Bài toán ghép cặp

Graph Matching

Bài toán ghép cặp trên đồ thị

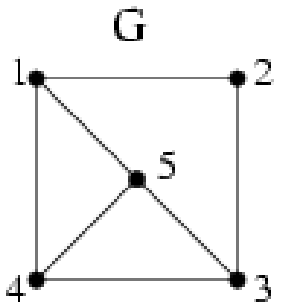
- Giả sử $G=(V,E)$ là đồ thị vô hướng, trong đó mỗi cạnh (v,w) đợc gán với một số thực $c(v,w)$ gọi là trọng số của nó.
- **Định nghĩa.** *Cặp ghép M trên đồ thị G là tập các cạnh của đồ thị trong đó không có hai cạnh nào có đỉnh chung.*
  - *Số cạnh trong M -* ***kích thớc****,*
  - *Tổng trọng số của các cạnh trong M -* ***trọng lợng*** *của cặp ghép.*
  - *Cặp ghép với kích thớc lớn nhất đợc gọi là* ***cặp ghép cực đại.***
  - *Cặp ghép với trọng lợng lớn nhất đợc gọi là* ***cặp ghép lớn nhất.***
  - *Cặp ghép đợc gọi là* ***đầy đủ (hoàn hảo)*** *nếu mỗi đỉnh của đồ thị là đầu mút của ít nhất một cạnh trong cặp ghép.*

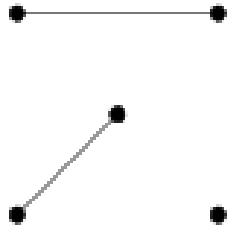
Hai bài toán

- **Bài toán cặp ghép cực đại**: *Tìm cặp ghép với kích thước lớn nhất trong đồ thị G.*
- **Bài toán cặp ghép lớn nhất:** *Tìm cặp ghép với trọng lợng lớn nhất trong đồ thị G.*

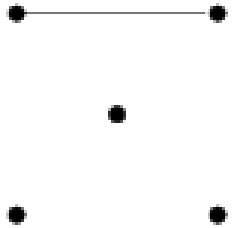
- *Ta hạn chế xét các bài toán đặt ra trên đồ thị hai phía* $G = (X \cup Y, E)$.

Ví dụ

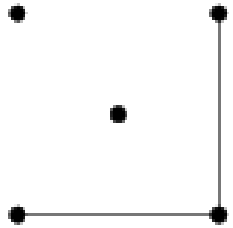




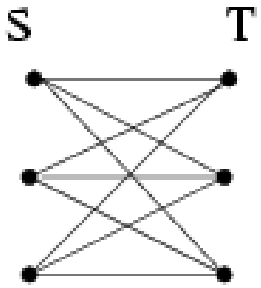
Cặp ghép cực đại
cặp ghép

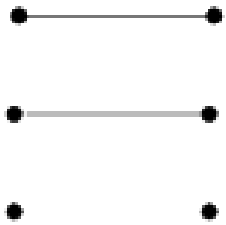


Cặp ghép

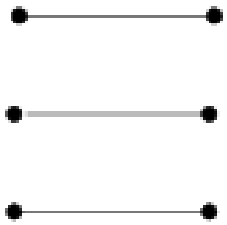


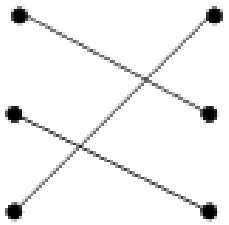
không là





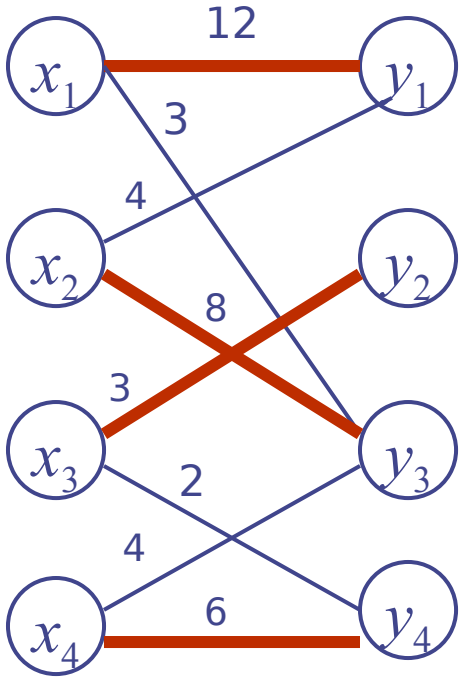
Cặp ghép





Cặp ghép hoàn hảo

Ví dụ



- Cặp ghép lớn nhất:

$$M = \{(x_1, y_1), (x_2, y_3), (x_3, y_2), (x_4, y_4)\}$$

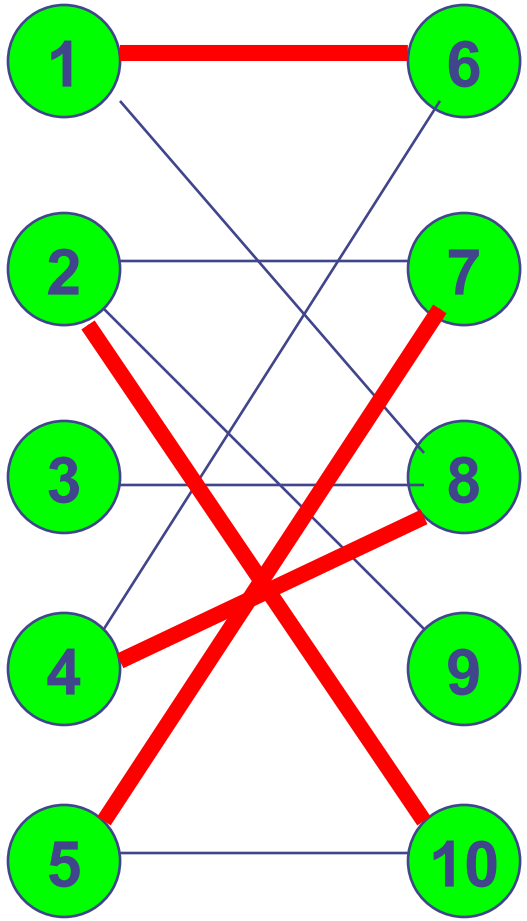
Có trọng lượng 29.

Bài toán cặp ghép cực đại trên đồ thị hai phía

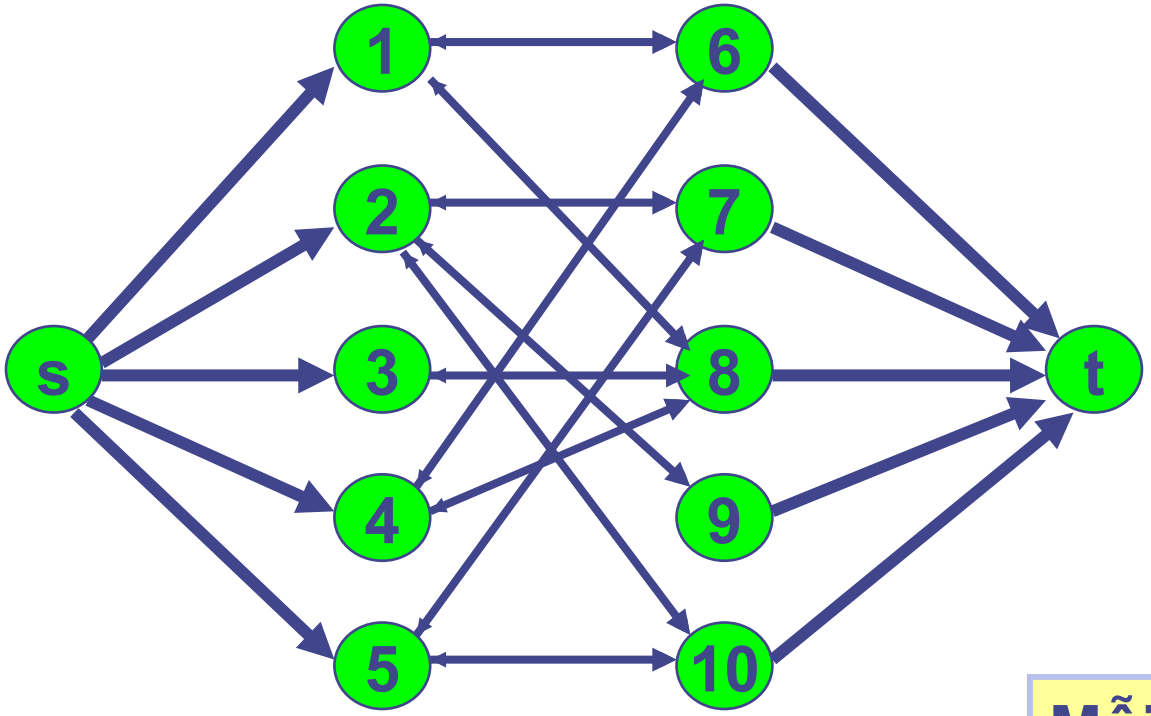
Xét đồ thị hai phía
$G = (X \cup Y, E)$.

