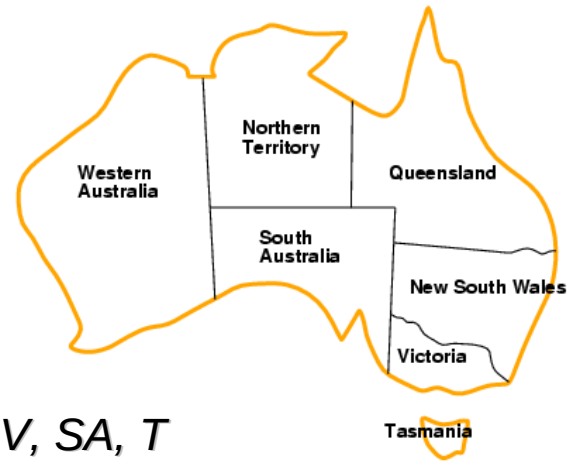
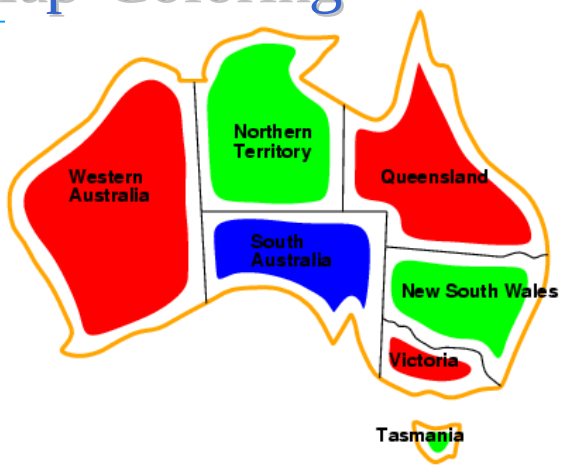
Esempio: Map-Coloring



- variabili *WA, NT, Q, NSW, V, SA, T*
- Domini D_i = {red,green,blue}
- vincoli: regioni adiacenti devono avere colori diversi
- e.g., WA ≠ NT, or (WA,NT) in {(red,green),(red,blue),(green,red), (green,blue),(blue,red),(blue,green)}

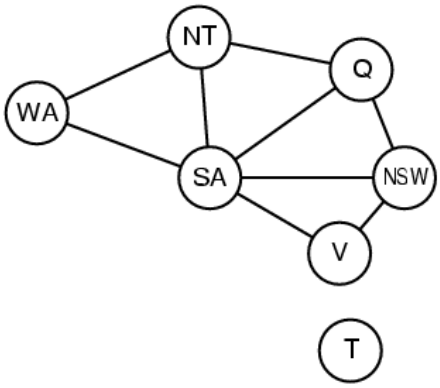
Esempio: Map-Coloring



- Le soluzioni sono assegnamenti completi e consistenti, e.g., WA = red, NT = green,Q = red,NSW = green,V = red,SA = blue,T = green

Constraint graph

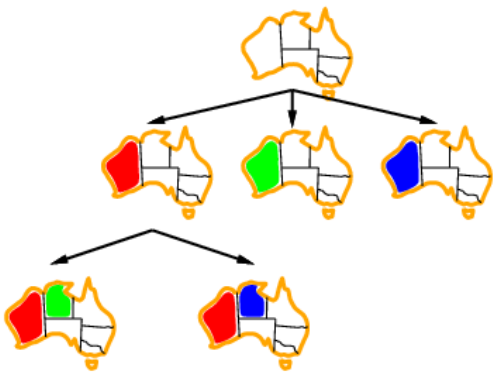
- CSP binario: ogni vincolo si correla a due variabili
- Constraint graph: nodi sono variabili, archi sono vincoli



