Introduction

Résolution des problèmes d'Optimisation

A. BENGHENI(2019/2020)

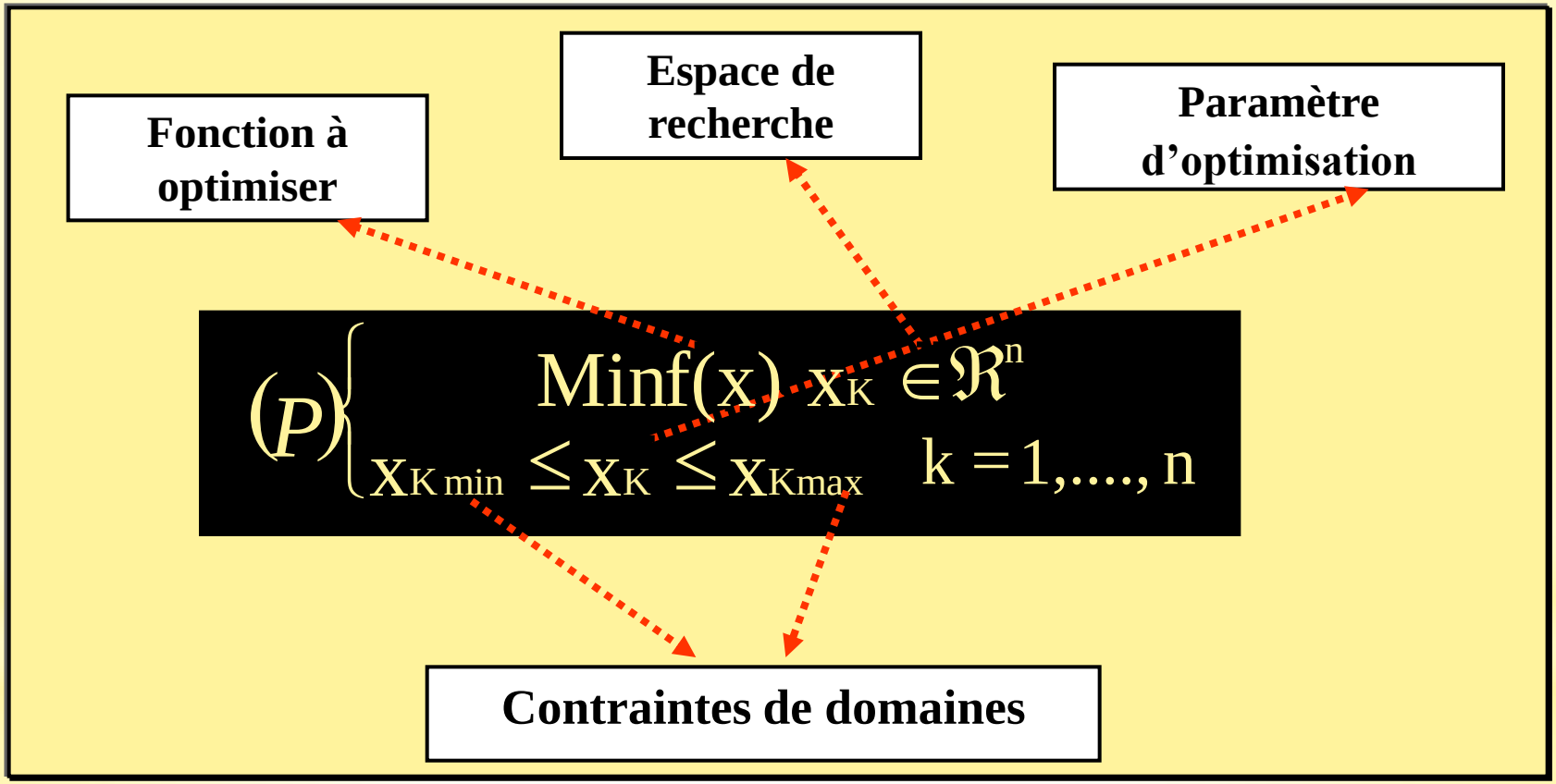
Université IBN Khaldoun - TIARET – Faculté MI –

Département informatique

Sommaire

- Problème d'optimisation
- Méthodes d'optimisation
- Méthodes d'optimisation locale
- Méthodes d'optimisation globale
  - Méthodes trajectoires
    - Méthode de descente
    - Recuit simulé
    - Recherche tabou
  - Méthodes de population
    - Algorithmes génétiques
    - Algorithmes des colonies de fourmis
- Conclusion

Problème d'optimisation



Optimisation combinatoire

- S : ensemble de solutions potentielles au plus *dénombrable* (souvent fini de grande taille) (l'ensemble de solutions réalisables (admissibles….)

- Problème combinatoire

  Trouver la ou les solutions de S convenable

- Optimisation combinatoire

  $f : S \rightarrow R$ fonction à optimiser (ou de coût)

