ÔN TẬP LÝ THUYẾT

CƠ SỞ TRÍ TUỆ NHÂN TẠO

# Mục lục

[Mục lục 2](#_Toc106449491)

[Các thuật toán tìm kiếm 3](#_Toc106449492)

[Thuật toán Breadth First Search 3](#_Toc106449493)

[Thuật toán Depth First Search 3](#_Toc106449494)

[Thuật toán Uniform Cost Search 4](#_Toc106449495)

[Thuật toán Lowest Cost First Search 5](#_Toc106449496)

[Thuật toán Iterative Deepening Search 5](#_Toc106449497)

[Heuristic search 6](#_Toc106449498)

[Giải thuật leo đồi 6](#_Toc106449499)

[Thuật toán Greedy Best-First Search 8](#_Toc106449500)

[Thuật toán A\* 8](#_Toc106449501)

[Bài toán phân việc - Nguyên lý thứ tự 9](#_Toc106449502)

[Tìm kiếm đối kháng 9](#_Toc106449503)

[Thuật toán tìm kiếm Minimax 9](#_Toc106449504)

[Cắt tỉa Alpha-Beta 11](#_Toc106449505)

[Bài toán bố trí 14](#_Toc106449506)

[Thuật toán tô màu đồ thị 15](#_Toc106449507)

[Thuật toán Welsh – Powell 16](#_Toc106449508)

[Logic 18](#_Toc106449509)

[Các luật logic 18](#_Toc106449510)

[Dạng chuẩn CNF 18](#_Toc106449511)

[Dùng hợp giải để chứng minh KB 18](#_Toc106449512)

[Thuật giải Davis-Putnam-Logemann-Loveland 19](#_Toc106449513)

[Thuật giải Vương Hạo 20](#_Toc106449514)

[Lượng từ và phép nối logic 21](#_Toc106449515)

[Học máy 22](#_Toc106449516)

[Thuật toán Quinlan 22](#_Toc106449517)

[Giải thuật Độ hỗn loạn trung bình 24](#_Toc106449518)

# Các thuật toán tìm kiếm

## Thuật toán Breadth First Search

Đối với mỗi bước, thêm lần lượt các nút kề vào cuối danh sách. Nút đầu tiên trong danh sách là nút duyệt ở bước tiếp theo. Dừng khi nút duyệt là đích.

A picture containing watch

Description automatically generated

Bảng duyệt đường đi:

|  |  |  |
| --- | --- | --- |
| **STT** | **Nút được mở rộng** | **Tập biên O (hàng đợi FIFO)** |
| 0 |  | S |
| 1 | S | **AS**, **ES**, **BS**, **CS** |
| 2 | AS | ES, BS, CS, **DA** |
| 3 | ES | BS, CS, DA, **GE** |
| 4 | BS | CS, DA, GE, **FB** |
| 5 | CS | DA, GE, FB, **HC** |
| 6 | DA | GE, FB, HC |
| 7 | GE | Đích |

Vậy đường đi là: S → E → G.

## Thuật toán Depth First Search

Đối với mỗi bước, thêm lần lượt các nút kề vào đầu danh sách. Nút đầu tiên trong danh sách là nút duyệt ở bước tiếp theo. Dừng khi nút duyệt là đích.

A picture containing watch

Description automatically generated

Bảng duyệt đường đi:

|  |  |  |
| --- | --- | --- |
| **STT** | **Nút được mở rộng** | **Tập biên O (hàng đợi FIFO)** |
| 0 |  | S |
| 1 | S | **AS**, **ES**, **BS**, **CS** |
| 2 | AS | **DA**, ES, BS, CS |
| 3 | DA | **ED**, ES, BS, CS |
| 4 | ED | **GE**, ES, BS, CS |
| 5 | GE | Đích |

Vậy đường đi là: S → A → D → E → G.

## Thuật toán Uniform Cost Search

Bước đầu tiên duyệt nút kề như duyệt BFS. Ở các bước tiếp theo, tìm nút sao cho tổng đường đi là ngắn nhất.

