

# AI

## HẠN CHẾ SỰ HÀI LÒNG CÁC VẤN ĐỀ

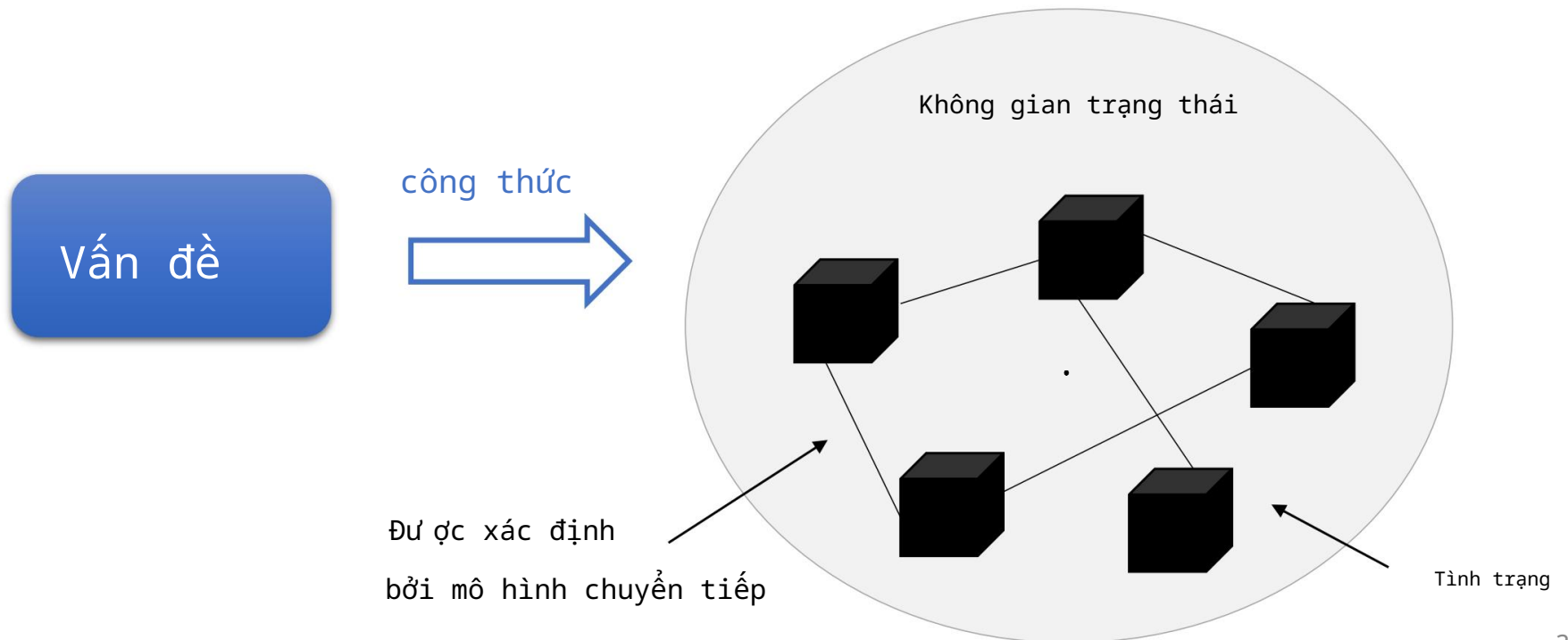
Nguyễn Ngọc Thảo - Nguyễn Hải Minh  
{nnthao, nhminh}@fit.hcmus.edu.vn

# Đề cương

- Các vấn đề về sự thỏa mãn ràng buộc (CSP)
- Tuyên truyền ràng buộc: Suy luận trong CSP
- Tìm kiếm ngược cho CSP
- Tìm kiếm CSP cục bộ (Tự học)
- Cấu trúc đề bài (Tự học)

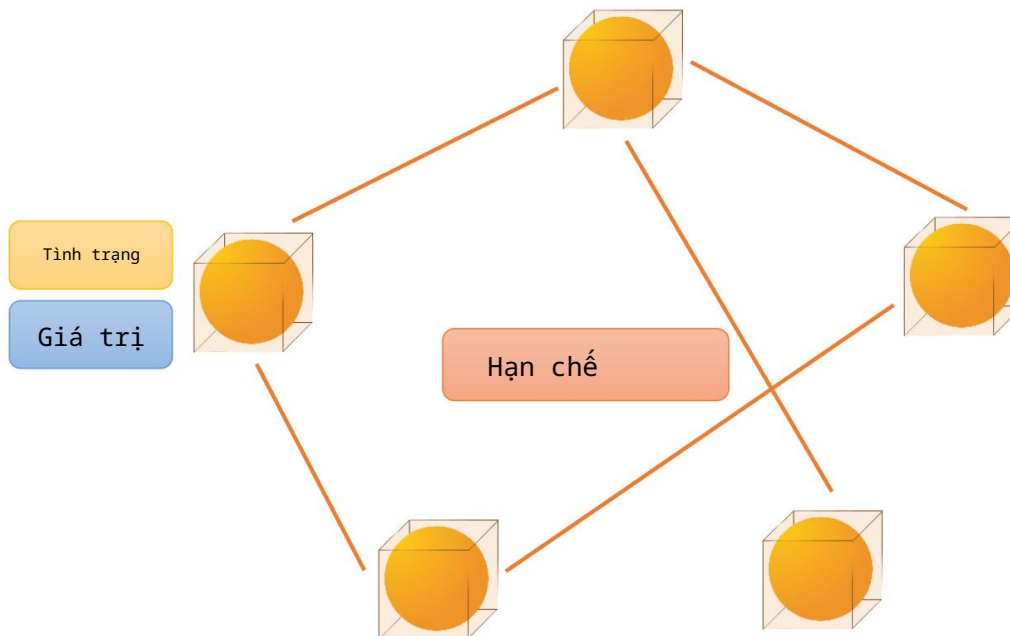
# Các bài toán tìm kiếm trong không gian trạng thái

- Mỗi trạng thái là nguyên tử hoặc không thể phân chia theo quan điểm của thuật toán tìm kiếm.
- Cần có mã theo miền cụ thể mô tả sự chuyển đổi giữa các trạng thái cho từng vấn đề.



# Đại diện nhân tố

- CSP phân tích từng trạng thái thành một tập hợp các biến, mỗi biến có một giá trị.
- Một vấn đề đư ợc giải quyết khi mọi biến đều có một giá trị thỏa mãn mọi ràng buộc của biến đó.





# Hạn chế vấn đề hài lòng

---

# Xây dựng vấn đề

- Một công thức **CSP** có ba thành phần chính.

$= \{ x_1, \dots, x_n \}$ : một tập hợp các biến

$= \{ D_1, \dots, D_n \}$ : một tập hợp các miền, một miền cho mỗi biến.

- $= \{ \text{values} \}$ : tập hợp các giá trị cho phép của biến

$R$ : một tập hợp các ràng buộc,  $= \{ R_1, \dots, R_m \}$ , trạng thái đó cho phép kết hợp các giá trị.

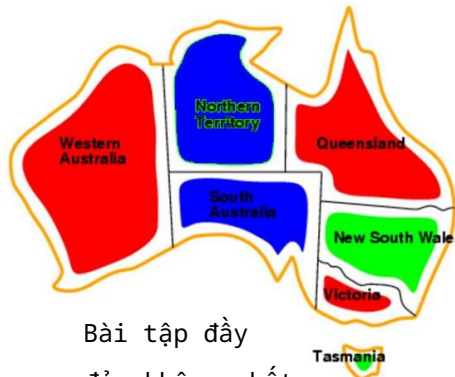
- $D$ : một bộ các biến tham gia vào ràng buộc
  - Một mối quan hệ  $R_D$  xác định các giá trị mà các biến tham gia có thể lấy
- CSP nói chung là một bài toán NP-đầy đủ.

# Bài tập trong CSP

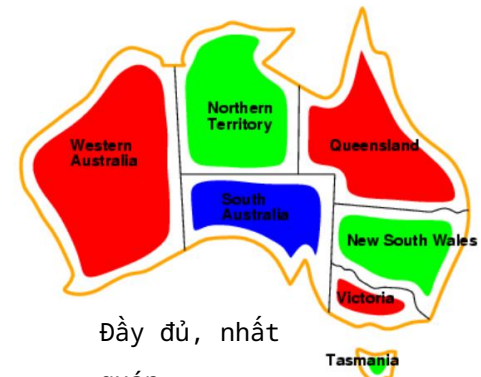
- CSP gán giá trị cho các biến,  $= \{ \quad , \quad = \quad , \quad . \}$ .
- **Giải pháp** là một **nhiệm vụ nhất quán và đầy đủ**.
  - Việc **phân công nhất quán** không vi phạm bất kỳ ràng buộc nào.
  - Một **phép gán hoàn chỉnh** có mỗi biến đư ợc gán một giá trị.
- **Giải pháp từng phần** là sự phân công từng phần một cách nhất quán.
  - **Phép gán một phần** là phép gán mà không gán một số biến.



Nhiệm vụ không  
đầy đủ, nhất  
quán



Bài tập đầy  
đủ, không nhất  
quán

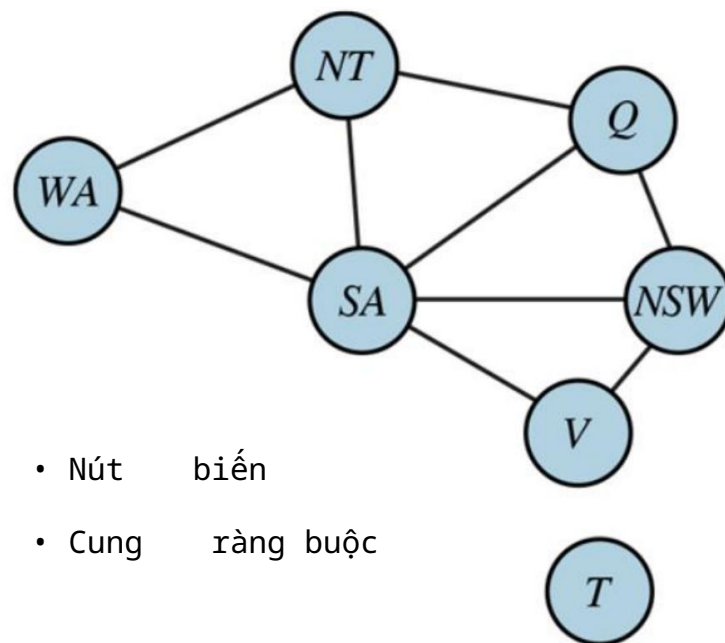


Đầy đủ, nhất  
quán  
phân công



# Bài toán ví dụ: Tô màu bản đồ

- Mục tiêu: Tô màu từng vùng là **đỏ**, **lục** hoặc **lam** sao cho không vùng lân cận nào có cùng màu



- Nút biến
- Cung ràng buộc

(a) Các bang và vùng lãnh thổ chính của Úc. (b)

Bài toán tô màu bản đồ được biểu diễn dưới dạng biểu đồ ràng buộc.



