Recherche locale

Rym Guibadj

LISIC, ULCO, EILCO

Objectif du cours

- La différence entre une recherche heuristique (par exemple A^*) et une recherche locale
- La méthode d'escalade : *hill-climbing*
- La méthode recuit simulé : *simulated annealing*
- Les algorithmes génétiques

Recherche locale

Définition

Une recherche locale décrit une technique algorithmique qui explore l'espace des solutions de manières **séquentielle** passant à chaque ?étape de **la solution courante** à une **solution** ”**voisine**”. Son objectif est de trouver rapidement une solution acceptable.

Principe

- cherche à optimiser une fonction objectif (maximiser ou minimiser)
- garde juste certains noeuds visités en mémoire
  - *hill-climbing* garde juste un noeud (le noeud courant)
  - un algorithme génétique garde un ensemble de noeuds (population)
- une recherche locale ne garantie pas de solution optimale

Recherche locale Vs A^*

	A^*	Recherche locale
fonction	$isGoal(n)$: fonction identifiant le noeud but	$f(n)$: fonction objectif à optimiser
solution	un chemin	juste le noeud optimal
mémoire	les noeuds rencontrés sont stockés pour éviter de les revisiter	l'espace d'états est trop grand pour enregistrer les noeuds visités

Motivation pour une recherche locale

une recherche locale est appliquée lorsqu'il y a une fonction objectif à optimiser, la solution recherchée est un noeud optimal et non pas le chemin qui y mène et que l'espace d'états est trop grand pour enregistrer les noeuds visiter

Recherche locale

- Problème : placer *N* reines sur un échiquier de taille *N*x*N* de sorte que deux reines ne s'attaquent pas mutuellement : jamais deux reines sur la même diagonale, ligne ou colonne.

Méthode *hill-climbing*

Principe :

- le noeud courant est initialisé avec le noeud initial
- on génére les sucesseurs (les voisins) immédiats du noeud courant
- le meilleur voisin n' avec $f(n') > f(n)$ devient le noeud courant
- s'il n'existe aucun noeud voisin n' meilleur que n, l'algorithme s'arrête et retourne le noeud courant comme solution

Méthode *hill-climbing*

Exemple : soit la fonction objectif suivante, définie pour les entiers de 1 à 16

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$f(n)$	4	6	15	5	3	2	4	5	6	7	8	10	9	8	7	3

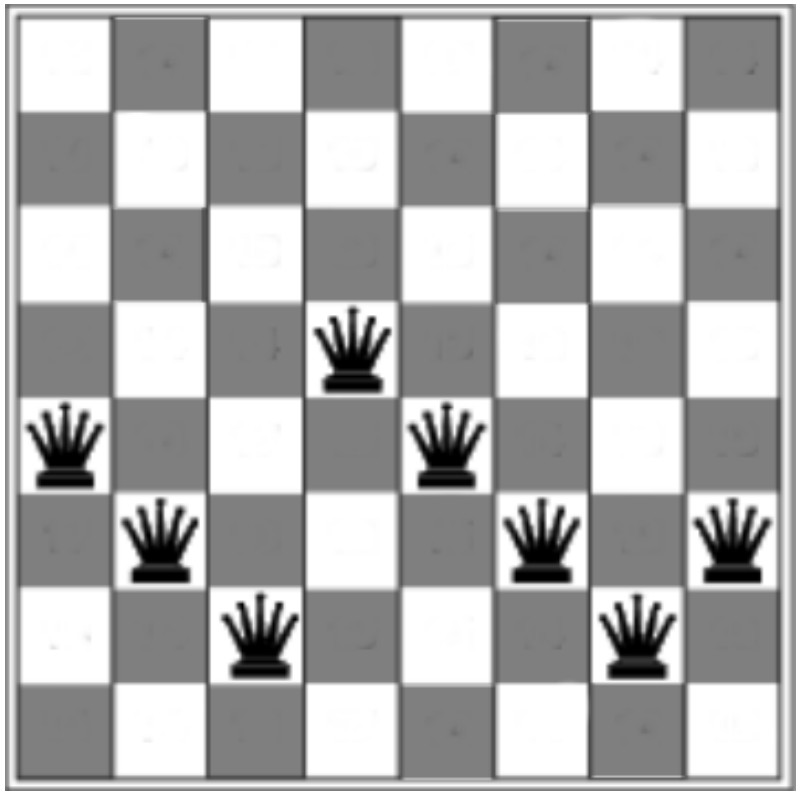
Quelle valeur de n trouverait la méthode hill-climbing si la valeur initiale de n était 6 et que les successeurs (voisins) utilisés étaient $n-1$ (seulement si $n > 1$ et $n+1$ (seulement si $n < 16$) ?)