Cặp ghép là tập cạnh mà không có hai cạnh nào có chung đỉnh

***Bài toán*: Tìm cặp ghép kích thước lớn nhất**



Qui về Bài toán luồng cực đại

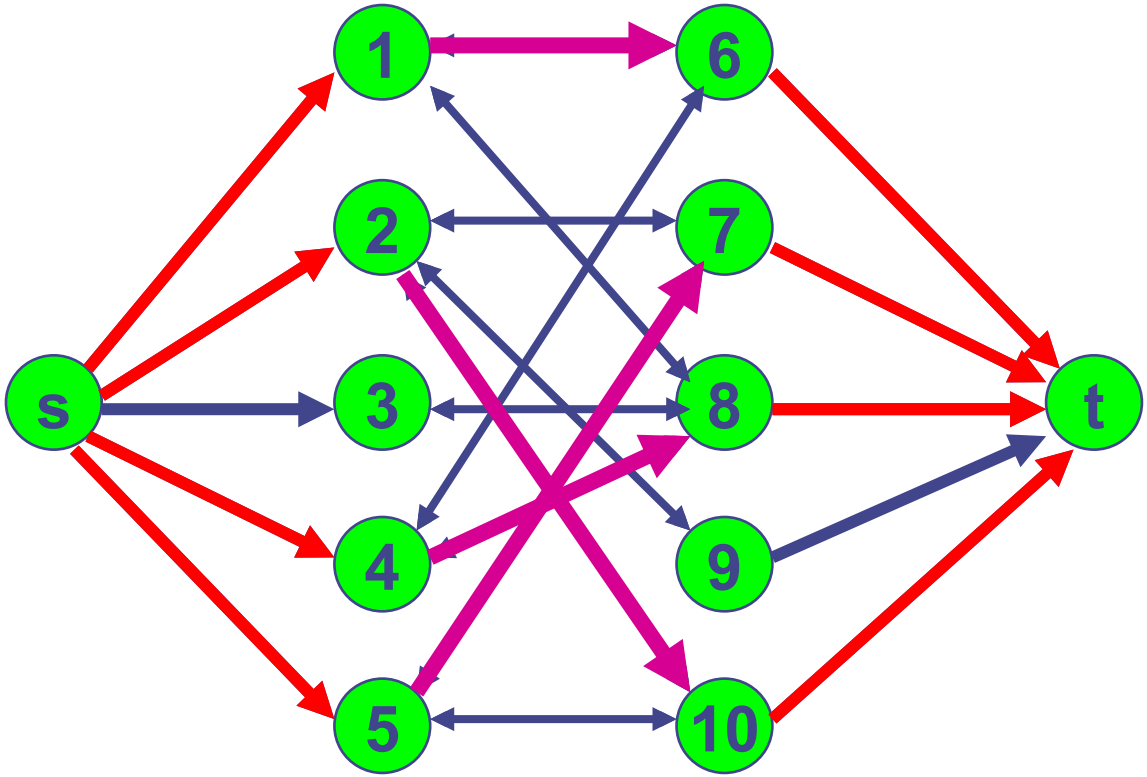


Mỗi cạnh được thay thế bởi cung có kntq 1.

Mỗi cung (s, i) có kntq 1.

Mỗi cung (j, t) có kntq 1.

Tìm luồng cực đại



Luồng cực đại từ s->t có giá trị 4.

Cặp ghép cực đại có kích thước 4.

Bài toán cặp ghép cực đại trên đồ thị hai phía

- Giả sử M là một cặp ghép trên G.
- Nếu cạnh $e = (x, y) \in M$, ta nói e là cạnh của cặp ghép (hay cạnh đậm) và các đỉnh x, y là các đỉnh đậm (hay không tự do).
- Nếu cạnh $e = (x, y) \notin M$, thì ta nói e là cạnh nhạt còn các đỉnh x, y là các đỉnh nhạt (hay tự do).

Đường tăng cặp ghép

- *Một đường đi trên đồ thị G mà trong đó hai cạnh liên tiếp là không cùng đậm hay nhạt sẽ đ ợc gọi là **đường đi luân phiên** đậm/nhạt (hay gọi ngắn gọn là đường đi luân phiên).*
- *Đường đi luân phiên bắt đầu từ một đỉnh tự do thuộc tập X và kết thúc ở một đỉnh tự do thuộc tập Y đợc gọi là **đường tăng cặp ghép.***

Định lý Berge

- **Định lý 1 (Berge C).** *Cặp ghép M là cực đại khi và chỉ khi không tìm đợc đờng tăng cặp ghép.*
- ***CM:***
- ***Điều kiện cần.*** *Bằng phản chứng. Giả sử M là cặp ghép cực đại nhng vẫn tìm đợc đờng tăng cặp ghép*

$$P \equiv x_0, y_1, x_1, y_2, \ldots, x_k, y_0$$

trong đó x_0 *và* y_0 *là các đỉnh tự do.*

Gọi E_P *là tập các cạnh của đồ thị nằm trên đờng đi P*

$$E_P = \{ (x_0,y_1), (y_1, x_1), \ldots, (x_k, y_0) \}.$$

Dễ thấy số lợng cạnh nhạt trong E_P *là bằng số lợng cạnh đậm của nó cộng với 1. Để đơn giản trong phần dới đây ta đồng nhất ký hiệu đờng đi P với tập cạnh* E_P *của nó. Xây dựng cặp ghép M' theo qui tắc:*

$$M' = (M \cup P) \setminus (M \cap P).$$

Dễ thấy M' cũng là cặp ghép và rõ ràng $|M'| = |M| + 1$. *Mâu thuẫn thu đợc đã chứng minh điều kiện cần.*

