Phương Pháp Nhánh Cận

(Branch and Bound)

Nhóm thực hiện

2. Trần Thị Kim Dung
3. Phan Thị Thu Hà
4. Nguyễn Thị Nga
5. Đoàn Thị Phương

A3K50 – Toán Tin ứng dụng

Nội dung

- Đặt vấn đề
- Ý tưởng
- Thuật giải
- Cài đặt
- Đánh giá
- Ví dụ minh họa

Đặt vấn đề

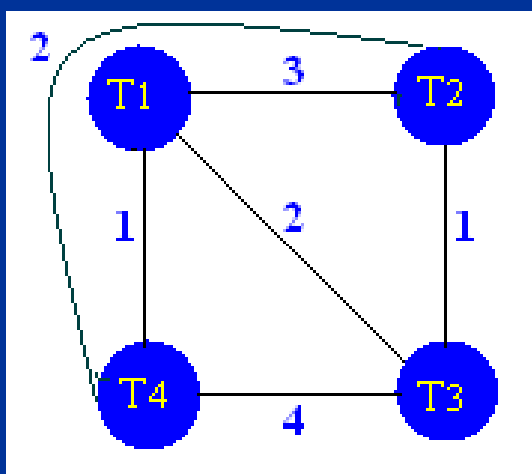
- **Bài toán thực tế:** Bài toán người giao hàng
  - Một người cần phải giao hàng tại N thành phố T1, T2, …, Tn
  - Cij: chi phí đi từ thành phố Ti đến thành phố Tj (i=1,2,…,N; j = 1,2,…,N)
  - **Yêu cầu:** xác định hành trình thỏa mãn
    - Đi qua tất cả các thành phố, mỗi thành phố qua đúng 1 lần, rồi quay trở lại thành phố xuất phát.
    - Chi phí nhỏ nhất

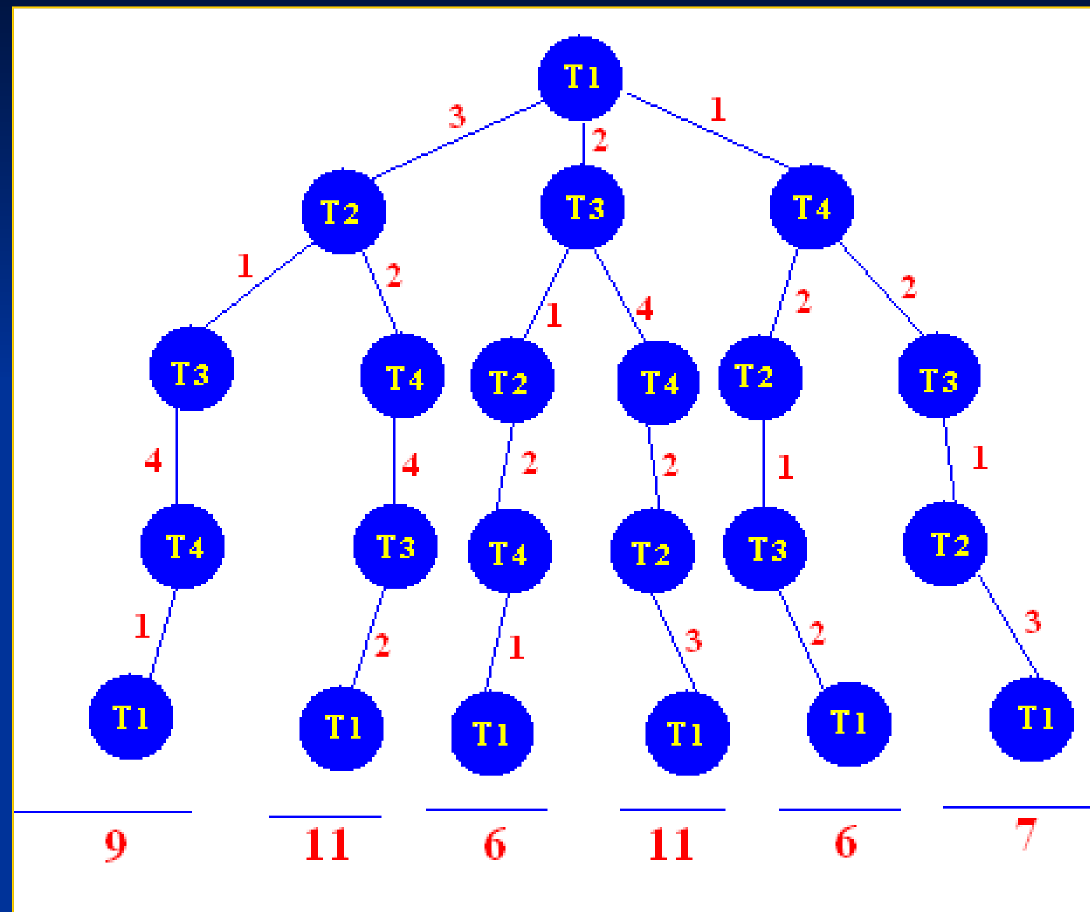
Đặt vấn đề [2]

