THUẬT TOÁN, THUẬT GIẢI
MỘT SỐ PHƯƠNG PHÁP GIẢI QUYẾT VẤN ĐỀ

Nội dung

Vấn đề, giải quyết vấn đề

Khái niệm về thuật toán, thuật giải

Các nguyên lý của thuật giải heuristic

Vấn đề?

Những vướng mắc khó khăn cần giải quyết

Một yêu cầu tìm kiếm xử lý trong một ngữ cảnh cụ thể

Bao gồm:

- các sự kiện;
- các thông tin ;
- những ràng buộc nhất định.

vấn đề = bài toán

Mô hình vấn đề

A ➔ B

A: giả thiết, điều kiện ban đầu

B: kết luận cần đạt đến

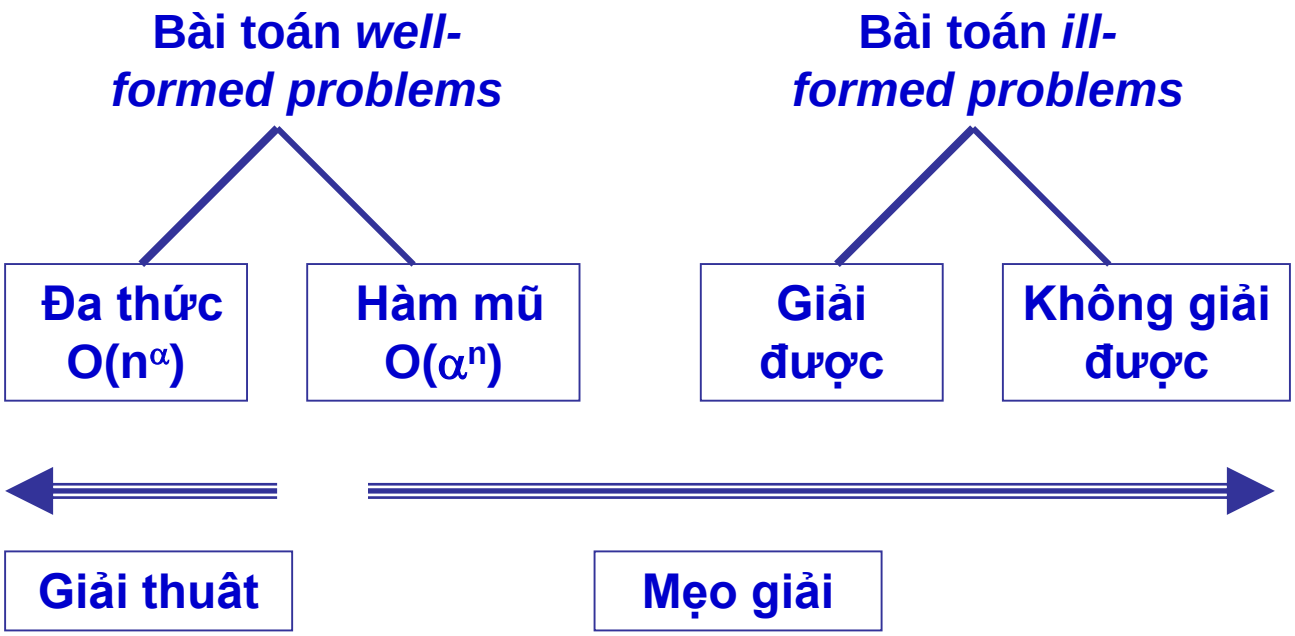
➔: suy luận hay giải pháp cần xác định = một số hữu hạn bước

Phân loại vấn đề & Các đặc trưng cơ bản của vấn đề

Vấn đề phát biểu chỉnh (well-formed problems): Là các bài toán có thể biết được hình trạng đầu, hình trạng đích và có thể quyết định khi nào vấn đề được coi là giải quyết xong. Các bài toán 1 - 2 là những vấn đề được phát biểu chỉnh.

Vấn đề phát biểu không chỉnh (ill-formed problems): Là các vấn đề được phát biểu chưa đầy đủ, thiếu dữ kiện. Các bài toán chẩn đoán và điều trị bệnh, bài toán xác định chất lượng sản phẩm là các bài toán phát biểu không chỉnh.

Phân loại vấn đề



Thuật toán

Thuật toán:

Là chuỗi hữu hạn các công việc trình tự xác định các thao tác để giải các bài toán.

Tính chất:

1)Tính xác định.

2)Tính đúng đắn.

3)Tính dừng

Thuật toán

Thuật toán có thể được thể hiện qua:

- Ngôn ngữ tự nhiên
- Lưu đồ
- Mã giả
- NN lập trình

Ngoài ra thuật toán còn phải đạt hiệu quả cao hay có độ phức tạp thấp

Thuật toán

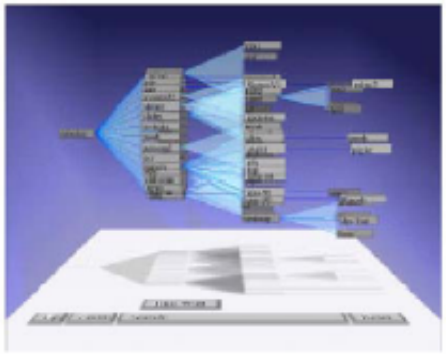
$O(\log_2 n)$
$O(n)$
$O(n \log_2 n)$
$O(n^2)$
$O(n^k)$

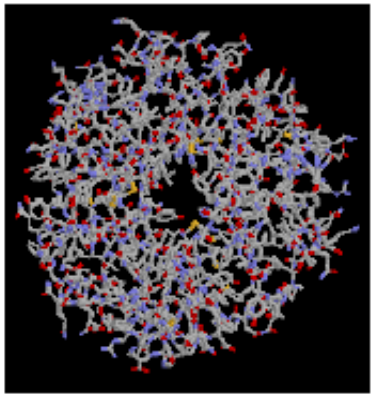
Độ phức tạp đa thức $\Rightarrow$ chấp nhận được

$O(2^n)$
$n!$

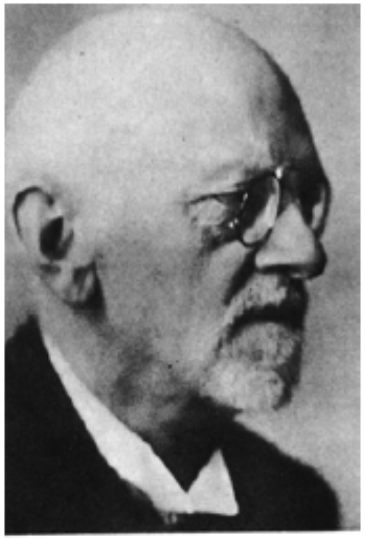
Độ phức tạp cao $\Rightarrow$ khả chấp nhận

Chiến đấu với độ phức tạp tính toán





23 bài toán của thế kỷ 20



Hilbert

- Tại Đại hội Toán học Thế giới lần thứ hai (Paris, tháng Năm 1900), Hilbert nêu ra 23 bài toán, thách thức các nhà toán học toàn thế giới giải trong thế kỷ 20.
- 12 bài toán đã được giải toàn bộ, 8 bài toán được giải từng phần, 3 bài vẫn chưa có lời giải.

7 bài toán của thế kỷ 21

- Vào lúc 4 giờ chiều Thứ tư ngày 24 tháng 5 năm 2000, Viện Toán học Clay công bố và thách thức 7 bài toán của thế kỷ 21 (1 triệu $ cho mỗi lời giải).
- Bài toán số 1: P versus NP
- Sáu bài toán khác:
  1. The Hodge Conjecture,
  2. The Poincaré Conjecture,
  3. The Riemann Hypothesis,
  4. Yang-Mills Existence and Mass Gap,
  5. Navier-Stokes Existence and Smoothness,
  6. The Birch and Swinnerton-Dyer Conjecture

Bài toán "P versus NP"

- Nếu ai đó hỏi rằng liệu 13.717.421 có là tích của hai số nhỏ hơn không, bạn sẽ cảm thấy rất khó trả lời là đúng hay sai.
- Nếu người đó bảo bạn rằng số này có thể là tích của 3607 và 3803, bạn có thể kiểm tra điều này thật dễ dàng.
- **Xác định xem với một bài toán cho trước, liệu có tồn tại một lời giải có thể kiểm chứng nhanh (bằng máy tính chẳng hạn), nhưng lại cần rất nhiều thời gian để giải từ đầu (nếu không biết lời giải)?**
- Có rất nhiều bài toán như vậy. Chưa ai có thể chứng minh được rằng, với bất kỳ bài toán nào như vậy, thực sự cần rất nhiều thời gian để giải. Có thể chỉ đơn giản là chúng ta vẫn chưa tìm ra được cách giải chúng nhanh chóng. Stephen Cook phát biểu bài toán **P versus NP** vào năm 1971.

