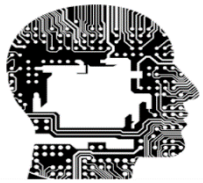
Université Constantine 2
جامعة قسنطينة 2

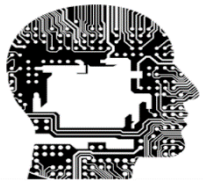


Systèmes Intelligents

CSPs (suite..)

Dr. NECIBI Khaled
Faculté des nouvelles technologies
Khaled.necibi@univ-constantine2.dz

Université Constantine 2
جامعة قسنطينة 2



Systèmes Intelligents

Les Problèmes à Satisfaction de Contraintes

Dr. NECIBI Khaled

Faculté des nouvelles technologies

Khaled.necibi@univ-constantine2.dz

Etudiants concernés

Faculté/Institut	Département	Niveau	Spécialité
Nouvelles technologies	IFA	Licence 3	Science de l'informatique SCI

CSP : Recherche en mode Backtracking (Backtracking Serch)

- Idée de base de Backtracking Search
- Représentation standard → état initial, fonction successeur
- `Select-Unassigned-Variable`, `Order-Domain-value` peuvent être utilisées pour implémenter des heuristiques générales
- Quand l'espace de recherche est très large, l'algorithme Backtracking Search n'est pas efficace
- Certaines améliorations sont possibles si on considère les points suivants :

Heuristiques pour la résolution des CSPs

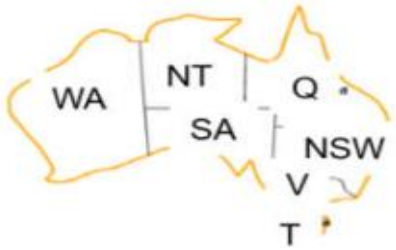
- Ordre des variables et des valeurs
- `var ← Select-Unassigned-Variable(variable[csp], assignment, csp);`
- Cette instruction permet de sélectionner une variable non traitée selon l'ordre établi dans `variable[csp]`
- Ceci conduit rarement à une recherche efficace
- Solution
  - Choisir la variable avec le nombre minimum de valeurs restantes
  - Minimum Remaining Value (MRV heuristic)
  - Connue aussi sous le nom de Most Constrained Variable

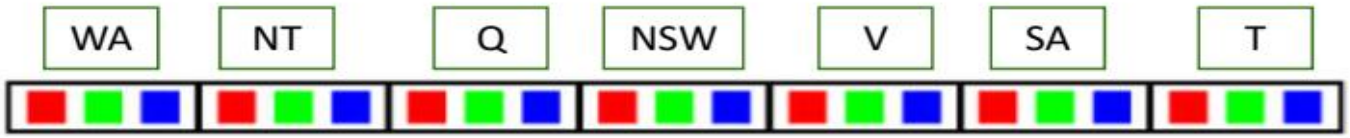
Heuristiques pour la résolution des CSPs

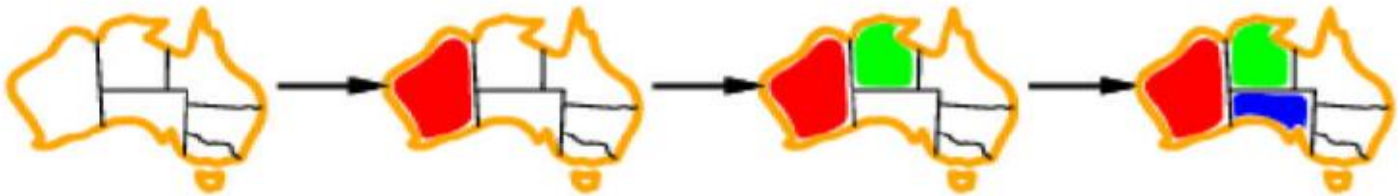
- Ordre des variables et des valeurs
- `var ← Select-Unassigned-Variable(variable[csp], assignment, csp);`
  - Dans quel ordre ces valeurs doivent être testées
- Remarque
  - Si une variable X à 0 valeurs restantes, l'heuristique MRV va sélectionner X
  - Un échec est immédiatement détecté
  - Gaspillage d'efforts pour tester une autre variable