Méthode *hill-climbing*

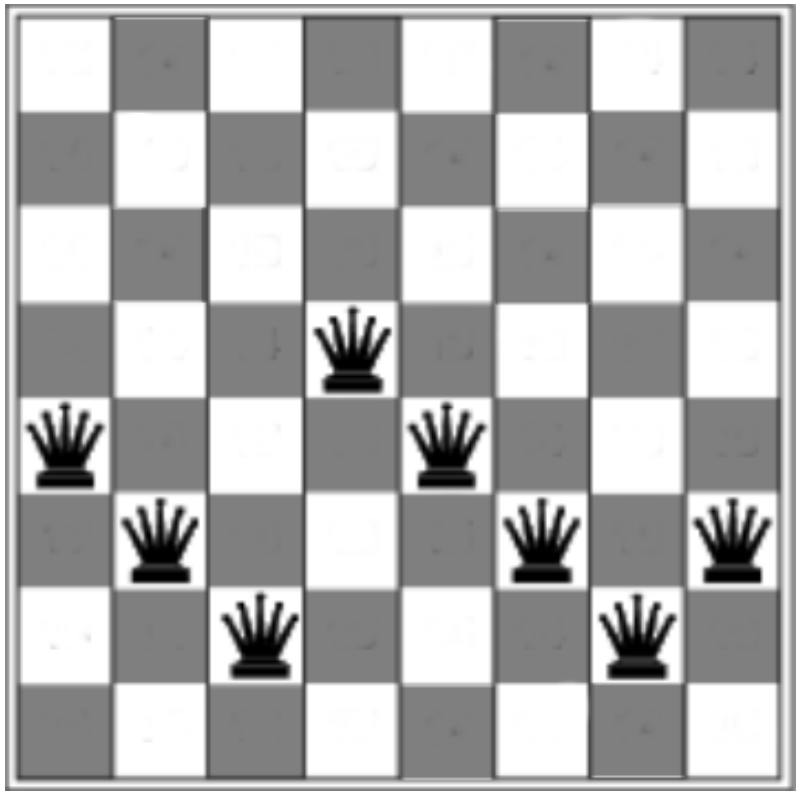
Exemple : *N-Queens*
Placer *N* reines sur un échiquier de taille *N*x*N* de sorte que deux reines ne s'attaquent pas mutuellement (jamais deux reines sur la même diagonale, ligne ou colonne).

Méthode *hill-climbing*

- n : une configuration de l'échiquier avec 8 reines
- $f(n)$: nombre de paires de reines qui s'attaquent mutuellement directement ou indirectement dans la configuration n
- On cherche à minimiser $f(n)$



Méthode *hill-climbing*

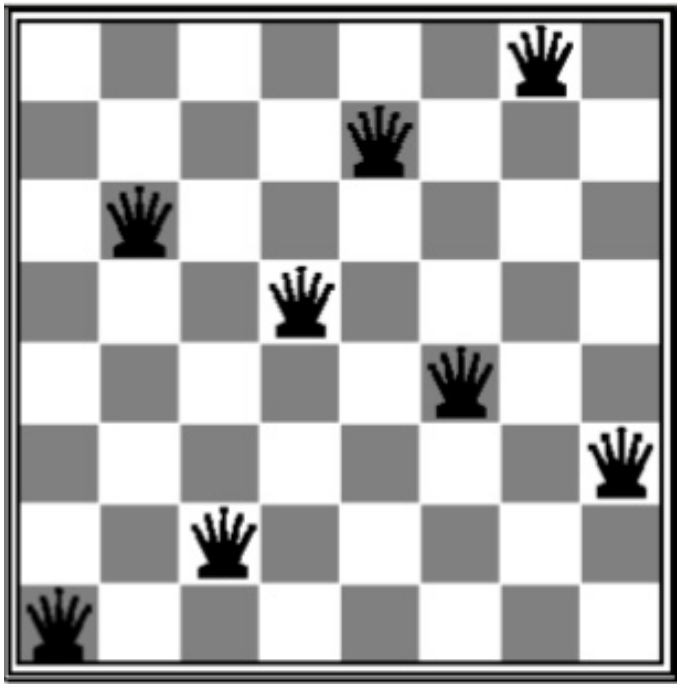


- Quelle est la valeur de la fonction objectif de la configuration *n* ci dessus ?
- Combien de successeurs immédiats existent pour la configuration *n* ?
- Quels sont les meilleurs successeurs ? C'est quoi leurs fonctions objectif ?