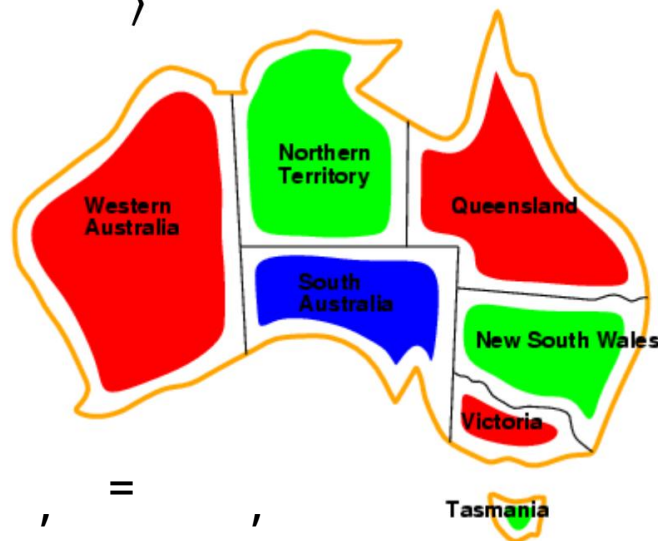
# Bài toán ví dụ: Tô màu bản đồ

- **Biến:**  $= \{ \quad , \quad , \quad , \quad , \quad , \quad \}$
- **Tên miền:**  $= \{ \quad , \quad , \quad \}$
- **Hạn chế:** Các vùng lân cận có màu sắc riêng biệt.

$$= \left\{ \quad , \quad , \quad , \quad , \quad , \quad \right\}$$

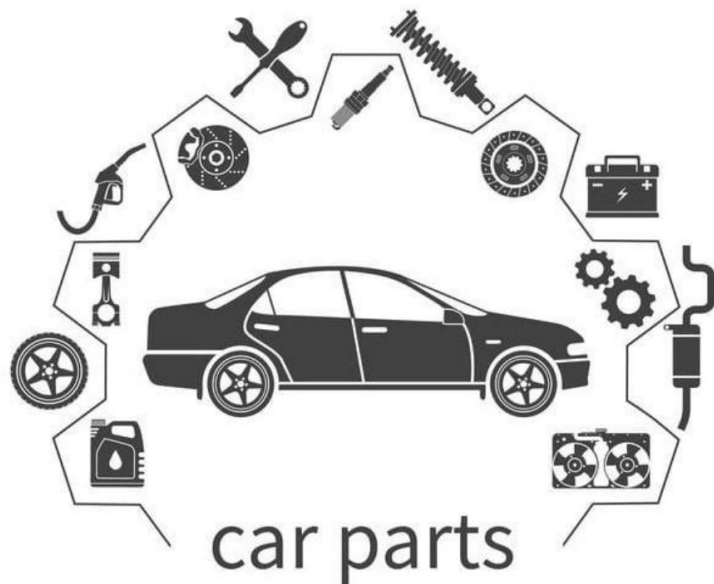
- là phép tắt của  $\langle ( \quad , \quad ), \quad \rangle$

- Có **nhiều** giải pháp khả thi.



$$\left\{ \begin{array}{l} = \\ = \end{array} \quad , \quad = \quad , \quad = \quad , \quad = \quad , \quad = \quad \right\}$$

# Vấn đề ví dụ: Lập kế hoạch cửa hàng việc làm



15 nhiệm vụ

- Lắp trục (trước và sau)
- Dán cả bốn bánh xe (phải và trái, trước và sau)
- Siết chặt đai ốc cho từng bánh xe
- Dán các nắp trục, và
- Kiểm tra việc lắp ráp cuối cùng

- Một số nhiệm vụ phải xảy ra trước nhiệm vụ khác và một số nhiệm vụ có thể tiếp tục ngay lập tức
  - Ví dụ: bánh xe phải được lắp trước khi lắp nắp trục
- Một nhiệm vụ cần một khoảng thời gian nhất định để hoàn thành.

- 11

# Vấn đề ví dụ: Lập kế hoạch cửa hàng việc làm

- Các trục phải được lắp vào đúng vị trí trước khi lắp bánh xe vào. Lắp đặt một trục mất 10 phút.
 

$+ 10 h$   
 $+ 10 h$

$+ 10 h$   
 $+ 10 h$
- Đối với mỗi bánh xe, cố định bánh xe (mất 1 phút), sau đó siết chặt đai ốc (2 phút) và cuối cùng gắn nắp trục (1 phút)
 

$h + 1 \leq$   
 $h + 1 h$

$h + 1 \leq$   
 $h + 1 \leq$

$+ 2 \leq$   
 $+ 2 \leq$

$+ 2 \leq$   
 $+ 2 \leq$
- Giả sử chúng ta có bốn công nhân để lắp bánh xe, nhưng họ phải dùng chung một công cụ đúng vị trí.
 

$+ 10 \leq$ 
 $+ 10 \leq$
- Việc kiểm tra diễn ra sau cùng và mất 3 phút cho mọi biến ngoại trừ việc thêm ràng buộc dạng
 

$+ \leq$
- Cuối cùng, giả sử có yêu cầu phải hoàn thành toàn bộ việc lắp ráp trong 30 phút
 

$giới hạn miền xác định của tất cả các biến là = \{1, 2, 3, \dots, 27\}.$

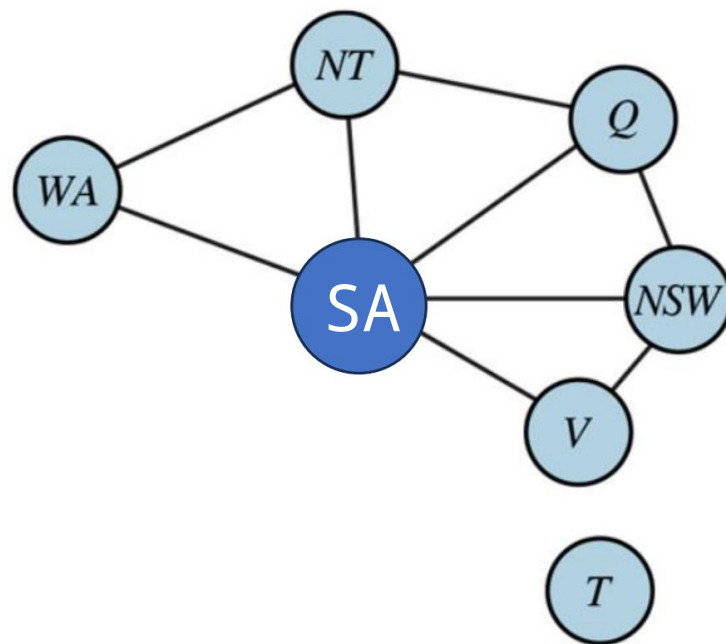
# Tại sao lại xây dựng một vấn đề như một CSP?

- CSP cung cấp **những hiểu biết sâu sắc hơn về vấn đề** và giải pháp.
- Bộ giải CSP có thể **nhạy** **chống thu gọn các vùng lớn của không gian** tìm kiếm mà bộ tìm kiếm trong không gian trạng thái nguyên tử không thể làm được.

Giả sử chúng ta đã chọn  $\{SA = \text{blue}\}$ .

Tìm kiếm: 35 = 243 bài tập = 32

bài tập      87% CSP: 2 5



- Nó mang **tính tổng quát** hơn các phương pháp phỏng đoán cụ thể cho từng vấn đề.

# Các loại biến trong CSP

- CSP đơn giản nhất bao gồm các biến có miền **hữu hạn và rời rạc**.
  - Ví dụ: bài toán tô màu bản đồ, lên lịch công việc với giới hạn thời gian
- **Miền rời rạc có thể là vô hạn**.
  - Ví dụ: bài toán tập số nguyên, lập lịch công việc không giới hạn thời gian
- **CSP có miền liên tục là phổ biến trong thực tế**
  - Ví dụ: việc lập kế hoạch thí nghiệm trên Kính viễn vọng Không gian Hubble sử dụng các biến có giá trị thực cho thời gian quan sát.
  - Chúng được nghiên cứu rộng rãi trong lĩnh vực nghiên cứu hoạt động, ví dụ như quy hoạch tuyến tính.

# Các loại ràng buộc trong CSP

- **Ràng buộc một ngôi hạn** chế giá trị của một **biến duy nhất**.
  - Ví dụ, người Nam Úc ghét màu xanh lá cây  $\langle (x, y), (x, z) \rangle$
- **Ràng buộc nhị phân** liên quan đến hai biến
  - Ví dụ: các vùng liền kề có màu khác nhau,  $\langle (x, y), (x, z) \rangle$
- Bất kỳ ràng buộc  $(n\text{-ary}) > 2$  nào cũng có thể chuyển thành ràng buộc nhị phân.
- **CSP nhị phân** là một CSP chỉ có các ràng buộc đơn nhất và nhị phân, có thể được biểu diễn dưới dạng biểu đồ ràng buộc.



# Các loại ràng buộc trong CSP

- Ràng buộc bậc cao hơn bao gồm ba biến trở lên.
  - Ví dụ, ràng buộc bậc ba  $(x, y, z)$  có thể được định nghĩa là  $\langle (x, y, z), < < \text{hoặc} > > \rangle$
- Ràng buộc toàn cục liên quan đến số lượng biến tùy ý.