Heuristiques pour la résolution des CSPs

- Exemple
- WA = Rouge, NT = Vert → SA = Bleu au lieu de traiter Q
- Après avoir traiter SA, les valeurs de Q, NSW et V sont définies
- La performance est meilleur que celle de l'algorithme de base





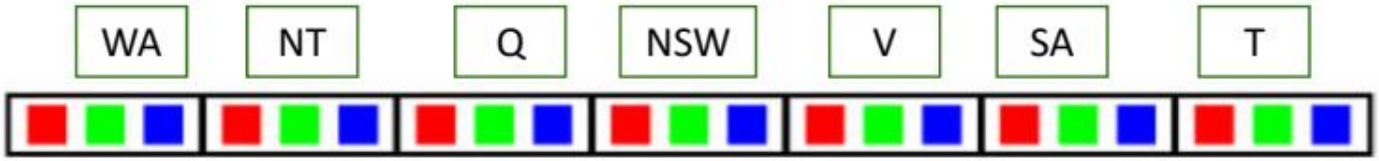


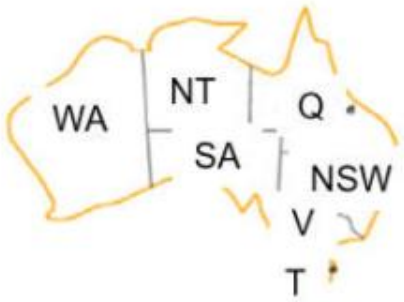
Heuristiques pour la résolution des CSPs

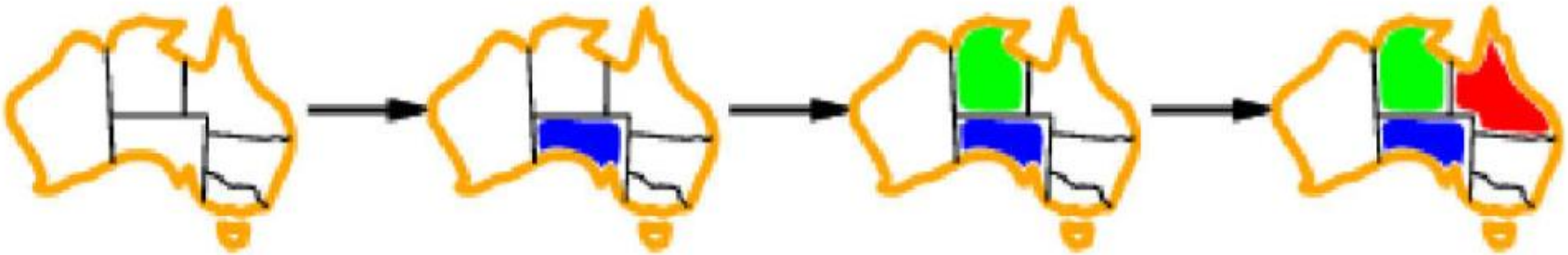
- Exemple
- Si le critère précédent donne des variables avec le même nombre de valeurs consistants restantes
  - → Choisir la variable ayant le plus de contraintes impliquant des variables non encore assignés → heuristique à degré
- Appelé : Degree Heuristic
- Exemple : degree heuristic de SA est 5 (Most Constraining Variable)
- C = {SA ≠ WA, SA ≠ NT, SA ≠ Q, SA ≠ NSW , SA ≠ V, WA ≠ NT, NT ≠ Q, Q ≠ NSW, NSW ≠ V}

Heuristiques pour la résolution des CSPs

- Exemple
- Exemple : degree heuristic de SA est 5 (Most Constraining Variable)
- C = {SA ≠ WA, SA ≠ NT, SA ≠ Q, SA ≠ NSW , SA ≠ V, WA ≠ NT, NT ≠ Q, Q ≠ NSW, NSW ≠ V}

