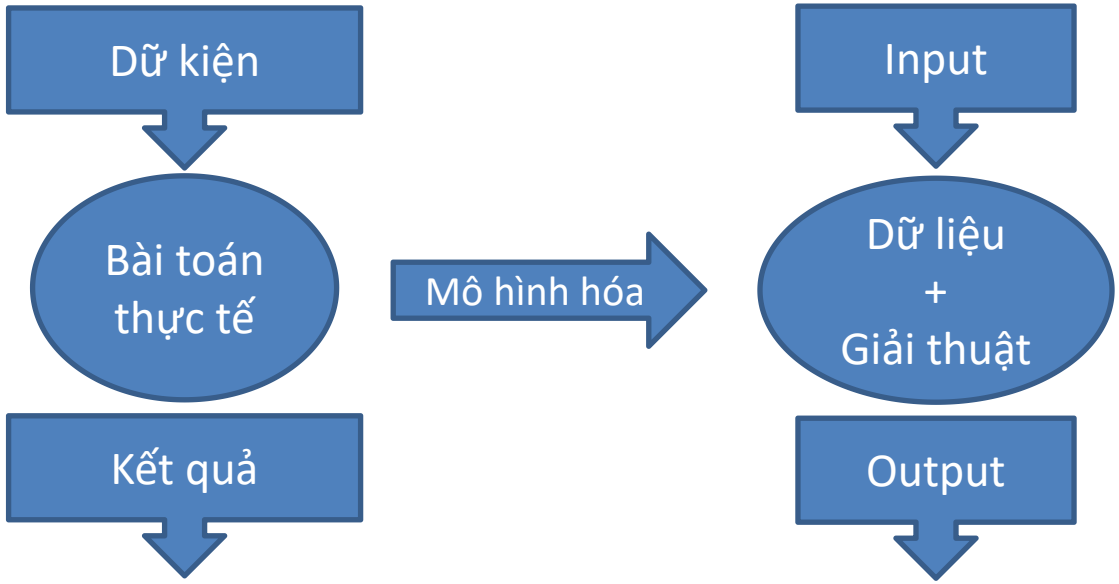
PHÂN TÍCH THIẾT KẾ GIẢI THUẬT

Nội dung

- Cách tiếp cận từ bài toán đến chương trình
- Kiểu dữ liệu trừu tượng (Abstract Data Type).
- Kiểu dữ liệu – Kiểu dữ liệu trừu tượng – Cấu trúc dữ liệu.

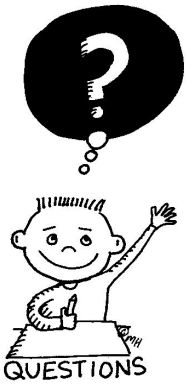
1. Mô hình hóa các bài toán

- Để giải một bài toán trong thực tế bằng máy tính ta phải bắt đầu từ việc xác định bài toán.
  - "phải làm gì?"
  - "làm như thế nào?"
- Hầu hết các bài toán là không đơn giản, không rõ ràng.
- Để giảm bớt sự phức tạp của bài toán thực tế ➔ hình thức hóa nó



Ví dụ: chọn lớp trưởng

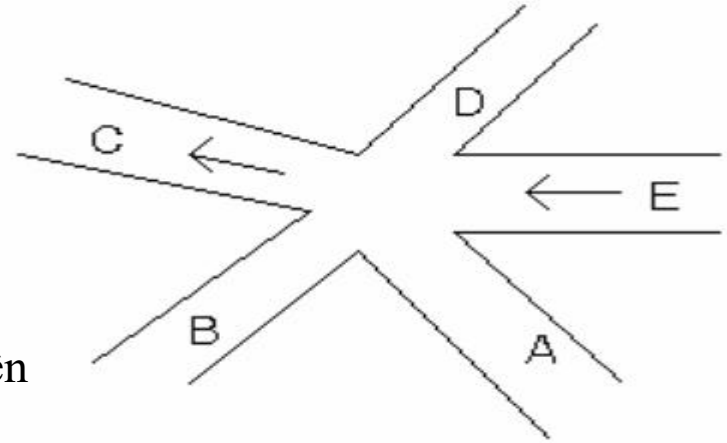
- Yêu cầu: chọn người có điểm cao nhất làm lớp trưởng
- Đánh giá:
  - Lập danh sách tất cả các học sinh trong lớp theo họ tên và điểm trung bình.
  - Sắp thứ tự các học viên giảm dần theo điểm trung bình (học viên có ĐTB bằng nhau thì có cùng hạng).
  - Chọn lọc lớp trưởng:
    - ➢Nếu chỉ có 1 người đứng đầu thì người đó làm lớp trưởng.
    - ➢Nếu hơn 1 người tiến hành bốc thăm.



Ví dụ: Tô màu bản đồ thế giới

- Phát biểu:
  - Ta cần phải tô màu cho các nước trên bản đồ thế giới.
  - Trong đó mỗi nước đều được tô một màu và hai nước láng giềng (cùng biên giới) thì phải được tô bằng hai màu khác nhau.
  - Hãy tìm một phương án tô màu sao cho số màu sử dụng là ít nhất.
- Giải pháp mô hình hóa:
  - Ta có thể xem mỗi nước trên bản đồ thế giới là một đỉnh của đồ thị, hai nước láng giềng của nhau thì hai đỉnh ứng với nó được nối với nhau bằng một cạnh.
  - Bài toán lúc này trở thành bài toán tô màu cho đồ thị như sau: Mỗi đỉnh đều phải được tô màu, hai đỉnh có cạnh nối thì phải tô bằng hai màu khác nhau và ta cần tìm một phương án tô màu sao cho số màu được sử dụng là ít nhất.