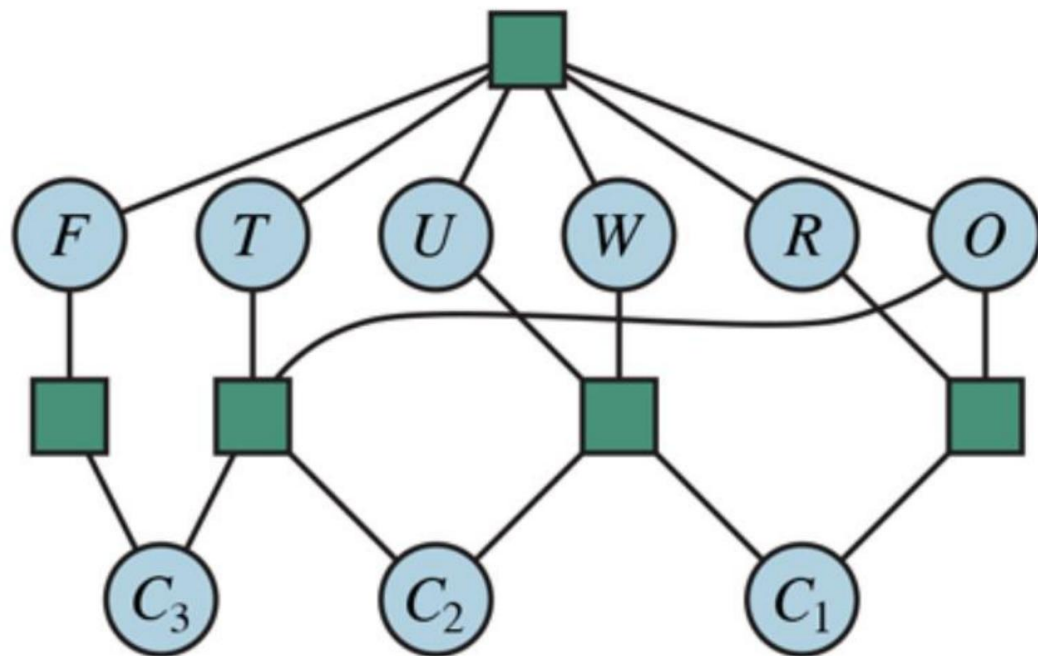
: tất cả các biến liên quan phải có giá trị khác nhau

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

$$\begin{array}{r} \text{E G G} \\ + \text{E G G} \\ \hline = \text{P A G E} \end{array}$$

# Bài toán ví dụ: Mật mã

$$\begin{array}{r} \text{TWO} \\ + \text{TWO} \\ \hline \text{FOUR} \end{array}$$



(Trái) Một bài toán mật mã giả định rằng mỗi chữ cái đại diện cho một chữ số riêng biệt và tìm sự thay thế các chữ số cho các chữ cái sao cho tổng kết quả đúng về mặt số học.

(Phải) Siêu đồ thị ràng buộc cho bài toán mật mã, hiển thị ràng buộc (hộp vuông ở trên cùng) và các ràng buộc cộng cột (bốn hộp vuông ở giữa). Các biến 3 đại diện cho các chữ số mang cho ba cột từ phải qua trái.

# Bài toán ví dụ: Mật mã

- Biến:  $1\ 2\ 3$
- Tên miền:  $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
- Hạn chế:

- $\text{Alldiff}( , , , , , )$

- $+ = + 10 \cdot$   $1$

- $1 + + = + 10 \cdot$   $2$

- $2 + + = + 10 \cdot$   $3$

- $=$   
 $3$

- $0, 0$

$$\begin{array}{r} \text{T W O} \\ + \text{T W O} \\ \hline \text{F O U R} \end{array}$$

Ràng buộc bổ sung cột

Không được phép có số 0 đứng đầu.

# Hạn chế ưu tiên

- Các ràng buộc được mô tả cho đến nay đều là **tuyệt đối những ràng buộc**, vi phạm sẽ loại trừ một giải pháp tiềm năng.
- Nhiều CSP trong thế giới thực bao gồm **các ràng buộc ưu tiên** cho biết giải pháp nào được ưu tiên hơn.
  - Ví dụ, Giáo sư R thích dạy vào buổi sáng. Lịch trình có lớp của Giáo sư R lúc 2 giờ chiều vẫn có thể chấp nhận được nhưng không tối ưu.
- Những hạn chế này thường được mã hóa dưới dạng chi phí biến đổi bài tập **Bài toán tối ưu có ràng buộc (COP)**
  - Ví dụ: suất buổi chiều dành cho Giáo sư R tốn 2 điểm, trong khi suất buổi sáng slot chỉ tốn 1 điểm.

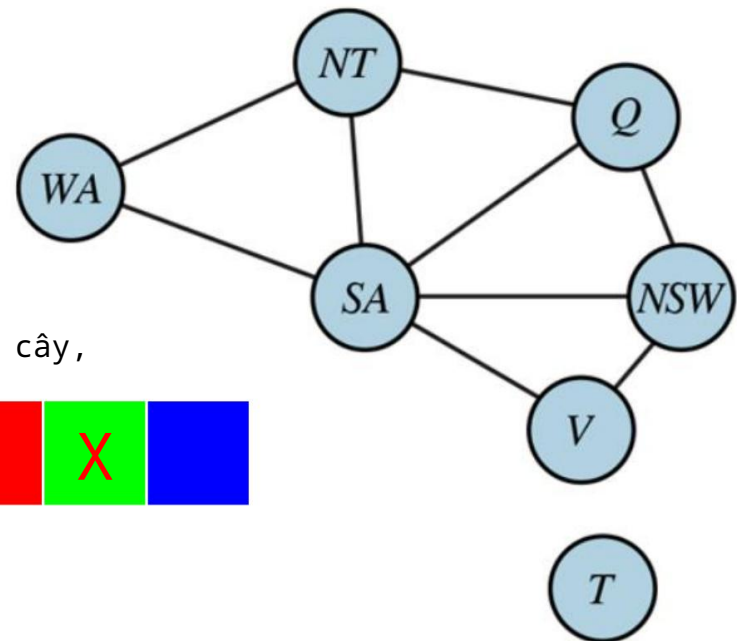
# Truyền bá ràng buộc: Suy luận trong CSP

# Sự lan truyền ràng buộc

- Giảm số lượng giá trị pháp lý cho một biến bằng cách sử dụng các ràng buộc giảm giá trị pháp lý cho các biến khác, v.v.
  - Điều này sẽ để lại ít lựa chọn hơn để xem xét khi chúng tôi thực hiện bước tiếp theo sự lựa chọn của một bài tập biến.
- Nó có thể được kết hợp với tìm kiếm hoặc được thực hiện như một bước tiền xử lý, tức là trước khi bắt đầu tìm kiếm.
  - Quá trình tiền xử lý đôi khi có thể giải quyết toàn bộ vấn đề mà không cần bất kỳ tìm kiếm thêm.

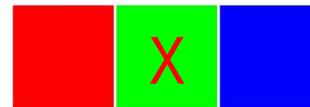
# Tính nhất quán của nút

- Một biến duy nhất **nhất quán với nút** nếu tất cả các giá trị trong miền xác định của biến đó thỏa mãn các ràng buộc đơ n nhất của biến đó .



Người Nam Úc không thích màu xanh lá cây,

Miền của { } sẽ là





# Tính nhất quán hồ quang (AC)

- Cho hai biến, và , tên miền của ai và , tư ơ ng ứng.
- phù hợp với **cung** nếu với mọi giá trị trong có một số giá trị trong đó **thỏa mãn ràng buộc nhị phân** ( , ).
  - ví  $\langle ( , ), =^2 \rangle$ , cả hai miền đều là tập hợp các chữ số less 's  
dụ: tên miền tới  $\{0, 1, 2, 3\}$  và 's đến  $\{0, 1, 4, 9\}$
- Tính nhất quán của hồ quang có thể **không có tác dụng trong một số trường hợp**.
  - Ví dụ, trong ràng buộc , bất kể giá trị nào đư ợc chọn cho , có một giá trị hợp lệ cho , đang theo dõi  
 $\{(\text{đỏ}, \text{xanh lá cây}), (\text{đỏ}, \text{xanh dư ơ ng}), (\text{xanh lá cây}, \text{đỏ}), (\text{xanh lục}, \text{xanh lam}), (\text{xanh lam}, \text{đỏ}), (\text{xanh lam}, \text{xanh lục})\}$

hàm AC-3(csp) trả về false nếu tìm thấy sự không nhất quán và đúng nếu không thì xếp hàng một hàng các cung, ban đầu tất cả các cung trong csp trong khi hàng đợi không

trống do  $(X_i, X_j)$  POP(queue) if

REVISE(csp,  $X_i, X_j$ ) thì

nếu kích thước của  $D_i = 0$  thì

trả về false cho mỗi  $X_k$  trong  $X_i$ .NEIGHBORS

-  $\{X_j\}$  do

thêm  $(X_k, X_i)$  vào hàng đợi

trả lại sự thật

- Độ phức tạp trong trường hợp xấu nhất là ( <sup>3</sup>
- ) : số ràng buộc nhị phân (cung)
- Mỗi biến có kích thước miền

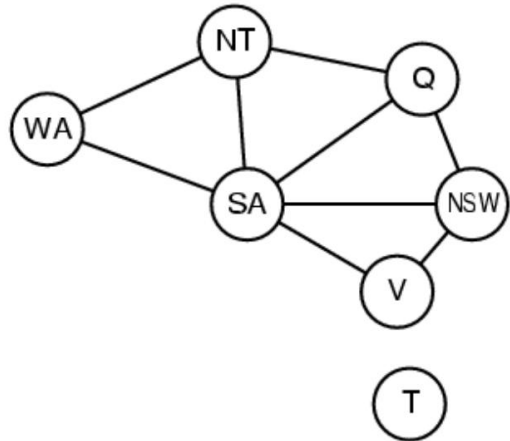
hàm REVISE(csp,  $X_i, X_j$ ) trả về true nếu chúng ta sửa đổi miền của  $X_i$  sửa đổi sai cho mỗi  $x$  trong  $D_i$  do

nếu không có giá trị  $y$  nào trong  $D_j$  cho phép  $(x, y)$  thỏa mãn ràng buộc giữa  $X_i$  và  $X_j$  xóa  $x$

