\*Matching (match making) problem trong đồ thị 2 phía:

Thế nào là match making problem( vấn đề kết hợp)

* Ta có hai tập các phần tử ( ví dụ như công việc và người thực hiện)
* Ta phải tìm một kết hợp( match up) thỏa mãn một số điều kiện nào đó

Nói 1 chút về đồ thị 2 phía :

* Là đồ thị mà tập các đỉnh V có thể chia làm 2 tập con rời nhau là X và Y nghĩa là:

Với mọi cạnh e = (a,b) thì a thuộc X và b thuộc Y

(mỗi cạnh bắt đầu ở một nút của tập này và kết thúc ở một nút ở tập còn lại)

Maching là những cặp phần tử từ tập X đến tập Y sao cho:

* Hai phần tử khác nhau tử tập X được ghép cặp với 2 phần tử khác nhau trong tập Y

Điều kiện cần và đủ: Định lý Kết hôn (Marriage theorem)

Cho n đàn ông và x phụ nữ ( x>= n) Tồn tại Matching của n đàn ông với n người phụ nữ khi và chỉ khi với bất kì tập con m đàn ông ( m <= n) thì m người đàn ông này biết tối thiểu m phụ nữ

* Chứng minh ( để đó)

\*Vấn đề phân công ( assignment problem)

Là một trong những vấn đề tối ưu hóa tổ hợp cơ bản. Giải quyết vấn đề phân công nghĩa là ta tìm kết hợp tối đa (maximum matching) sao cho kết hợp đó có tổng trọng số trong đồ thị trọng số 2 phía là nhỏ nhất

* Cách giải quyết ngây thơ:
* Chúng ta tìm tất cả các maximum matchings
* Tính tổng trọng số của mỗi maximum matching tìm được
* Chọn maximum matching có chi phí ít nhất có thể
* Hungarian method

\*Hungarian Algorithm:

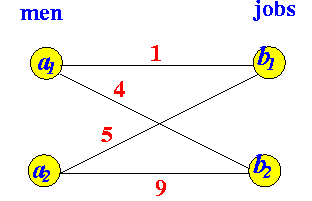
Mục tiêu: Tìm một kết hợp hoàn toàn (kết hợp tối đa, maximum matching, complete matching) với chi phí tối thiểu trong đồ thị 2 phía có trọng số.

Pseudo Code

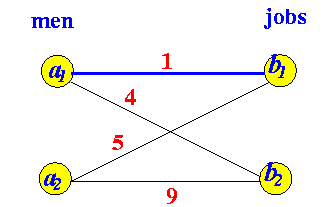
|  |
| --- |
| **Construct a subgraph graph G consisting of the "best cost edges";**  **// We will discuss what is a "best cost edge" later**  **Find a maximal matching M in subgraph G**  **repeat until M is a complete matching**  **{**  **Add the "next best cost edges" to G;**  **// Notice the quotes: it's more complex that just**  **// looking at the cost of an edge**  **Find a maximal matching M in (modified) subgraph G;**  **}** |

* Thế nào là best cost edges (cạnh có chi phí tốt nhất)?
* Chi phí tốt nhất không có nghĩa là thấp nhất!

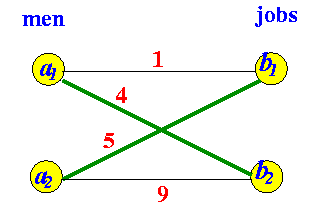
Xét ví dụ



Cạnh có chi phí nhỏ nhất là **(*a1, b1*)**



Nhưng giải pháp để có được chi phí tối thiểu không chứa **(*a1, b1*)**



* Vậy làm thế nào để tìm best cost edges?

Ta có ví dụ sau:

|  |  |
| --- | --- |
| http://www.mathcs.emory.edu/~cheung/Courses/323/Syllabus/Assignment/FIGS/assign40.gif | **Cost matrix:**  **| b1 b2**  **----+----------**  **a1 | 1 4**  **a2 | 5 9** |

Chi phí tối thiểu phát sinh để nối ***a1* = 1**  (bằng cách nối ***a1*** với ***b1***)

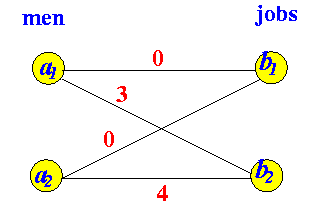
Chi phí tối thiểu phát sinh để nối ***a2* = 5**  (bằng cách nối ***a2*** với ***b1***)

Ta tính chi phí phát sinh thêm khi sử dụng cạnh không tối ưu bằng cách trừ chi phí tối thiểu cho mỗi cạnh liên thuộc với cùng một đỉnh

|  |  |
| --- | --- |
| **Subtract min cost:** | **Result:** |
| http://www.mathcs.emory.edu/~cheung/Courses/323/Syllabus/Assignment/FIGS/assign43a.gif | http://www.mathcs.emory.edu/~cheung/Courses/323/Syllabus/Assignment/FIGS/assign43b.gif |