Ví dụ: Đèn giao thông

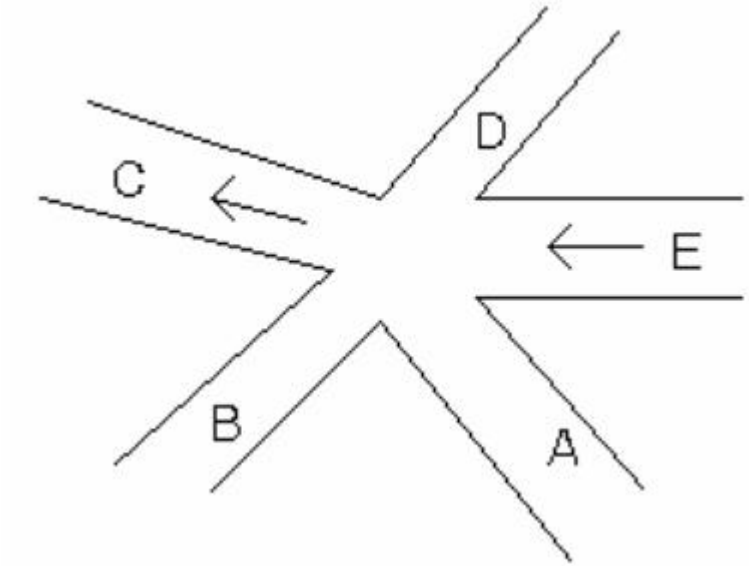


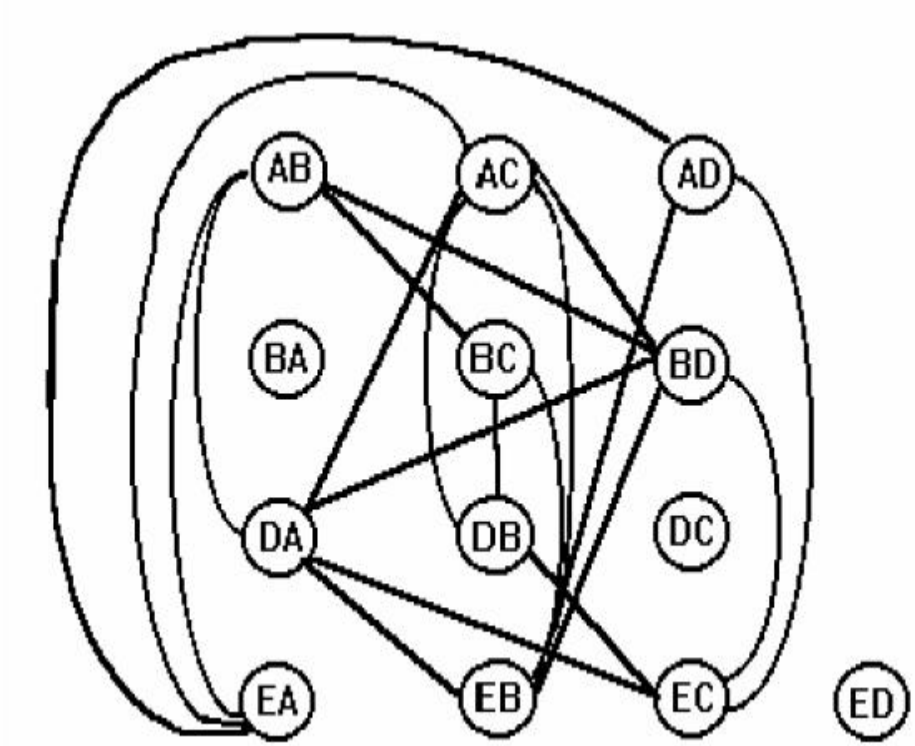
- Phát biểu:
  - Cho một ngã năm trong đó:
    - C và E là các đường một chiều theo chiều mũi tên
    - Các đường khác là hai chiều.
  - Hãy thiết kế một bảng đèn hiệu điều khiển giao thông tại ngã năm này một cách hợp lý
  - Nghĩa là: phân chia các lối đi tại ngã năm này thành các nhóm, mỗi nhóm gồm các lối đi có thể cùng đi đồng thời nhưng không xảy ra tai nạn giao thông (các hướng đi không cắt nhau), và số lượng nhóm là ít nhất có thể được.
- Phân tích:
  - Tại ngã năm này có 13 lối đi: AB, AC, AD, BA, BC, BD, DA, DB, DC, EA, EB, EC, ED.
  - Xác định các lối có thể và không thể đi đồng thời.
  - Vẽ sơ đồ trực quan.
    - Viết tên của 13 lối đi được lên mặt phẳng
    - Hai lối đi nào nếu đi đồng thời sẽ xảy ra đụng nhau (tức là hai hướng đi cắt qua nhau) sẽ được nối lại với nhau.

➔ Ta đã mô hình hoá bài toán giao thông ở trên theo mô hình đồ thị.

