Christophe Rodrigues

DataScience & IA

Plan du cours

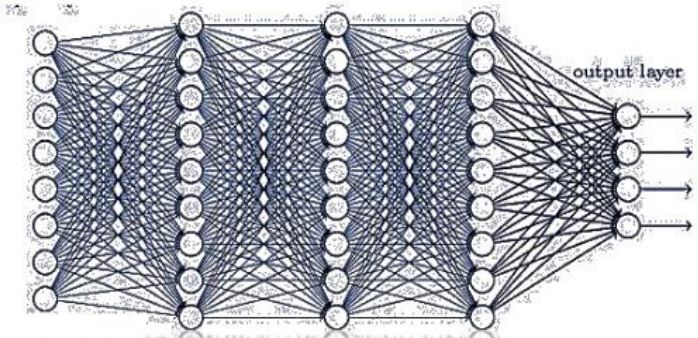
- IA et Deep Learning, une révolution?
- Rappels et relations entre les algorithmes déjà abordés
- Problèmes de satisfaction de contraintes (CSP)

Turc mécanique

- Automate capable de jouer aux échecs et de gagner contre des joueurs humains
- présenté pour la première fois en 1770.
- ~~Superbe IA~~
- Superbe Escroquerie

Deep Learning

- **Success of Deep Learning techniques attributable to concurrence of big data sets, scalable hardware, and high-level software**