- ***Giải quyết bài toán:***
  - Phương pháp vét cạn
  - Phương pháp vét cạn quay lui
  - Một số phương pháp khác
- ***Nhược điểm:***
  - Phải xét cả những phương án không khả thi (gây bùng nổ tổ hợp khi dữ liệu đầu vào n lớn)

Đặt vấn đề [3]

Mô hình thành phố





Sử dụng thuật toán quay lui

Ý tưởng

- ***Ý tưởng thuật toán:***
  - Giữ lại 1 phương án mẫu.
  - Tính chi phí của các phương án khác ngay trong quá trình xây dựng.
    - Tốt hơn: Cập nhật lại phương án mẫu và đi tiếp
    - Không tốt hơn: Quay lại bước trên xét phương án khác

Thuật giải

- **Bài toán tối ưu:** Tìm min{f(x): x $\in$ D}

với X={a=(a1, a2, …, an) $\in$ ΠAi (i=1, 2, …n) : P(x)}

|Ai< ∞| $\forall$ i=1, 2, …, n với P là một tính chất trên tập Ai

- **Nghiệm bài toán** có dạng x= (x1,x2,…,xn).
- **Bước 1:**
  - Xuất phát từ x1, xây dựng một phương án mẫu f*
- **Bước i:**
  - Đã xây dựng được nghiệm thành phần (x1, x2,…, xi-1)
  - Đánh giá cận: tìm g xác định trên Xi:

    g(x1,…,xi) < Min { f(a): a=(a1,…,an) thuộc X, xi=ai, i=1,…,n}
  - Giả sử x* là lời giải tốt nhất tại thời điểm đó, f* là giá trị tốt nhất f*=f(x*)
    - Nếu f*<g có thể bỏ đi không cần phát triển lời giải bộ phận (x1,…,xi)
    - Ngược lại: tiến hành bước i+1 để xác định xi+1

Cài đặt

```
Try(i) {
        for ( j=1->n )
                if ( chấp nhận được ) {
                        Xác định xi theo j;
                        Ghi nhận trạng thái mới;
                        if ( i=n )
                                Cập nhật lời giải tối ưu;
                        else  {
                        Xác định cận g(x1,…,xi);
                        if ( g(x1,…,xi) < f* )
                                Try(i+1)
                        }
                }
}
```

Đánh giá

- Ưu điểm:

  Giảm được chi phí: do loại bỏ được những bước đi không cần thiết (nhờ đánh giá cận)

- Nhược điểm:

  Việc xây dựng hàm g phụ thuộc vào từng bài toán tối ưu tổ hợp cụ thể. Hàm g phải đảm bảo điều kiện:

  - Việc tính giá trị của g phải đơn giản hơn việc giải bài toán tổ hợp tìm min= min{f(a): a=(a_1,…,a_n) thuộc X, x_i=a_i, i=1,…,n}
  - Giá trị của g(a_1, a_2,…, a_k) phải sát với các giá trị của min.