Ví dụ: Đèn giao thông

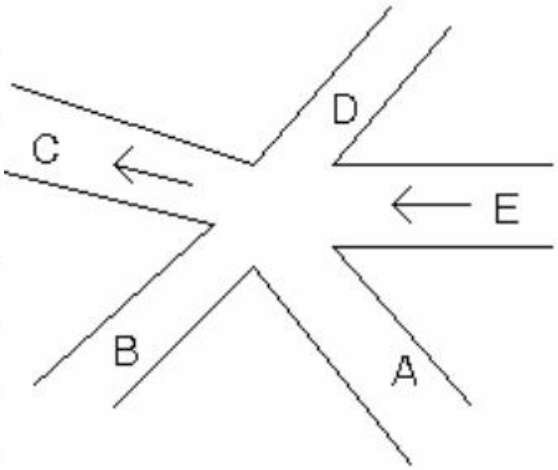
- Giải pháp: Ta sẽ dùng màu tô lên các đỉnh của đồ thị này sao cho:
  - Các lối đi có thể đi đồng thời sẽ có cùng một màu: Tức hai đỉnh có cạnh nối nhau sẽ không được tô cùng màu.
  - Số nhóm là ít nhất: Tức số màu được dùng là ít nhất.

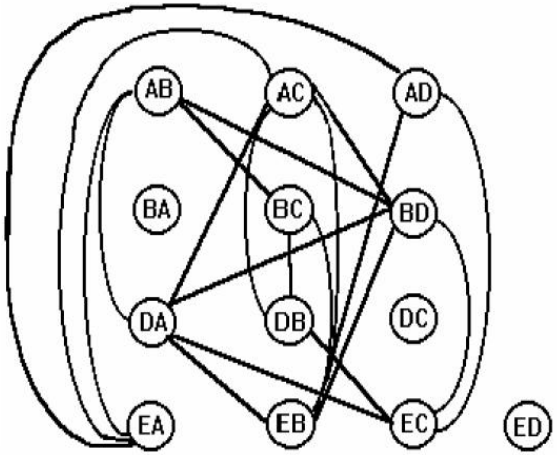




Ví dụ: Đèn giao thông

	AB	AC	AD	BA	BC	BD	DA	DB	DC	EA	EB	EC	ED
AB					1	1	1			1			
AC						1	1	1		1	1		
AD										1	1	1	
BA													
BC	1							1			1		
BD	1	1					1				1	1	
DA	1	1				1					1	1	
DB		1			1							1	
DC													
EA	1	1	1										
EB		1	1		1	1	1						
EC			1			1	1	1					
ED													





2. Thuật toán (Algorithms)

- Thuật toán là một chuỗi hữu hạn các thao tác để giải một bài toán nào đó (Knuth (1973)).
- Tính chất:
  - Hữu hạn (finiteness): giải thuật phải luôn luôn kết thúc sau một số hữu hạn bước. (phải có điểm dừng)
  - Xác định (definiteness): mỗi bước của giải thuật phải được xác định rõ ràng và phải được thực hiện chính xác, nhất quán.
  - Hiệu quả (effectiveness): các thao tác trong giải thuật phải được thực hiện trong một lượng thời gian hữu hạn.
  - Phải có đầu vào (input) và đầu ra (output).
- Biểu diễn giải thuật:
  - Dùng ngôn ngữ tự nhiên.
  - Dùng lưu đồ-sơ đồ khối (flowchart).
  - Dùng mã giả (pseudocode).

(Sẽ trình bày các cách biểu diễn giải thuật ở phần sau)

3. Heuristic

Thuật giải Heuristic là một sự mở rộng khái niệm thuật toán. Nó thể hiện cách giải bài toán với các đặc tính sau:

- *Thường* tìm được lời giải tốt (nhưng không chắc là lời giải tốt nhất)
- Giải bài toán theo thuật giải Heuristic thường dễ dàng và nhanh chóng đưa ra kết quả hơn so với giải thuật tối ưu, vì vậy chi phí thấp hơn.
- Thuật giải Heuristic thường thể hiện khá tự nhiên, gần gũi với cách suy nghĩ và hành động của con người.

Nguyên lý xây dựng Heuristic

- **Nguyên lý vét cạn thông minh:**

Trong một bài toán tìm kiếm nào đó, khi không gian tìm kiếm lớn, ta thường tìm cách giới hạn lại không gian tìm kiếm hoặc thực hiện một kiểu dò tìm đặc biệt dựa vào đặc thù của bài toán để nhanh chóng tìm ra mục tiêu.

- **Nguyên lý tham lam (Greedy):**

Lấy tiêu chuẩn tối ưu (trên phạm vi toàn cục) của bài toán để làm tiêu chuẩn chọn lựa hành động cho phạm vi cục bộ của từng bước (hay từng giai đoạn) trong quá trình tìm kiếm lời giải.

- **Nguyên lý thứ tự:**

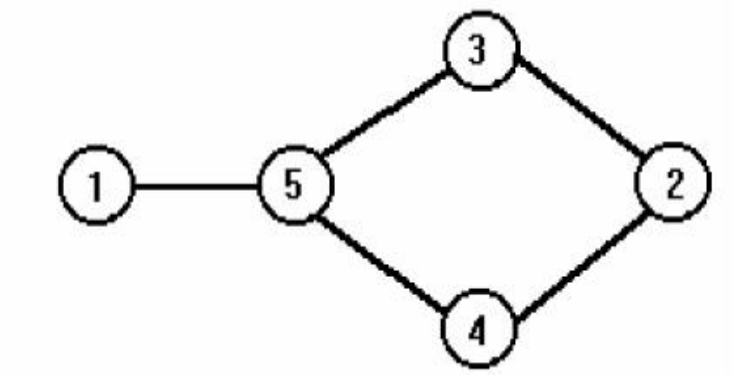
Thực hiện hành động dựa trên một cấu trúc thứ tự hợp lý của không gian khảo sát nhằm nhanh chóng đạt được một lời giải tốt.