DEEP NEURAL NETWORK*

Data

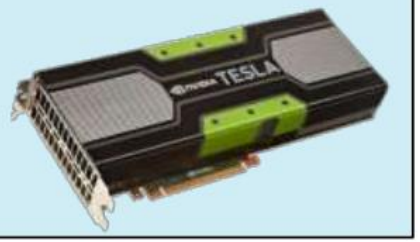
Software

NVIDIA CUDA

theano

TensorFlow

Hardware

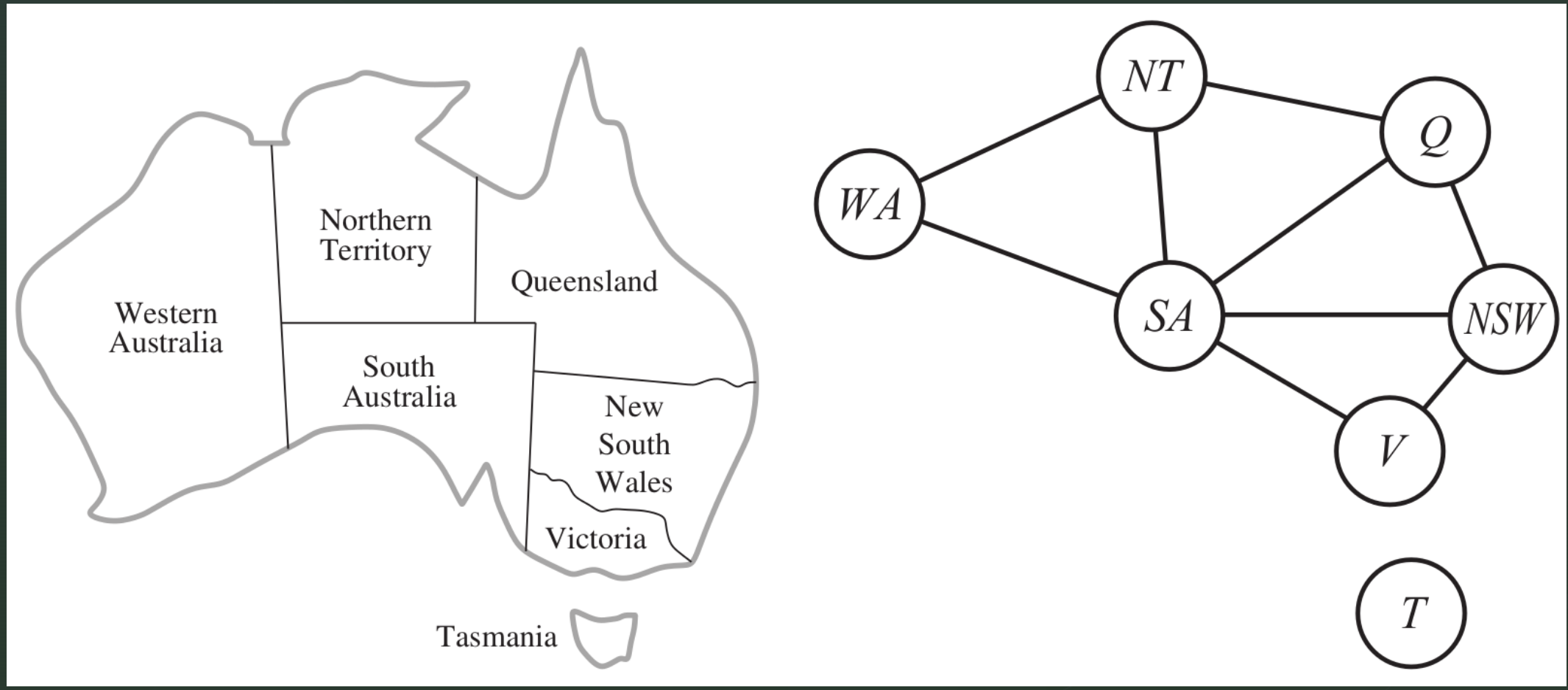


*Modeling high-level abstractions in data by using multiple non-linear transformations

Rappels et relations entre méthodes abordées

- Algorithmes évolutionnaires (ou génétiques)
- Découverte de règles d'association (pattern mining)
- A*
- Problèmes de satisfaction de contraintes

- Problèmes de satisfaction de contraintes CSP par l'exemple: 3 couleurs possibles, aucun voisin ne doit avoir la même



Définitions

- X un ensemble de variables : {X1,…,Xn}
- D un ensemble de domaines : {D1,…,Dn} (un pour chaque variable X)
  - Avec Di = {v1,…,vn} pour Xi
- C un ensemble de contraintes spécifiant des combinaisons de valeurs. Chaque contrainte Ci est une paire: <porté, relation>
  - Avec porté un tuple de variables participant à la contrainte Ci
  - Avec relation définissant les valeurs possibles de ces variables

Définitions

- Une relation peut être définie de deux façons, par extension et par intension:
- Exemple: soit les variables X1 et X2 ayant le même domaine {A,B}, la contrainte imposant qu'elles soient différentes devient alors:
  - Par extension:

    <(X1,X2), [(A,B),(B,A)]>
  - Par intension:

    <(X1,X2), X1 ≠ X2>

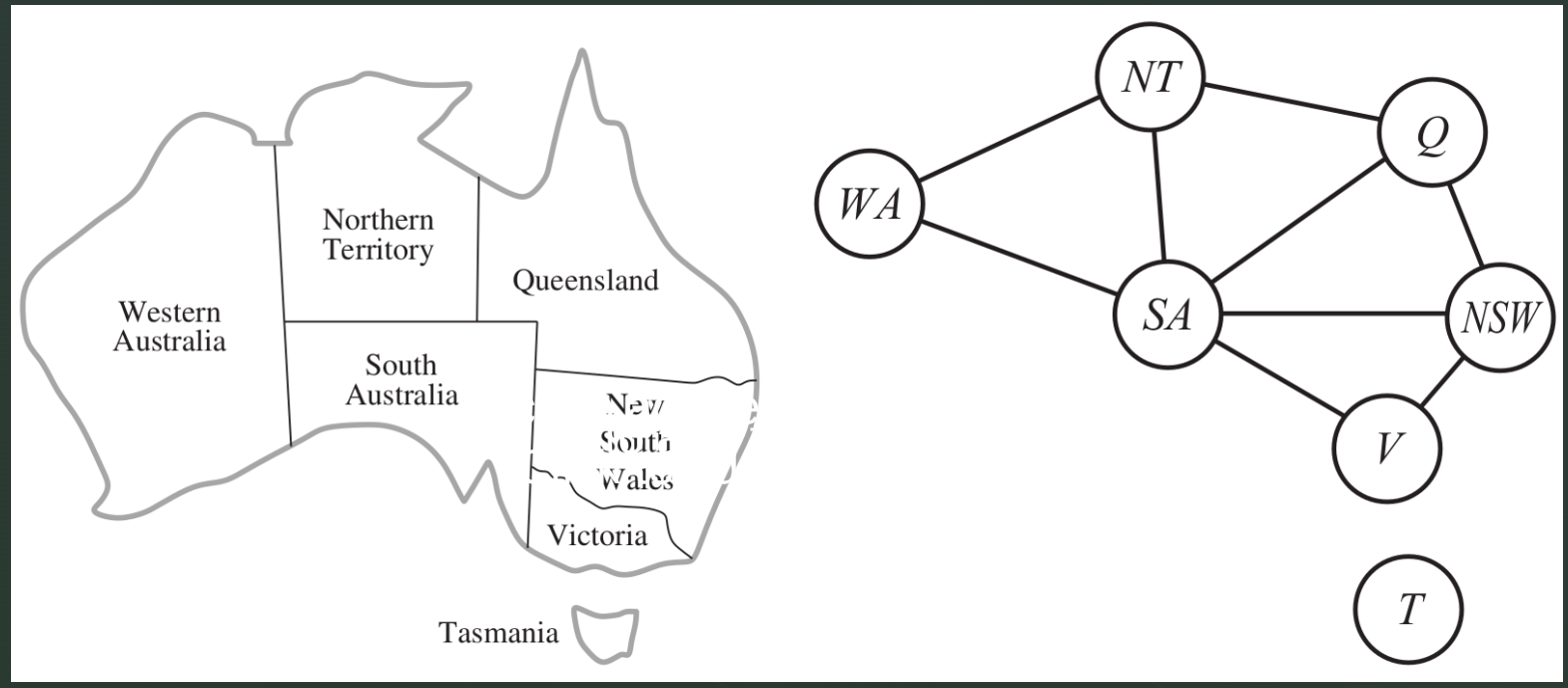
Définitions

- Afin de résoudre un CSP, il est nécessaire d'affecter des valeurs aux variables en respectant les contraintes.
- Une affectation qui ne viole aucune contrainte est dite cohérente.
- Une affectation est dite complète si chaque variable est affectée.
- La solution d'un CSP est une affectation cohérente et complète.

Différents types de contraintes

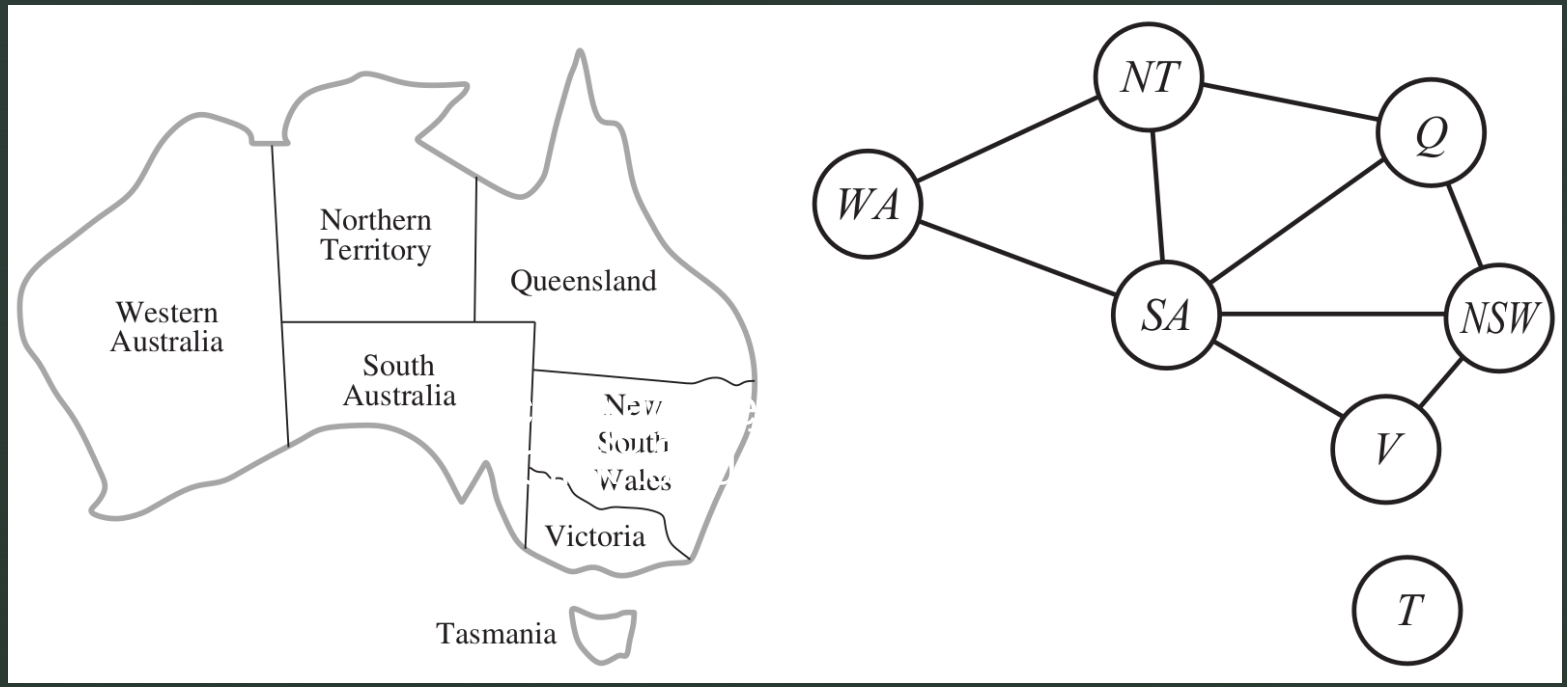
- Contrainte Unaire : contrainte sur une seule variable
  - Exemple : <(SA), SA ≠ vert>
- Contrainte Binaire : contrainte sur deux variables
  - Exemple : SA ≠ NSW
- Contrainte Ternaire :
  - Exemple : entre(X,Y, Z)
- Contrainte Globale (avec un nombre arbitraire de variables):
  - Exemple: tousDifférents(SA, WA, NT, Q, NSW, V)