Ví dụ minh họa

- Bài toán người đưa hàng
  - Ý tưởng
  - Thuật giải và đánh giá
  - Cài đặt
  - Minh họa

Ví dụ minh họa[2]

- **Ý tưởng**
  - Gọi p là 1 hoán vị của {1,…,n} ta được hành trình $T_{p(1)} \to T_{p(2)} \to \ldots \to T_{p(n)}$
  - Có n! hành trình
  - Nếu cố định đỉnh xuất phát là đỉnh 1 thì có (n-1)! hành trình, bài toán trở thành :
    - Tìm Min{f(a2,…,an): (a2,…,an) là hoán vị của {2,…,n}} với $f(a_1,..,a_n)=C_{1,a2}+C_{a2,a3}+\ldots+C_{an-1,an}+C_{an,1}$
    - Ta sẽ kết hợp đánh giá nhánh cận trong quá trình liệt kê phương án của thuật toán quay lui

Ví dụ minh họa [3]

- **Thuật giải và đánh giá:**
  - Cố định đỉnh xuất phát là đỉnh 1, duyệt vòng lặp từ j=2
  - Tại bước i:
    - Đánh giá cận:
      - Đặt $C_{min}=Min\{C_{ij}: i,j=\{1,..,n\}\}$
      - Giả sử đã đi đoạn đường T1->T2->…->Ti với chi phí: $S_i=C_{1,x2}+C_{x2,x3}+\ldots+C_{xi-1,xi}$
      - Hàm cận: $g(x_1,\ldots,x_i)=S_i+(n-i+1)C_{min}$
      - Lưu dấu bằng mảng logic Daxet[]:
        - $Daxet[j]=\begin{cases} 1 \text{ nếu } T[j] \text{ đã qua} \\ 0 \text{ nếu } T[j] \text{ chưa qua} \end{cases}$
      - Xác định $x_i=j$, cập nhật Daxet[j]=1và $S=S+C_{xi-1,xi}$
      - Nếu i=n, $Tong=S+C_{xn,1}$;
        - Nếu ($Tong< f^*$)thì lời giải tối ưu=x; $f^*=Tong$;
      - Nếu Daxet[j]=0 thì $S=S-C_{xi-1,xi}$

Ví dụ minh họa [4]

- **Cài đặt:**

```
Try(i)
 for ( j= 2 -> n )
   if ( !Daxet[j] )
    {
      x[i]=j;
      Daxet[j]=1;
      S=S+C[x[i-1]] [x[i]];
      if (i= =n) //cap nhat toi uu
      {
          tong=S+C[x[n]] [x[1]];
          if  ( tong< f* )  {
             Lgtu=x;//loi giai toi uu
             f*=tong;
          }
      }
      else {
         g=S+(n-i+1)*Cmin; //đánh
                          giá cận
         if (g < f* )
           Try ( i +1 );
      }
      S=S-C[x[i-1]] [x[i]];
      Daxet[j]=0;
    }
```

Ví dụ minh họa [5]