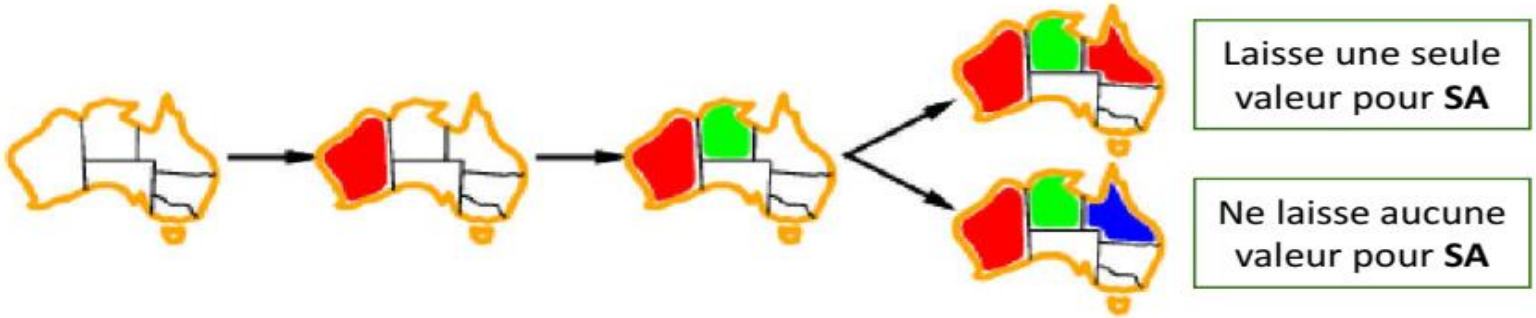






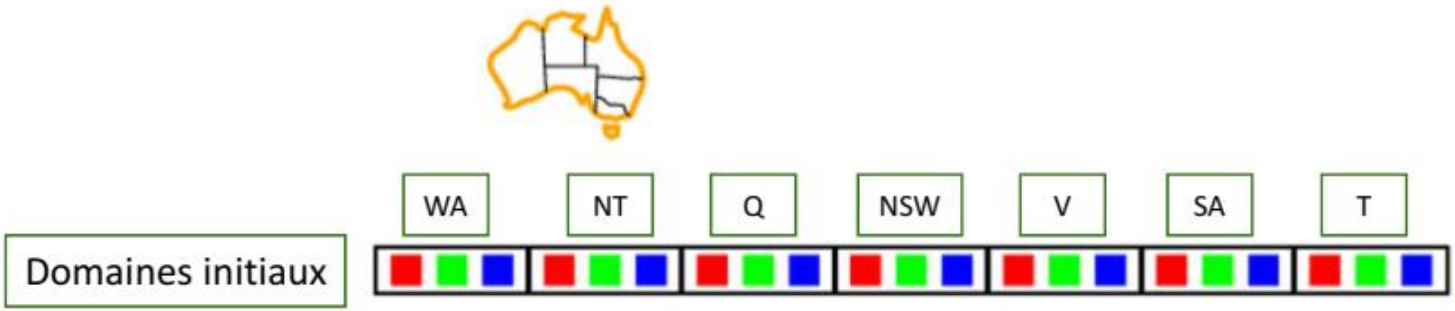
Heuristiques pour la résolution des CSPs

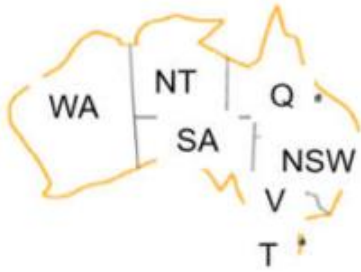
- Valeur moins contraignante (Least constraining value)
  - Une fois qu'une variable est sélectionnée, dans quel ordre faut-il essayer ses valeurs ?
- Solution
  - Choisir la valeur la moins contraignante qui laisse plus de flexibilité aux autres variables