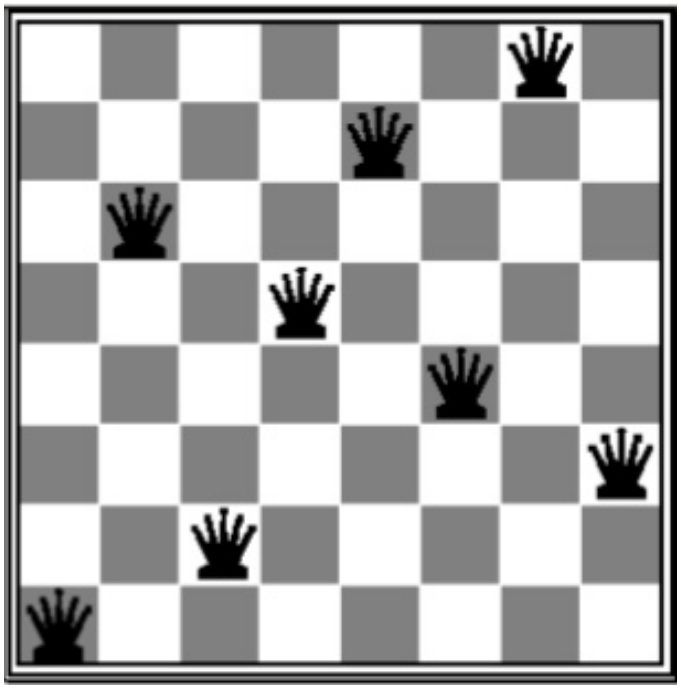
Méthode *hill-climbing*

- La valeur de la fonction objectif de la configuration $f(n) = 17$
- Il y a $7 * 8 = 56$ successeurs immédiats de la configuration n
- Les meilleurs successeurs ont une valeur égale à 12