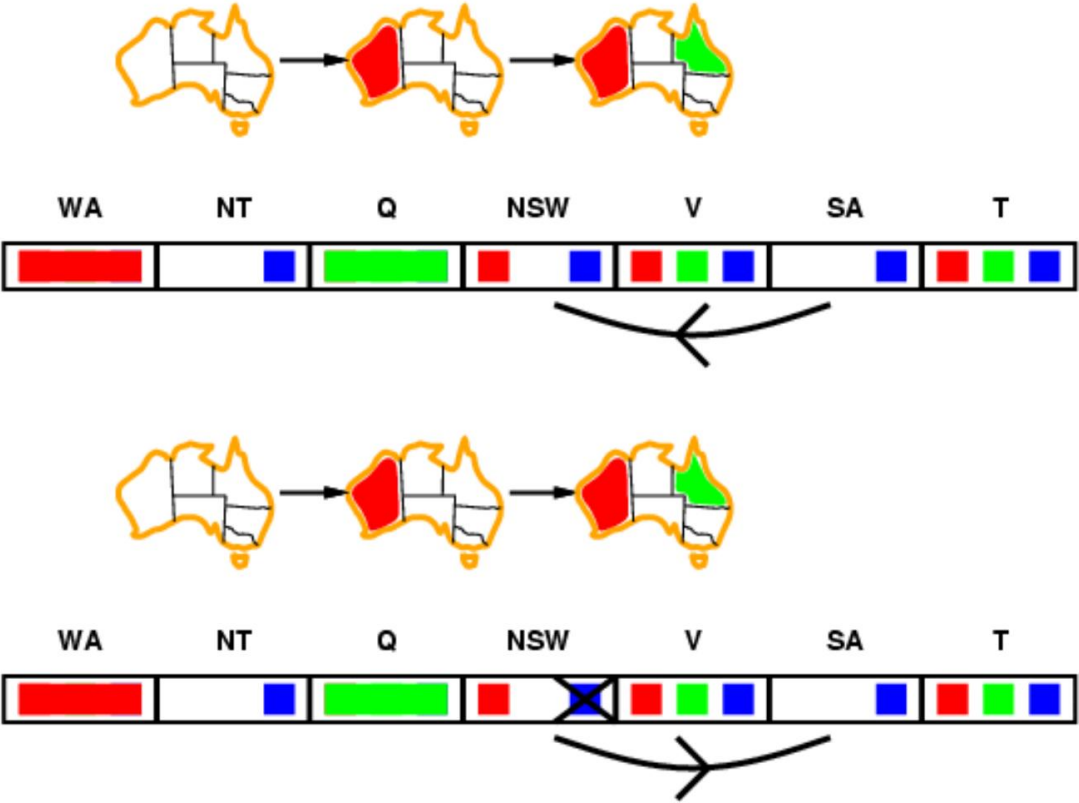
khỏi  $D_i$  sửa đổi

đúng

trở lại sửa đổi



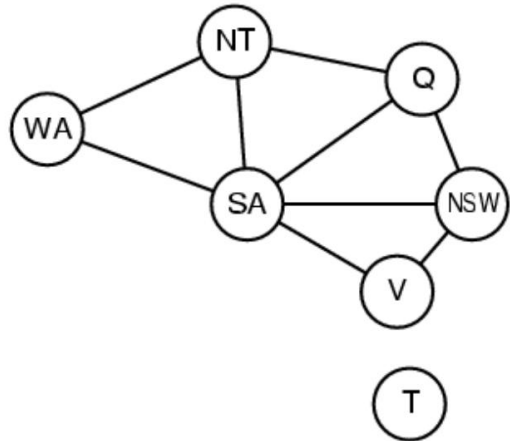
Xem xét trạng thái tìm kiếm sau và đã được chỉ định.



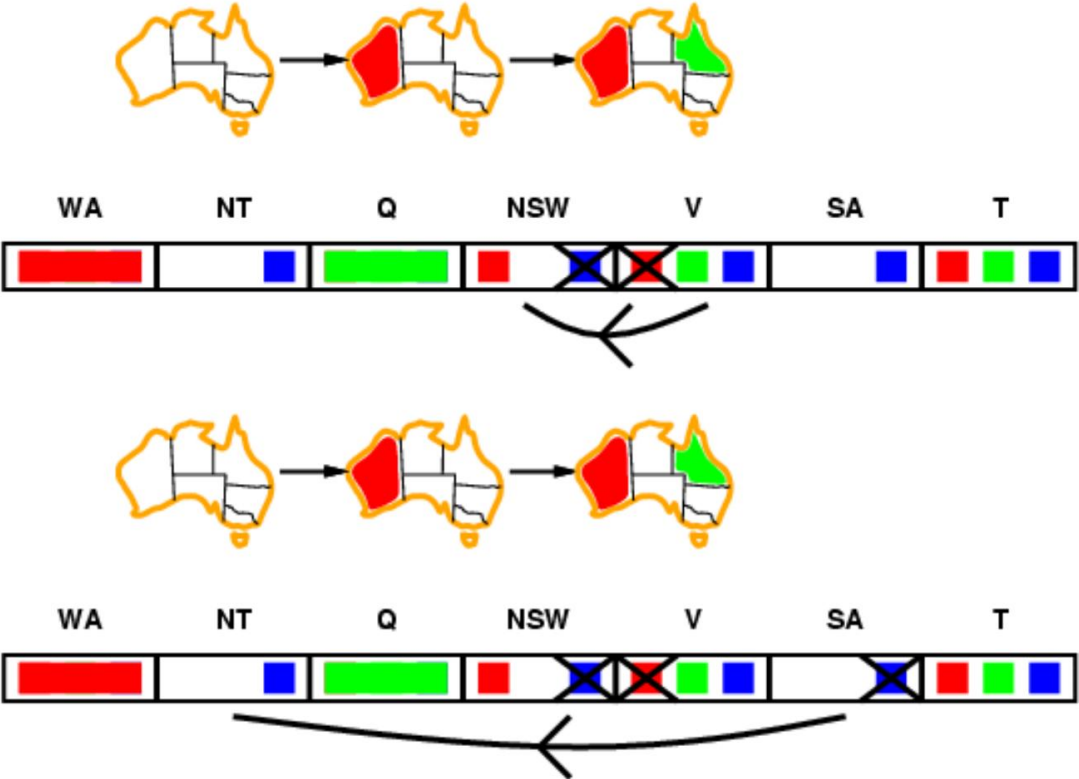
• là nhất quán nếu  
= và =

• nhất quán nếu  
= và =  
= và =? ? ?

Tính nhất quán của hồ quang có thể được thực hiện bằng cách loại bỏ từ



Nếu mất một giá trị, hàng xóm cần được kiểm tra lại



- Kiểm tra
- Không nhất quán đối với =  
loại bỏ từ
- Kiểm tra
- Không nhất quán đối với =  
loại bỏ từ
- Miền của trống.

Tính nhất quán của hồ quang phát hiện lỗi sớm hơn việc kiểm tra chuyển tiếp.

# Tính nhất quán của vòng cung: Một đánh giá

- Tính nhất quán của Arc có thể chạy trước khi bắt đầu tìm kiếm hoặc sau mỗi lần gán, lặp đi lặp lại cho đến khi không còn sự mâu thuẫn.
  - Đánh đổi: Loại bỏ các phần lớn không nhất quán của không gian trạng thái trong khi yêu cầu một số chi phí để thực hiện
- AC thực hiện một phương pháp có hệ thống để kiểm tra hồ quang.
  - Các cung đến có thể trở nên không nhất quán, trong khi các cung đi ra vẫn đứng yên.

# Câu đố 01: Lên thời gian biểu

- Bạn đang lên lịch cho các lớp khoa học máy tính vào các ngày thứ Hai, thứ Tư và thứ Sáu.
- Có 5 lớp và 3 giáo sư sẽ dạy các lớp này. • Bạn bị hạn chế rằng mỗi giáo sư chỉ có thể dạy một lớp tại một thời điểm.
- Các lớp học là:
  - Lớp 1 - Giới thiệu về Lập trình: họp từ 8:00-9:00 sáng •
  - Lớp 2 - Giới thiệu Trí tuệ nhân tạo: họp từ 8:30-9:30 sáng • Lớp
  - 3 - Xử lý ngôn ngữ tự nhiên: họp từ 9:00-10 :00 sáng
  - Lớp 4 - Thị giác máy tính: học từ 9h00-10h00 sáng
  - Lớp 5 - Machine Learning: họp từ 9h30-10h30 sáng
- Các giáo sư là:
  - Giáo sư A hiện đang dạy Lớp 3 và Lớp 4.
  - Giáo sư B hiện đang dạy Lớp 2, 3, 4 và 5.
  - Giáo sư C hiện đang dạy các lớp 1, 2, 3, 4 và 5.

# Câu đố 01: Lên thời gian biểu

- Xây dựng bài toán này dư ới dạng CSP trong đó có **một biến cho mỗi lớp**, nêu rõ các lĩnh vực (tức là các giáo sư có sẵn) và các ràng buộc.
  - Các ràng buộc cần đư ợc xác định một cách hình thức và chính xác như ng có thể ngầm định hơ n là rõ ràng.
- Vẽ đồ thị ràng buộc liên quan đến CSP của bạn.
- Hiện thị miền xác định của các biến sau khi chạy tính nhất quán cung trên biểu đồ ban đầu này (sau khi đã thực thi bất kỳ ràng buộc đờ n phân nào).
- Đư a ra một giải pháp cho CSP này.



# Câu đố 02: Hình vuông kỳ diệu bậc 3

- Bình phương ma thuật cấp  $n$  là sự sắp xếp của  $n^2$  các số nguyên dương khác nhau trong một hình vuông sao cho tổng các số ở tất cả các hàng, tất cả các cột và cả hai đường chéo về cùng một hằng số.
- Đối với hình vuông ma thuật bậc 3, có một công thức tổng quát do Édouard Lucas nghĩ ra.

Bàn bên cạnh được làm bằng tích cực số nguyên  $p$ ,  $q$  và  $z$ .  
Chín số này sẽ tạo thành một bình phương ma thuật với hằng số ma thuật  $3(p+q)$  miễn là  $0 < q < z < p$  và  $z \neq 2q$ .

$p + q - z$	$p + 2q + z$	$p$
$p + z$	$p + q$	$p + 2q - z$
$p + 2q$	$p - z$	$p + q + z$

- Xây dựng phương pháp xây dựng hình vuông ma thuật ở trên dưới dạng CSP trên các biến  $p$ ,  $q$  và  $z$ , mỗi biến có miền khởi đầu là  $\{1, \dots, 9\}$ .
- Gọi hằng số ảo là 15. Giải CSP.