Định lý Berge

- **Điều kiện đủ.** Giả sử cặp ghép M cha là cặp ghép cực đại. Gọi M^* là cặp ghép cực đại. Xét đồ thị $G' = (V, M \cup M^*)$. Rõ ràng hai cạnh liên tiếp trong mỗi đờng đi cũng nh mỗi chu trình trong G' không thể thuộc cùng một cặp ghép M hoặc M^*. Vì vậy, mỗi đờng đi cũng nh mỗi chu trình trong G' đều là đờng luân phiên M/M^*. Do $|M^*| > |M|$, nên rõ ràng là luôn tìm đợc ít nhất một đờng đi luân phiên M/M^* mà trong đó số lợng cạnh thuộc M^* là lớn hơn số lợng cạnh thuộc M. Đờng đi đó chính là đờng tăng cặp ghép trên đồ thị G.
- Định lý đợc chứng minh.
- **Chú ý:** Trong chứng minh định lý ta không sử dụng tính hai phía của G. Do đó, Định lý 1 là đúng với đồ thị vô hớng bất kỳ.

Thuật toán tìm cặp ghép cực đại

- *Đầu vào:* Đồ thị vô hướng $G = (V, E)$.
- *Bước khởi tạo*. Xây dựng cặp ghép M trong đồ thị G (có thể bắt đầu từ $M = \varnothing$).
- *Bước lặp*.
  - Kiểm tra tiêu chuẩn tối u: Nếu đồ thị G không chứa đờng tăng cặp ghép thì M là cặp ghép cực đại, thuật toán kết thúc.
  - Ngược lại, gọi P là một đờng tăng cặp ghép xuất phát từ đỉnh tự do $x_0 \in X$, kết thúc ở đỉnh tự do $y_0 \in Y$. Tăng cặp ghép theo qui tắc

    $$M := (M \cup P) \setminus (M \cap P),$$

    rồi lặp lại bớc lặp.

Tìm đường tăng

- Từ đồ thị G ta xây dựng đồ thị có hướng $G_M = (X \cup Y, E_M)$ với tập cung E_M đợc bằng cách định hướng lại các cạnh của G theo quy tắc sau:

  i) Nếu $(x,y) \in M \cap E$, thì $(y,x) \in E_M$;

  ii) Nếu $(x,y) \in E \setminus M$, thì $(x,y) \in E_M$.

  Đồ thị G_M sẽ đợc gọi là ***đồ thị tăng cặp ghép***.

- Dễ thấy:
  - Đờng tăng cặp ghép tơng ứng với một đờng đi xuất phát từ một đỉnh tự do $x_0 \in X$ kết thúc tại một đỉnh tự do $y_0 \in Y$ trên đồ thị G_M.
  - Ngợc lại, một đờng đi trên đồ thị G_M xuất phát từ một đỉnh tự do $x_0 \in X$ kết thúc tại một đỉnh tự do $y_0 \in Y$ sẽ tơng ứng với một đờng tăng cặp ghép trên đồ thị G.
- Vì vậy, để xét xem đồ thị G có chứa đờng tăng cặp ghép hay không, có thể thực hiện thuật toán tìm kiếm theo chiều rộng trên đồ thị G_M bắt đầu từ các đỉnh tự do thuộc tập X.

Thuật toán

- Sử dụng cách tìm đờng tăng cặp ghép theo nhận xét vừa nêu, từ sơ đồ tổng quát dễ dàng xây dựng thuật toán để giải bài toán tìm cặp ghép cực đại trên đồ thị hai phía với thời gian tính $O(n^3)$, trong đó $n = \max(|X|, |Y|)$.

Cài đặt

Cấu trúc dữ liệu

```
Var
    A : Array[1..100,1..100] of Byte; (* Ma trận kề của đồ thi hai phía G *)
    Truoc,    (* Ghi nhận đường đi *)
    Vo,                                    (* Vo[x]- đỉnh được ghép với x∈X *)
    Chong : Array[1..100] of Byte;    (* Chong[y]-đỉnh được ghép với y ∈Y *)
    N, x0, y0, Cnt : Byte;
    Stop : Boolean;

(* Nếu (x, y) ∈ M thì Vo[x]=y; Chong[y]=x.
   Vo[x]=0 => x là đỉnh nhạt; Chong[y]=0 => y là đỉnh nhạt *)
```

Tìm đường tăng

```
Procedure Tim(x:Byte);
  var y: Byte;
   begin
     For y:=1 to N do
       If (A[x,y]=1)and(Truoc[y]=0)and(y0=0)
     then
      begin
           Truoc[y]:=x;
           If Chong[y]<>0 then Tim(Chong[y])
           else
           begin
                   y0:=y;
                   Exit;
           end;
       end;
   end;
```

```
Procedure
Tim_Duong_Tang;
begin

Fillchar(Truoc,Sizeof(Truoc),0);
     y0:=0;
     For x0:=1 to N do
     begin
        If Vo[x0]=0 then
Tim(x0);
        If y0<>0 then exit;
     end;
     Stop:=true;
end;
```