Retour à l'exemple (3 couleurs possibles)



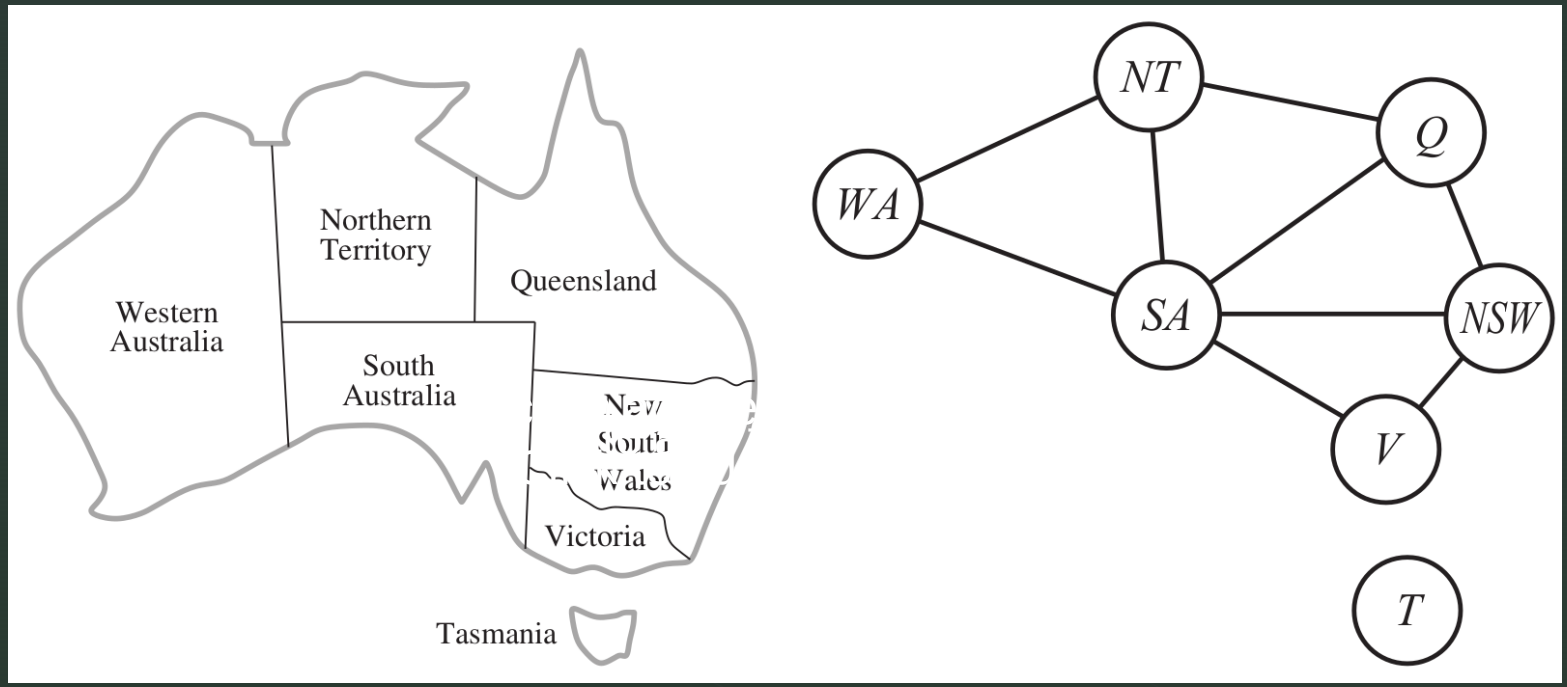
X=?, D=?