Quay lại Tìm kiếm CSP

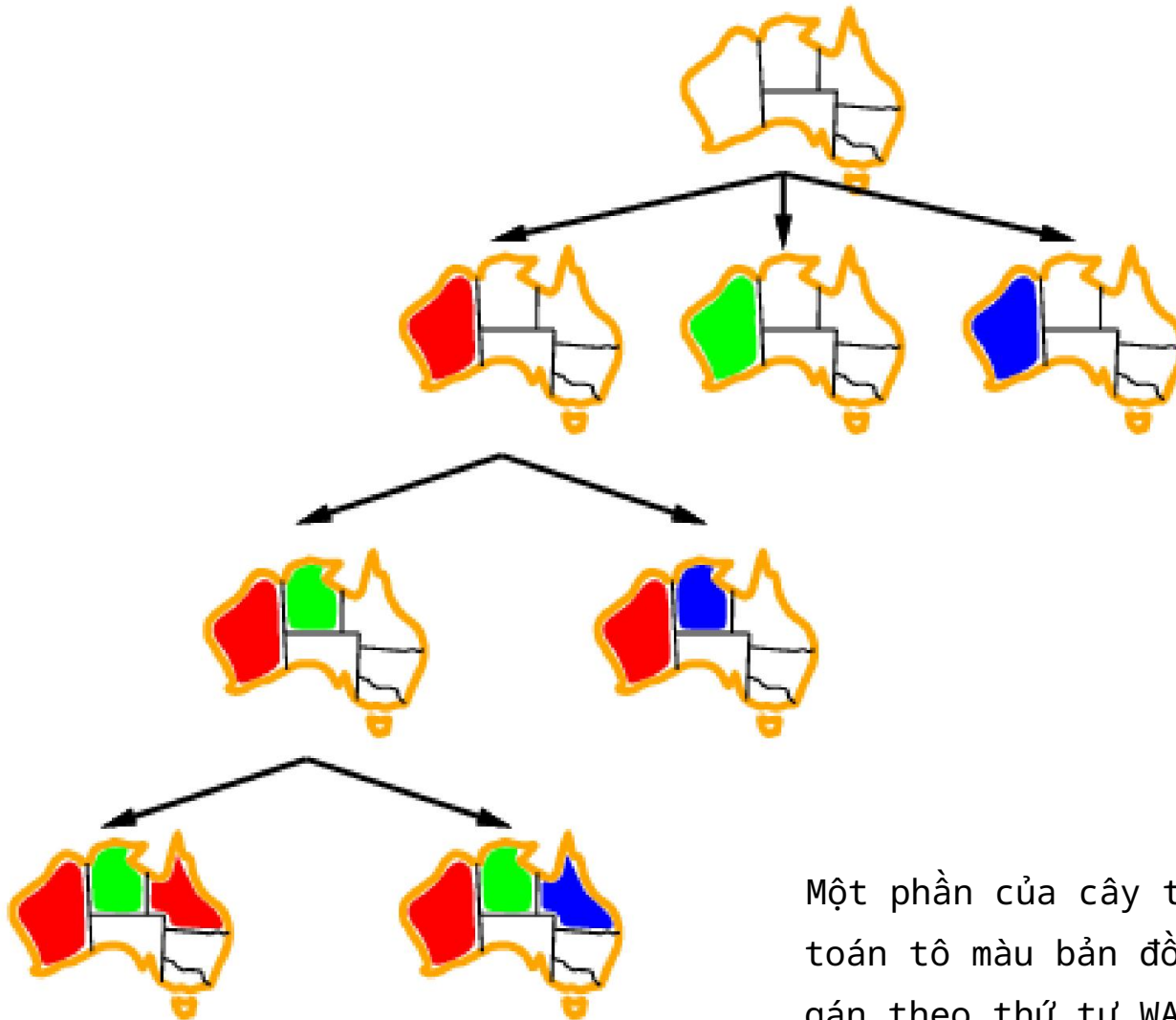
# CSP là một vấn đề tìm kiếm

- Để một trạng thái được gán một phần và một hành động mở rộng phân công.
- Tìm kiếm giới hạn độ sâu tiêu chuẩn có thể được sử dụng để giải các CSP.
  - Trạng thái ban đầu: gán trống { }
  - Hàm kế thừa: gán giá trị cho một biến chưa được gán đồng ý với nhiệm vụ hiện tại thất bại nếu không có nhiệm vụ hợp pháp
  - Kiểm tra mục tiêu: bài tập hiện tại đã hoàn thành
- Tất cả các phép gán đầy đủ của các biến có kích thước miền xuất hiện ở độ sâu trên cây tìm kiếm.
  - Hệ số phân nhánh = ( ) ở độ sâu , có ! chỉ với những bài tập hoàn chỉnh! lá

# Tính giao hoán của CSP

- Một bài toán có tính giao hoán nếu thứ tự áp dụng bất kỳ tập hợp các hành động nhất định không thành vấn đề.
- Phép gán biến trong CPS có tính chất giao hoán.
  - Ví dụ:  $[x = \text{thì} = \dots] \quad [x = \dots \text{thì} =]$
- Chúng ta chỉ cần xem xét một biến duy nhất tại mỗi nút trong cây tìm kiếm
  - Hệ số phân nhánh =  $\dots$ , và có lá.

# Tìm kiếm quay lui: Một ví dụ



Một phần của cây tìm kiếm cho bài toán tô màu bản đồ. Các biến được gán theo thứ tự WA, NT, Q,.

hàm BACKTRACKING-SEARCH(csp) trả về giải pháp hoặc trả về lỗi BACKTRACK(csp, { })

hàm BACKTRACK(csp, gán) trả về một giải pháp hoặc thất bại nếu việc gán hoàn thành

sau đó trả về phép gán var SELECT-UNASSIGNED-VARIABLE(csp, gán) cho mỗi giá trị trong ORDER-DOMAIN-VALUES(csp, var, task) do if

value phù hợp với phép gán thì thêm {var = value} vào suy luận phép gán

INFERENCE(csp, var, transfer) nếu suy luận failed

thì thêm suy luận vào kết quả csp

BACKTRACK(csp, var, task) nếu kết quả failed thì trả

về kết quả

Biến nào sẽ được chỉ định tiếp theo?

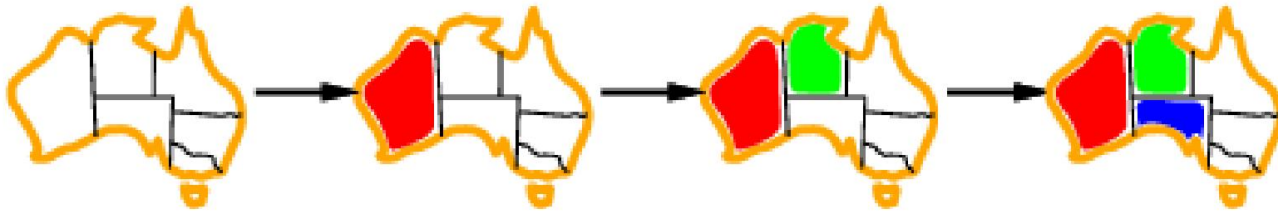
Các giá trị của nó nên được thử theo thứ tự nào?

Những suy luận nào nên được thực hiện?

xóa suy luận khỏi csp xóa {var = value} khỏi lỗi trả về bài tập

# Heuristic giá trị còn lại tối thiểu

- **MRV heuristic** lấy biến có ít giá trị pháp lý nhất.
  - Nó chọn một biến có nhiều khả năng gây ra lỗi sớm nhất, do đó tĩa cây tìm kiếm.



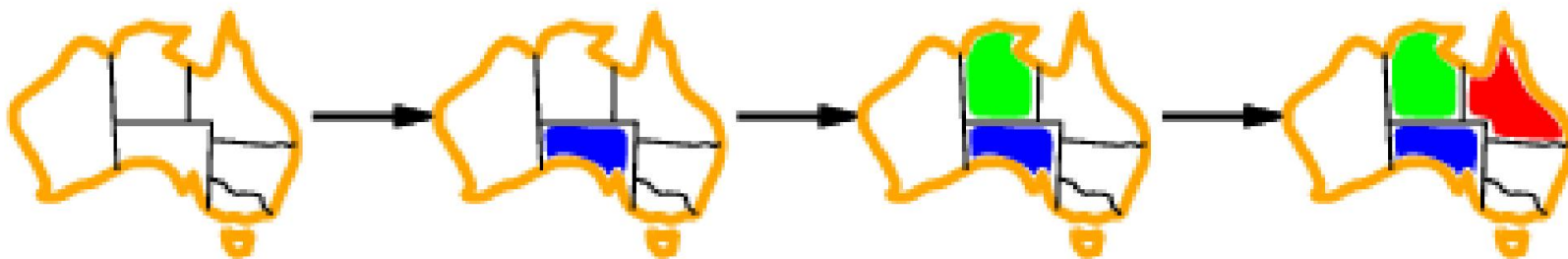
Chỉ có một giá trị có thể có cho sau [ = , = ].

- Nó thường **hoạt động tốt hơn** so với **thứ tự ngẫu nhiên hoặc tĩnh**.
  - Ưu điểm đôi khi nằm ở mức độ lớn, tuy nhiên kết quả khác nhau tùy theo vấn đề.



# Heuristic độ

- DH heuristic chọn biến có liên quan đến giá trị lớn nhất số ràng buộc trên các biến chưa được gán khác.
- Nó cố gắng giảm yếu tố phân nhánh trong các lựa chọn trong tương lai.

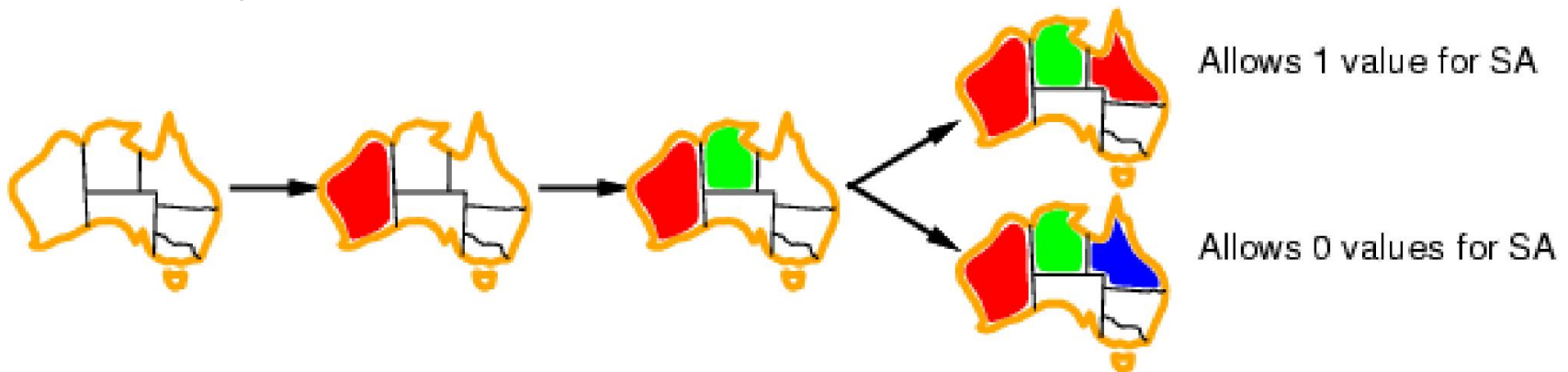


có bậc cao nhất là 5, các biến khác ngoại trừ có bậc 2 hoặc 3.

- DH là yếu tố quyết định hầu hết các biến bị ràng buộc.