Diagram

Description automatically generated

Bảng duyệt đường đi:

|  |  |  |
| --- | --- | --- |
| **STT** | **Nút được mở rộng** | **Tập biên O** |
| 0 |  | S |
| 1 | S | AS(55), ES(72), **BS(42)**, CS(48) |
| 2 | BS | AS(55), ES(72), **CS(48)**, FB(82) |
| 3 | CS | **AS(55)**, ES(72), FB(82), HC(121) |
| 4 | AS | **ES(72)**, FB(82), HC(121), DA(100) |
| 5 | ES | **FB(82)**, HC(121), DA(100), GE(154) |
| 6 | FB | HC(121), **DA(100)**, GF(132) |
| 7 | DA | **HC(121)**, GF(132) |
| 8 | HC | **GF(132)** |
| 9 | GF | Đích |

Tại bước 6, do có xuất hiện GF(132) với đường đi nhỏ hơn GE(154) nên ta có thể lược bớt GE(154) ra khỏi danh sách. Chỉ để lại một G trong danh sách.

## Thuật toán Lowest Cost First Search

Tương tự như BFS, tuy nhiên cuối mỗi bước chọn ra nút có độ dài nhỏ nhất. Ở bước tiếp theo, xóa nút đã chọn, thêm các nút kề không trùng.

Ví dụ:

* Tập mở rộng đang có: A(10), B(5), C(7)
* B là nút có độ dài ngắn nhất.
* Đỉnh kề với B lần lượt là M(12) và P(15).
* Xóa B, mở rộng tại B các nút liền kề với B, khi đó ta được: A(10), M(12), P(15), C(7).

## Thuật toán Iterative Deepening Search

Tương tự như DFS, tuy nhiên DFS này có độ sâu duyệt của nhánh. Giả sử cho độ sâu nhánh d = 2 thì ta có cách duyệt như sau:

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# Heuristic search

## Giải thuật leo đồi

Cho bài toán 8-Puzzle như sau, trong đó:

* h(n) là số nút sai vị trí so với trạng thái đích.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | 2 | 8 | 3 | | 1 | 6 | 4 | | 7 |  | 5 |   **Nguồn**  h(n) = 4 | |  |  |  | | --- | --- | --- | | 1 | 2 | 3 | | 8 |  | 4 | | 7 | 6 | 5 |   **Đích**  h(n) = 0 |

Chi tiết giải thuật:

Bước 1:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | 2 | 8 | 3 | | 1 |  | 4 | | 7 | 6 | 5 |   **Lên**  h(n) = 3 | |  |  |  | | --- | --- | --- | |  |  |  | |  |  |  | |  |  |  |   **Xuống** | |  |  |  | | --- | --- | --- | | 2 | 8 | 3 | | 1 | 6 | 4 | | 7 | 5 |  |   **Phải**  h(n) = 5 | |  |  |  | | --- | --- | --- | | 2 | 8 | 3 | | 1 | 6 | 4 | |  | 7 | 5 |   **Trái**  h(n) = 5 |

Bước 2:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | 2 |  | 3 | | 1 | 8 | 4 | | 7 | 6 | 5 |   **Lên**  h(n) = 3 | |  |  |  | | --- | --- | --- | | 2 | 8 | 3 | | 1 | 6 | 4 | | 7 |  | 5 |   **Xuống (=Nguồn)**  h(n) = 4 | |  |  |  | | --- | --- | --- | | 2 | 8 | 3 | | 1 | 4 |  | | 7 | 6 | 5 |   **Phải**  h(n) = 4 | |  |  |  | | --- | --- | --- | | 2 | 8 | 3 | |  | 1 | 4 | | 7 | 6 | 5 |   **Trái**  h(n) = 4 |

Bước 3:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | |  |  |  | |  |  |  |   **Lên** | |  |  |  | | --- | --- | --- | | 2 | 8 | 3 | | 1 |  | 4 | | 7 | 6 | 5 |   **Xuống (=Lên1)**  h(n) = 3 | |  |  |  | | --- | --- | --- | | 2 | 3 |  | | 1 | 8 | 4 | | 7 | 6 | 5 |   **Phải**  h(n) = 4 | |  |  |  | | --- | --- | --- | |  | 2 | 3 | | 1 | 8 | 4 | | 7 | 6 | 5 |   **Trái**  h(n) = 2 |

Bước 4:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | |  |  |  | |  |  |  |   **Lên** | |  |  |  | | --- | --- | --- | | 1 | 2 | 3 | |  | 8 | 4 | | 7 | 6 | 5 |   **Xuống**  h(n) = 1 | |  |  |  | | --- | --- | --- | | 2 |  | 3 | | 1 | 8 | 4 | | 7 | 6 | 5 |   **Phải (=Lên2)**  h(n) = 4 | |  |  |  | | --- | --- | --- | |  |  |  | |  |  |  | |  |  |  |   **Trái** |