- **Hàm Heuristic:**

Trong việc xây dựng các thuật giải Heuristic, người ta thường dùng các hàm Heuristic. Đó là các hàm đánh giá thô, giá trị của hàm phụ thuộc vào trạng thái hiện tại của bài toán tại mỗi bước giải. Nhờ giá trị này, ta có thể chọn được cách hành động tương đối hợp lý trong từng bước của thuật giải.

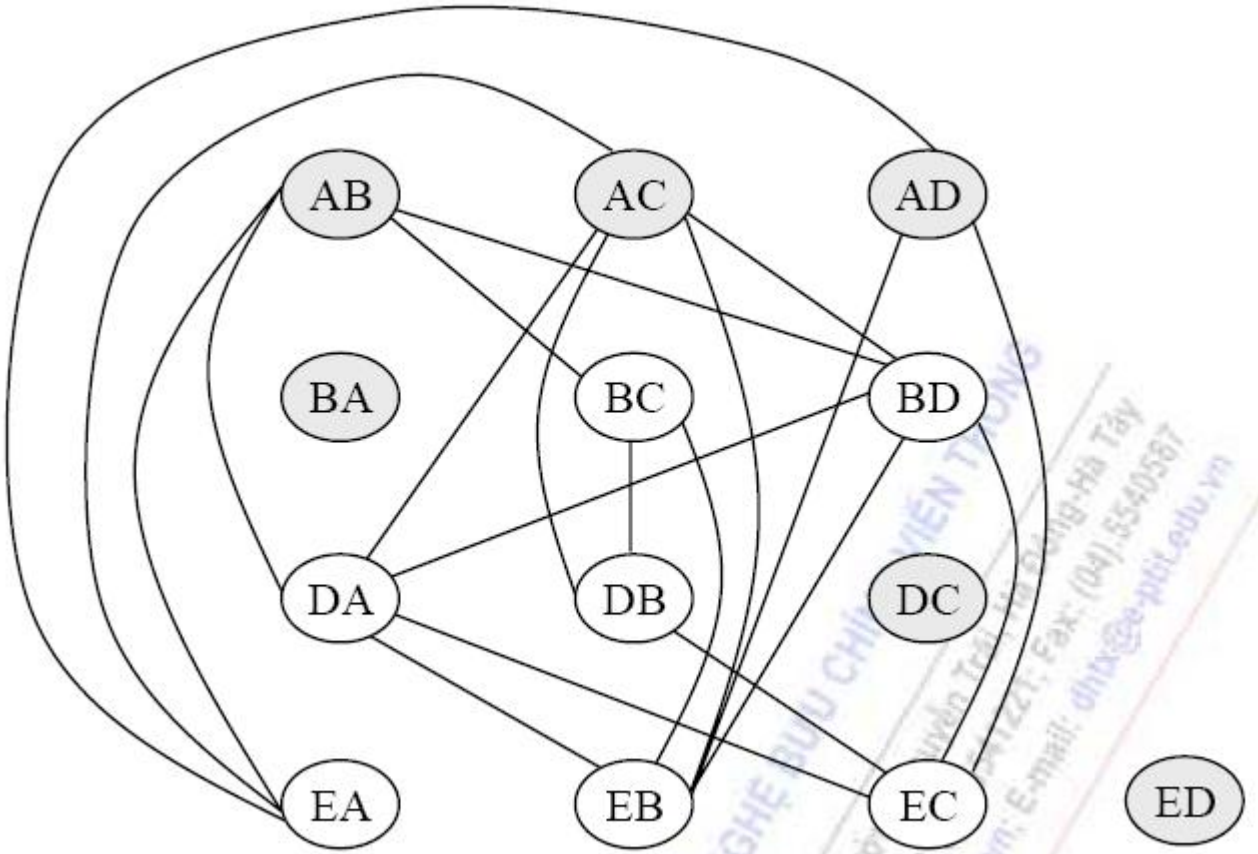
Heuristic: Greedy: Tô màu đồ thị

- Thường gọi là giải thuật "háu ăn" (GREEDY) :
  - Chọn một đỉnh chưa tô màu và tô nó bằng một màu mới C nào đó.
  - Duyệt danh sách các đỉnh chưa tô màu. Đối với một đỉnh chưa tô màu, xác định xem nó có kề với một đỉnh nào được tô bằng màu C đó không. Nếu không có, tô nó bằng màu C đó.