# Heuristic giá trị ít ràng buộc nhất

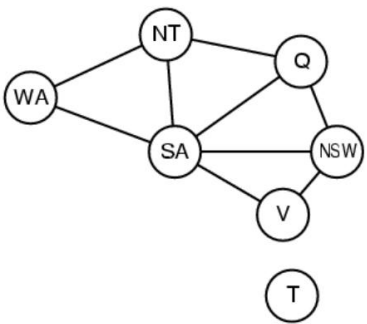
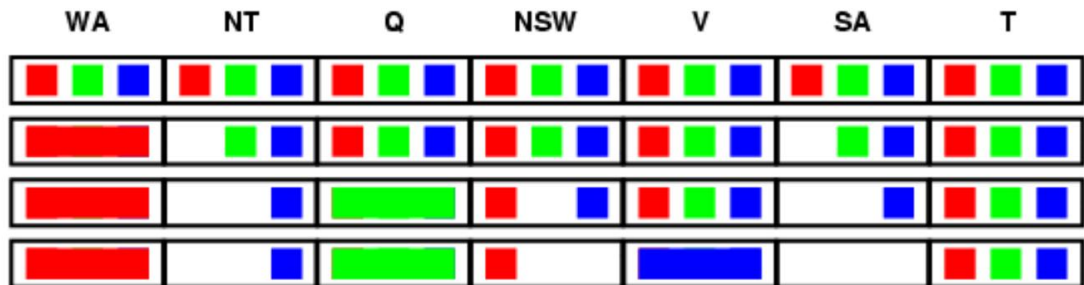
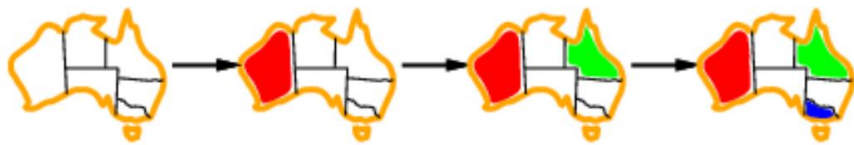
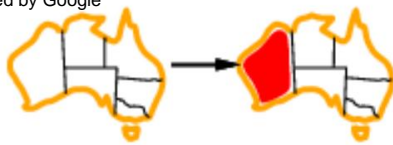
- **LCV heuristic** ưu tiên giá trị loại trừ ít lựa chọn nhất cho các biến lân cận trong biểu đồ ràng buộc.
- Nó cố gắng để lại sự linh hoạt tối đa cho biến tiếp theo bài tập.



- Tóm lại, việc lựa chọn biến phải là thất bại trước, trong khi lựa chọn giá trị phải là thất bại cuối cùng. Tại sao?

# Tìm kiếm và suy luận xen kẽ

- Gọi là biến vừa đư ợc gán và là mỗi biến chư a đư ợc gán đư ợc kết nối bởi một ràng buộc nào đó.
- Kiểm tra chuyển tiếp (FC) xóa mọi giá trị khỏi miền của không nhất quán với giá trị đư ợc chọn cho .
- Kết hợp phư ơ ng pháp phỏng đoán MRV với kiểm tra chuyển tiếp sẽ cải thiện hiệu quả việc tìm kiếm trong nhiều vấn đề.
- FC phát hiện nhiều điểm không nhất quán, như ng không phải tất cả.
  - Nó làm cho biến hiện tại nhất quán như ng không nhìn về phía trư ớc.



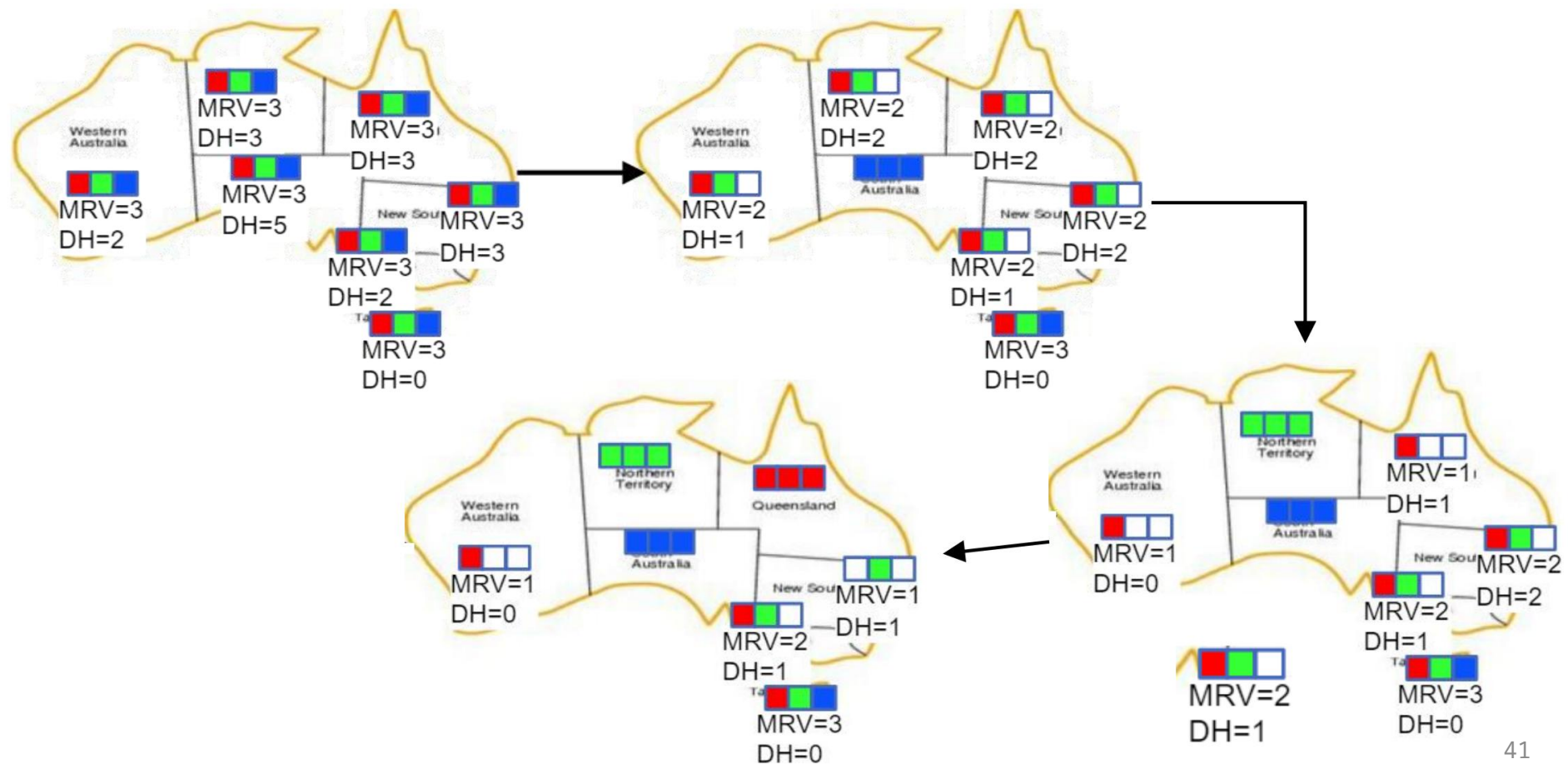
- Gán { = } và
- không còn màu đỏ

- Chỉ định { = }
- , và không thể xanh đư ợc nữa.

- Gán = không }
- còn màu xanh. trông
- r ồng

# Ví dụ: Tô màu bản đồ nước Úc

- Vấn đề tô màu bản đồ có thể được giải quyết dễ dàng bằng cách sử dụng kiểm tra chuyển tiếp và chặn đoán MRV.

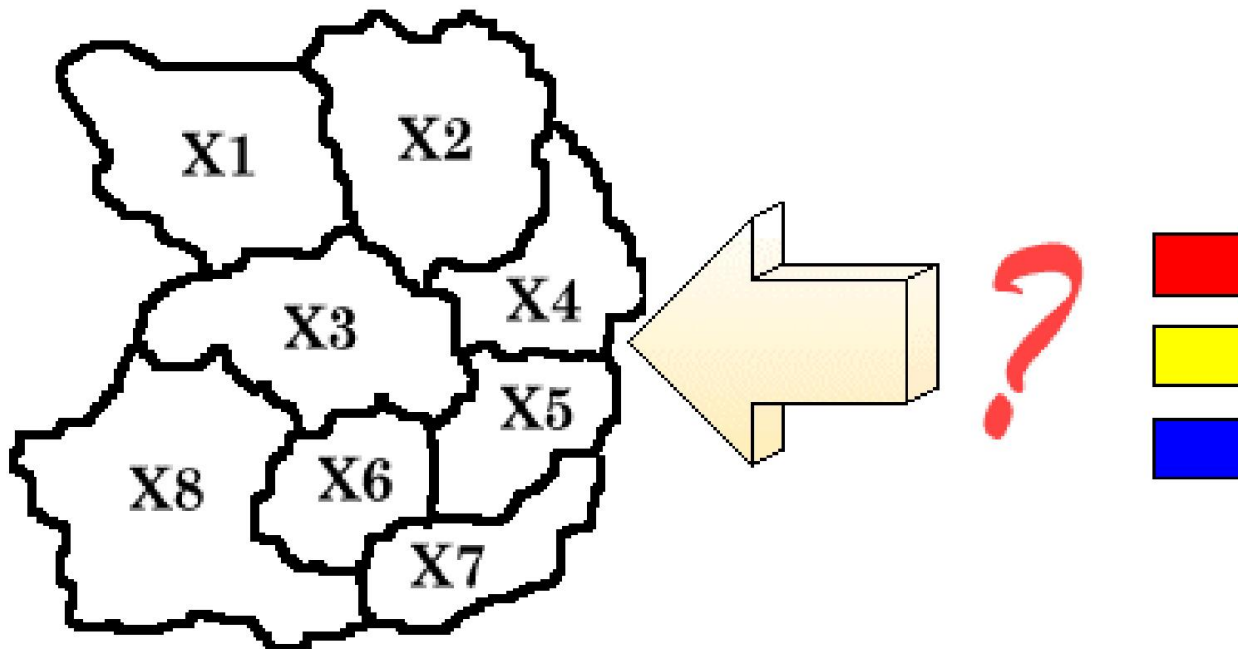


# Duy trì tính nhất quán hồ quang (MAC)

- Xét biến cung (có hướng) và là  $(\quad, \quad)$ , nhiệm vụ được giao ở đâu mỗi lần cận chương được gán của .
- Thuật toán MAC chỉ bắt đầu bằng , để truyền  $(\quad)$  và áp dụng AC-3 bá ràng buộc.
- Nếu bất kỳ biến nào có miền xác định bị rút gọn về tập trống, AC-3 thất bại và việc quay lại xảy ra ngay lập tức.

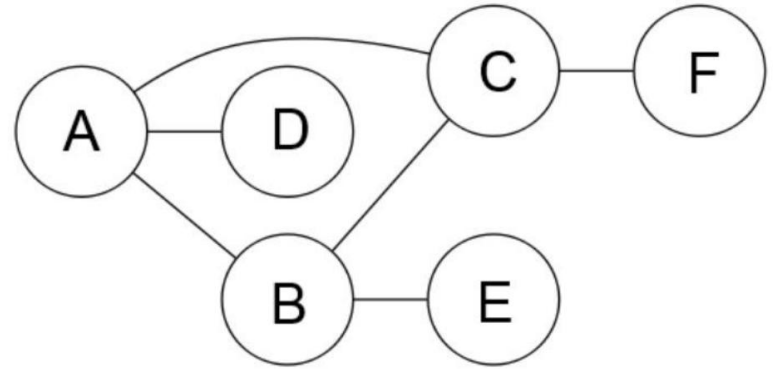
## Câu đố 03: Bài toán tô màu bản đồ

- Tô màu từng vùng là **đỏ**, **vàng** hoặc **xanh lam** sao cho không vùng lân cận nào có cùng màu



# Câu đố 04: AC vs FC

- Biểu đồ bên cạnh là biểu đồ ràng buộc cho CSP chỉ có ràng buộc nhị phân. Ban đầu, không có biến nào được gán.