Bước 5:

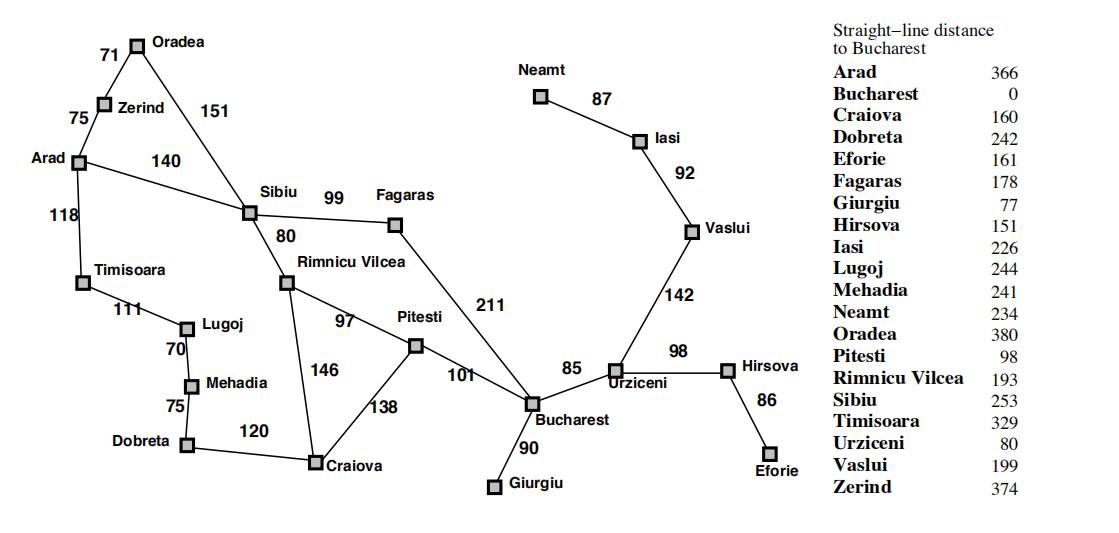
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  | 2 | 3 | | 1 | 8 | 4 | | 7 | 6 | 5 |   **Lên (=Trái3)**  h(n) = 2 | |  |  |  | | --- | --- | --- | | 1 | 2 | 3 | | 7 | 8 | 4 | |  | 6 | 5 |   **Xuống**  h(n) = 1 | |  |  |  | | --- | --- | --- | | 1 | 2 | 3 | | 8 |  | 4 | | 7 | 6 | 5 |   **Phải (=Đích)**  h(n) = 0 | |  |  |  | | --- | --- | --- | |  |  |  | |  |  |  | |  |  |  |   **Trái** |

Vậy phải di chuyển ô lần lượt theo các bước: Lên → Lên → Trái → Xuống → Phải.

Nếu sử dụng thuật toán A\* cho bài toán này, thì g(n) bằng bước đang xét trừ 1.

## Thuật toán Greedy Best-First Search

Tại mỗi bước, tìm ra các nút kề so với nút đang xét. Sau đó chọn nút kề nào có đường đi ngắn nhất.



Tìm đường đi từ Arad đến Bucharest.

Bảng duyệt đường đi:

|  |  |  |
| --- | --- | --- |
| **STT** | **Nút được mở rộng** | **Tập biên O** |
| 0 |  | Arad(366) |
| 1 | Arad | Zerind(374), **Sibiu(253)**, Timisoara(329) |
| 2 | Sibiu | Arad(366), Oradea(380), **Fagaras(178)**, Rimnicu Vilcea(193) |
| 3 | Fagaras | Sibiu(253), **Bucharest(0)** |
| 4 | Bucharest | Đích |

## Thuật toán A\*

Tìm độ dài đường đi theo công thức f(n) = g(n) + h(n). Trong đó, g(n) là tổng độ dài đường đi từ gốc đến đỉnh, còn h(n) là độ dài đường đi dự kiến.

Ta lại có: g(A) = g(B) + g(B → A), với g(B) là tổng độ dài từ gốc đến B, còn g(B → A) là độ dài đoạn đường từ B đến A.

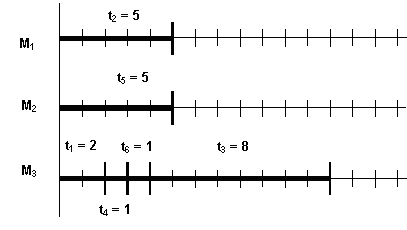
Theo ví dụ ở bài GBFS, h(n) chính là độ dài đường chim bay tính từ Bucharest.