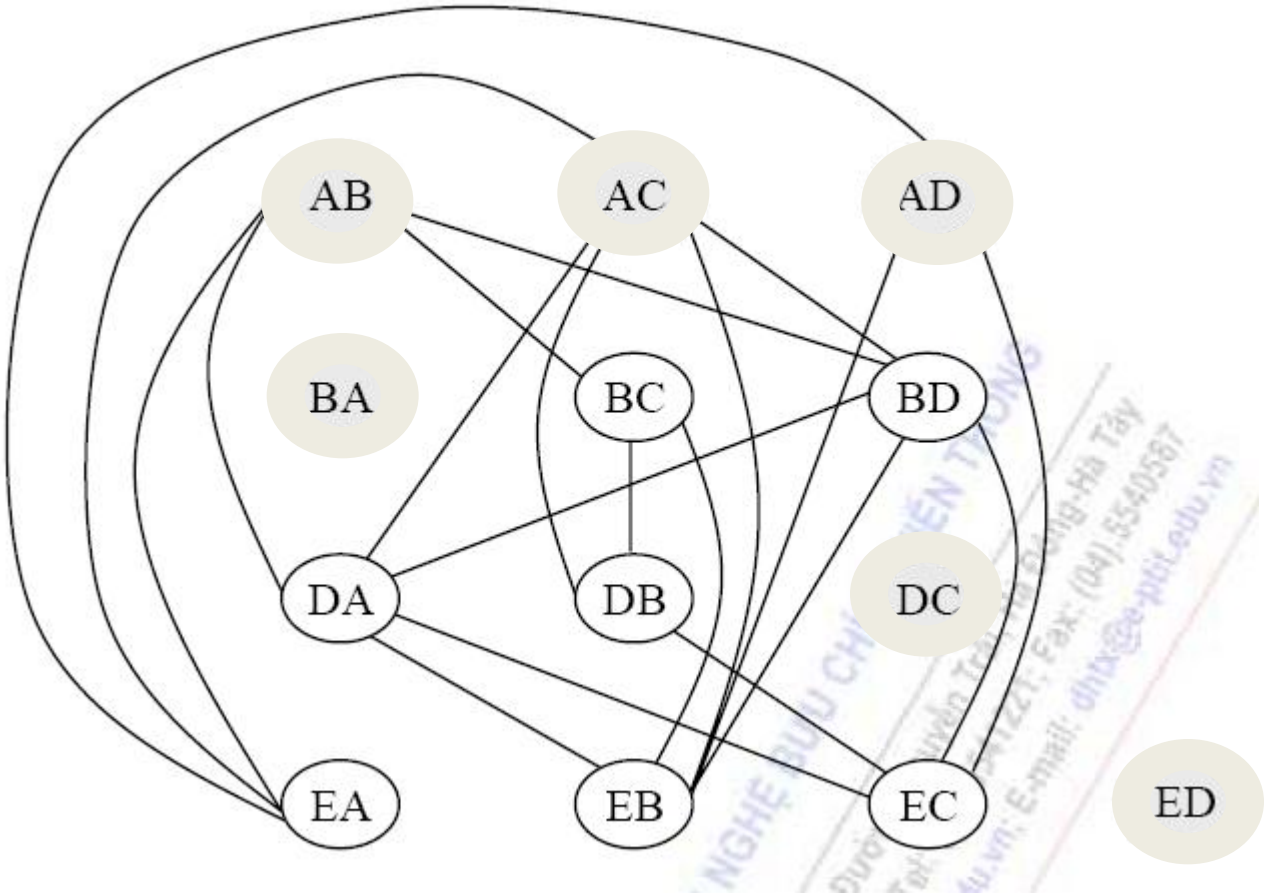


Tô theo GREEDY	Tối ưu
1, 2: đỏ 3, 4: xanh 5: vàng	1, 3, 4: đỏ 2, 5: xanh

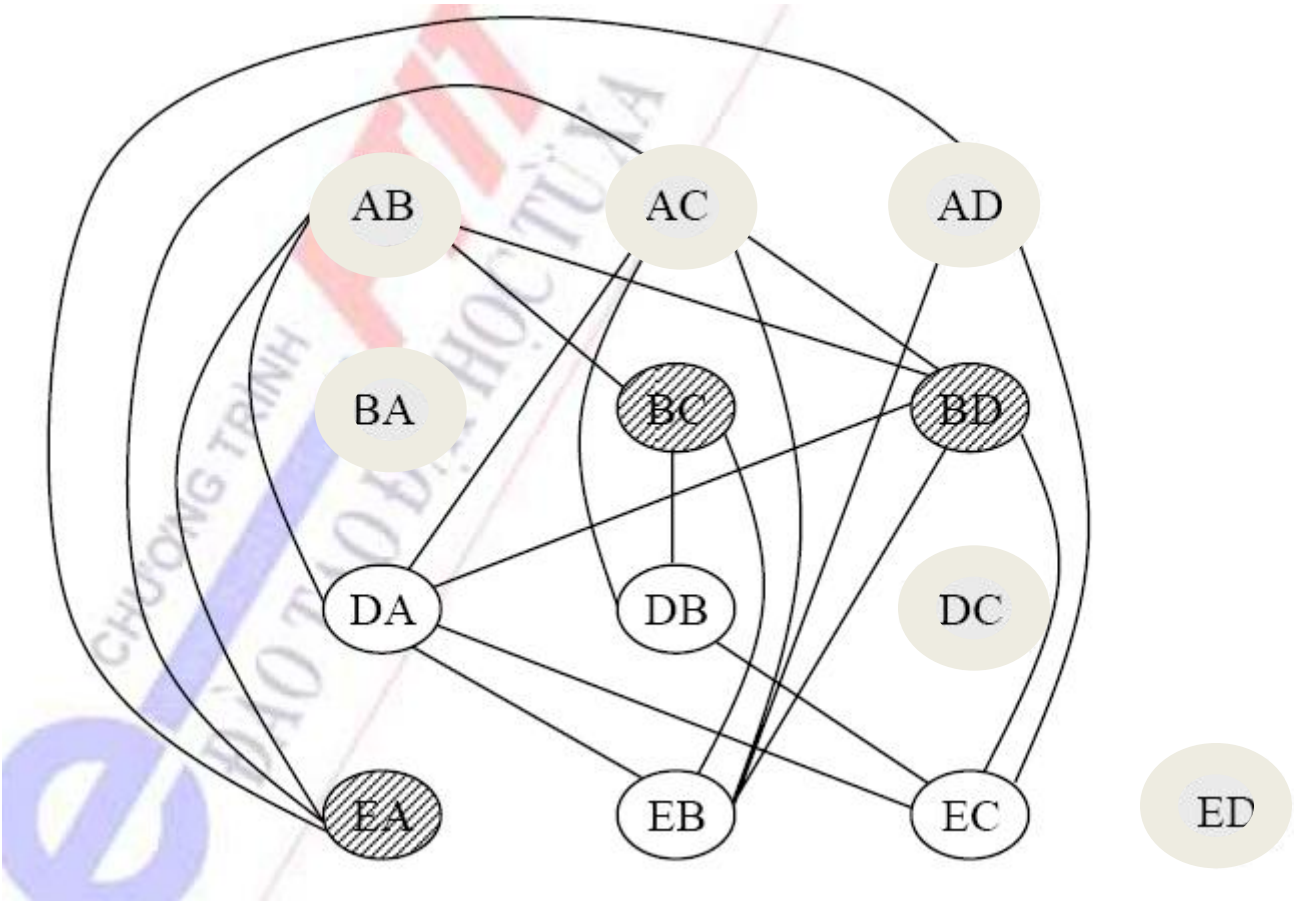
Heuristic: Greedy: Bài toán giao thông



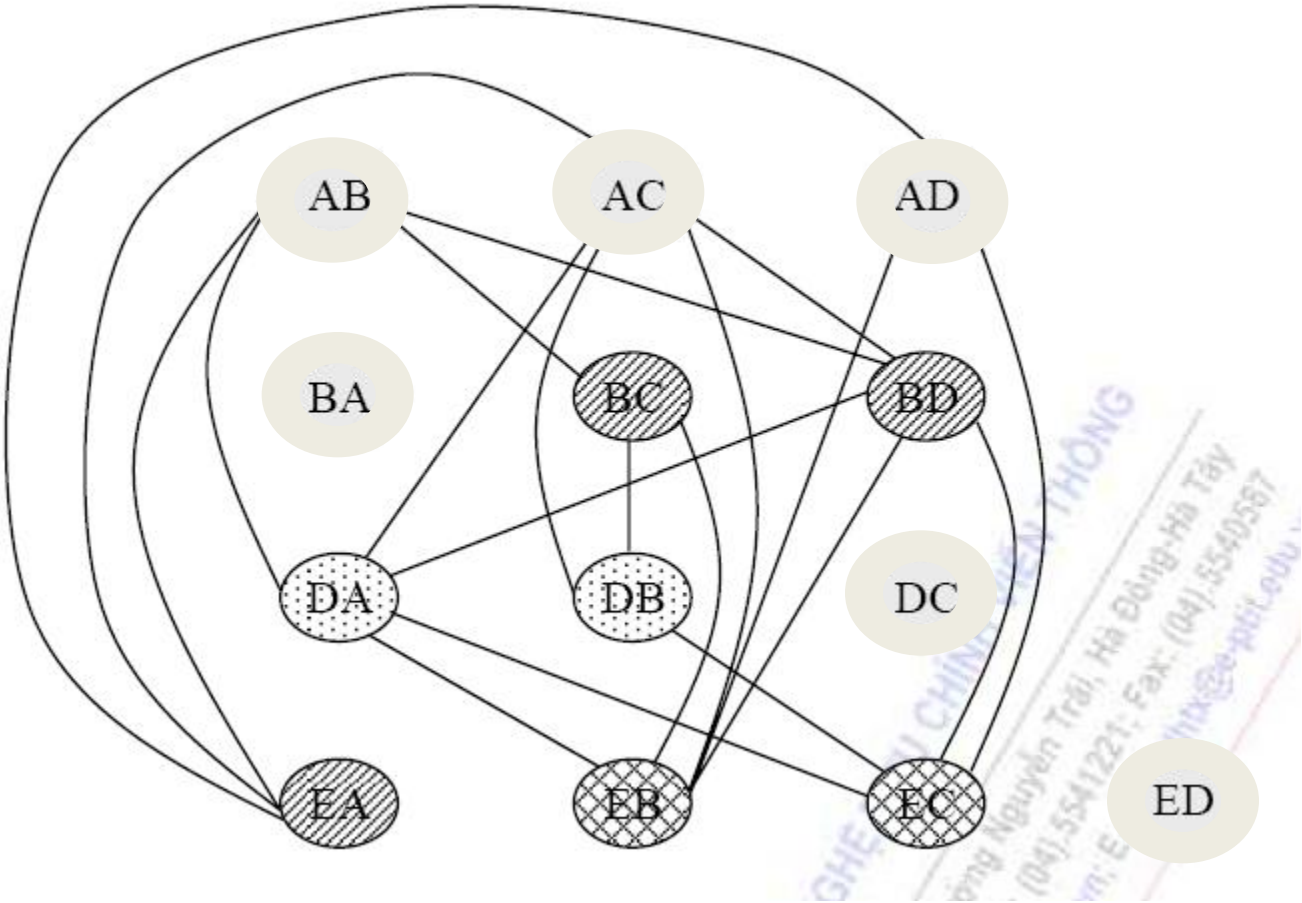
Heuristic: Greedy: Bài toán giao thông



Heuristic: Greedy: Bài toán giao thông



Heuristic: Greedy: Bài toán giao thông



Biểu diễn thuật toán

- Dùng ngôn ngữ tự nhiên
- Dùng mã giả (pseudocode)
- Dùng ngôn ngữ lập trình (Java, C, C++, C#, …)
- Lưu đồ - sơ đồ khối (flowchart)