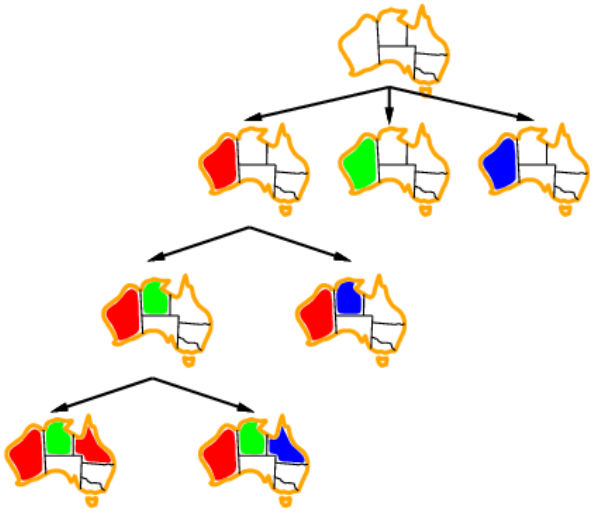
Backtracking : esempio

Backtracking : esempio

Backtracking : esempio



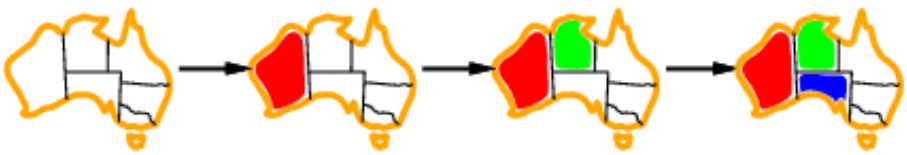
Backtracking : esempio



Most constrained variable

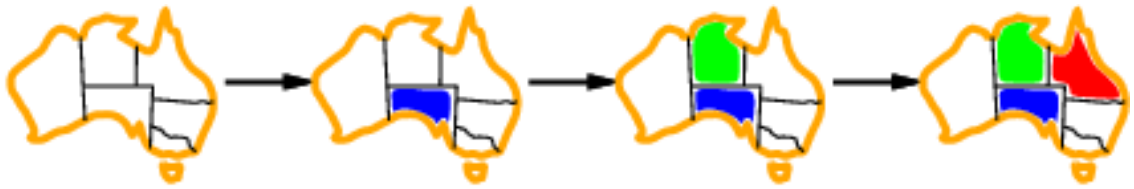
- Most constrained variable:
  Scegli la variabile con meno valori ammessi

- minimum remaining values (MRV) o first fail

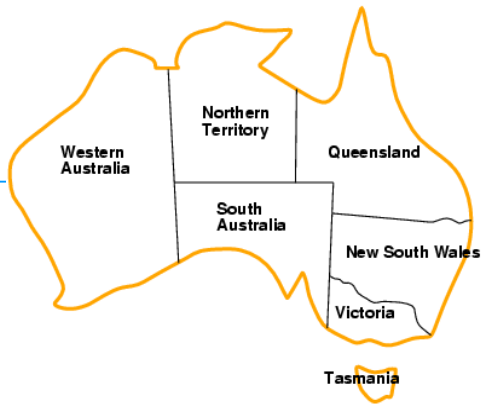


Most constraining variable

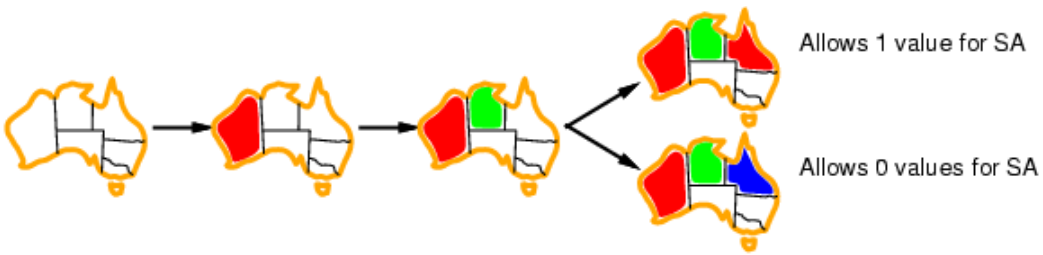
- Most constraining variable:
  - Scegli la variabile con il maggior numero di vincoli sulle altre variabili rimaste (si applica quando ci sono piu' variabili con uguale numero di valori nel dominio).