Độ phức tạp tính toán—Sự tồn tại các bài toán giải được nhưng vô cùng khó giải

- Độ phức tạp tính toán: **P** (thời gian đa thức) và **non-P** (thời gian hàm mũ). Bài toán kiểu P có thể giải dễ dàng (sắp xếp dãy số theo thứ tự), bài toán kiểu non-P rất khó giải (tìm các thừa số nguyên tố của một số nguyên cho trước).
- Người ta tin rằng có rất nhiều bài toán thuộc kiểu non-P, nhưng chưa bao giờ chứng minh được chính chúng là như vậy (hết sức khó).
- **NP** (Nondeterministic Polynomial) là một họ đặc biệt các bài toán kiểu non-P: nếu bất kỳ trong chúng có nghiệm thời gian đa thức thì tất cả sẽ có nghiệm thời gian đa thức.
- **P = NP?** Các bài toán kiểu P và NP là như nhau?

Thời gian đa thức và hàm mũ

Time complexity function	n = 10	n = 20	n = 30	n = 40	n = 50	n = 60
n	0.0001 second	0.0002 second	0.0003 second	0.0004 second	0.0005 second	0.0006 second
n^2	0.001 second	0.002 second	0.003 second	0.004 second	0.005 second	0.006 second
n^3	0.01 second	0.02 second	0.03 second	0.04 second	0.05 second	0.06 second
n^5	1 second	3.2 second	24.3 second	1.7 minutes	5.2 minutes	13.0 minutes
2^n	0.01 second	1.0 second	17.9 second	12.7 days	35.7 years	336 centuries
3^n	0.059 second	58 minutes	6.5 years	3855 centuries	$2x10^8$ centuries	$1.3x10^{13}$ centuries

Thời gian đa thức và hàm mũ

Time complexity function	With present computer	With computer 100 times faster	With computer 1000 times faster
n	N_1	$100\ N_1$	$1000\ N_1$
n^2	N_2	$10\ N_2$	$31.6\ N_2$
n^3	N_3	$4.64\ N_3$	$10\ N_3$
n^5	N_4	$2.5\ N_4$	$3.98\ N_4$
2^n	N_5	$N_5 + 6.64$	$N_5 + 9.97$
3^n	N_6	$N_6 + 4.19$	$N_6 + 6.29$

Một số ví dụ về bài toán có độ phức tạp cao

Bài toán phân công công việc

Một đề án gồm n công việc và các việc sẽ được thực hiên bởi m máy như nhau.

Giả sử biết thời gian để 1 máy thực hiện viêc thứ j là tj.

Yêu cầu: Tìm phương án phân công sao cho thời gian hoàn thành toàn bộ công việc là thấp nhất.

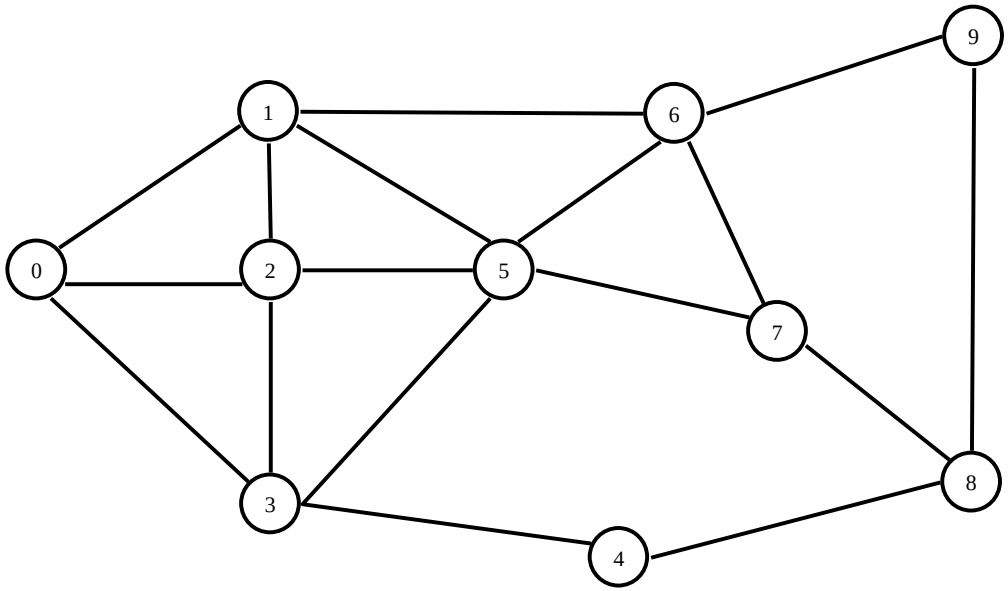
Mẫu số liệu: n=10, m=3, tj = 4 9 5 2 7 6 10 8 7 5

Một số ví dụ về bài toán có độ phức tạp cao

Bài toán tô màu

Giả sử ta có bản đồ các quốc gia trên thế giới, ta muốn tô màu các quốc gia này sao cho các nước khác nhau được tô khác màu.

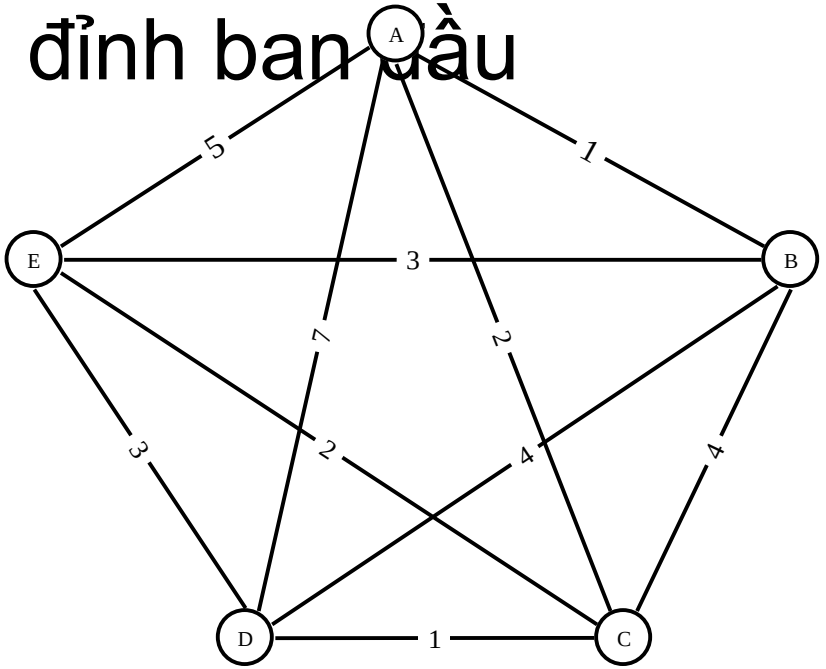
Yêu cầu tìm cách tô sao cho số màu sử dụng là ít nhất.



Một số ví dụ về bài toán có độ phức tạp cao

Bài toán người đưa thư

Giả sử có một đồ thị có trọng số dương, tìm đường đi nhắn nhất qua tất cả các đỉnh của đồ thị rồi trở về đỉnh ban đầu



Thuật giải

Thuật giải:

Giải pháp được viết dưới dạng thủ tục tương tự như thuật toán nhưng không đòi hỏi các tiêu chuẩn như thuật toán.

Tính đúng: chấp nhận các thuật giải đơn giản có thể cho kết quả đúng hay gần đúng nhưng có khả năng thành công cao hơn.