- Đối với mỗi kịch bản nhất định, hãy đánh dấu tất cả các biến mà việc lọc được chỉ định có thể dẫn đến miền của chúng bị thay đổi. Lưu ý rằng mọi kịch bản đều độc lập với các kịch bản khác.



# Câu đố 04: AC vs FC

- Một giá trị được gán cho A. Những miền nào có thể bị thay đổi do chạy tiếp để kiểm tra A?

A

B

C

D

E

F

- Một giá trị được gán cho A, sau đó việc kiểm tra chuyển tiếp được thực hiện cho A. Sau đó, một giá trị được gán cho B. Những miền nào có thể bị thay đổi do chạy tiếp để kiểm tra B?

A

B

C

D

E

F

- Một giá trị được gán cho A. Những miền nào có thể bị thay đổi do thực thi tính nhất quán của vòng cung sau nhiệm vụ này?

A

B

C

D

E

F

- Một giá trị được gán cho A và sau đó tính nhất quán của cung được thực thi. Sau đó một giá trị được gán cho B. Những miền nào có thể bị thay đổi do thực thi tính nhất quán của cung sau khi gán cho B?

A

B


C

D

E

F

# Tìm kiếm cục bộ cho CSP



# Tìm kiếm cục bộ cho CSP

- Công thức trạng thái hoàn chỉnh

- Trạng thái ban đầu gán giá trị cho mọi biến vi phạm ràng buộc
- Việc tìm kiếm thay đổi giá trị của một biến tại một thời điểm giải quyết sự xung đột

- Min-conflicts heuristic: số lượng xung đột tối thiểu

với các biến khác

- Xung đột tối thiểu có hiệu quả đáng kinh ngạc đối với nhiều CSP.
  - Bài toán triệu quân hậu có thể được giải quyết ~ 50 bước
  - Kính viễn vọng Không gian Hubble: thời gian lên lịch quan sát trong một tuần giảm từ 3 tuần (!) xuống còn ~10 phút

# Thuật toán MIN-CONFLICTS

hàm MIN-CONFLICTS(*csp*, các bước tối đa) trả về một giải pháp hoặc các đầu vào

lỗi : *csp*, một vấn đề thỏa mãn ràng buộc các bước

tối đa, số bước được phép trừ 0 khi từ bỏ hiện tại một

phép gán hoàn chỉnh ban đầu cho *csp* với  $i = 1$  đến các

bước tối đa làm

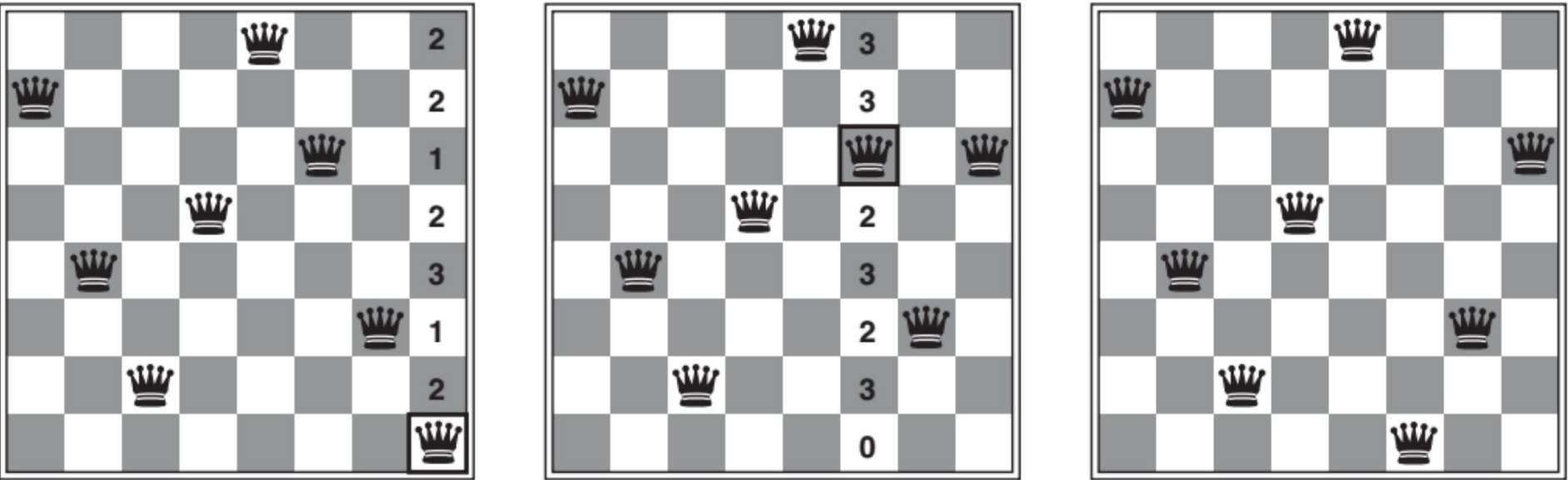
nếu hiện tại là một giải pháp cho *csp* thì trả về hiện tại *var*

một biến xung đột được chọn ngẫu nhiên từ giá trị *csp.VARIABLES* giá trị *v* cho *var*

giúp giảm thiểu CONFLICTS(*var*, *v*, *current*, *csp*) đặt *var* = value in *current*

trả lại thất bại

# XUNG ĐỘT TỐI THIỂU: 8 quân hậu



Một giải pháp hai bước sử dụng xung đột tối thiểu cho bài toán 8 quân hậu.

Ở mỗi giai đoạn, một quân hậu được chọn để tái bổ nhiệm vào cột của nó.

Số lượng quân hậu tấn công (tức là xung đột) được hiển thị trong mỗi ô vuông.

Thuật toán di chuyển quân hậu đến ô xung đột tối thiểu, phá vỡ các mối quan hệ một cách ngẫu nhiên.

# Tìm kiếm cục bộ cho CSP

- Bối cảnh của CSP theo phương pháp phỏng đoán xung đột tối thiểu thường có một loạt cao nguyên.
  - Có hàng triệu bài tập khác nhau mà chỉ cần một xung đột là có thể giải quyết được.
- Tìm kiếm cao nguyên: cho phép di chuyển ngang sang trạng thái khác với điểm giống nhau
- Tìm kiếm Tabu: giữ một danh sách nhỏ các trạng thái được truy cập gần đây và cấm thuật toán quay trở lại trạng thái đó
- Ủ mô phỏng cũng có thể được sử dụng

# Trọng số ràng buộc

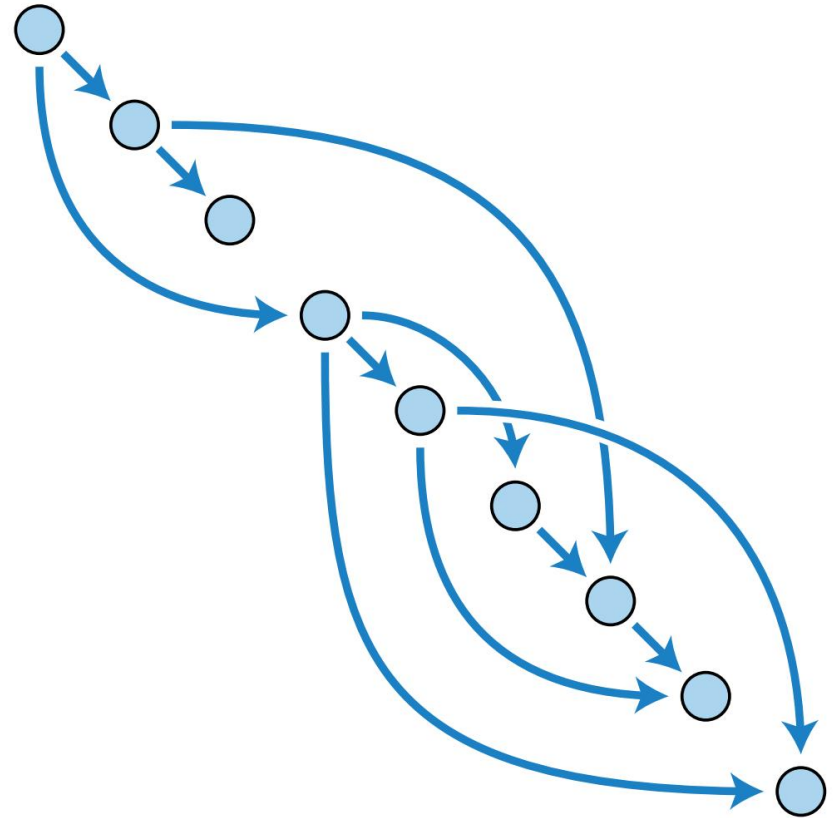
- Tập trung tìm kiếm vào các ràng buộc quan trọng
- Mỗi ràng buộc được gán một trọng số, , ban đầu tất cả đều là 1.
- Ở mỗi bước, chọn một cặp biến/giá trị để thay đổi có tổng trọng số thấp nhất trong tất cả các ràng buộc bị vi phạm
- Tăng trọng số của từng ràng buộc bị vi phạm bởi phép gán hiện tại

# Tìm kiếm cục bộ trong cài đặt trực tuyến

- Vấn đề lập kế hoạch: cài đặt trực tuyến
  - Lịch trình hàng tuần của hãng hàng không có thể bao gồm hàng ngàn chuyến bay và hàng chục của hàng ngàn nhiệm vụ nhân sự
  - Thời tiết xấu tại một sân bay có thể khiến lịch trình không thể thực hiện được.
- Lịch trình cần được sửa chữa với số lượng tối thiểu những thay đổi.
  - Thực hiện dễ dàng với tìm kiếm cục bộ bắt đầu từ lịch trình hiện tại
  - Tìm kiếm quay lui với tập hợp ràng buộc mới thường đòi hỏi nhiều thời gian hơn và có thể tìm ra giải pháp có nhiều thay đổi so với lịch trình hiện tại



# Các cấu trúc của vấn đề



# Các bài toán con độc lập

- Nếu bài tập là một giải pháp của  $A$ , vậy thì  $A$  là một giải pháp của  $A$ .

- Ví dụ: tô màu bản đồ Úc: Tasmania và đất liền

- Giả sử từng bài  $A_i$  có các biến từ các biến.

toán • Khi đó có  $n$  bài toán con, mỗi bài toán có hầu hết công việc để giải quyết.

- ở đâu là hằng số và là kích thước của miền.

- Do đó, tổng công là  $(n+1) \cdot 2^n$ , là tuyến tính trong  $n$ .