Least constraining value

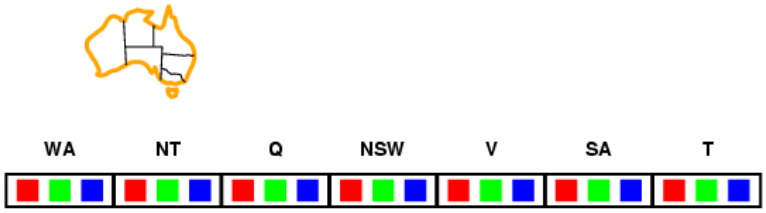


- Data una variabile, scegli il valore meno vincolante, cioe' che rende impossibili o inconsistenti meno assegnamenti dellle variabili rimaste.



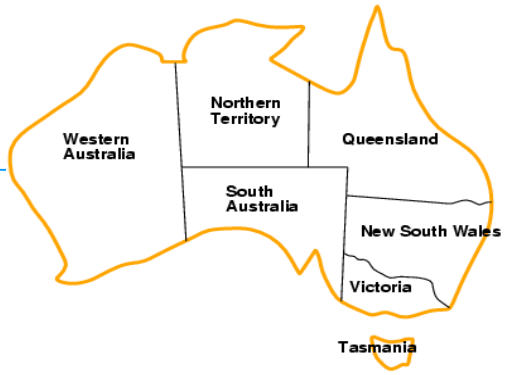
Forward checking

- Idea:
  - Tenere traccia dei valori legali per le variabili rimanenti
  - Fallire quando non ci sono piu' valori legali.

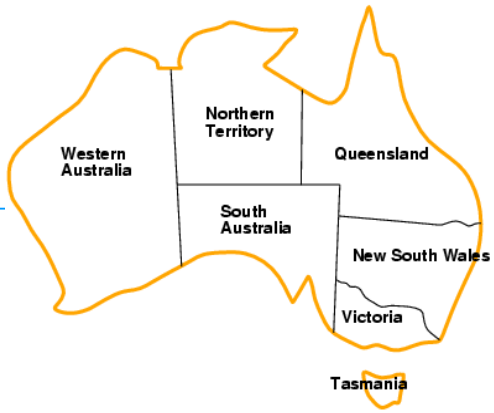


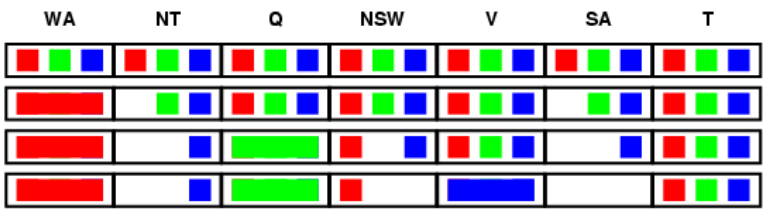
Forward checking

Forward checking



Forward checking





Constraint Propagazione di vincoli e forward checking

- Forward checking propaga l'informazione da variabili assegnate a variabili non assegnate, ma non consente di individuare subito situazioni inconsistenti.
- NT e SA non possono essere entrambe blue
- Constraint propagation fra variabili non assegnate!

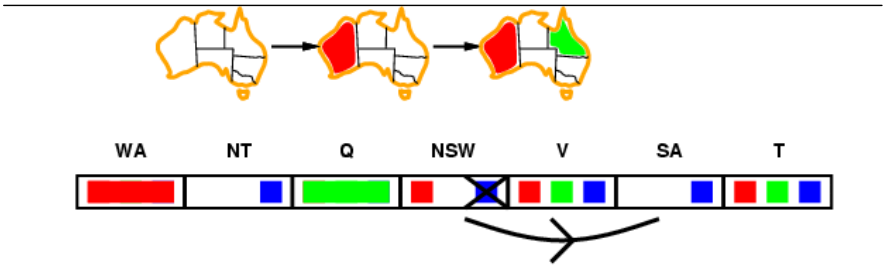
Arc consistency

- La piu' semplice forma di propagazione, rende ogni arco consistente.