Thuật giải *(tt)*

Để có thể được chấp nhận thuật giải phải thể hiện một giải pháp hợp lý nhất có thể trong tình huống hiện tại bằng cách:

- Tận dụng mọi thông tin hữu ích
- Sử dụng tri thức, kinh nghiệm trực giác của con người
- Tự nhiên đơn giản nhưng cho kết quả chấp nhận được

Thuật giải Heuristic:

Là mở rộng khái niệm thuật toán.

- Thường tìm lời giải tốt nhưng không tốt nhất.
- Nhanh chóng tìm ra kết quả hơn so với giải thuật tối ưu, vì vậy chi phí thấp hơn.
- Thường thể hiện khá tự nhiên, gần gũi với cách suy nghĩ và hành động của con người.

Các nguyên lý của thuật giải heuristic

Vét cạn thông minh

Nguyên lý thứ tự

Nguyên lý tham lam

Hàm heuristic

Kỹ thuật Heuristics

- Theo Từ điển tiếng Anh Oxford: “Heuristics là nghệ thuật tìm kiếm chân lý. Nói riêng, heuristics là đặc trưng của quá trình học nhờ đó các học sinh học được cách tự tìm ra cách giải thích các hiện tượng tự nhiên”.
- Từ “Heuristics” có cùng một gốc tiếng Hy Lạp như từ Eureka. Feigenbaum Feldman đã đưa ra định nghĩa :
- “Heuristics (Các quy tắc heuristics, các phương pháp heuristics) là các quy tắc, phương pháp, chiến lược, mẹo giải hay phương cách nào đó nhằm làm giảm khối lượng tìm kiếm lời giải trong không gian bài toán cực lớn”.

Các nguyên lý của thuật giải heuristic

1.Vét cạn thông minh

Hạn chế vùng không gian tìm kiếm và có sự định hướng để nhanh chóng tìm đến mục tiêu.

Tạo miền D’ rất nhỏ so với D

Vét cạn trên D’

Các nguyên lý của thuật giải heuristic

2.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.

a)Thuật giải GTS1: (Greedy-Traveling Saleman)

Xây dựng một lịch trình du lịch có chi phí Cost tối thiểu cho bài toán trong trường hợp phải qua n thành phố với ma trận chi phí C và bắt đầu tại một đỉnh U nào đó.

Các nguyên lý của thuật giải heuristic

Thuật giải:
Bước 1: {Khởi đầu}
Đặt Tour := {};
Cost := 0;
V := U; {V là đỉnh hiện tại đang làm việc}
Bước 2: {Thăm tất cả các thành phố}
For k := 1 To n Do
qua bước 3;

Bước 3: {Chọn cung kế tiếp}

Đặt (V, W) là cung có chi phí nhỏ nhất tình từ V đến các đỉnh W chưa dùng:

Tour := Tour + {(V,W)};

Cost := Cost + Cost(V,W);

Nhãn W được sử dụng

Đặt V := W; {Gán để xét bước kế tiếp}

Bước 4: {Chuyến đi hoàn thành}

Đặt Tour := Tour + {(V,U)};

Cost := Cost + Cost(V,U);

Dừng.

Ví dụ :

$$C = \begin{bmatrix} \infty & 1 & 2 & 7 & 5 \\ 1 & \infty & 4 & 4 & 3 \\ 2 & 4 & \infty & 1 & 2 \\ 7 & 4 & 1 & \infty & 3 \\ 5 & 3 & 2 & 3 & \infty \end{bmatrix}$$

U = A

Tour = {}

Cost = 0

V = A

W $\in$ {B, C, D, E}{Các đỉnh có thể đến từ A}

$\rightarrow$ W = B{Vì qua B có giá thành bé nhất}

Tour = {(A, B)}

Cost = 1

V = B

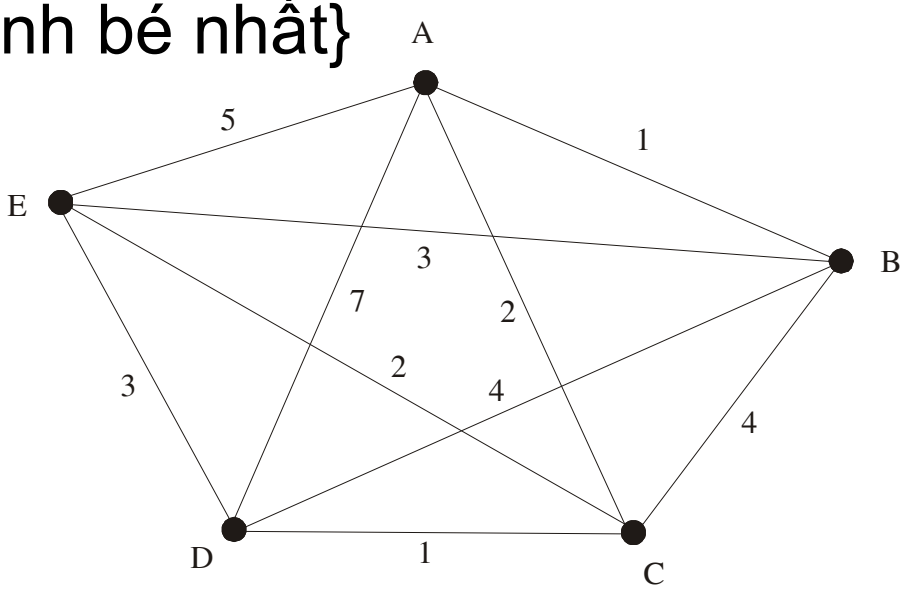
W $\in$ {C, D, E}$\rightarrow$ W = E

Tour = {(A, B),(B, E)}

Cost = 1 + 3 = 4

V = E

W $\in$ {C, D} $\rightarrow$ W = C



Ví dụ :

Tour = {(A, B), (B, E), (E, C)}

Cost = 4 + 2 = 6

V = C

$W \in \{D\}$

$\rightarrow$ W = D

Tour = {(A, B), (B, E), (E, C), (C, D)}

Cost = 6 + 1 = 7

V = D

Tour = {(A, B), (B, E), (E, C), (C, D), (D, A)}

Cost = 7 + 7 = 14

Kết quả:Tour du lịch $A \rightarrow B \rightarrow E \rightarrow C \rightarrow D \rightarrow A$ với giá thành Cost = 14.

Nhận xét: Tuy nhiên kết quả nhỏ nhất sẽ là $A \rightarrow B \rightarrow D \rightarrow C \rightarrow E \rightarrow A$ với Cost=13. Sở dĩ không tối ưu do "háu ăn": cứ hướng nào có chi phí thấp thì đi, bất chấp về sau.

b)Thuật giải GTS2:

Tạo ra lịch trình từ p thành phố xuất phát riêng biệt. Tìm chu trình của người bán hàng qua n thành phố (1<p<n) và p chu trình được tạo ra và chỉ chu trình tốt nhất trong p chu trình được giữ lại mà thôi (thuật giải này đòi hỏi phải nhập n, p và C)

Thuật giải:

Bước 1: {Khởi đầu}

k := 0; {Đếm số thành phố đi qua}

Best := {}; {Ghi nhớ chu trình tốt nhất tìm thấy có chi phí là Cost}

Cost := ∞;

b)Thuật giải GTS2:(tt)

Bước 2: {Bắt đầu chu trình mới}

Chuyển qua bước 3 khi k<p, ngược lại dừng.

Bước 3: {Tạo chu trình mới}

k := k + 1;

Call (GTS1(Vk)) : Trả về một chu trình T(k) ứng với chi phí C(k).