Méthode *hill-climbing*



Méthode *hill-climbing*



Exemple d'un minimum local avec $f(n) = 1$

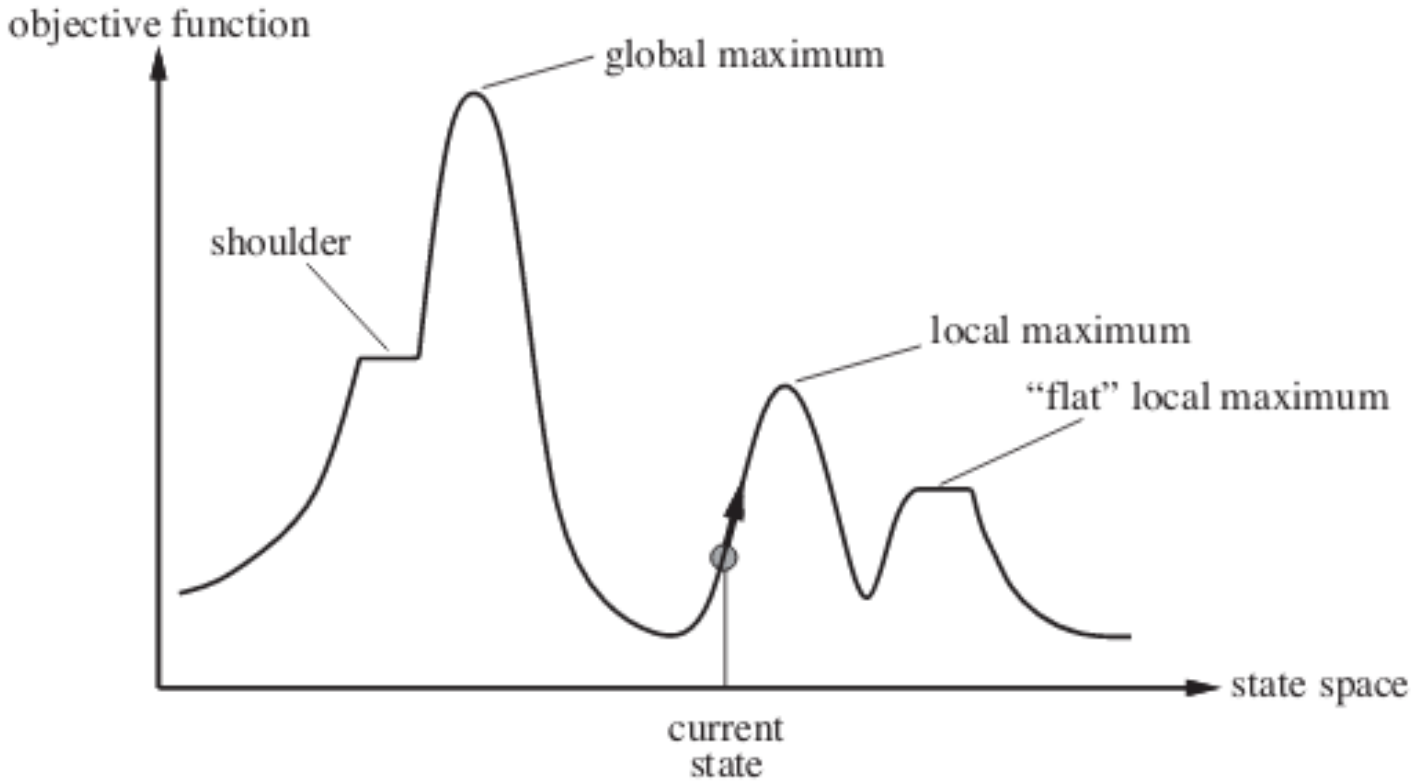
Méthode *hill-climbing*

Algorithme :

Algorithm *hill-climbing*

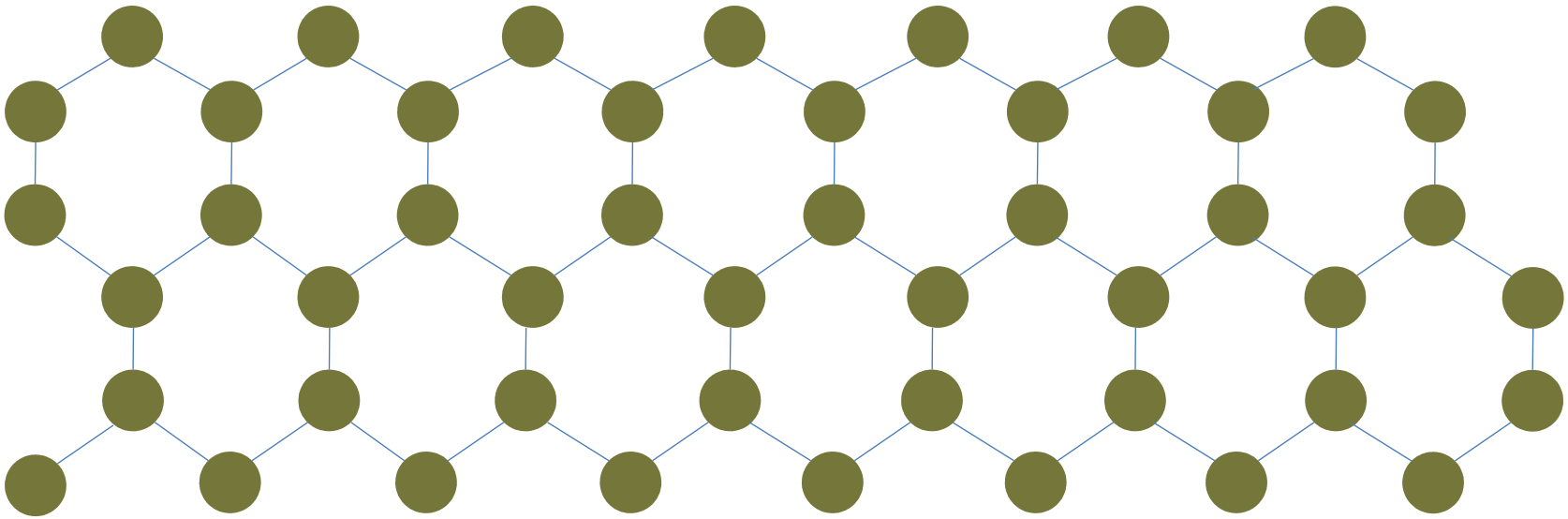
Argument : noeud initial
Sortie : noeud solution
Initialiser noeud courant *n* avec le noeud initial
Boucler
    n' = meilleurVoisin(n)
    Si $f(n') \geq f(n)$ **alors**
        n = n '
    Sinon
        Retourner n
    Fin si
Fin Boucler