Heuristiques pour la résolution des CSPs

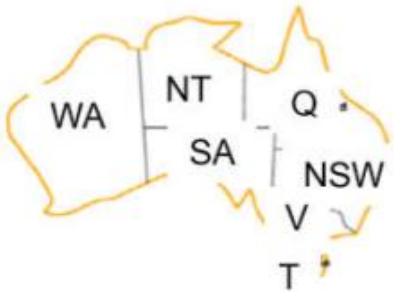
- Vérification anticipative (forward-checking)
- Vérifier chaque variable Y connectée à X
- Enlever de la variable Y les valeurs dont les contraintes ne sont pas vérifiées i.e. non consistante avec la valeur choisie pour X
- Exemple

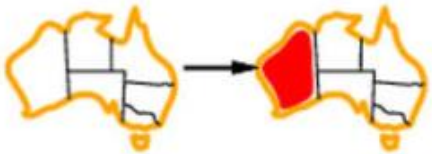


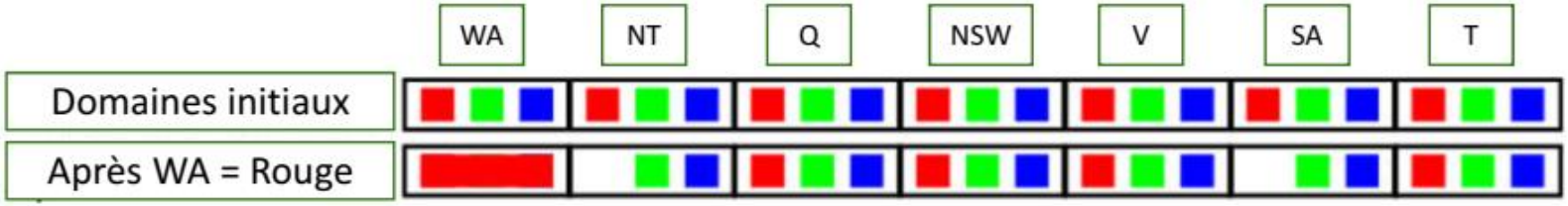


Heuristiques pour la résolution des CSPs

- Vérification anticipative (forward-checking)
- Exemple : Choix de départ WA
- Avec l'assignation WA = Rouge
- Résultat

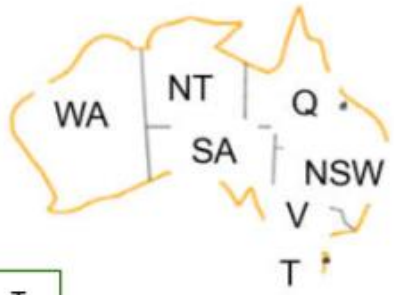






Heuristiques pour la résolution des CSPs

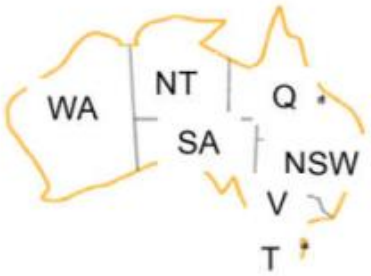
- Vérification anticipative (forward-checking)
- Ensuite soit la variable Q choisie
- Avec l'assignation Q = Vert
- Résultat



	WA	NT	Q	NSW	V	SA	T
Domaines initiaux							
Après WA = Rouge							
Après Q = Vert							

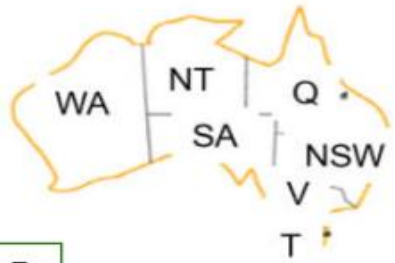
Heuristiques pour la résolution des CSPs

- Vérification anticipative (forward-checking)
- Ensuite soit la variable V choisie
- Avec l'assignation V = Bleu
- Résultat



Heuristiques pour la résolution des CSPs

- Vérification anticipative (forward-checking)
- Après WA = Rouge et Q = Vert, NT et SA ont une seule valeur possible qui est Bleu
- Froward Checking ne peut pas détecter toutes les inconsistances
- i.e. Il ne peut pas propager l'effet des affectations sur les toutes les variables