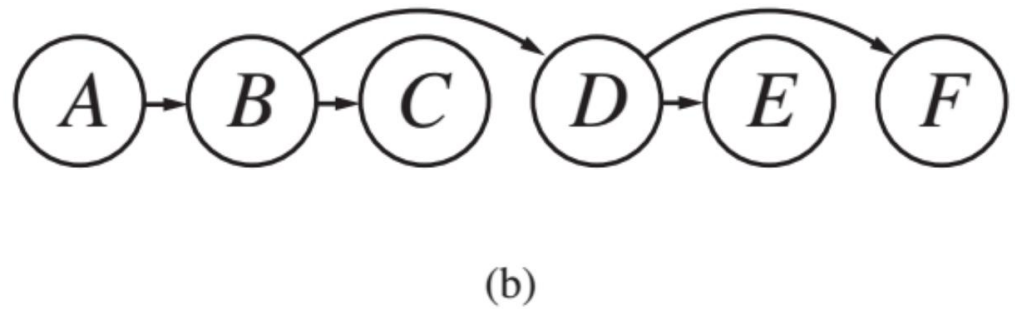
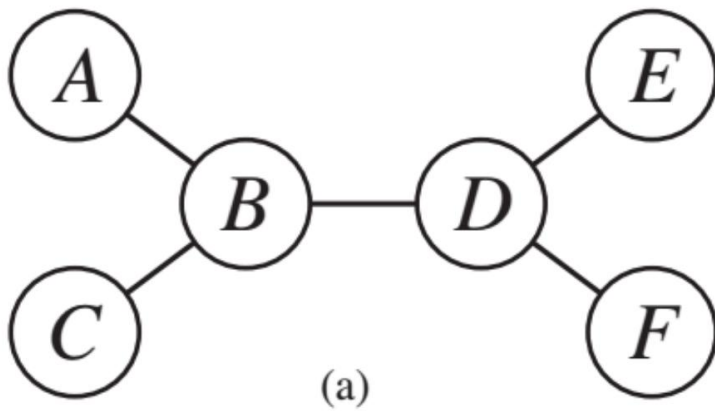
- Nếu không phân rã thì tổng công là  $(n+1) \cdot 2^n$ .

# CSP có cấu trúc cây

- Đồ thị ràng buộc là một cây khi có hai biến bất kỳ chỉ được kết nối bằng một con đường.
- Bất kỳ CSP có cấu trúc cây nào cũng có thể được giải tuyến tính theo thời gian trong số lượng biến
- Tính nhất quán hồ quang có hướng (DAC): Một CSP được tính nhất quán hồ quang có hướng theo thứ tự của các biến iff  $1, 2, \dots, n$  mọi thứ đều phù hợp với từng cái cho  $i > 1$ .

# CSP có cấu trúc cây

- Sắp xếp tôpô: đầu tiên chọn bất kỳ biến nào làm gốc của cây và chọn thứ tự của các biến sao cho mỗi biến xuất hiện sau cha của nó trong cây.

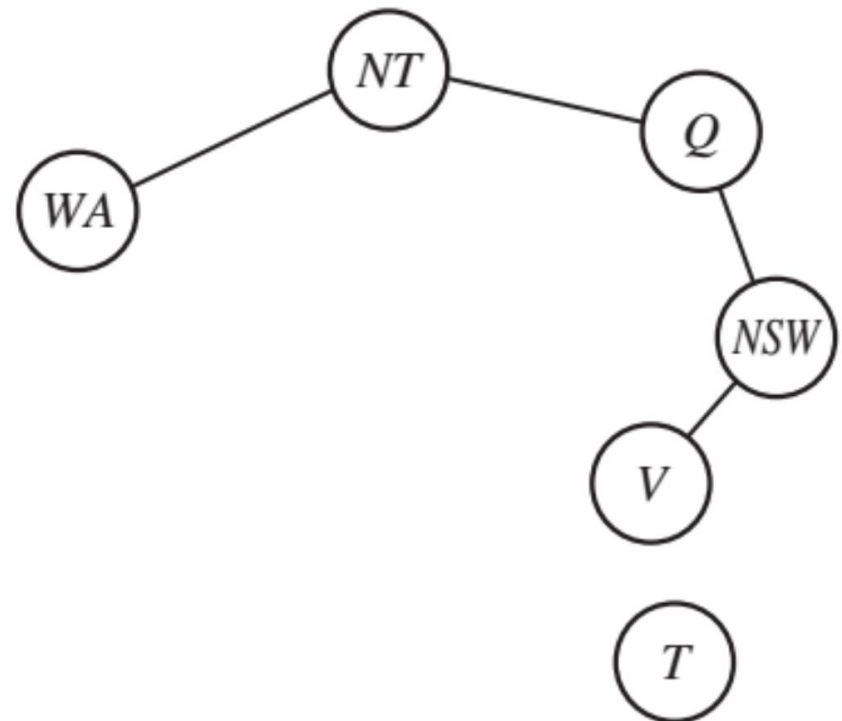
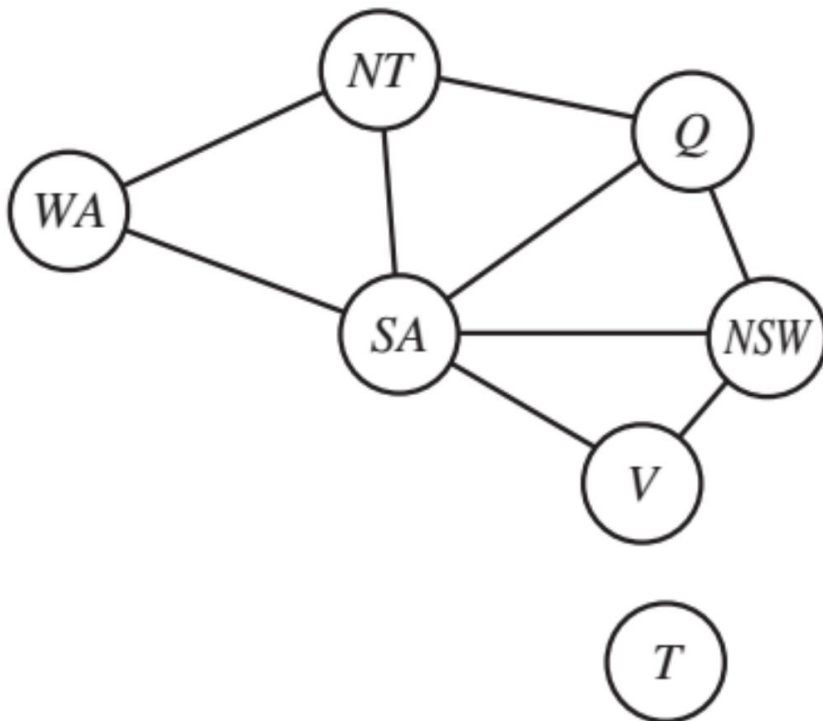


(a) Biểu đồ ràng buộc của CSP có cấu trúc cây.

(b) Thứ tự tuyến tính của các biến phù hợp với cây có *A* là gốc.

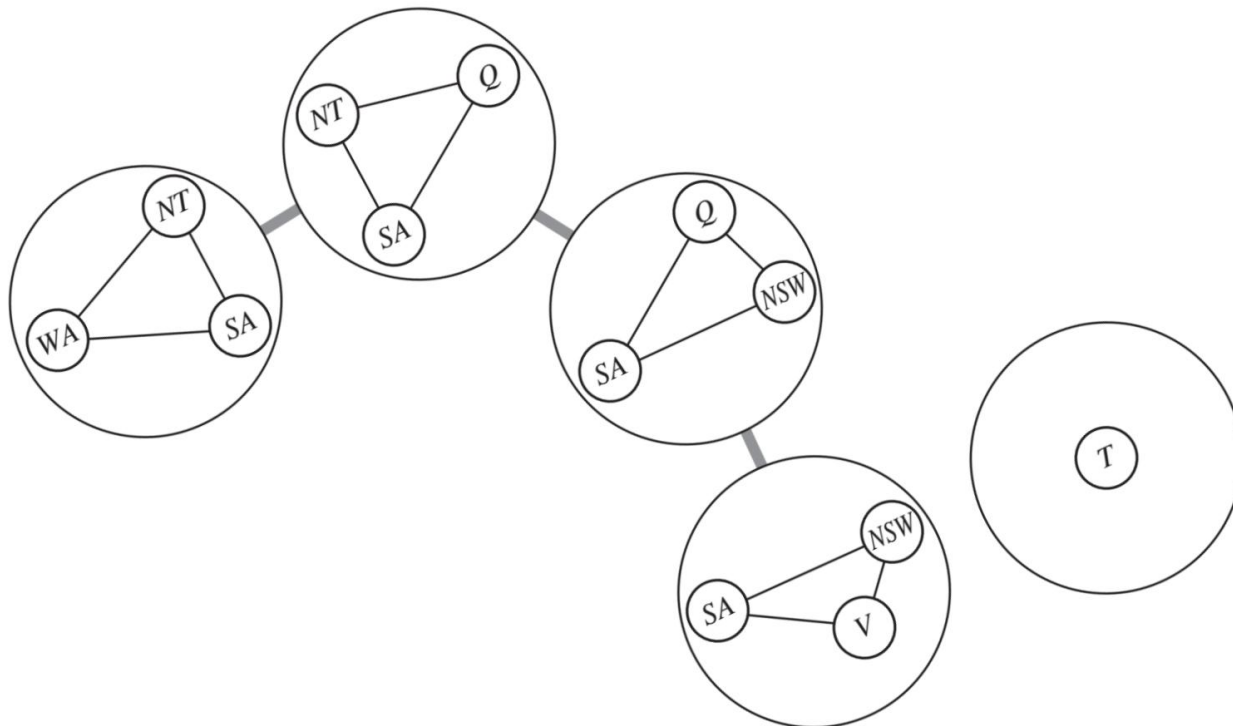
# Giảm đồ thị thành cây

- Gán giá trị cho một số biến sao cho những biến còn lại các biến tạo thành một cây
  - Ví dụ: sửa một giá trị và xóa khỏi miền của các biến khác bất kỳ giá trị không phù hợp với giá trị đã chọn



# Giảm đồ thị thành cây

- Xây dựng sự phân rã cây của đồ thị ràng buộc thành một tập hợp các bài toán con đư ợc kết nối.
- Mỗi bài toán con đư ợc giải độc lập và kết quả sau đó các giải pháp đư ợc kết hợp.



# Cấu trúc của các giá trị

- Xét bài toán tô màu bản đồ bằng màu sắc.
- Đối với mỗi giải pháp nhất quán, có một bộ ! các giải pháp được hình thành bằng cách hoán vị tên màu.
  - Ví dụ, , và tất cả đều phải có màu sắc khác nhau, nhưng có 3! cách gán ba màu cho ba vùng này.
- **Ràng buộc phá vỡ tính đối xứng:** Áp đặt một thứ tự tùy ý ràng buộc yêu cầu các giá trị phải theo thứ tự bảng chữ cái
  - Ví dụ, < < chỉ có một giải pháp khả thi: { = = } ,  
= ,