Méthode *hill-climbing*



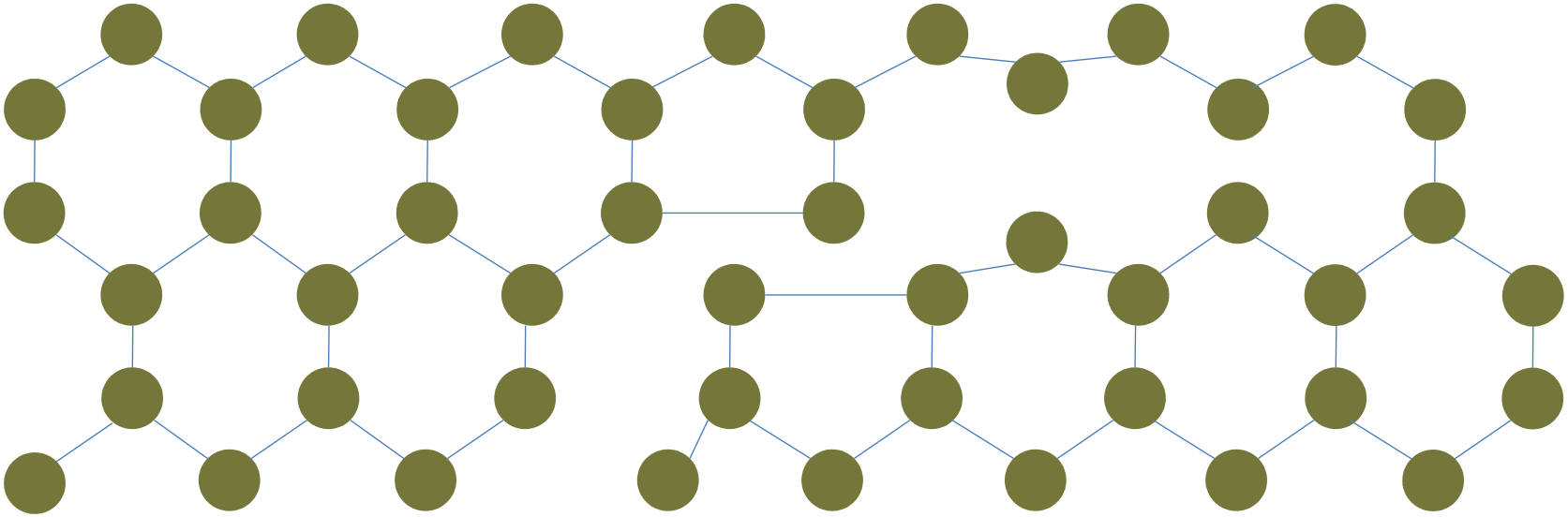
Méthode *simulated annealing* (recuit simulé)

- Un procédé issu de la métallurgie qui permet de trouver les optima d'une fonction.
- Un métal possède une structure cristalline : les atomes y sont assemblés de manière (plus ou moins) régulière.



Méthode *simulated annealing* (recuit simulé)

- Si le métal a refroidi trop vite ou s'il a été sa structure cristalline peut présenter des défauts (comme des fractures).



Méthode *simulated annealing* (recuit simulé)

- Le recuit consiste à réchauffer le métal pour redonner de l'énergie à ses atomes qui ≪ sortent ≫ de leur position puis à le laisser refroidir lentement pour permettre aux atomes de revenir vers de meilleures positions.

Méthode *simulated annealing* (recuit simulé)

Motivation

Méthode inspirée d'un processus utilisé en métallurgie. Alterner des cycles de refroidissement lent et de réchauffage (recuit) pour minimiser l'énergie du matériau.

Principe

- au lieu de prendre le meilleur voisin immédiat, on va choisir **un moins bon voisin** avec une certaine **probabilité**
- la probabilité de prendre un moins bon voisin est élevée au début, puis diminue graduellement durant la recherche
- la diminution des probabilités est définie à l'aide d'un schéma de "température" en ordre décroissant.

Méthode *simulated annealing* (recuit simulé)

Algorithm *Simulated-Annealing*

Argument : noeud initial , T_0
Sortie : noeud solution
Initialiser n = noeud initial
Initialiser $t = T_0$
Tant que $t > 0$ **faire**
    $n' =$ voisin aléatoire de n
    $\Delta E = f(n') - f(n)$
    Si $\Delta E > 0$ **alors**
      $n = n'$
    Sinon
      $n = n'$ avec une probabilité de $e^{\Delta E/t}$
    Fin Si
    Diminuer t
Fin Tant que
retourner n

Recherche taboue (*Tabu Search*)

Le recuit simulé

- + minimise le risque d'être piégé dans des optima locaux
- - n'élimine pas la possibilité d'osciller indéfiniment en revenant à un noeud antérieurement visité

Idée

- enregistrer les noeuds visités $\rightarrow$ on revient à l'algorithme A^*
- impraticable si l'espace d'état est trop grand

Recherche taboue

- sauvegarde en mémoire seulement les k derniers noeuds visités
- la liste des k noeuds visités est appelée **liste taboue**
- le paramètre k est choisi empiriquement
- elle n'élimine pas les oscillations mais les réduit

Algorithme génétique

- Inspiré du processus de l'évolution naturelle des espèces
- après tout l'intelligence est le résultat d'un processus d'évolution sur des millions d'années :
  - théorie de l'évolution (Darwin, 1858)
  - théorie de la sélection naturelle (Weismann)
  - concepts de génétiques (Mendel)
- La simulation de l'évolution n'a pas besoin de durer des millions d'années sur un ordinateur