Bảng duyệt đường đi:

|  |  |  |
| --- | --- | --- |
| **STT** | **Nút được mở rộng** | **Tập biên O** |
| 0 |  | Arad(366=0+0+366) |
| 1 | Arad(0) | Zerind(449=0+75+374), **Sibiu(393=0+140+253)**, Timisoara(447=0+118+329) |
| 2 | Sibiu(140) | Arad(680=140+140+366), Oradea(671=151+140+380), Fagaras(415=99+140+178), **Rimnicu Vilcea(413=80+140+193)** |
| 3 | Rimnicu Vilcea(220) | Sibiu(553=220+80+253), **Pitesti(415=220+97+98)**, Craiova(526=220+146+160) |
| 4 | Pitesti(317) | **Bucharest(418=317+101+0)** |

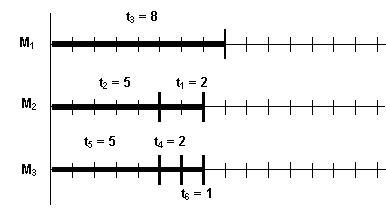
## Bài toán phân việc - Nguyên lý thứ tự

Chúng ta xét bài toán trong trường hợp có 3 máy P1, P2, P3 và 6 công việc với thời gian là t1 = 2, t2 = 5, t3 = 8, t4 = 1, t5 = 5, t6 = 1. Ta có một phương án phân công (L) như hình sau:



Ý tưởng giải bài toán như sau:

* Sắp xếp các công việc theo thứ tự giảm dần về thời gian gia công.
* Lần lượt sắp xếp các việc theo thứ tự đó vào máy còn dư nhiều thời gian nhất.



# Tìm kiếm đối kháng

## Thuật toán tìm kiếm Minimax

Giả sử ta có cây tìm kiếm đối kháng như sau:



Bước tiếp theo, ta chọn max từ các nhánh dưới của nút D, E, F, G. Khi đó:

* Nút D max(-1, 4) = 4
* Nút E max(2, 6) = 6
* Nút F max(-3, -5) = -3
* Nút G max(0, 7) = 7



Tương tự, tiếp theo ta chọn min từ các nhánh dưới của nút B, C. Khi đó:

* Nút B min(4, 6) = 4
* Nút C min(-3, 7) = -3



Bước cuối cùng, ta chỉ còn lại nút A, ta chọn max từ các nhánh dưới của A là nút B, C. Khi đó:

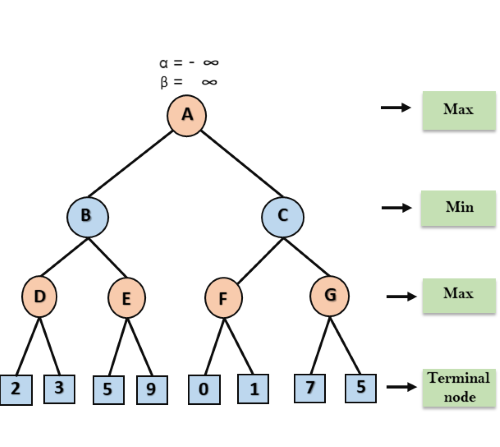
* Nút A max(4, -3) = 4



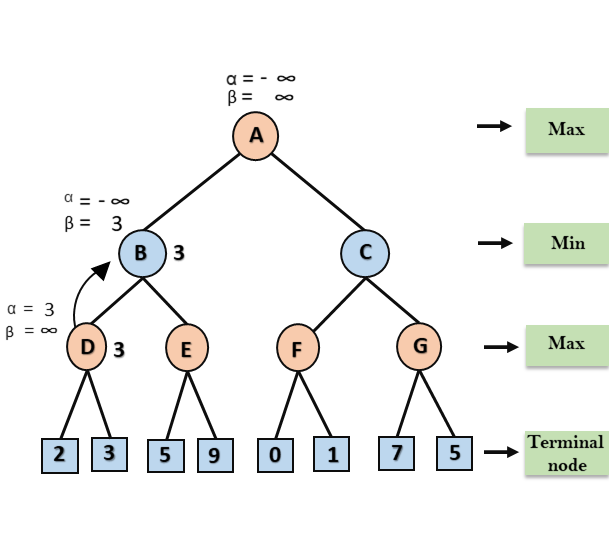
## Cắt tỉa Alpha-Beta

Đầu tiên, ta gán α, β của nút A lần lượt là α = -∞, β = +∞ với α, β lần lượt là giá trị min, max của nút đó.