Thủ tục MaxMatching

```
Procedure Tang;
var temp: Byte;
begin
    Inc(Cnt);
    While Truoc[y0]<>x0 do
    begin
        Chong[y0]:=Truoc[y0];
        Temp:=Vo[Truoc[y0]];
        Vo[Truoc[y0]]:=y0;
        y0:=Temp;
    end;
    Chong[y0]:=x0;
    Vo[x0]:=y0;
end;
```

```
Procedure MaxMatching;
begin
     Stop:=false;
    Fillchar(Vo,Sizeof(Vo),0);
    Fillchar(Chong,Sizeof(Chong),0);
    Cnt:=0;
    While not Stop do
     begin
         Tim_duong_tang;
         If not Stop then Tang;
     end;
end;
```

Bài toán phân công

Có n công việc và n thợ. Mỗi thợ đều có khả năng thực hiện tất cả các công việc. Biết

w_{ij} - hiệu quả phân công thợ i làm việc j,

$(i, j = 1, 2,..., n)$.

Cần tìm cách phân công thợ thực hiện các công việc sao cho mỗi thợ chỉ thực hiện một việc và mỗi việc chỉ do một thợ thực hiện, đồng thời tổng hiệu quả thực hiện các công việc là lớn nhất.

Qui về bài toán cặp ghép lớn nhất

Xây dựng đồ thị hai phía đầy đủ $G = (X \cup Y, E)$

- $X = \{x_1, x_2, ..., x_n\}$ tơng ứng với các thợ,
- $Y = \{y_1, y_2, ..., y_n\}$ - tơng ứng với các công việc.

Mỗi cạnh (x_i, y_j) đợc gán cho trọng số $w(x_i, y_j) = w_{ij}$.

Khi đó trong ngôn ngữ đồ thị, bài toán phân công có thể phát biểu nh sau: Tìm trong đồ thị G cặp ghép đầy đủ có tổng trọng số là lớn nhất. Cặp ghép nh vậy đợc gọi là cặp ghép tối u.

Cơ sở thuật toán

Ta gọi một ***phép gán nhãn chấp nhận đợc*** *cho các đỉnh của đồ thị* $G=(X \cup Y, E)$ *là một hàm số* f *xác định trên tập đỉnh* $X \cup Y$: $f: X \cup Y \rightarrow R$, *thoả mãn*

$$f(x) + f(y) \geq w(x,y), \ \forall x \in X, \ \forall y \in Y.$$

Một phép gán nhãn chấp nhận đợc nh vậy dễ dàng có thể tìm đợc, chẳng hạn phép gán nhãn sau đây là chấp nhận đợc

$$f(x) = \max \{ w(x,y): \ y \in Y \}, \ x \in X,$$

$$f(y) = 0 , \ y \in Y.$$

Đồ thị cân bằng

- Giả sử có f là một phép gán nhãn chấp nhận đợc, ký hiệu

$$E_f = \{(x,y) \in E:\ f(x) + f(y) = w(x,y)\}.$$

- Ký hiệu G_f là đồ thị con của G sinh bởi tập đỉnh $X \cup Y$ và tập cạnh E_f. Ta sẽ gọi G_f là ***đồ thị cân bằng***.

Tiêu chuẩn tối ưu

Định lý 2. *Giả sử f là phép gán nhãn chấp nhận đợc. Nếu G_f chứa cặp ghép đầy đủ M*, thì M* là cặp ghép tối u.*

Chứng minh.

Giả sử G_f chứa cặp ghép đầy đủ M^*. Khi đó từ định nghĩa G_f suy ra M^* cũng là cặp ghép đầy đủ của đồ thị G. Gọi $w(M^*)$ là trọng lợng của M^*:

$$w(M^*) = \sum_{e \in M^*} w(e)$$

Do mỗi cạnh $e \in M^*$ đều là cạnh của G_f và mỗi đỉnh của G kề với đúng một cạnh của M^*, nên

$$w(M^*) = \sum_{e \in M^*} w(e) = \sum_{v \in V} f(v)$$

Giả sử M là một cặp ghép đầy đủ tuỳ ý của G, khi đó

$$w(M) = \sum_{e \in M} w(e) \le \sum_{v \in V} f(v)$$

Suy ra $w(M^*) \ge w(M)$. Vậy M^* là cặp ghép tối u.

Sơ đồ thuật toán

- Ta sẽ bắt đầu từ một phép gán nhãn chấp nhận đợc f. Xây dựng đồ thị G_f. Bắt đầu từ một cặp ghép M nào đó trong G_f ta xây dựng cặp ghép đầy đủ trong G_f. Nếu tìm đợc cặp ghép đầy đủ M^*, thì nó chính là cặp ghép tối u. Ngợc lại, ta sẽ tìm đợc cặp ghép cực đại không đầy đủ M'. Từ M' ta sẽ tìm cách sửa phép gán nhãn thành f' sao cho M' vẫn là cặp ghép của G_f' và có thể tiếp tục phát triển M' trong G_f'., v.v...
- Quá trình đợc tiếp tục cho đến khi thu đợc cặp ghép đầy đủ trong đồ thị cân bằng.