Retour à l'exemple



X = {WA, NT, Q, NSW, V, SA, T}

Retour à l'exemple



X = {WA, NT, Q, NSW, V, SA, T}

D = {rouge, vert, bleu}

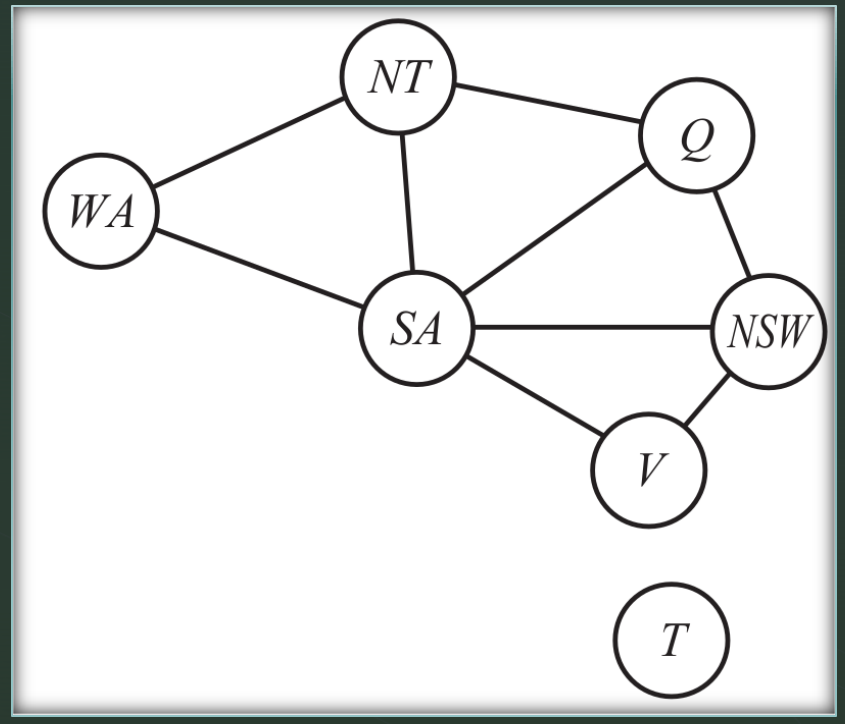
C = {SA ≠ WA, SA ≠ NT, SA ≠ Q, SA ≠ NSW, SA ≠ V, WA ≠ NT, NT ≠ Q, Q ≠ NSW, NSW ≠ V}

Pourquoi formuler un problème sous forme d'un CSP?

- 1) Répresentation naturelle/direct pour beaucoup de problèmes
  - Exemple: le problème des 8 reines déjà abordé
  - Pas besoin de développer une solution ad hoc, il suffit de formaliser les variables et contraintes et de demander au solveur de trouver une solution.
- 2) L'élagage du CSP peut le rendre plus puissant qu'une méthode cherchant dans l'espace d'états car il est possible d'éliminer rapidement de larges fractions de l'espace

Exemple d'élagage par CSP

- Une fois {SA = bleu} fixé au sait qu'aucun des voisins ne peut prendre la couleur bleu.
- Sans prise en compte de cette information il y a: $3^5 = 243$ affectations possibles pour les 5 voisins de SA
- Avec prise en compte de cette information il n'y a plus que $2^5=32$ possibilités!
- SOIT UNE REDUCTION DE 87% DE L'ESPACE DE RECHERCHE