Những cạnh có trọng số = 0 là những cạnh tốt nhất để nối với a1 và a2 khi không có những vấn đề khác xảy ra. Tuy vậy, chúng ta có một vấn đê là a1 và a2 không thể cùng được match tới cũng 1 đỉnh b1

Ta xét đồ thị chi phí đã được chuẩn hóa:



3 và 4 ở đây có nghĩa là:

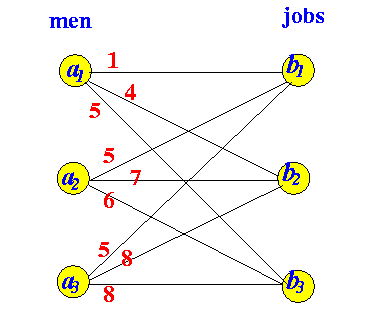
Sẽ tốn thêm 3$ ( tốn hơn 3$ so với cạnh tốt nhất) để nối a1 và b2

Sẽ tốn thêm 4$ ( tốn hơn 4$ so với cạnh tốt nhất) để nối a2 và b2

* Sẽ là rẻ hơn nếu ta nối a1 với b2
* Cạnh (a1, b2) cũng là một trong những cạnh có chi phí tốt nhất

Thuật toán Hungarian

Xét 1 vì dụ về vấn đề phân công



Bước khởi tạo:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| * + Với **mỗi đỉnh ∈ *X* (men)**:  |  | | --- | | * + - 1. Tìm **cạnh có chi phí nhỏ nhất** và       2. **Trừ trọng số của nó** cho tất cả trọng số của các cạnh **nối với đỉnh đó**.   Chúng ta sẽ thu được **tối thiểu là 1** **cạnh có trọng số 0** với mỗi đỉnh. |  * + **Ví dụ:**  |  |  | | --- | --- | | **Cạnh có trọng số thấp nhất với mỗi đỉnh ∈ *X*** | **Sau khi trừ chi phí cho những đỉnh cạnh còn lại** | | http://www.mathcs.emory.edu/~cheung/Courses/323/Syllabus/Assignment/FIGS/assign10.gif | http://www.mathcs.emory.edu/~cheung/Courses/323/Syllabus/Assignment/FIGS/assign11.gif |  * + Với **mỗi đỉnh ∈ *Y* (jobs)**:  |  | | --- | | * + - 1. Tìm cạnh có chi phí tối thiểu và       2. **Trừ trọng số của nó** cho tất cả trọng số của các cạnh nối với đỉnh đó   Ta ***có thể*** thu được nhiều **cạnh có trọng số 0** |  * + **Ví dụ:**  |  |  | | --- | --- | | **Cạnh có trọng số thấp nhất với mỗi đỉnh ∈ *Y*** | **Sau khi trừ chi phí cho những đỉnh cạnh còn lại** | | http://www.mathcs.emory.edu/~cheung/Courses/323/Syllabus/Assignment/FIGS/assign12.gif | http://www.mathcs.emory.edu/~cheung/Courses/323/Syllabus/Assignment/FIGS/assign13.gif | |

Maximum weighted Bipartite Matching (Kuhn Munkras algorithm/Hungarian Method)

-Vấn đề: Số matching tối đa trong một đồ thị 2 phía(Bipartite graph).

-Thuật toán Hungary là một thuật toán tối ưu hóa tổ hợp (combinational optimization algorithm). Nó giải quyết “vấn đề phân công” trong thời gian đa thức. Được phát triển và công bố bởi Harold Kuhn, ông lấy tên là “Hungarian Method” bởi vì phần lớn thuật toán dựa tên nghiên cứu của 2 nhà toán học người Hungary là Dénes Kőnig và Jenő Egerváry (Theo Wikipedia).

- Các định nghĩa:

Matching, matched, unmatched:

Một tập các cạnh M thuộc đồ thị G = (V,E) được gọi là matching của tập U (là tập con của tập các đỉnh V) nếu mỗi đỉnh(vertex) thuộc U đều gắn liền với một cạnh trong M. Những đỉnh trong tập U được gọi là matched; những đỉnh k gắn với bất kì cạnh nào trong M được gọi là unmatched.

Alternating path:

Một path (đường) thuộc đồ thị G, bắt đầu tại đỉnh unmatched trong A và chứa một cách xen kẽ những cạnh thuộc tập E\M và tập M được gọi là alternating path ( đường xen kẻ) đối với M

Augmenting path:

Một alternating path P kết thúc tại 1 đỉnh unmatched thuộc B được gọi là augmenting path( đường tăng cường) bởi vì chúng ta có thể sử dụng nó để khiến M trở thành một matching rộng hơn.