Bước 1: Tính Δ

$$\Delta = b^2 - 4ac$$

Bước 2: Xét dấu Δ

- Nếu $\Delta > 0$:

$$x1 = \left(-b - \sqrt{\Delta}\right)/2a$$

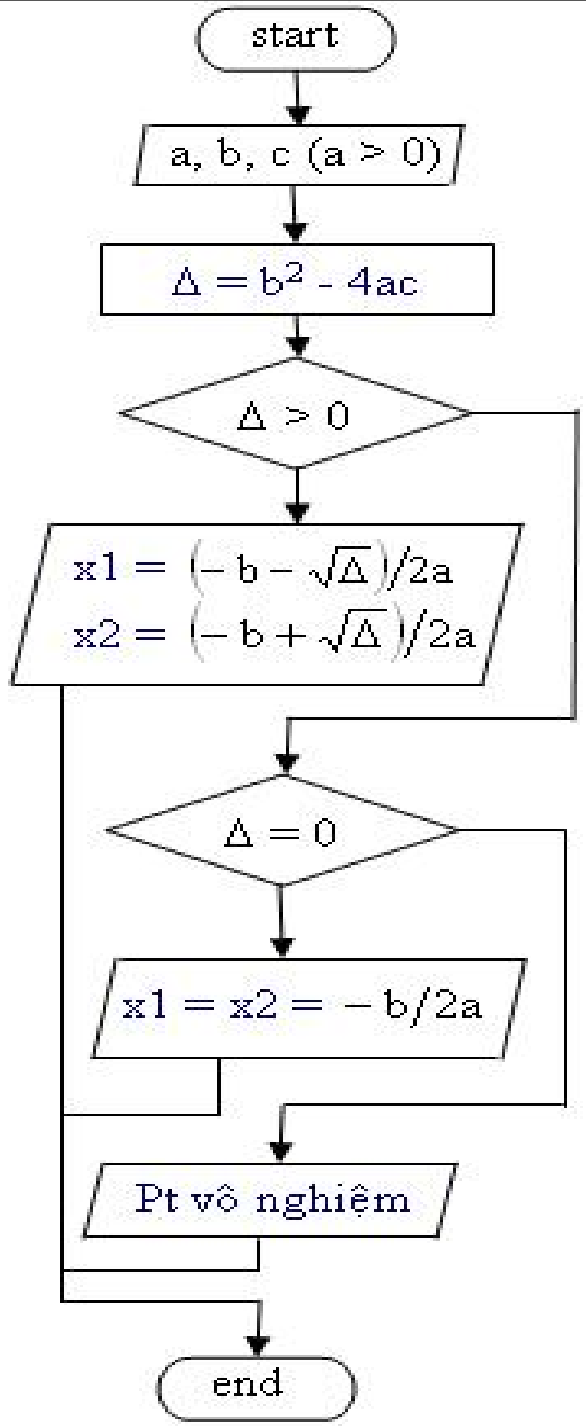
$$x2 = \left(-b + \sqrt{\Delta}\right)/2a$$

- Nếu $\Delta = 0$

$$x1 = x2 = -b/2a$$

- Nếu $\Delta < 0$

Phương trình vô nghiệm.



Ngôn ngữ giả & tinh chế từng bước

- Khi đã có *mô hình thích hợp* cho bài toán ➔cần *hình thức hoá một giải thuật* trong thuật ngữ của mô hình đó.
- Ví dụ GREEDY:

```
void Greedy(GRAPH G, SET mauMoi){
    mauMoi = Tập rỗng;
    for mỗi đỉnh v chưa được tô màu thuộc G
        If v không được nối tới đỉnh nào trong tập mauMoi{
            Tô màu mới cho đỉnh v;
            Đưa v vào tập mauMoi;
    }
}
```

Ngôn ngữ giả & tinh chế từng bước

- Sau đó tinh chỉnh từng bước ta có:

```
void Greedy(GRAPH g, SET mauMoi){
    int tonTai;
    int v, w;
    mauMoi = tập rỗng;
    v = đỉnh chưa tô màu đầu tiên trong G;
    while v !=null{
       tonTai = 0;
       w = đỉnh đầu tiên trong mauMoi;
       while w != null{
       if tồn tại cạnh nối v và w trong G;
       tonTai = 1;
       w = đỉnh tiếp theo trong mauMoi;      }
       If tonTai == 1;{
       tô màu mới cho đỉnh v;
       đưa v vào tập hợp mauMoi;    }
       v = đỉnh chưa tô màu tiếp theo trong G;}}
```

Các bước tiếp cận với một bài toán

- **Mô hình hoá** bài toán bằng một mô hình toán học thích hợp.
- **Tìm giải thuật** trên mô hình này.
  - Giải thuật có thể mô tả một cách không hình thức.
  - Nó chỉ nêu phương hướng giải hoặc các bước giải một cách tổng quát.
- **Hình thức hoá** giải thuật bằng cách viết một thủ tục bằng ngôn ngữ giả
  - Sau đó: chi tiết hoá dần + kiểu dữ liệu trừu tượng + các cấu trúc điều khiển trong ngôn ngữ lập trình ➔ mô tả giải thuật.
- **Cài đặt** giải thuật trong một ngôn ngữ lập trình cụ thể