Điều chỉnh nhãn

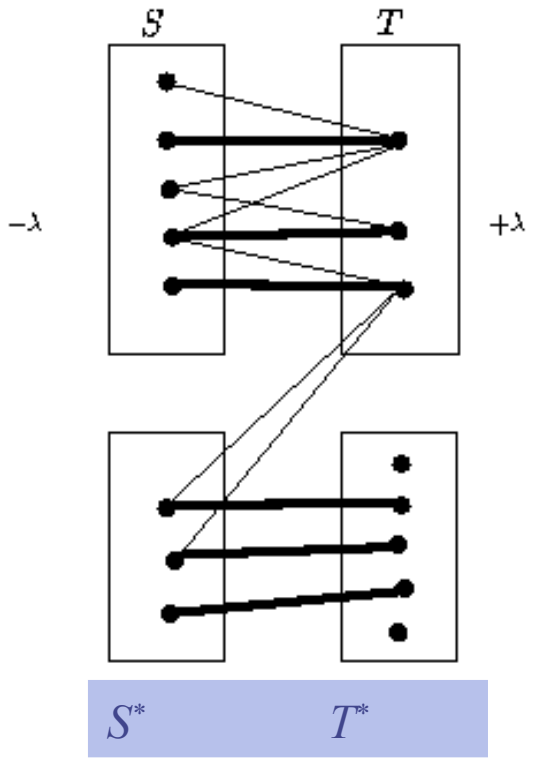
- Giả sử M là cặp ghép cực đại trong đồ thị G_f và M ch a là cặp ghép đầy đủ của G. Ta cần tìm cách điều chỉnh phép gán nhãn f thoả mãn các yêu cầu đặt ra.
- Thực hiện tìm kiếm theo chiều rộng từ các đỉnh tự do trong X. Gọi S là các đỉnh đợc thăm trong X, còn T là các đỉnh đợc thăm trong Y trong quá trình thực hiện tìm kiếm. Ký hiệu

$$S^* = X \setminus S; \quad T^* = Y \setminus T.$$

$|S| > |T|$ (do mỗi đỉnh trong T đạt đợc từ một đỉnh nào đó trong S).

Điều chỉnh nhãn

- Từ tính chất của thuật toán tìm kiếm theo chiều rộng, rõ ràng, không có cạnh nào từ S đến T^*. Để sửa chữa nhãn, chúng ta sẽ tiến hành giảm đồng loạt các nhãn trong S đi cùng một giá trị λ nào đó, và đồng thời sẽ tăng đồng loạt nhãn của các đỉnh trong T lên λ. Điều đó đảm bảo các cạnh từ S sang T (nghĩa là những cạnh mà một đầu mút thuộc S còn một đầu mút thuộc T) không bị loại bỏ khỏi đồ thị cân bằng



Các tập S và T trong thực hiện thuật toán. Chỉ vẽ các cạnh trong G_f.

Điều chỉnh nhãn

- Khi các nhãn trong S bị giảm, các cạnh trong G từ S sang T^* sẽ có khả năng gia nhập vào đồ thị cân bằng G_f. Ta sẽ tăng λ đến khi có thêm ít nhất một cạnh mới gia nhập đồ thị cân bằng. Có hai khả năng:
  - Nếu cạnh mới gia nhập đồ thị cân bằng giúp ta thăm đợc một đỉnh không tự do $y \in T^*$ thì từ nó ta sẽ thăm đợc một đỉnh đợc ghép với nó trong cặp ghép $x \in S^*$, và cả hai đỉnh này đợc bổ sung vào S và T tơng ứng, và nh vậy việc tìm kiếm đờng tăng sẽ đợc tiếp tục mở rộng.
  - Nếu cạnh mới gia nhập đồ thị cân bằng cho phép thăm đ ợc một đỉnh tự do $y \in T^*$ thì ta tìm đợc đờng tăng cặp ghép, và kết thúc một pha điều chỉnh nhãn.

Điều chỉnh nhãn

Ta gọi ***một pha điều chỉnh*** là tất cả các lần sửa nhãn cần thiết để tăng đợc kích thớc của cặp ghép M.

- Vì sau mỗi pha điều chỉnh kích thớc của cặp ghép tăng lên 1, nên ta phải thực hiện nhiều nhất n pha điều chỉnh.
- Trong mỗi pha điều chỉnh, do sau mỗi lần sửa nhãn có ít nhất hai đỉnh mới đợc bổ sung vào danh sách các đỉnh đợc thăm, nên ta phải thực hiện việc sửa nhãn không quá n lần. Mặt khác, trong thời gian $O(n^2)$ ta có thể xác định đợc cạnh nào từ S sang T^* là cạnh gia nhập đồ thị cân bằng (bằng việc duyệt hết các cạnh). Từ đó suy ra đánh giá thời gian tính của thuật toán là $O(n^4)$.