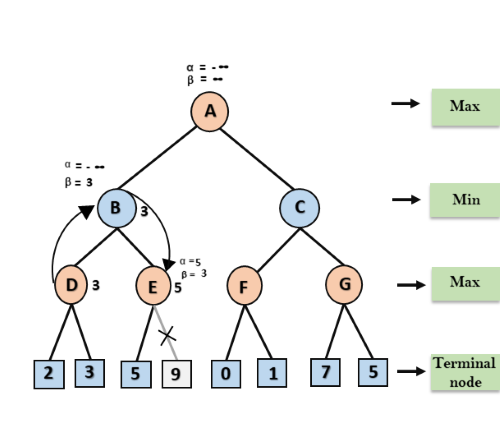
Mẹo: Cập nhật lên thì α chuyển qua β. Cập nhật xuống thì giữ nguyên.



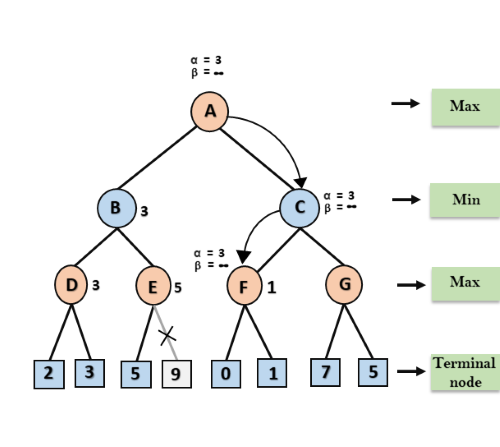
Bước tiếp theo, ta tìm được giá trị nút D là 3. Vì D là max nên chặn trên αD = 3. Max của D sẽ là min của B, do đó βB = 3. Nút E cũng sẽ được cập nhật lại từ nút B với αE = 3.



Ở nút E, ta duyệt lần lượt từng nhánh. Khi duyệt tại giá trị 5, thì nút E được cập nhật lại thành αE = 5. Duyệt nút dưới E tìm max, đầu tiên ta duyệt được βE = 5. Nút tiếp theo 9 > 5 nên ta bỏ.

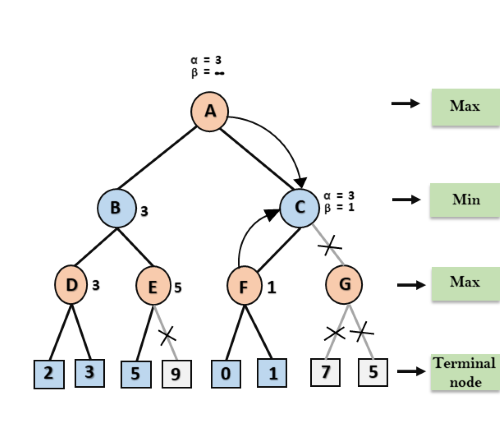


Ta tìm được giá trị B là 3, do đó cập nhật lên A là αA = 3. Duyệt xuống nhánh C, cập nhật lại αC = 3. Nhánh F cũng tương tự, αF = 3.

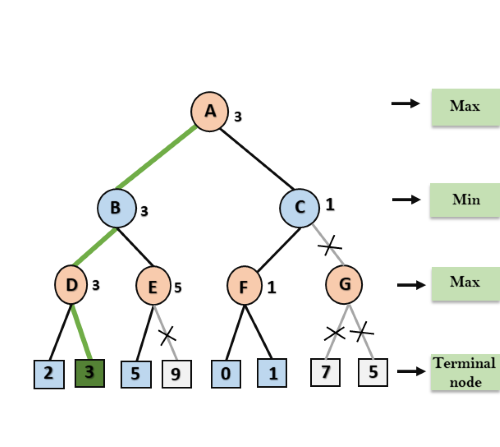


Tiếp theo duyệt các nhánh dưới F, ta thấy 0, 1 đều nhỏ hơn 3. Khi đó ta tìm được F là αF = 1. Cập nhật lên C là βC = 1.

Duyệt G tiếp theo, do các nút dưới G đều lớn hơn 1 nên ta bỏ luôn nhánh G.