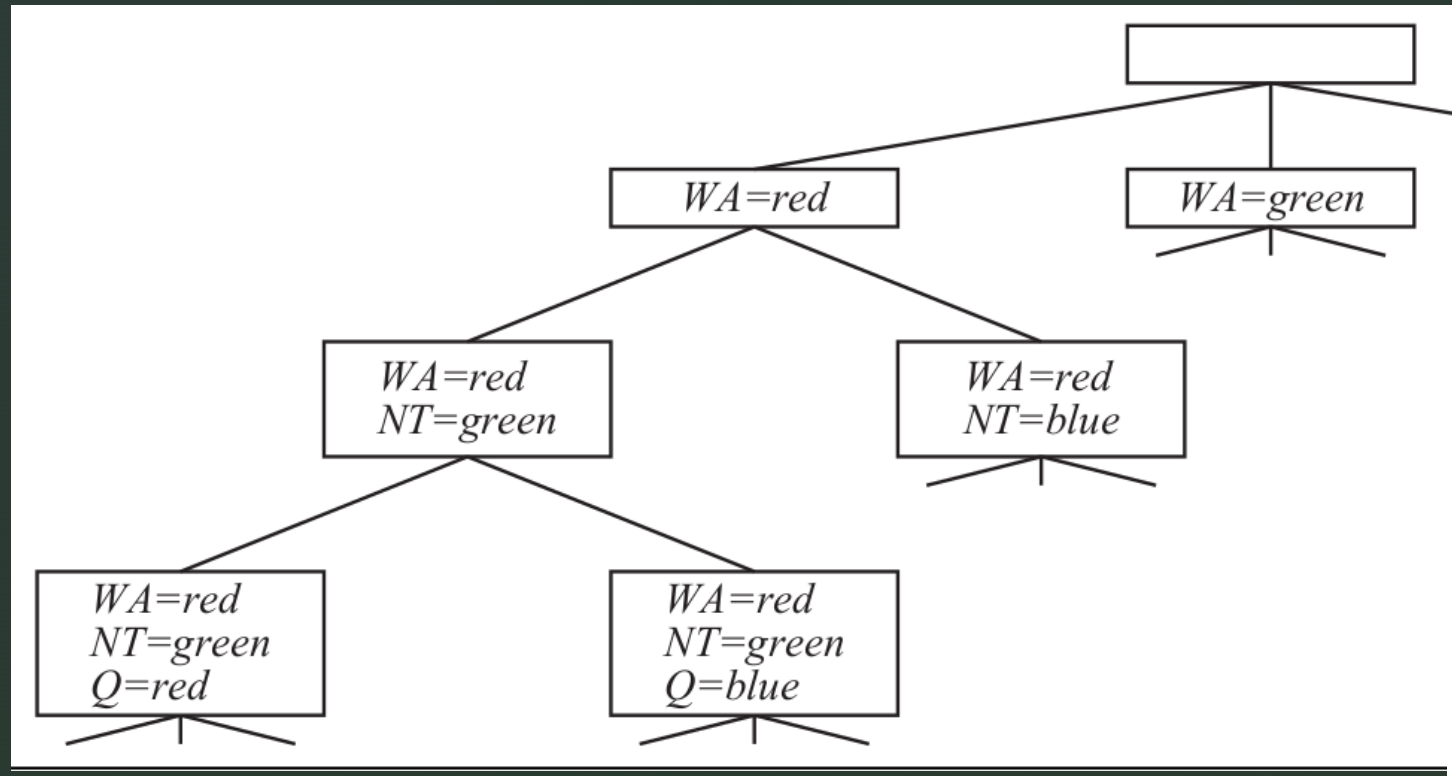


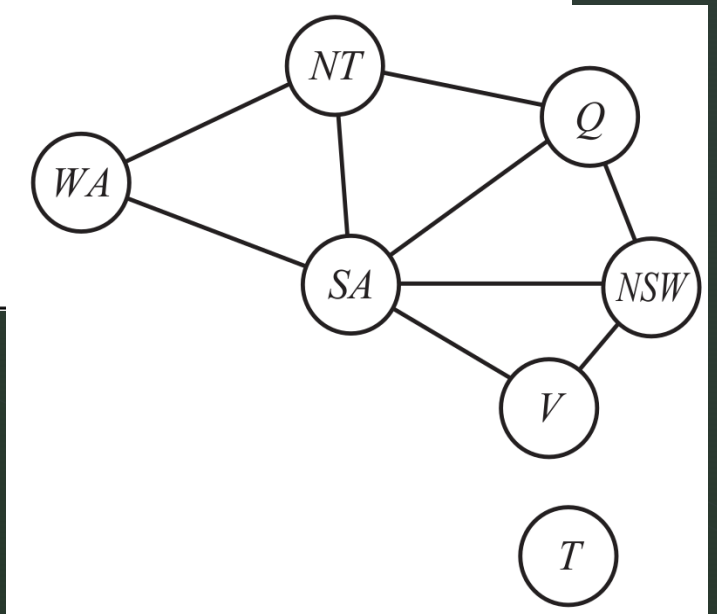
Backtracking Search (parcours en profondeur)

```
function BACKTRACKING-SEARCH(csp) returns a solution, or failure
    return BACKTRACK({ }, csp)

function BACKTRACK(assignment, csp) returns a solution, or failure
    if assignment is complete then return assignment
    var ← SELECT-UNASSIGNED-VARIABLE(csp)
    for each value in ORDER-DOMAIN-VALUES(var, assignment, csp) do
        if value is consistent with assignment then
            add {var = value} to assignment
            inferences ← INFERENCE(csp, var, value)
            if inferences ≠ failure then
                add inferences to assignment
                result ← BACKTRACK(assignment, csp)
                if result ≠ failure then
                    return result
        remove {var = value} and inferences from assignment
    return failure
```

Exemple de parcours par Backtrack



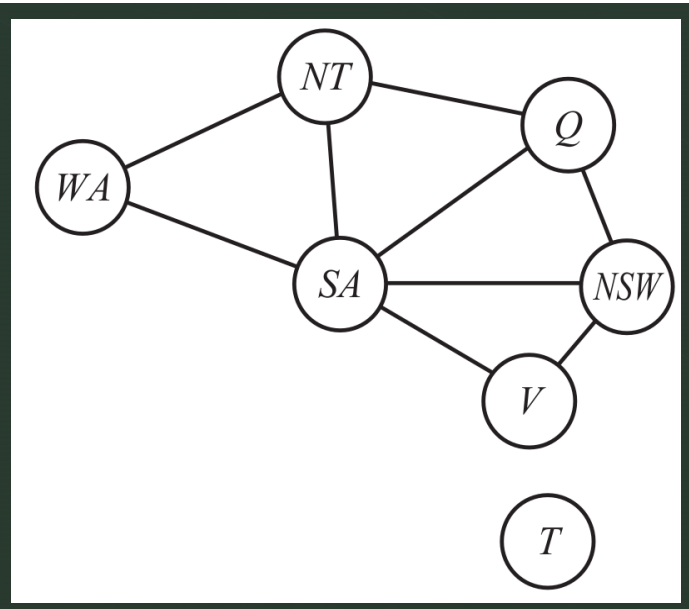


Heuristiques possibles

- L'ordre dans lequel les variables sont choisies influence la recherche:
- MRV : Minimum Remaining values : on privilégiera la variable avec le moins de valeurs possibles restantes