Thuật toán

- **Bớc 0:** Tìm một phép gán nhãn chấp nhận đợc f.
- **Bớc 1:** Xây dựng đồ thị cân bằng G_f.
- **Bớc 2:** Tìm cặp ghép cực đại M trong G_f.
- **Bớc 3:** Nếu M là cặp ghép đầy đủ thì nó là cặp ghép lớn nhất cần tìm. Thuật toán kết thúc.
- **Bớc 4:** Gọi S là tập các đỉnh tự do trong X. Thực hiện tìm kiếm từ các đỉnh trong S. Gọi T là tập các đỉnh của Y đợc thăm trong quá trình tìm kiếm. Bổ sung các đỉnh trong X đợc thăm trong quá trình tìm kiếm vào S.
- **Bớc 5:** Tiến hành điều chỉnh nhãn f ta sẽ bổ sung đợc các cạnh vào G_f cho đến khi tìm đợc đờng tăng, bổ sung các đỉnh mới đợc thăm vào S và T tơng ứng nh đã mô tả ở trên. Tăng cặp ghép M và quay lại bớc 3.

Tăng hiệu quả

- Để có đợc thuật toán với đánh giá thời gian tính tốt hơn, vấn đề đặt ra là làm thế nào có thể tính đợc giá trị λ tại mỗi lần sửa nhãn của pha điều chỉnh một cách nhanh chóng.
- Ta xác định độ lệch của các cạnh theo công thức

$$slack(x, y) = f(x) + f(y) — c(x, y).$$

Tăng hiệu quả

- Khi đó

$$\lambda = \min_{x \in S, y \in T^*} slack(x, y)$$

- Rõ ràng việc tính trực tiếp λ theo công thức đòi hỏi thời gian $O(n^2)$. Bây giờ, nếu với mỗi đỉnh trong T^* ta ghi nhận lại cạnh với độ lệch nhỏ nhất

$$slack(y_j) = \min_{x_i \in S} slack(x_i, y_j).$$

Tăng hiệu quả

- Việc tính giá trị độ lệch $slack(y_j)$ đòi hỏi thời gian $O(n^2)$ ở đầu pha điều chỉnh. Khi tiến hành pha điều chỉnh ta có thể sửa lại tất cả các độ lệch trong thời gian $O(n)$ do chúng bị thay đổi cùng một giá trị (do nhãn của các đỉnh trong S giảm đồng loạt đi cùng một giá trị λ). Khi một đỉnh x đợc chuyển từ S^* sang S ta cần tính lại các độ lệch của các đỉnh trong T^*, việc đó đòi hỏi thời gian $O(n)$. Tuy nhiên sự kiện một đỉnh đợc chuyển từ S^* sang S chỉ xảy ra nhiều nhất n lần.
- Nh vậy, mỗi pha điều chỉnh có thể cài đặt với thời gian $O(n^2)$. Do có không quá n pha điều chỉnh trong thuật toán, nên cách cài đặt này cho ta thuật toán với thời gian tính $O(n^3)$.

Ví dụ

- Xét bài toán với ma trận hiệu quả

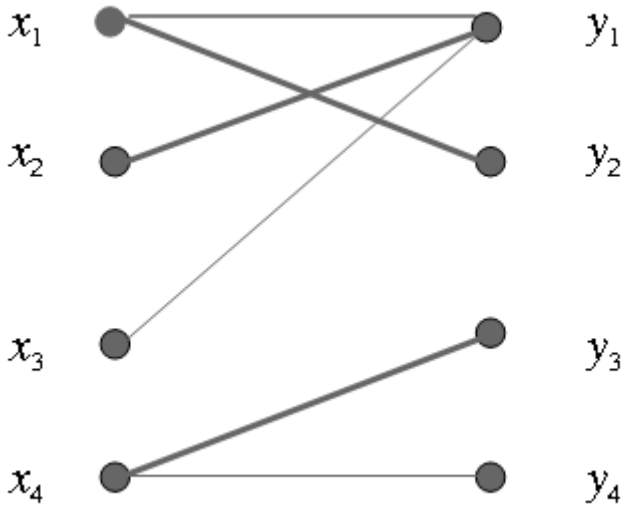
$$W = \begin{vmatrix} 4 & 4 & 1 & 3 \\ 3 & 2 & 2 & 1 \\ 5 & 4 & 4 & 3 \\ 1 & 1 & 2 & 2 \end{vmatrix}$$

