánh dấu. Chỉ số các số 0 được đánh dấu cho ta trị tối ưu :

$a_{15} = 0$ nghĩa là thùng 1 được vận chuyển tới địa điểm 5

$a_{24} = 0$ nghĩa là thùng 2 được vận chuyển tới địa điểm 4

$a_{32} = 0$ nghĩa là thùng 3 được vận chuyển tới địa điểm 2

$a_{46} = 0$ nghĩa là thùng 4 được vận chuyển tới địa điểm 6

$a_{53} = 0$ nghĩa là thùng 5 được vận chuyển tới địa điểm 3

$a_{61} = 0$ nghĩa là thùng 6 được vận chuyển tới địa điểm 1

Chương trình viết theo thuật toán trên như sau :

Chương trình 14-5

```
// van_tru;
#include <conio.h>
#include <stdio.h>

void main()
{
    int a[20][20], z[20][20], p[20][2];
    float x[20][20], w[20][20];
    float c[20], r[20];
    int t, c1, i, j, k, k2, k3, k5, l, l1, m, n, r1, s;
    float m1, q;

    clrscr();
    printf("Cho so an so n = ");
    scanf("%d", &n);
    printf("Cho cac he so cua ma tran x\n");
    for (i=1; i<=n; i++)
        for (j=1; j<=n; j++)
        {
            printf("x[%d][%d] = ", i, j);
            scanf("%f", &x[i][j]);
            w[i][j] = x[i][j];
        }

    for (i=1; i<=n; i++)
    {
        c[i] = 0.0;
```

```

        r[i]=0.0;
        p[i][1]=0.0;
        p[i][2]=0.0;
        a[i][1]=0.0;
        a[i][2]=0.0;
    }

    for (i=1;i<=2*n;i++)
    {
        z[i][1]=0.0;
        z[i][2]=0.0;
    }

    for (i=1;i<=n;i++)
    {
        m1=9999.0;
        for (j=1;j<=n;j++)
            if (x[i][j]<m1)
                m1=x[i][j];
        for (j=1;j<=n;j++)
            x[i][j]=x[i][j]-m1;
    }

    for (j=1;j<=n;j++)
    {
        m1=9999.0;
        for (i=1;i<=n;i++)
            if (x[i][j]<m1)
                m1=x[i][j];
        for (i=1;i<=n;i++)
            x[i][j]=x[i][j]-m1;
    }

    l=1;
    for (i=1;i<=n;i++)
    {
        j=1;
mot:   if (j>n)
            continue;
        if (x[i][j]!=0)
        {
            j=j+1;
            goto mot;
        }
        else
            if (i==1)
            {

```



```

        a[l][1]=i;
        a[l][2]=j;
        c[j]=1.0;
        l=l+1;
    }
else
{
    l1=l-1;
    for (k=1;k<=l1;k++)
    {
        if (a[k][2]!=j)
            continue;
        else
        {
            j=j+1;
            goto mot;
        }
    }
}

}

l=l-1;
if (l!=n)
{
    m=1;
hai:  for (i=1;i<=n;i++)
    {
        j=1;
ba:    if (j>n)
            continue;
        else
            if ((x[i][j]!=0)||(c[j]!=0)||(r[i]!=0))
            {
                j=j+1;
                goto ba;
            }
        else
        {
            p[m][1]=i;
            p[m][2]=j;
            m=m+1;
            for (k=1;k<=l;k++)
                if (a[k][1]!=i)
                    continue;
            else
            {
                r[i]=1.0;

```

```

                                c[a[k][2]]=0.0;
                                goto sau;
                                }
                                }
                                }
                                k2=m-1;
                                r1=p[k2][1];
                                c1=p[k2][2];
                                k3=1;
                                k=1;
                                s=1;
bon:  if (k==1)
        {
            z[k][1]=r1;
            z[k][2]=c1;
            k=k+1;
            goto bon;
        }
    else
    {
        if (s==1)
        {
            for (j=1;j<=k3;j++)
                if (a[j][2]==c1)
                {
                    r1=a[j][1];
                    s=2;
                    z[k][1]=r1;
                    z[k][2]=c1;
                    k=k+1;
                    goto bon;
                }
            k=k-1;
        }
        else
        {
            for (j=1;j<=k2;j++)
                if (p[j][1]==r1)
                {
                    c1=p[j][2];
                    s=1;
                    z[k][1]=r1;
                    z[k][2]=c1;
                    k=k+1;
                    goto bon;
                }
            else

```

```

                                continue;
                                k=k-1;
                                }
                                }
                                k5=1;
nam:   if (k5==k)
        {
            l=l+1;
            a[l][1]=z[k][1];
            a[l][2]=z[k][2];
            if (l!=n)
            {
                for (i=1;i<=n;i++)
                {
                    r[i]=0.0;
                    c[i]=0.0;
                    p[i][1]=0;
                    p[i][2]=0;
                }
                for (i=1;i<=l;i++)
                    c[a[i][2]]=1.0;
                m=1;
                goto hai;
sau:    m1=9999;
                for (i=1;i<=n;i++)
                    if (r[i]==0.0)
                        for (j=1;j<=n;j++)
                            if (c[j]==0.0)
                                if (x[i][j]<m1)
                                    m1=x[i][j];
                for (i=1;i<=n;i++)
                    for (j=1;j<=n;j++)
                        {
                            if ((r[i]!=0.0)||(c[j]!=0.0))
                                if ((r[i]!=1.0)||(c[j]!=1.0))
                                    continue;
                                else
                                    x[i][j]=x[i][j]+m1;
                                else
                                    x[i][j]=x[i][j]-m1;
                        }
                goto hai;
            }
        }
    else
    {
        for (i=1;i<=l;i++)

```

```

        if ((a[i][1]==z[k5+1][1]))
            if ((a[i][2]==z[k5+1][2]))
                break;
        a[i][1]=z[k5][1];
        a[i][2]=z[k5][2];
        k5=k5+2;
        goto nam;
    }
}
q=0.0;
for (i=1;i<=n;i++)
    q+=w[a[i][1]][a[i][2]];
printf("Gia thanh cuc tieu : %10.5f\n",q);
printf("\n");
printf("Cuc tieu hoa\n");
for (i=1;i<=n;i++)
    printf("%d%10c%d\n",a[i][1], ' ',a[i][2]);
getch();
}

```

Chạy chương trình ta nhận được giá thành cực tiểu là 181