- *X* → *Y* e' consistente se e solo se

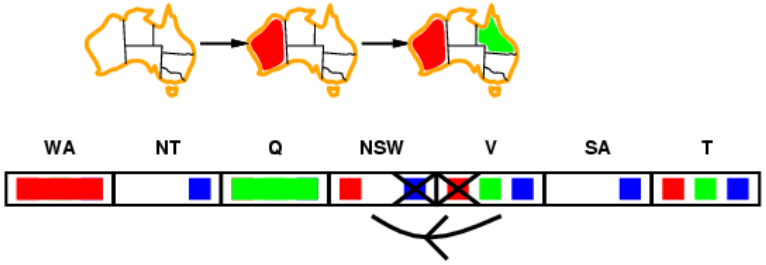
  Per ogni valore di *X* *c'e' almeno un* valore ammissibile per Y

Arc consistency

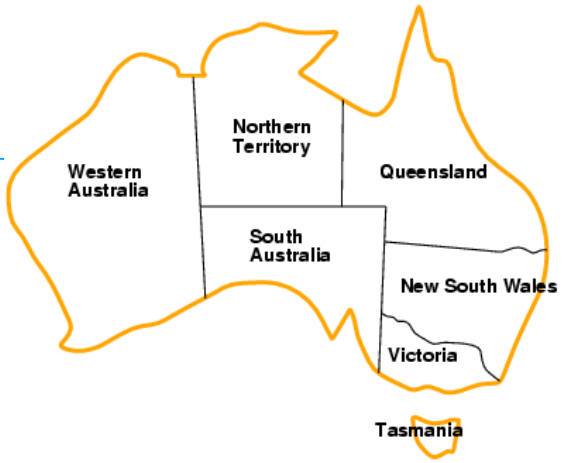


Arc consistency

- SE *X* perde un valore, devo ricontrollare I "vicini" di X.

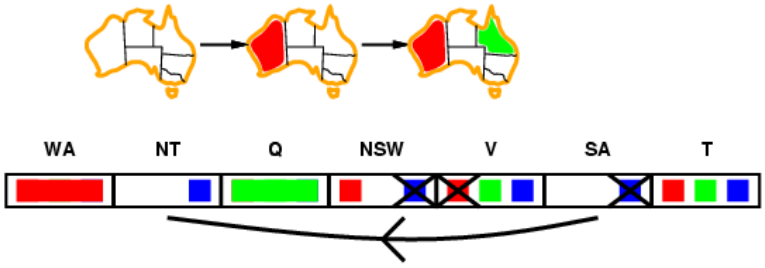


Arc consistency



I fallimenti sono trovati con l'Arc consistency prima che con il forward checking

Puo' girare come pre-processore, oppure dopo ogni assegnamento.



Arc-Consistency: Algoritmo

```
function AC-3( csp) returns the CSP, possibly with reduced domains
  inputs: csp, a binary CSP with variables {X1, X2, ..., Xn}
  local variables: queue, a queue of arcs, initially all the arcs in csp

  while queue is not empty do
    (Xi, Xj) ← REMOVE-FIRST(queue)
    if RM-INCONSISTENT-VALUES(Xi, Xj) then
      for each Xk in NEIGHBORS[Xi] do
        add (Xk, Xi) to queue

function RM-INCONSISTENT-VALUES( Xi, Xj) returns true iff remove a value
  removed ← false
  for each x in DOMAIN[Xi] do
    if no value y in DOMAIN[Xj] allows (x,y) to satisfy constraint(Xi, Xj)
      then delete x from DOMAIN[Xi]; removed ← true
  return removed
```