Bước 4: {Cập nhật chu trình tốt nhất}

Nếu C(k)< Cost thì

Best := T(k);

Cost := C(k);

b)Thuật giải GTS2:(tt)

Ví dụ: (Giải lại ví dụ trên) p=3

$$C = \begin{bmatrix} \infty & 25 & 40 & 31 & 27 \\ 5 & \infty & 17 & 30 & 25 \\ 19 & 15 & \infty & 6 & 1 \\ 9 & 30 & 24 & \infty & 6 \\ 22 & 8 & 7 & 10 & \infty \end{bmatrix}$$

k=0 Best={} Cost=∞

k=0 < p V_0=A Call(GTS1(A))

=>T(0) = {(A,B),(B,E),(E,C),(C,D),(D,A)},

C(0) = 14 < Cost =>Best = T(0) Cost = 14

k=1 < p V_1=B Call(GTS1(B))

=>T(1) = {(B,A),(A,C),(C,D),(D,E),(E,B)},

C(1) = 10 < Cost=>Best = T(1) Cost = 10

k=2 < p V_2 = C Call(GTS1(C))

=>T(2) = {(C,D),(D,E),(E,B),(B,A),(A,C)},

C(2) = 10 >= Cost

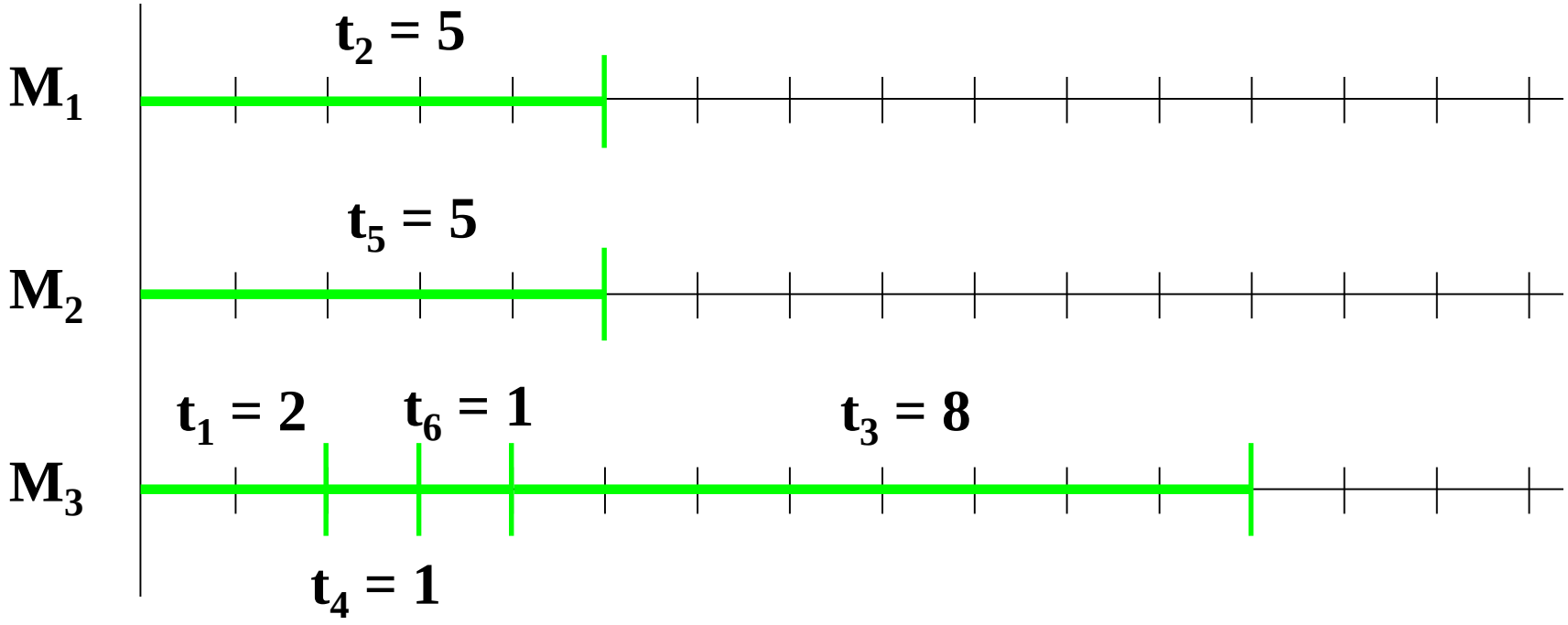
k=3 >= p Dừng.

3.Nguyên lý thứ tự

Bài toán phân công công việc : Cho M máy có cùng công suất như nhau và n công việc . Thực hiện công việc i trên bất kỳ máy nào cũng tốn thời gian là ti. Hãy phân công các công việc trên các máy sao cho tổng thời gian để hoàn thành tất cả công việc là thấp nhất**.**

Ví dụ:

Có 3 máy M1, M2, M3 và 6 công việc t1 = 2, t2 = 5, t3 = 8, t4 = 1, t5 = 5, t6 = 1.



Nhận xét độ phức tạp

Thứ tự của các công việc trên một máy là không quan trọng *(vì không làm thay đổi tổng thời gian thực hiện trên máy đó)*

Công việc	**1**	**2**	**3**	**4**	**...**	**n**
Máy	**1**	**1**	**3**	**2**	**...**	

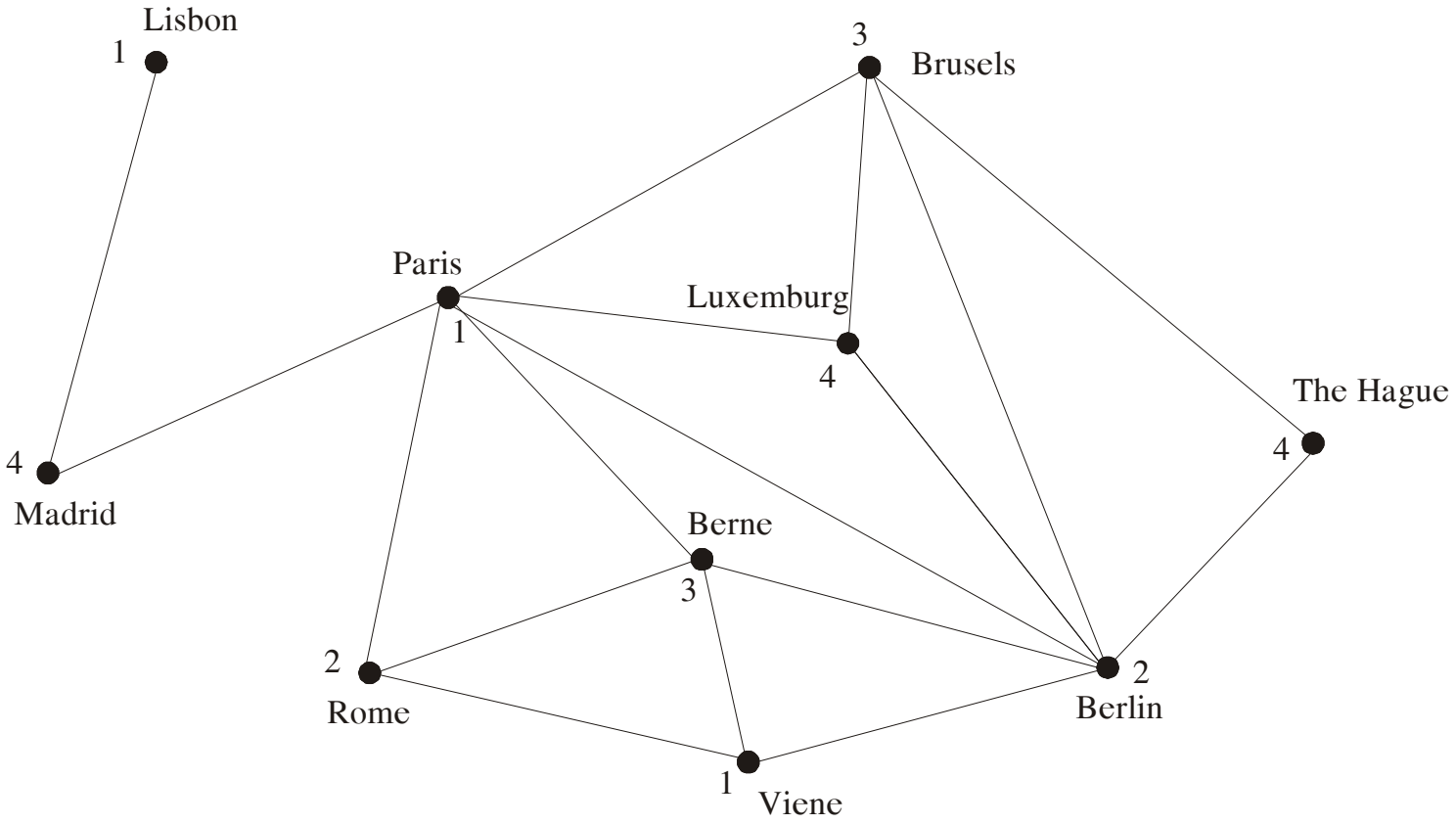
Độ phức tạp : $O(M^n)$

4.Thuật toán tô màu tối ưu trên đồ thị

Giả thiết 4 màu: Chúng ta nói 2 nước trên bản đồ vẽ trên mặt cầu hoặc là mặt phẳng là láng giềng của nhau nếu như chúng có chung đường biên giới (chỉ xét những nước có đường biên giới là một đường cong khép kín).