Algorithme génétique

- L'algorithme commence avec un ensemble de N solutions choisies aléatoirement. Cet ensemble est appelé **population**
- Un successeur (enfant) est généré en combinant deux parents
- Une solution est représentée par une chaîne (mot) sur un alphabet : c'est le code génétique du noeud
- la fonction d'évaluation d'une solution est appelée fonction d'adaptation ou **fitness** function
- la prochaine génération est produite par :
  - sélection
  - croisement
  - mutation

Algorithme génétique

Algorithm *Algorithme génétique*

Arguments : k , *nbIterations*
Sortie : *solution*
Déclarer : *population* = ensemble de k chromosomes générés aléatoirement
Pour $t = 1...$*nbIteraions* **faire**
    nouvelle_population $=\{\}$
    Pour $i = 1...k$ **faire**
        $parent_1$ = selection(*population*)
        $parent_2$ = selection(*population*)
        child = croisement($parent_1$, $parent_2$)
        Avec une petite probabilité mutation(*child*)
        ajouter *child* à la *nouvelle_population*
    Fin Pour
    population = *nouvelle_population*
Fin Pour
retourner l'individu n ayant le fitness $f(n)$ le plus élevé de la population

Algorithme génétique

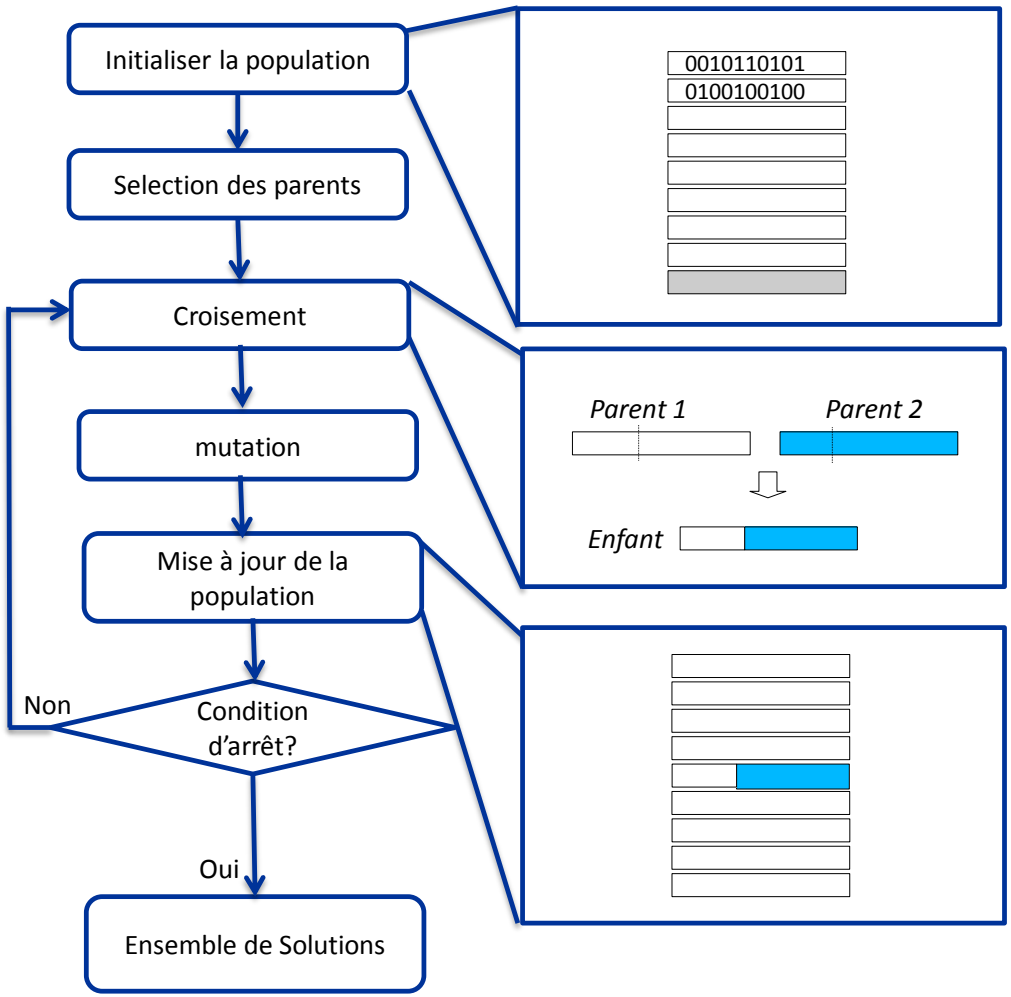
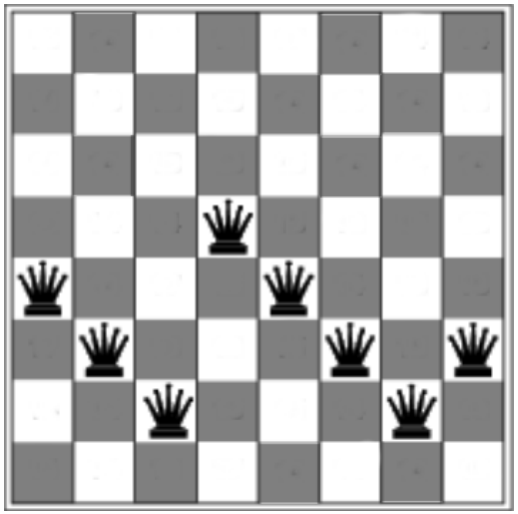
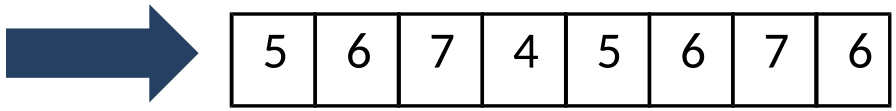