Cuối cùng ta được đường đi như sau:



# Bài toán bố trí

Dạng bài toán thường gặp là sắp xếp công việc sao cho không trùng nhau, sắp xếp các quyển sách trên kệ…

## Thuật toán tô màu đồ thị

A picture containing watch

Description automatically generated

Đỉnh có bậc cao nhất là 4, khi đó màu tối đa có thể sử dụng là 4. Giả sử ta cho 4 màu R(đỏ), G(xanh lá), B(xanh lam), C(xanh cyan).

Bước đầu ta tô đỉnh 2 màu đỏ:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| GBC | **R** | GBC | GBC | RGBC | RGBC | RGBC |
| 2 |  | **G** | **R** | GBC | GBC | RGBC | RGBC | RGBC |
| 3 |  | **G** | **R** | **G** | GBC | RGBC | RGBC | RGBC |
| 4 |  | **G** | **R** | **G** | **G** | BC | RBC | RGBC |
| 5 |  | **G** | **R** | **G** | **G** | **B** | RBC | RGBC |
| 6 |  | **G** | **R** | **G** | **G** | **B** | **R** | GBC |
| 7 |  | **G** | **R** | **G** | **G** | **B** | **R** | **G** |

## Thuật toán Welsh – Powell

Cho đồ thị sau:

Diagram, schematic

Description automatically generated

Đầu tiên ta sắp xếp đỉnh bậc theo thứ tự giảm dần

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Đỉnh** | A | B | C | D | E | F | G | H | I | J | K |
| **Bậc** | 2 | 5 | 3 | 2 | 2 | 5 | 3 | 3 | 4 | 1 | 2 |
|  | | | | | | | | | | | |
| **Đỉnh** | B | F | I | C | G | H | A | D | E | K | J |
| **Bậc** | 5 | 5 | 4 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 1 |

Đỉnh bậc cao nhất là 5 nên ta có 4 màu. Ta tô B màu đỏ trước tiên. Sau đó tô các đỉnh không kề B theo thứ tự đã sắp xếp. Lúc này ta có I, D, E, K, J, mà E, K, J lại kề với I, D nên ta chỉ có I, D là tô.

Diagram, schematic

Description automatically generated

Tiếp theo ta loại bỏ B, D, I ra khỏi danh sách.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Đỉnh** | F | C | G | H | A | E | K | J |
| **Bậc** | 5 | 3 | 3 | 3 | 2 | 2 | 2 | 1 |

Làm tương tự. Tô F, sau đó tô đỉnh không kề bao gồm H, A, K, J, ta bỏ K vì K trùng màu với A.

Diagram, schematic

Description automatically generated

Tiếp tục ta bỏ F, H, A, J ra khỏi danh sách.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Đỉnh** | C | G | E | K |
| **Bậc** | 3 | 3 | 2 | 2 |

Cuối cùng ta được:

Diagram

Description automatically generated

# Logic

## Các luật logic

* Luật Modus Ponens (MP):
* Luật Modus Tollens (MT):
* Luật Hội:
* Luật đơn giản:
* Luật cộng:
* Luật tam đoạn luận tuyển:
* Luật tam đoạn luận giả thiết:
* Luật hợp giải:

## Dạng chuẩn CNF

Giả sử ta có:

1. Loại bỏ mũi tên:
2. Đưa phủ định vào:
3. Phân phối:

Phép ta có thể tách ra thành từng dòng riêng biệt.

## Dùng hợp giải để chứng minh KB

Cho tập: . Chứng minh R.

Trước khi thực hiện giải bài toán ta cần chuyển về dạng chuẩn CNF.

|  |  |
| --- | --- |
| **Biểu thức gốc** | **Chuẩn hóa** |
|  |  |
|  |  |
|  |  |
|  |
|  |  |
|  |  |

Tiến hành hợp giải:

|  |  |  |
| --- | --- | --- |
| **STT** | **Biểu thức** | **Ghi chú** |
| 1 |  | Tiền đề (p) |
| 2 |  | P |
| 3 |  | P |
| 4 |  | P |
| 5 |  | P |
| 6 |  | P |
| 7 |  | 1, 2 |
| 8 |  | 4, 5 |
| 9 |  | 7, 8 |
| 10 | 0 | 6, 9 |

Vậy R có thể được suy dẫn từ tập tri thức KB.

## Thuật giải Davis-Putnam-Logemann-Loveland

Áp dụng thuật toán quay lui khi gặp conflict. Duyệt các nút theo thứ tự bảng chữ cái. Duyệt cho đến khi nào tồn tại một mô hình mà tập hợp các mệnh đề đều trả ra TRUE.