Yêu cầu: tô toàn bộ bản đồ mà chỉ sử dụng 4 màu sao cho không có bất kỳ 2 nước láng giềng nào có cùng chung một màu

Đỉnh	Lisbon L	Madrid M	Paris P	Berne Be	Rome R	Viene V	Berlin Ber	Luxemburg Lx	Brusen Bru	Hague H
Bậc	1	2	6	4	3	3	6	3	4	2



4.Thuật toán tô màu tối ưu trên đồ thị (tt)

Thuật toán: Lặp lại các bước sau cho đến khi nào tô màu hết các đỉnh

Bước 1: Chọn đỉnh có bậc lớn nhất tô màu i.

Bước 2: Hạ bậc:

- Đỉnh đã tô màu: bậc = 0
- Những đỉnh có liên hệ: bậc := bậc – 1

Bước 3: Đánh dấu các đỉnh liên hệ (bậc vừa trừ đi 1) cấm tô màu i.

4. Thuật toán tô màu tối ưu trên đồ thị (tt)

Màu tô

Màu tô lần 7 (Các nước có bậc là 0 mà chưa tô màu)		2				1		4		1
Màu tô lần 6				3		3				
Màu tô lần 5	1	1								
Màu tô lần 4								3	3	3
Màu tô lần 3				2	2	2				
Màu tô lần 2				2		2	2	2	2	2
Màu tô lần 1		1	1	1	1		1	1	1	
Đỉnh	L	M	P	Be	R	V	Ber	Lx	Bru	H
Bậc	1	2	6*	4	3	3	6	3	4	2
Hạ bậc lần 1	1	1	0	3	2	3	5*	2	3	2
Hạ bậc lần 2	1	1	0	2	2*	2	0	1	2	1
Hạ bậc lần 3	1	1	0	1	0	1	0	1	2*	1
Hạ bậc lần 4	1*	1	0	1	0	1	0	0	0	0
Hạ bậc lần 5	0	0	0	1*	0	1	0	0	0	0
Hạ bậc lần 6	0	0	0	0	0	0	0	0	0	0

Ví dụ 2: Phân công, lịch công tác, lịch thi đấu:

- Có một cuộc hội thảo khoa học với 9 chủ đề khác nhau, mỗi chủ đề diễn ra trong một buổi.
- Các chủ đề sau không được đồng thời: AE, BC, CD, ED, ABD, AHI, BHI, DFI, DHI, FGH.
- Xây dựng lịch sao cho số buổi diễn ra là ít nhất.
- Gợi ý: số màu = số buổi.

AE, BC, CD, ED, ABD, AHI, BHI, DFI, DHI, FGH.

	-1	-1	-1	1	-1	-1		-1	-1
dinh	A	B	C	D	E	F	G	H	I
bac	5	5	2	7	2	4	2	6	5
	4	4	1	0	1	3	2	5	4

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.

Thuật giải A^T ,A^{KT}

Thuật giải A^T (Algorithm for Tree):
Mỗi đỉnh n tương ứng với một số g(n): giá thành của đường đi từ đỉnh ban đầu đến đỉnh n.
Đỉnh:
+ Đỉnh đóng (Closed) : là những đỉnh đã được xem xét.
+Đỉnh mở (Open) : là những đỉnh giả thiết sẽ được xem xét ở bước sau.
+ Đỉnh ẩn (Hiden) : là những đỉnh mà tại đó hàm g(n) chưa được xác định.

Thuật giải A^T

Bước 1:

+ Mọi đỉnh n, mọi giá trị g(n) đều là ẩn.
+ Mở đỉnh đầu tiên và gọi đó là đỉnh S. Đặt g(S) = 0.

Bước 2 : Chọn đỉnh mở với giá thành g tương ứng là nhỏ nhất và gọi đó là đỉnh N.

+ Nếu N là mục tiêu: đường đi từ đỉnh ban đầu đến N là đường đi ngắn nhất và bằng g(N). Dừng (Success).
+ Nếu không tồn tại một đỉnh mở nào nữa: cây biểu diễn vấn đề không có đường đi tới mục tiêu. Dừng (Fail).
+ Nếu tồn tại nhiều hơn 1 đỉnh N (nghĩa là có 2 đỉnh N trở lên) mà có cùng giá thành g(N) nhỏ nhất. Kiểm tra xem trong số đó có đỉnh nào là đích hay không.

Nếu có: đường đi từ đỉnh ban đầu đến đỉnh N là ngắn nhất và bằng g(N), dừng (Success).

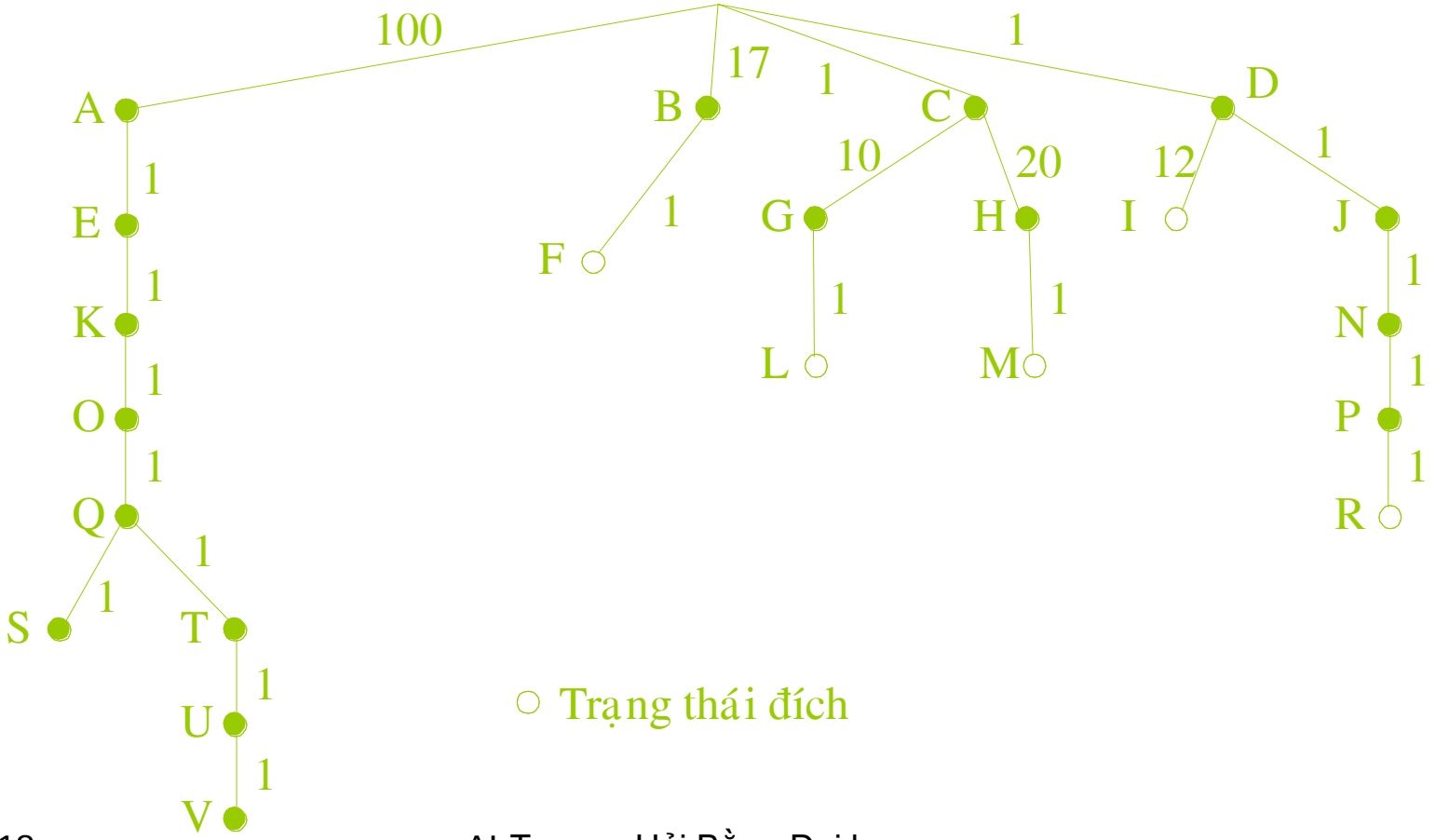
Nếu không có: Chọn ngẫu nhiên một trong các đỉnh đó và gọi là đỉnh N.