FIGURE – algorithme génétique de base

Algorithme génétique

Exemple : codage





- La fonction d'adaptation (fitness) = nombre de paires de reines qui ne s'attaquent pas.
- max = (8*7/2) = 28 ; min = 0
- f(exemple) = (28-17) = 11

Algorithme génétique

Sélection

- Sélectionner un sous ensemble d'individus qui constitueront des parents pour l'étape du croisement ou de la mutation.
- La probabilité de sélection d'un individu sera proportionnelle à sa fitness
- Il existe différents mécanismes de sélection
  - La méthode de la "loterie biaisée" (roulette wheel),
  - La méthode "élitiste",
  - La sélection par tournois,
  - La sélection universelle stochastique.

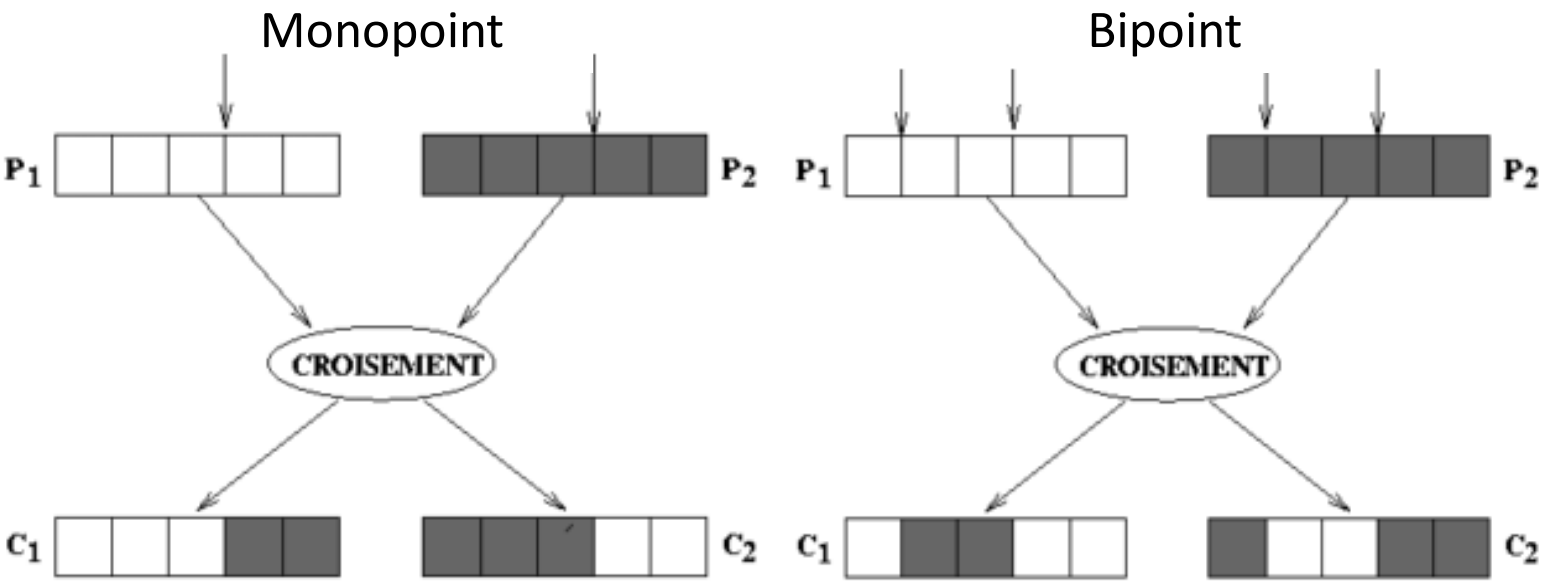
Algorithme génétique

Croisement

- Produire de nouveaux individus (les enfants) à partir d'un sous ensemble d'individus de la population (les parents).
- Préserver les meilleurs gènes des parents.
- Le croisement peut se faire de différentes manières :
  - Uni-point
  - Bipoint
  - Avec Masque