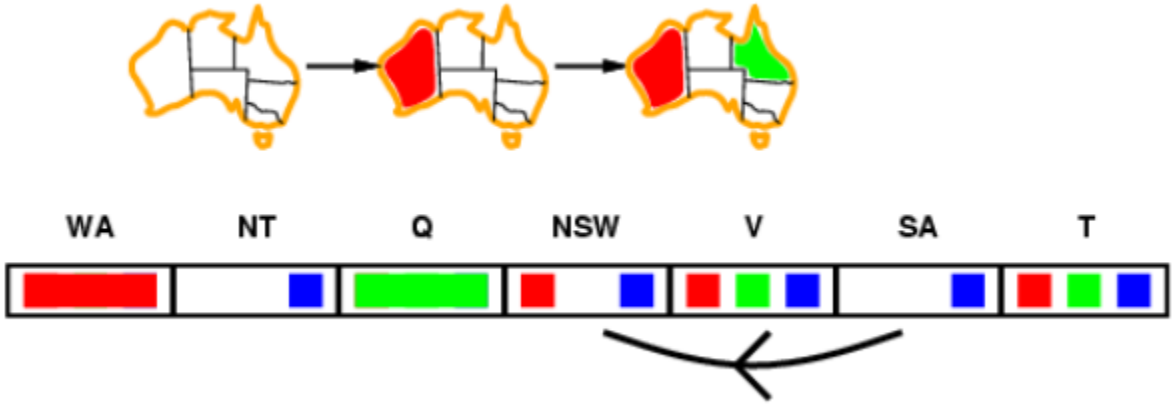
	WA	NT	Q	NSW	V	SA	T
Domaines initiaux							
Après WA = Rouge							
Après Q = Vert							

Heuristiques pour la résolution des CSPs

- Consistance d'Arc (AC)
- Solution : Propager les effets d'une affectation sur les autres variables → Consistance d'Arc
- La Consistance d'Arc (Arc Consistency) est la forme de propagation de contraintes la plus simple
- La Consistance d'Arc (AC) est plus forte que l'algorithme FC
- L'arc X → Y est consistent si et seulement si :
  - Pour tout valeur x de X il existe au moins une valeur y consistante avec x

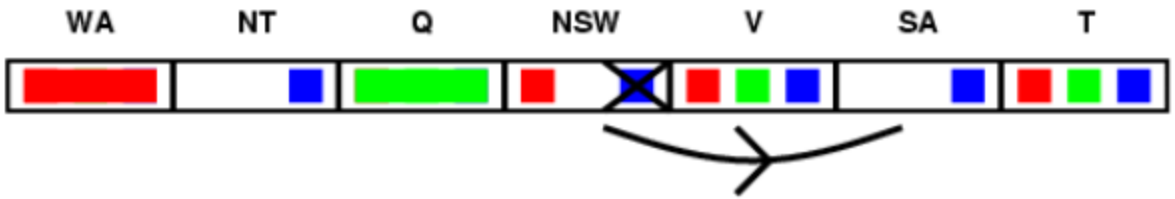
Heuristiques pour la résolution des CSPs

- Consistance d'Arc (AC)
- Solution : Propager les effets d'une affectation sur les autres variables
- L'arc X → Y est consistent si et seulement si :
  - Pour tout valeur x de X il existe au moins une valeur y permise de Y



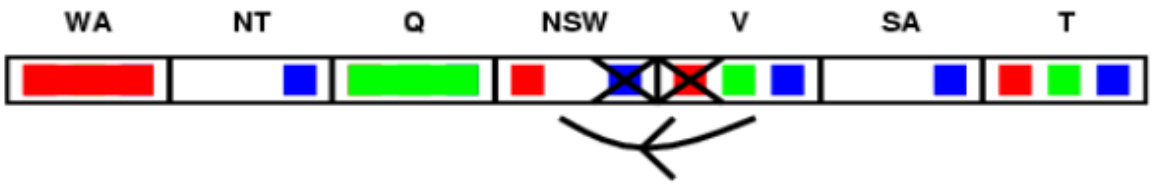
Heuristiques pour la résolution des CSPs