- Degré : on privilégiera la variable impliquée dans le plus de contraintes
- Valeurs moins contraintes : on choisira une valeur laissant plus de possibilités pour les variables suivantes

Forward Checking

	WA	*NT*	*Q*	*NSW*	*V*	*SA*	*T*
Initial domains	R G B	R G B	R G B	R G B	R G B	R G B	R G B
After *WA*=*red*	(R)	G B	R G B	R G B	R G B	G B	R G B
After *Q*=*green*	(R)	B	(G)	R B	R G B	B	R G B
After *V*=*blue*	(R)	B	(G)	R	(B)		R G B



Min-Conficts

function MIN-CONFLICTS(csp, max_steps) **returns** a solution or failure
  inputs: csp, a constraint satisfaction problem
    max_steps, the number of steps allowed before giving up

  $current \leftarrow$ an initial complete assignment for csp
  for i = 1 to max_steps **do**
    if $current$ is a solution for csp **then return** $current$
    $var \leftarrow$ a randomly chosen conflicted variable from csp.VARIABLES
    $value \leftarrow$ the value v for var that minimizes CONFLICTS(var, v, $current$, csp)
    set $var = value$ in $current$
  return *failure*

Exemple du nombre d'itérations par méthode

	Backtracking	Forward Checking	Min Conflicts
100-reines	40 000 000	40 000 000	4 000
états USA	1 000 000	2 000	64