Ví dụ

- Bắt đầu từ phép gán nhãn

$f(y)$ / $f(x)$	0	0	0	0
4	4	4	1	3
3	3	2	2	1
5	5	4	4	3
2	1	1	2	2

Đồ thị cân bằng G_f



Cặp ghép cực đại tìm đợc

$$M = \{(x_1,y_2), (x_2,y_1), (x_4, y_4)\}.$$

Tìm kiếm theo chiều rộng bắt đầu từ đỉnh tự do x_3 ta có

$$S = \{ x_2 , x_3 \}, \; T = \{y_1\}$$

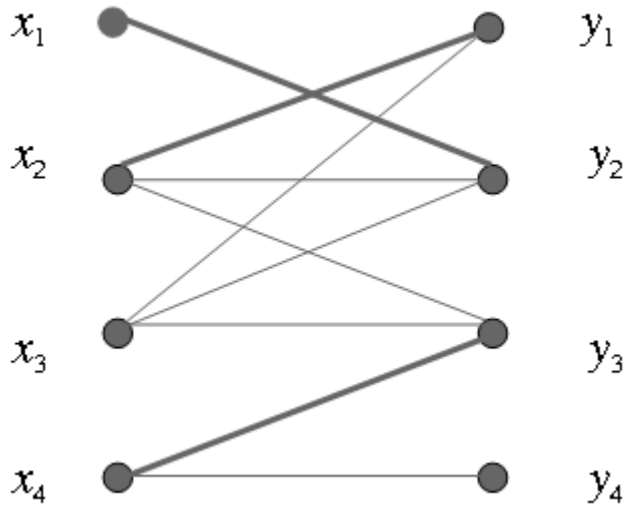
Ví dụ

- Tính

  $\lambda = \min \{f(x)+f(y)-w(x,y): x \in \{x_2, x_3\}, y \in \{y_2, y_3, y_4\}\} = 1.$

- Tiến hành sửa nhãn, ta đi đến phép gán nhãn mới

$f(x)$ \ $f(y)$	1	0	0	0
4	4	4	1	3
2	3	2	2	1
4	5	4	4	3
2	1	1	2	2



Ví dụ

- Theo đờng tăng cặp ghép

$$x_3, y_3, x_4, y_4$$

ta tăng cặp ghép M thành cặp ghép đầy đủ

$$M = \{(x_1,y_2), (x_3,y_1), (x_2,y_3), (x_4,y_4)\},$$

đồng thời là cặp ghép tối u với trọng lợng

$$w(M) = 4 + 2 + 5 + 2 = 13.$$

Cài đặt trên Pascal

```
const maxn = 170;

type    data1=array [1..maxn,1..maxn] of integer;
        data2=array [1..2*maxn] of integer;
        data3=array [1..2*maxn] of longint;

var     c: data1;
        px, py, q, queue: data2;
        a, b, f: data3;
        n, n2, k, u, z: integer;
```

Khởi tạo

procedure init;

```
var i, j: integer;
begin
     n2:= n+n; fillchar(f,sizeof(f),0);
     for i:=1 to n do
      for j:=1 to n do
       if f[i]<c(i,j) then f[i]:=c(i,j);
     k:=0;
     fillchar(px,sizeof(px),0); fillchar(py,sizeof(py),0);
     for i:=1 to n do
      for j:=1 to n do
       if (py[j]=0) and (f[i]+f[j+n]=c(i,j)) then
       begin
          px[i]:=j; py[j]:=i;  inc(k);
          break;
       end;
end;
```

Tìm đường tăng

```
function FoundIncPath: boolean;
var dq, cq, v, w: integer;
begin
   fillchar(q,sizeof(q),0);
   dq:=1; cq:=1; queue[dq]:=u; q[u]:=u;
   while dq<=cq do
   begin
      v:=queue[dq]; inc(dq);
      if v<=n then
      begin
         for w:=n+1 to n2 do
          if (f[v]+f[w]=c(v,w-n)) and (q[w]=0) then
          begin  inc(cq); queue[cq]:=w; q[w]:=v; end;
      end  else
      if (py[v-n]=0) then begin  FoundIncPath:=true;z:=v;exit; end
      else  begin w:=py[v-n];  inc(cq); queue[cq]:=w; q[w]:=v; end;
   end;
   FoundIncPath:=false;
end;
```

Tìm đỉnh tự do

```
function FreeNodeFound :boolean;
var i:integer;
begin
   for i:=1 to n do
      if px[i]=0 then
      begin
          u:=i;
          FreeNodeFound:=true;
          exit;
      end;
  FreeNodeFound :=false;
end;
```

Tăng cặp ghép và Sửa nhãn

```
procedure Tangcapghep;
var i, j: integer;
    ok: boolean;
begin
   j:=z; ok:=true;
   while j<>u do
   begin
      i:=q[j];
      if ok then
      begin
         px[i]:=j-n;
         py[j-n]:=i;
      end;
      j:=i;
      ok:= not ok;
   end;
   inc(k);
end;
```

```
procedure Suanhan;
var i, j: integer;
    d: longint;
begin
     d:= maxlongint;
     for i:=1 to n do
        if q[i]>0 then
           for j:=n+1 to n2 do
              if q[j]=0 then
                 if d>longint(f[i]+f[j]-c(i,j-n))
then
                        d:=longint(f[i]+f[j]-c(i,j-n));
     for i:=1 to n do
        if q[i]>0 then dec(f[i],d);
     for j:=n+1 to n2 do
        if q[j]>0 then inc(f[j],d);
end;
```

Main Procedure

```
procedure Solve;
begin
    init;
    while FreeNodeFound do
    begin
        while not FoundIncPath do suanhan;
        Tangcapghep;
    end;
end;
```