Bước 3: Đóng đỉnh N và mở các đỉnh sau N (là những đỉnh có cung hướng từ N tới). Tại mọi đỉnh S sau N tính :

$g(S) = g(N) + cost(N \rightarrow S)$

Bước 4: Quay lại bước 2

Thuật giải A^{T-} Ví dụ



Thuật giải A^{T-} Ví dụ

Mọi đỉnh n, g(n) chưa biết.

B1: Mở S, đặt g(S) = 0.

B2: Đóng S; mở A, B, C, D

g(A) = g(S) + gt(S→A) = 0 + 100 = 100

g(B) = 0 + 17 = 17

g(C) = g(D) = 0 + 1 = 1 (min)

Chọn ngẫu nhiên giữa C, D: chọn C

B3: Đóng C, mở G, H:

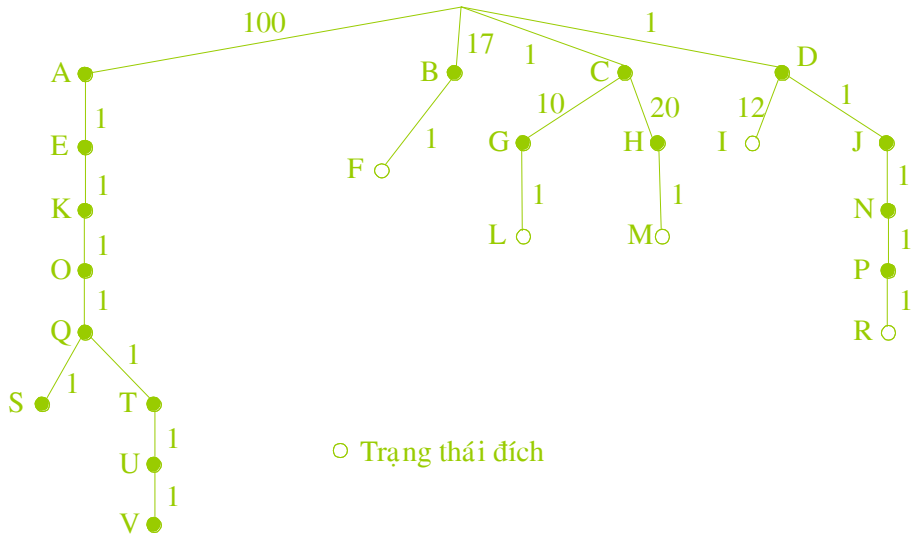
g(A) = 100

g(B) = 17

g(D) = 1 (min)

