Recherche locale pour les CSP

- Le chemin à la solution est sans importance
  - on peut utiliser une méthode de recherche locale (*hill-climbing*, etc.)
  - on peut travailler avec des états qui sont des assignations complètes (compatibles ou non)

- Le problème de la recherche locale est qu'elle peut tomber dans des optima locaux

- Par contre, pour *N-Queens*, l'algorithme *min-conflicts* fonctionne étonnamment bien
  - fonction objectif : on minimise le nombre de conflits
  - ressemble à *hill-climbing*, mais avec un peu de stochasticité

Algorithme *min-conflicts*

Algorithme MIN-CONFLICTS (*csp, nb_iterations*)

1. *assignation* = une assignation aléatoire complète (probablement pas compatible) de *csp*
2. pour *i* = 1 ... *nb_iterations*
    3. si *assignation* est compatible, retourner *assignation*
    4. *X* = variable choisie aléatoirement dans VARIABLES(csp)
    5. *v* = valeur dans DOMAINE(*X*, *csp*) satisfaisant le plus de contraintes de *X*
    6. assigner (*X* = *v*) dans *assignation*
7. retourner faux

- Peut résoudre un problème *1,000,000-Queens* en 50 étapes!
- La raison du succès de la recherche locale est qu'il existe plusieurs solutions possibles, « éparpillées » dans l'espace des états
- A été utilisé pour générer les horaires d'observation du *Hubble Space Telescope* (roule en 10 minutes, plutôt que 3 semaines!)