Sử dụng thủ tục Davis-Putnam để tính toán các mô hình cho các tập hợp mệnh đề sau hoặc để chứng minh rằng không tồn tại mô hình nào. Ở mỗi bước, hãy cho biết bạn đã áp dụng quy tắc nào?

Shape

Description automatically generated

Vậy tập hợp mệnh đề đã cho có tồn tại mô hình SAT.

b)

Diagram

Description automatically generated

Vậy tập hợp mệnh đề đã cho có không tồn tại mô hình SAT.

## Thuật giải Vương Hạo

Cho tập: . Chứng minh .

Chuyển về dạng chuẩn:

Vẽ cây, tách tại dấu theo hai phía:

Chuyển vế

(đã chứng minh)

(đã chứng minh)

(đã chứng minh)

(không thể chứng minh)

(đã chứng minh)

Kết luận: Do có một nhánh lá không thể chứng minh được nữa nên ta kết luận bài toán không được chứng minh.

## Lượng từ và phép nối logic

Ví dụ 1: Đặt C(x): “x có một con mèo”,

D(x): “x có một con chó”,

F(x): “x có một con chồn”.

Biểu diễn các phát biểu sau theo C(x), D(x), F(x), các lượng từ và các phép nối logic. Xét không gian biến là các sinh viên trong lớp.

1. Một sinh viên trong lớp có một con mèo, một con chó hay một con chồn.

→

1. Tất cả sinh viên trong lớp đều có một con mèo, một con chó hay một con chồn.

→

1. Một sinh viên nào đó có một con mèo và một con chồn nhưng không có con chó.

→

1. Không có sinh viên nào trong lớp có một con mèo, một con chó và một con chồn.

→

1. Với mỗi loại con vật trên, có một sinh viên trong lớp có một con.

→

Ví dụ 2: Đặt L(x): “x là một nhà logic”,  
 C(x): “x uống cà phê”,  
 W(x): “x làm việc chăm chỉ”,  
 T(x): “x phát biểu định lý”,  
 f(x): hàm trả ra giá trị là bạn của x.

1. Không nhà logic nào uống cà phê.
2. Bất kỳ ai là một nhà logic cũng đều là bạn của ai đó.
3. Không người nào phát biểu được định lý lại có một người bạn biết uống cà phê.
4. Ai có một người bạn làm việc chăm chỉ thì hoặc là một nhà logic hoặc là một người làm việc chăm chỉ.
5. Mọi người bạn là một nhà logic.
6. Bất kỳ ai không phát biểu được định lý đều không uống cà phê.
7. Có một số người mà bạn của họ là nhà logic.

# Học máy

## Thuật toán Quinlan

Cho bảng mô tả sau:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **STT** | **Tên** | **Màu tóc** | **Chiều cao** | **Cân nặng** | **Dùng kem** | **Kết quả** |
| 1 | Sarah | Vàng | Trung bình | Nhẹ | Không | Cháy nắng |
| 2 | Dana | Vàng | Cao | Trung bình | Có | Không cháy nắng |
| 3 | Alex | Nâu | Thấp | Trung bình | Có | Không cháy nắng |
| 4 | Annie | Vàng | Thấp | Trung bình | Không | Cháy nắng |
| 5 | Emily | Đỏ | Trung bình | Nặng | Không | Cháy nắng |
| 6 | Peter | Nâu | Cao | Nặng | Không | Không cháy nắng |
| 7 | John | Nâu | Trung bình | Nặng | Không | Không cháy nắng |
| 8 | Katie | Vàng | Thấp | Nhẹ | Có | Không cháy nắng |

Hãy sử dụng thuật toán Quinlan để xác định xem một người có bị cháy nắng hay không?

Xét thuộc tính Màu tóc:

* Vector đặc trưng thuộc tính Màu tóc:

Do đó:



Tổng số vector đơn vị của thuộc tính Màu tóc là 2.

Xét thuộc tính Chiều cao:



Tổng số vector đơn vị của thuộc tính Chiều cao là 1.

Xét thuộc tính Cân nặng:



Tổng số vector đơn vị của thuộc tính Cân nặng là 0.

Xét thuộc tính Dùng kem:



Tổng số vector đơn vị của thuộc tính Dùng kem là 1.