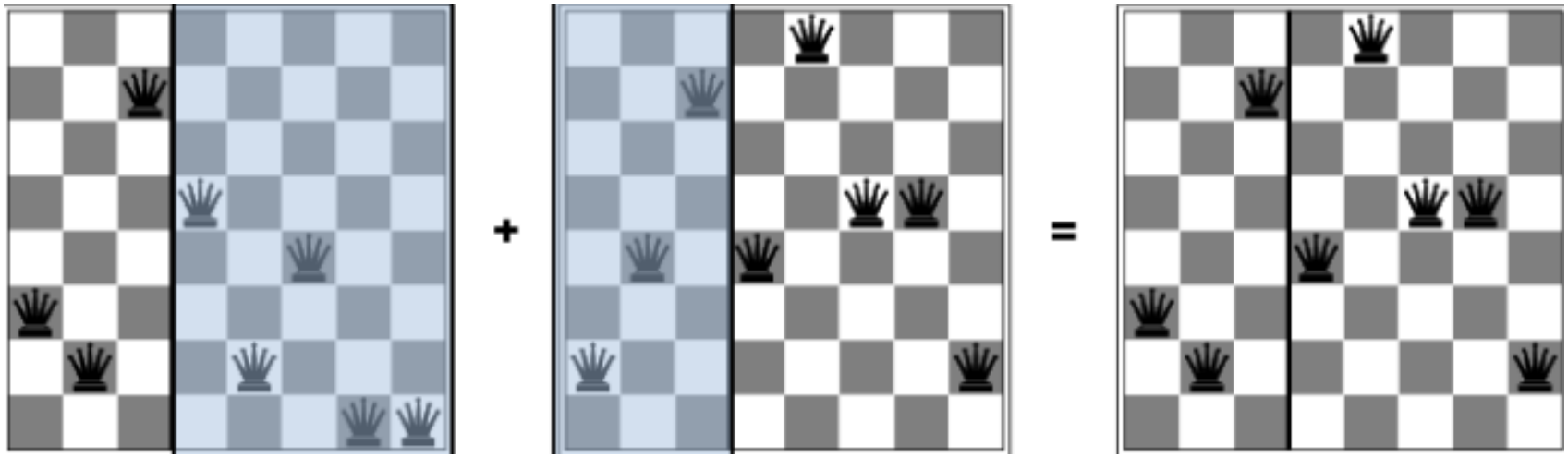
Algorithme génétique

Exemple : croisement



Algorithme génétique

Exemple : croisement 8-reines

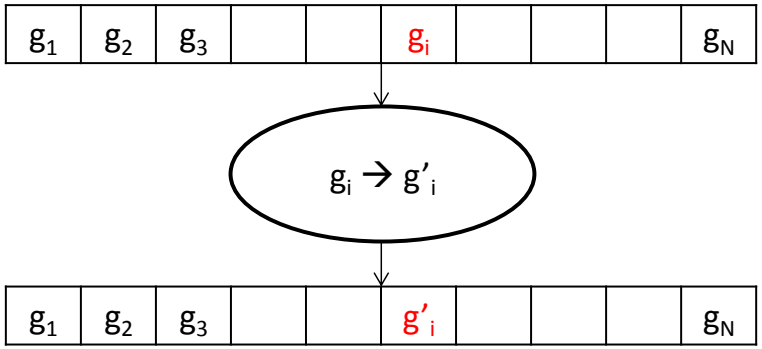


672**47588** + **752**51448 = **67251448**

Algorithme génétique

Mutation

- Un gène du chromosome (solution) est modifié.
- Taux de mutation : Qualifie la probabilité qu'une mutation se produit dans une même itération ou génération.



Algorithme génétique

Exemple : 8-reines

Critère d'arrêt

- Un nombre maximal d'itérations atteint
- La solution optimale trouvée dans la population courante
- Stagnation de la population au bout d'un certain nombre d'itération :
  - La population après remplacement est égal à la population au début de l'itération
  - La population reste inchangée au bout d'un certain nombre d'itérations (nombre de tentatives autorisées)
  - Les individus de la population possède la même évaluation par rapport à la fonction objective.

Algorithme génétique