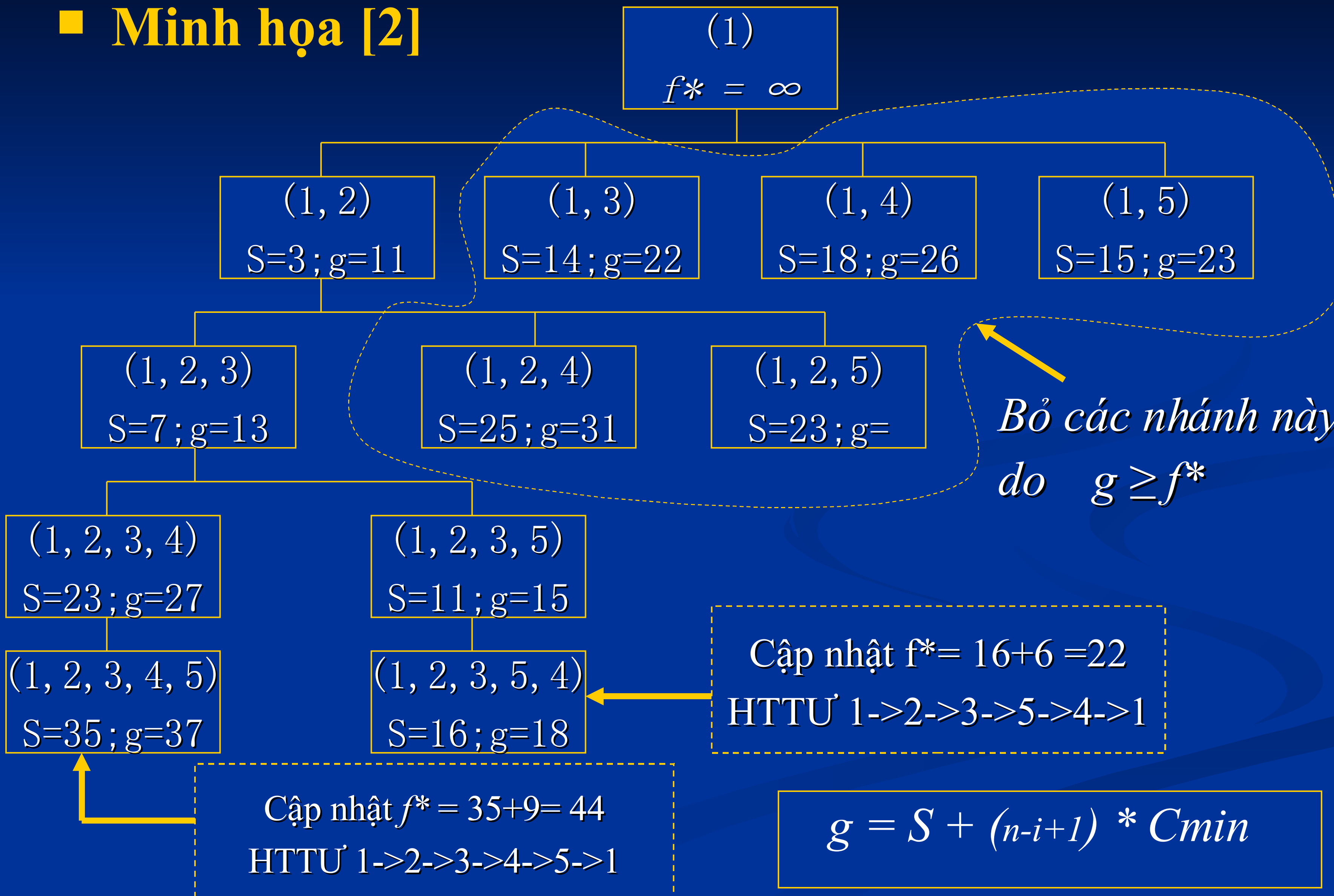
- **Minh họa**

  Giải bài toán người đưa hàng với ma trận chi phí như sau:

$$C = \begin{vmatrix} 0 & 3 & 14 & 18 & 15 \\ 3 & 0 & 4 & 22 & 20 \\ 17 & 9 & 0 & 16 & 4 \\ 6 & 2 & 7 & 0 & 12 \\ 9 & 15 & 11 & 5 & 0 \end{vmatrix}$$

Ví dụ minh họa [6]

- **Minh họa [2]**



Một số bài toán khác

- Bài toán cái túi xách:
  - Có n loại đồ vật, mỗi loại có khối lượng không hạn chế
  - Đồ vật loại i được đặc trưng bởi:
    - Trọng lượng Wi
    - Giá trị sử dụng Vi
  - Chọn đặt vào trong 1 túi xách
    - Tổng trọng lượng m
    - Tổng giá trị sử dụng của các vật trong túi là lớn nhất