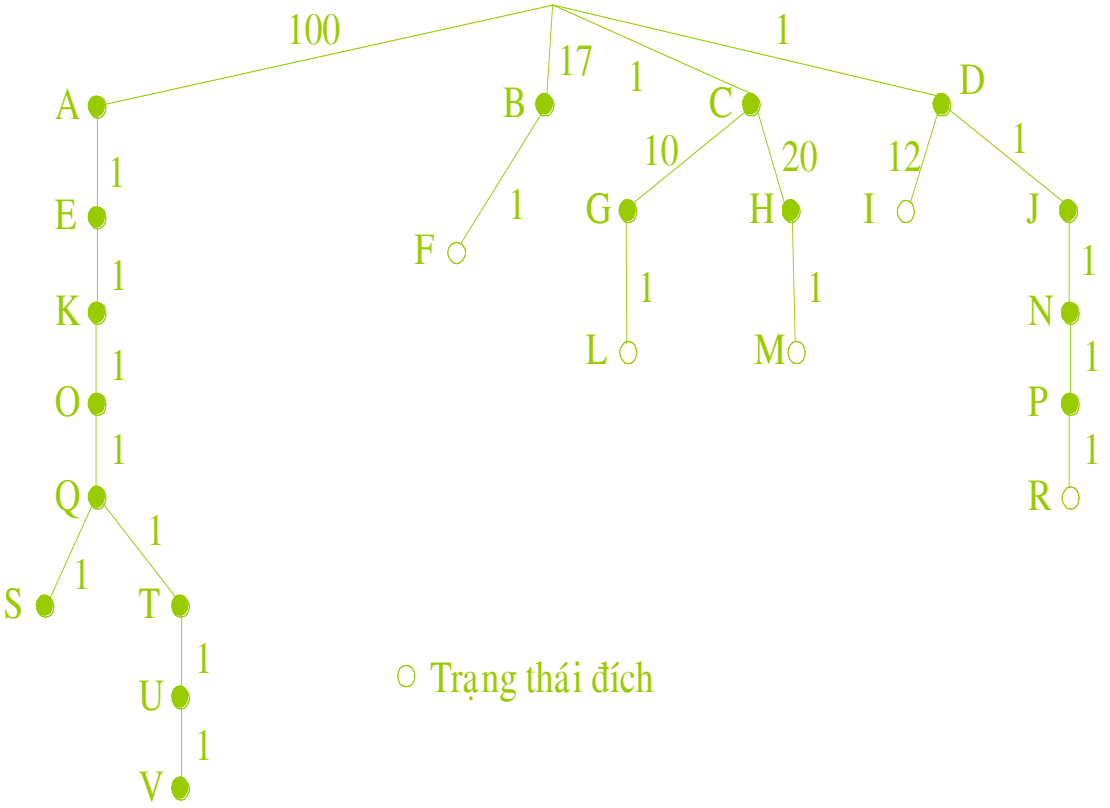
g(G) = 11

g(H) = 21



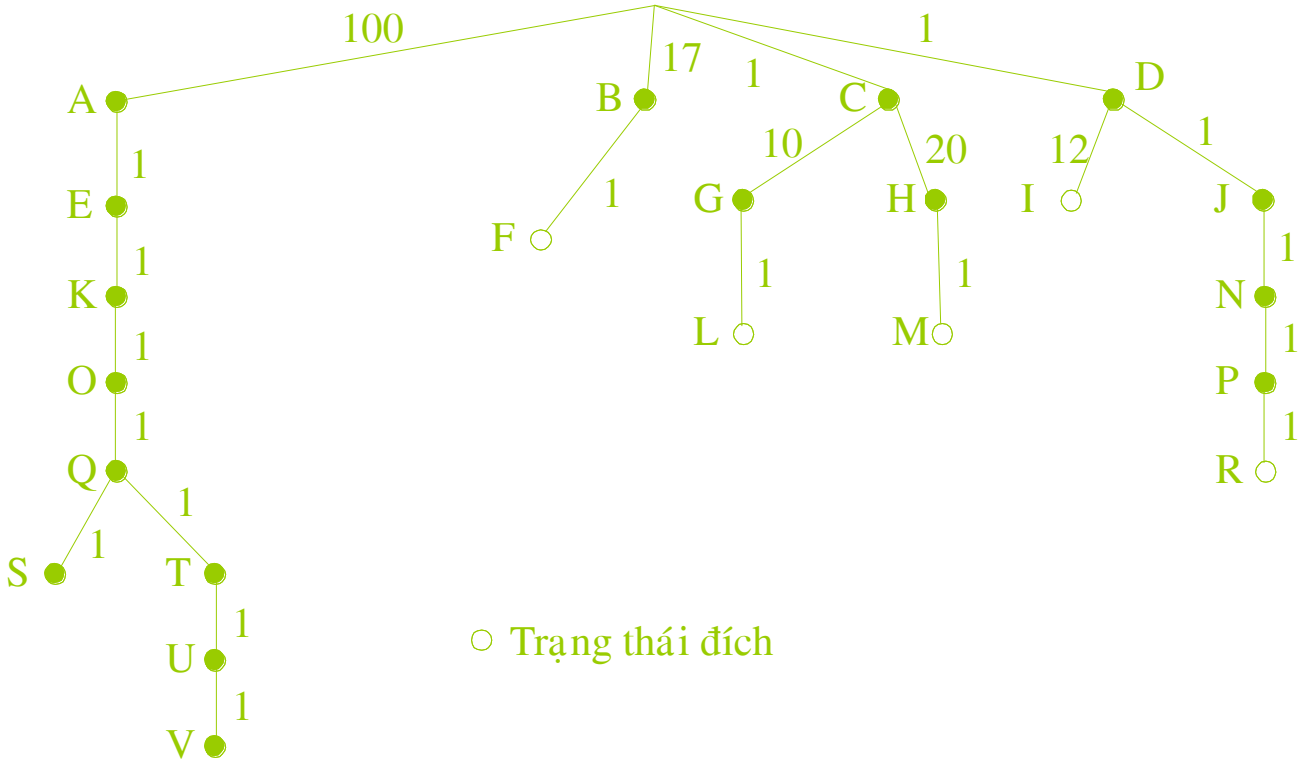
Thuật giải A^{T-} Ví dụ

B4: Đóng D, mở I, J:
g(A) = 100
g(B) = 17
g(I) = 12
g(J) = 2 (min)
g(G) = 11
g(H) = 21
B5: Đóng J, mở N:
g(A) = 100
g(B) = 17
g(I) = 12
g(G) = 11
g(H) = 21
g(N) = 3 (min)



Thuật giải A^{T-} Ví dụ

B6: Đóng N, mở P:
g(A) = 100
g(B) = 17
g(I) = 12
g(G) = 11
g(H) = 21
g(P) = 4 (min)
B7: Đóng P, mở R:
g(A) = 100
g(B) = 17
g(I) = 12
g(G) = 11
g(H) = 21
g(R) = 5 (min)
R là đích. Vậy đường đi là:



Nhận xét: Thuật toán này chỉ sử dụng 3 thông tin: đỉnh, cung và giá thành của cung.

$$S \xrightarrow{1} D \xrightarrow{1} J \xrightarrow{1} N \xrightarrow{1} P \xrightarrow{1} R$$

Thuật giải A^{KT} – Tìm kiếm với tri thức bổ sung (Algorthm for Knowledgeable Tree Search):

Thuật giải A^T là thuật giải tìm kiếm đường đi tốt nhất đối với cây chỉ có các thông tin về đỉnh, cung và giá trị của cung. Trong nhiều trường hợp việc tìm kiếm đường đi sẽ được định hướng rõ thêm nếu sử dụng các tri thức thu được dựa trên các hiểu biết về tình huống vấn đề ở mỗi bước.

Tri thức bổ sung ở mỗi đỉnh được tương ứng với một giá trị h(n). Chẳng hạn đó là ước lượng giá thành đường đi từ n đến mục tiêu. Ở ví dụ của giải thuật AT, ở bước đầu tiên :g(c) = g(d) = 1

Thuật giải A^{KT} – Tìm kiếm với tri thức bổ sung (Algorthm for Knowledgeable Tree Search):

A^T chọn tùy ý một trong hai đỉnh c và d để xét tiếp. Nhưng thay vì chọn tùy ý chúng ta có thể đặt câu hỏi "Đỉnh nào trong các đỉnh c và d gần mục tiêu hơn", chúng ta ước lượng được:

h(c) = 11

h(d) = 4

thì việc chọn đỉnh kế tiếp sẽ là d chứ không phải c.

Do vậy tri thức bổ sung sẽ dựa trên cơ sở cực tiểu hóa giá thành f ở mỗi bước :

f(n) = g(n) + h(n)

Thuật giải A^{KT}

Bước 1:

Mọi đỉnh, cũng như các hàm g, h, f chưa biết.

Mở đỉnh đầu tiên S, gán g(S) = 0

Sử dụng tri thức bổ sung để ước tính hàm h(S)

Tính f(S) = g(S) + h(S)

Bước 2: Chọn đỉnh mở có f là nhỏ nhất và gọi là đỉnh N

Nếu N là đích: đường đi từ đỉnh ban đầu đến đỉnh N là ngắn nhất và và bằng g(N). Dừng (Success).

Nếu không tồn tại đỉnh mở nào: cây biểu diễn vấn đề không tồn tại đường đi tới mục tiêu. Dừng (Fail).

Thuật giải A^{KT}

Nếu có 2 đỉnh mở trở lên có cùng giá trị f nhỏ nhất: Chúng ta phải kiểm tra xem những đỉnh đó có đỉnh nào là đích hay không.

+ Nếu có: đường đi từ đỉnh ban đầu đến đỉnh N là ngắn nhất và bằng g(N). Dừng (Success).

+ Nếu không có: chọn ngẫu nhiên một trong các đỉnh đó và gọi đỉnh đó là N.

Bước 3:

Đóng đỉnh N, mở mọi đỉnh sau N. Với mỗi đỉnh S sau N, tính:

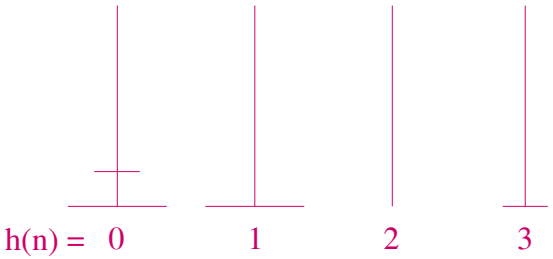
$g(S) = g(N) + cost(S \rightarrow N)$

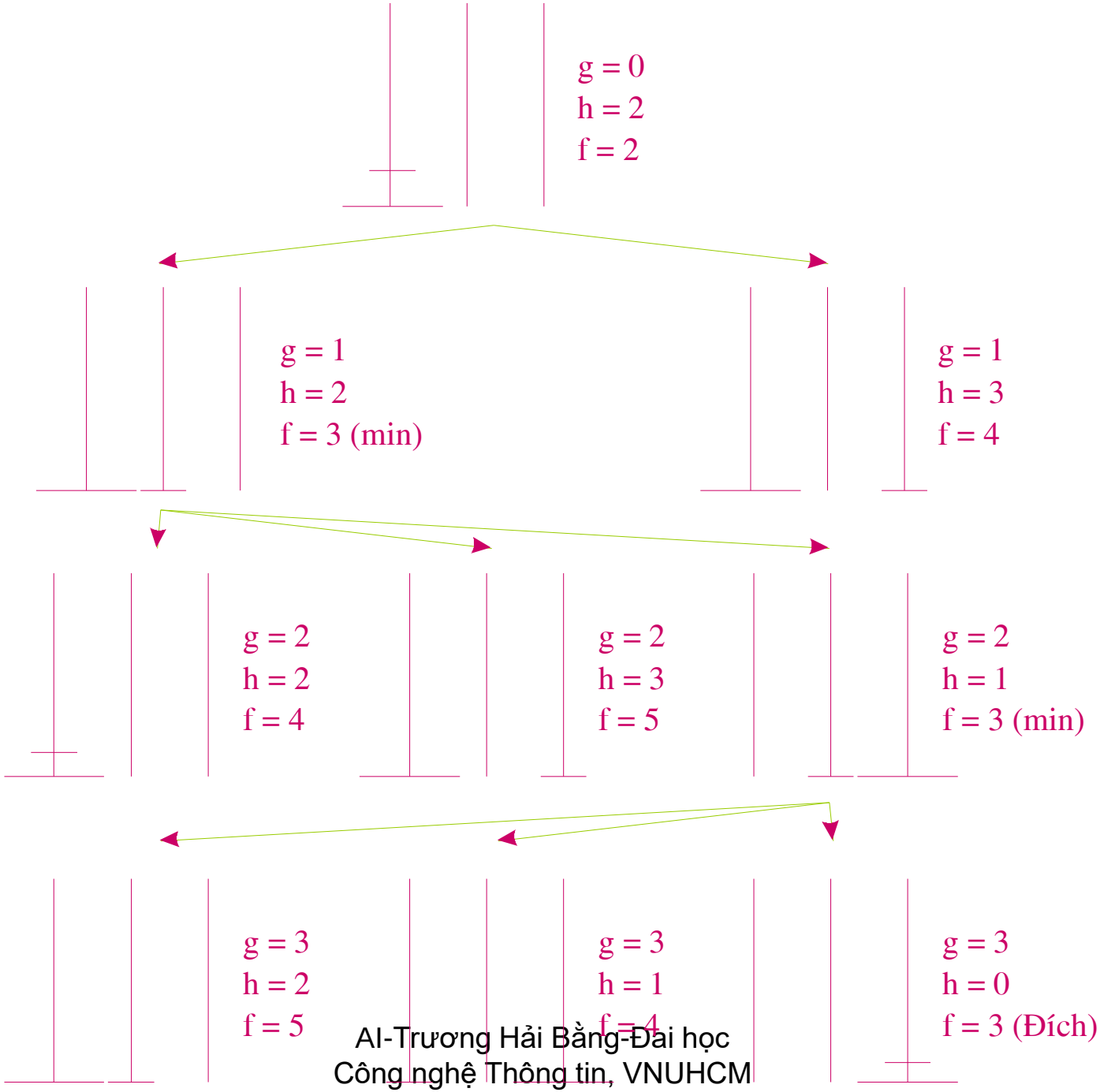
Sử dụng tri thức bổ sung để tính h(S) và f(S): $f(S) = g(S) + h(S)$

Bước 4: Quay lại bước 2.

Bài toán Tháp Hà Nội với n = 2

Các trường hợp của bài toán là với trạng thái cột thứ ba:

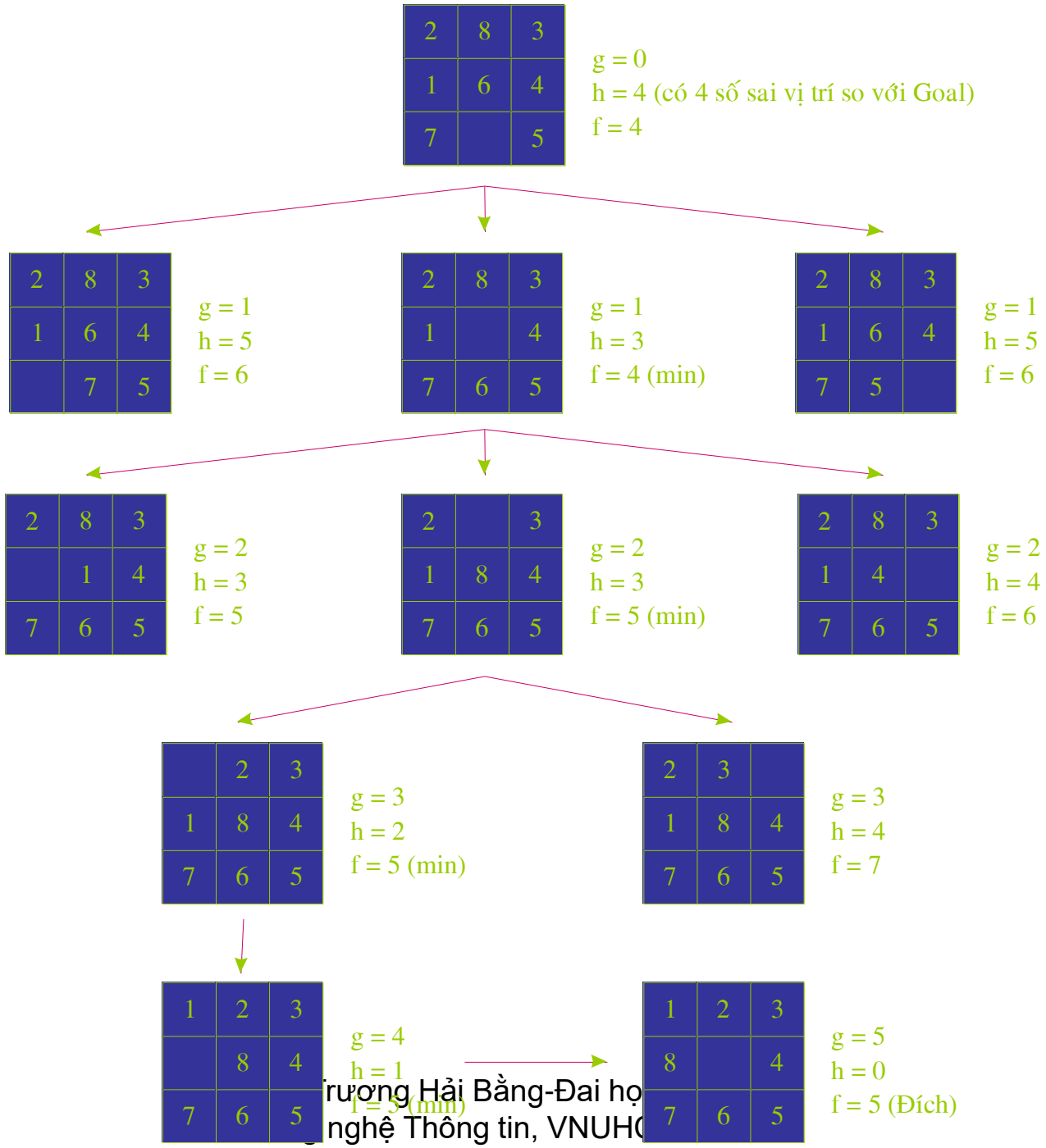



g = 0
h = 2
f = 2
g = 1
h = 2
f = 3 (min)
g = 1
h = 3
f = 4
g = 2
h = 2
f = 4
g = 2
h = 3
f = 5
g = 2
h = 1
f = 3 (min)
g = 3
h = 2
f = 5
g = 3
h = 1
f = 4
g = 3
h = 0
f = 3 (Đích)

Bài toán taci

☞**Cách 1**:

$$H = \sum_{i=1}^{t} \delta(a_i, b_i) \text{ với } \delta(a_i, b_i) = \begin{cases} 0 \text{ nếu } a_i = b_i \\ 1 \text{ nếu } a_i \neq b_i \end{cases}$$


2 8 3
1 6 4
7 5
g = 0
h = 4 (có 4 số sai vị trí so với Goal)
f = 4
2 8 3
1 6 4
7 5
g = 1
h = 5
f = 6
2 8 3
1 4
7 6 5
g = 1
h = 3
f = 4 (min)
2 8 3
1 6 4
7 5
g = 1
h = 5
f = 6
2 8 3
1 4
7 6 5
g = 2
h = 3
f = 5
2 3
1 8 4
7 6 5
g = 2
h = 3
f = 5 (min)
2 8 3
1 4
7 6 5
g = 2
h = 4
f = 6
2 3
1 8 4
7 6 5
g = 3
h = 2
f = 5 (min)
2 3
1 8 4
7 6 5
g = 3
h = 4
f = 7
1 2 3
8 4
7 6 5
g = 4
h = 1
f = 5 (min)
1 2 3
8 4
7 6 5
g = 5
h = 0
f = 5 (Đích)

Bài toán taci

☞**Cách 2:** $$H = \sum_{i=1}^{t} \eta(a_i, b_i)$$

với $\eta(a_i,b_i)$ là số lần ít nhất phải đẩy ô a_i = a theo chiều dọc hay ngang về đúng vị trí b_i = b.

Giả sử:

2	8	3
1	6	4
7		5

Số	1	2	3	4	5	6	7	8	Cộng
Vị trí	1	1	0	0	0	1	0	2	5