- Consistance d'Arc (AC)
- Solution : Propager les effets d'une affectation sur les autres variables
- L'arc X → Y est consistent si et seulement si :
  - Pour tout valeur x de X il existe au moins une valeur y permise de y
- Si une variable perd une valeur, ses variables voisines doivent être revérifiées (notifiées)



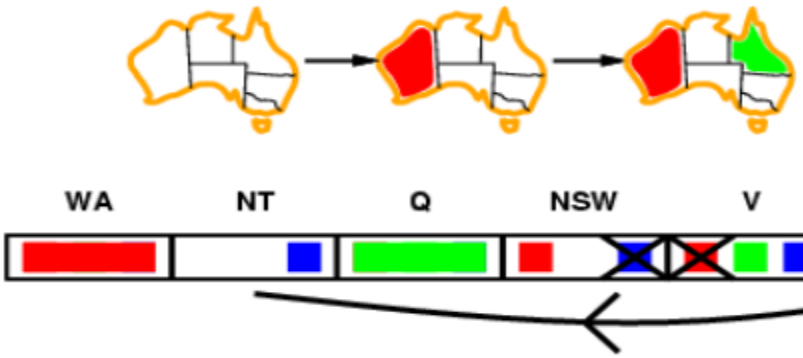
Heuristiques pour la résolution des CSPs

- Consistance d'Arc (AC)
- Solution : Propager les effets d'une affectation sur les autres variables
- L'arc X → Y est consistent si et seulement si :
  - Pour tout valeur x de X il existe au moins une valeur y permise de y
- Si une variable perd une valeur, ses variables voisins doivent être revérifiées (notifiées)



Heuristiques pour la résolution des CSPs

- Consistance d'Arc (AC)
- Solution : Propager les effets d'une affectation sur les autres variables
- L'arc X → Y est consistent si et seulement si :
  - Pour tout valeur x de X il existe au moins une valeur y permise de y
- Si une variable perd une valeur, ses variables voisins doivent être revérifiées (notifiées)



Heuristiques pour la résolution des CSPs

Arc Consistency 3 (AC-3)

```
Function AC-3(csp) return CSP, Possibly with reduced domains

Inputs : csp, a binary CSP with variables {X1, X2,…., Xn}
Variables Local : queue, a queue of arcs, initially all the
arcs in csp

      While queue is not empty do
      (Xi, Xj) ← Remove-first(queue)
      if Remove-Inconsistent-Values(Xi, Xj) then
      for each Xk in neighbors[Xi]-{Xi} do
      add(Xk, Xi) to queue
```

Heuristiques pour la résolution des CSPs

Arc Consistency 3 (AC-3)

```
Function Remove-Inconsistent-Values(Xi, Xj) returns true if
we remove a value {

Removed ← false;
   for each x in Domain[Xi] do
        if no value y in Domain[Xi] allows (x, y) to satisfy
   the        constraints between Xi and Xj then
                                   delete x from Domain[Xi];
                                   Removed ← true;

Return Removed;
}
```

Heuristiques pour la résolution des CSPs

- Conclusion
- CSP : De quoi s'agit-il ? : Variables, domaines de valeurs et contraintes
- Objectif : Un état final dont tout les variables sont affectées et que tout les affectations (ou les assignations) sont consistantes
- Comment ? La recherche se fait en utilisant une formulation incrémentale ou complète
- Formulation Incrémentale : Backtracking Search amélioré :
  - Avec des heuristiques MRV Minimum Remaining Values (ou Most Constraining Variable) et Least Constrained Value (la valeur de la variable la moins contraignante)
  - Avec l'utilisation de la vérification anticipative ou Forward Checking
  - Avec la propagation de contraintes ou l'algorithme AC3
- Formulation complète : Une recherche locale peut être effectuée