Như vậy thuộc tính Màu tóc có số vector đơn vị nhiều nhất nên sẽ được chọn để phân hoạch. Sau khi phân hoạch ta thu được bảng rút gọn sau:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Tên** | **Chiều cao** | **Cân nặng** | **Dùng kem** | **Kết quả** |
| 1 | Sarah | Trung bình | Nhẹ | Không | Cháy nắng |
| 2 | Dana | Cao | Trung bình | Có | Không cháy nắng |
| 4 | Annie | Thấp | Trung bình | Không | Cháy nắng |
| 8 | Katie | Thấp | Nhẹ | Có | Không cháy nắng |

Xét thuộc tính Chiều cao:



Tổng số vector đơn vị của thuộc tính Chiều cao là 2.

Xét thuộc tính Cân nặng:



Tổng số vector đơn vị của thuộc tính Cân nặng là 0.

Xét thuộc tính Dùng kem:



Tổng số vector đơn vị của thuộc tính Dùng kem là 2.

Ta lại thấy lần này có đến hai thuộc tính có cùng max vector đơn vị. Tuy nhiên ta sẽ chọn thuộc tính Dùng kem vì thuộc tính này không còn phân hoạch những người cháy nắng và không cháy nắng.

Diagram

Description automatically generated

Vậy ta rút ra tập lập luận là:

If (MauToc == "Nâu")

KQ = "Không cháy nắng";

If (MauToc == "Đỏ")

KQ = "Cháy nắng";

If (MauToc == "Vàng" AND DungKem == "Có")

KQ = "Không cháy nắng";

If (MauToc == "Vàng" AND DungKem == "Không")

KQ = "Cháy nắng";

## Giải thuật Độ hỗn loạn trung bình

Dựa vào bảng mô tả sau, hãy rút ra tập luật số 9 mang kết quả nhóm gì.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Vóc dáng** | **Quốc tịch** | **Gia cảnh** | **Nhóm** |
| 1 | Nhỏ | Đức | Độc thân | A |
| 2 | Lớn | Pháp | Độc thân | A |
| 3 | Lớn | Đức | Độc thân | A |
| 4 | Nhỏ | Ý | Độc thân | B |
| 5 | Lớn | Đức | Có gia đình | B |
| 6 | Lớn | Ý | Độc thân | B |
| 7 | Lớn | Ý | Có gia đình | B |
| 8 | Nhỏ | Đức | Có gia đình | B |
| 9 | Nhỏ | Pháp | Có gia đình | ? |

*Test lần 1:*

Độ hỗn loạn trung bình của Vóc dáng:

* Lớn: 2A, 3B
* Nhỏ: 1A, 2B

Độ hỗn loạn trung bình của Quốc tịch:

* Đức: 2A, 2B
* Ý: 0A, 3B
* Pháp: 1A, 0B

Độ hỗn loạn trung bình của Gia cảnh:

* Độc thân: 3A, 2B
* Có gia đình: 0A, 3B

So sánh ta thấy thuộc tính Quốc tịch có độ hỗn loạn trung bình thấp nhất nên ta chọn thuộc tính này làm nút gốc.

Diagram

Description automatically generated

*Test lần 2:*

Sau khi test lần 1, ta có bảng dữ liệu nhỏ hơn như sau:

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Vóc dáng** | **Gia cảnh** | **Gia cảnh** |
| 1 | Nhỏ | Độc thân | A |
| 3 | Lớn | Độc thân | A |
| 5 | Lớn | Có gia đình | B |
| 8 | Nhỏ | Có gia đình | B |

Độ hỗn loạn trung bình của Vóc dáng:

* Lớn: 1A, 1B
* Nhỏ: 1A, 1B

Độ hỗn loạn trung bình của Gia cảnh:

* Độc thân: 2A, 0B
* Có gia đình: 0A, 2B

Dễ dàng nhận thấy rằng độ hỗn loạn của Gia cảnh là thấp nhất nên ta chọn thuộc tính này làm nút nhánh.

Diagram

Description automatically generated

Cuối cùng ta được cây quyết định:

Diagram

Description automatically generated

Vậy ta có tập luật:

If (QuocTich == "Pháp")

Nhom = A;

If (QuocTich == "Ý")

Nhom = B;

If (QuocTich == "Đức" AND GiaCanh == "Độc thân")

Nhom = A;

If (QuocTich == "Đức" AND GiaCanh == "Có gia đình")

Nhom = B;

Từ tập luật trên ta có thể suy ra: Người có vóc dáng nhỏ, mang quốc tịch Pháp, có gia đình thì thuộc nhóm A.