Tóm tắt các bước

Mô hình toán học	Kiểu dữ liệu trừu trượng	Cấu trúc dữ liệu
Giải thuật không hình thức	Chương trình ngôn ngữ giả	Chương trình Java, C, Pascal…

Kiểu dữ liệu trừu tượng (ADT)

- Khái niệm trừu tượng hóa:
  - Trong tin học, trừu tượng hóa nghĩa là đơn giản hóa, làm cho nó sáng sủa hơn và dễ hiểu hơn.
  - Cụ thể trừu tượng hóa là che đi những chi tiết, làm nổi bật cái tổng thể.
  - Trừu tượng hóa có thể thực hiện trên hai khía cạnh:
    - Trừu tượng hóa dữ liệu
    - Trừu tượng hóa chương trình.

Trừu tượng hóa chương trình

- Trừu tượng hóa chương trình là sự định nghĩa các chương trình con để tạo ra các phép toán trừu tượng (sự tổng quát hóa của các phép toán nguyên thủy).
- Trừu tượng hóa chương trình cho phép phân chia chương trình thành các chương trình con ➔ che dấu tất cả các lệnh cài đặt chi tiết trong các chương trình con.
- Ví dụ:

```
void Main() {
Nhap( Lop);
Xu_ly (Lop);
Xuat (Lop);}
```

Trừu tượng hóa dữ liệu

- Trừu tượng hóa dữ liệu là định nghĩa các kiểu dữ liệu trừu tượng.
- *Một kiểu dữ liệu trừu tượng là một mô hình toán học cùng với một tập hợp các phép toán (operator) trừu tượng được định nghĩa trên mô hình đó.*
  - Ví dụ tập hợp số nguyên cùng với các phép toán hợp, giao, hiệu là một kiểu dữ liệu trừu tượng.

Trừu tượng hóa dữ liệu

- Ví dụ: giải thuật GREEDY

Câu lệnh	Mệnh đề hình thức
MAKENULL(mauMoi)	mauMoi = tập rỗng
w=FIRST(mauMoi)	w = đỉnh đầu tiên trong tập mauMoi
w=NEXT(w, mauMoi)	w = đỉnh kế tiếp trong tập mauMoi
INSERT(v, mauMoi)	Đưa v vào tập hợp mauMoi

Trừu tượng hóa dữ liệu

- Thuận lợi khi dùng ADT:
  - Có thể định nghĩa một kiểu dữ liệu tuỳ ý cùng với các phép toán cần thiết trên nó rồi chúng ta dùng như là các đối tượng nguyên thuỷ.
  - Có thể cài đặt một ADT bằng bất kỳ cách nào, chương trình dùng chúng cũng không thay đổi, chỉ có các chương trình con biểu diễn cho các phép toán của ADT là thay đổi.

Kiểu dữ liệu – Cấu trúc dữ liệu – Kiểu dữ liệu trừu tượng

- Kiểu dữ liệu là một tập hợp các giá trị và một tập hợp các phép toán trên các giá trị đó.
  - Kiểu Boolean là một tập hợp có 2 giá trị TRUE, FALSE và các phép toán trên nó như OR, AND, NOT ….
  - Kiểu Integer là tập hợp các số nguyên có giá trị từ -32768 đến 32767 cùng các phép toán cộng, trừ, nhân, chia, Div, Mod…
- Kiểu dữ liệu có hai loại:
  - Kiểu dữ liệu sơ cấp là kiểu dữ liệu mà giá trị của nó là đơn nhất.
    - Ví dụ: kiểu Boolean, Integer….
  - Kiểu dữ liệu có cấu trúc hay còn gọi là cấu trúc dữ liệu là kiểu dữ liệu mà giá trị dữ liệu của nó là sự kết hợp của các giá trị khác.
    - Ví dụ: ARRAY là một cấu trúc dữ liệu

Kiểu dữ liệu – Cấu trúc dữ liệu – Kiểu dữ liệu trừu tượng

- Một kiểu dữ liệu trừu tượng là một mô hình toán học cùng với một tập hợp các phép toán trên nó.
- Có thể nói kiểu dữ liệu trừu tượng là một kiểu dữ liệu do chúng ta định nghĩa ở mức khái niệm (conceptual), nó chưa được cài đặt cụ thể bằng một ngôn ngữ lập trình.
- Khi cài đặt một kiểu dữ liệu trừu tượng trên một ngôn gnữ lập trình cụ thể, chúng ta phải thực hiện hai nhiệm vụ:
  - Biểu diễn kiểu dữ liệu trừu tượng bằng một cấu trúc dữ liệu hoặc một kiểu dữ liệu trừu tượng khác đã được cài đặt.
  - Viết các chương trình con thực hiện các phép toán trên kiểu dữ liệu trừu tượng mà ta thường gọi là cài đặt các phép toán.

Tóm tắt

- Trong chương này, chúng ta cần phải nắm vững các vấn đề sau:
  - Các bước phân tích và lập trình để giải quyết một bài toán thực tế.
  - Hiểu rõ khái niệm về kiểu dữ liệu, kiểu dữ liệu trừu tượng và cấu trúc dữ liệu.

HỎI ĐÁP