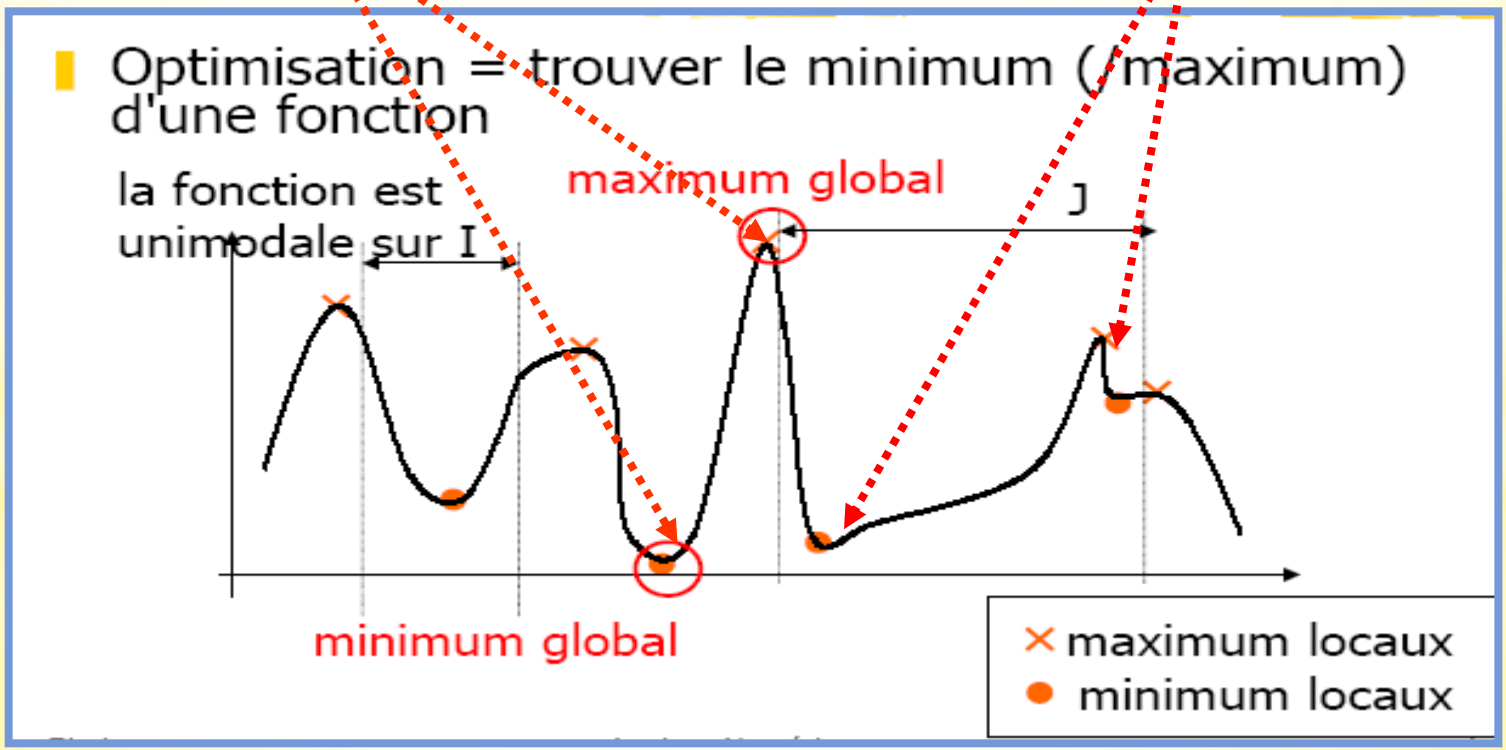
Trouver la ou les solutions de S donnant la ou les plus grandes (ou plus petites) valeurs pour f.

Optimisation combinatoire

- Exemples :
  - Le Voyageur de commerce
  - Le problème du sac-à-dos
  - Routage dans les réseaux
  - Routage de véhicules
  - Affectation de fréquence en téléphonie
  - ……

Méthodes d'optimisation

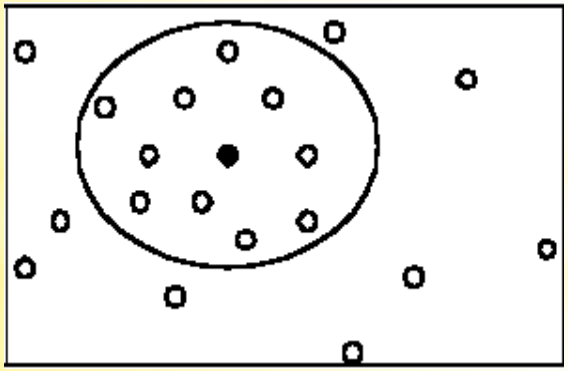
- Méthodes d'optimisation globale
- Méthodes d'optimisation locale



Notion de voisinage : l'espace est structuré. A chaque élément *x* de E, on associe un voisinage.

- Un voisinage est un ensemble d'éléments de E
- Le voisinage d'une solution s $\in$ S est un sous-ensemble de S: N(s) = {s1,s2,…,sk} solutions voisines de s

Exemple 01



Exemple 02

Voisinage de x: inverse successivement chacun des bits

0	1	0	1	0	1
1	1	0	1	0	1
0	0	0	1	0	1
0	1	1	1	0	1
0	1	0	0	0	1
0	1	0	1	1	1
0	1	0	1	0	0

L'espace de recherche peut être vu comme un graphe où les sommets représentent les solutions et les arêtes connectent les paires de solutions voisines.

Un **chemin** dans l'espace de recherche est une séquence de solutions, deux solutions consécutives étant toujours voisines.

Optimum local: une solution aussi bonne que toutes les voisines

Problème de minimisation:

s^+ est un optimum local

$\Uparrow\Downarrow$

$f(s^+) \leq f(s), \forall s \in N(s^+)$

Optimum global (solution optimale) s*:

$f(s^*) \leq f(s), \quad \forall s \in S$

Algorithmes d'optimisation locale

- Les algorithmes de recherche locale sont conçus comme une stratégie pour explorer l'espace de recherche.
- Départ: solution initiale obtenue par une méthode constructive
- Itération: amélioration de la solution courante par une recherche dans son voisinage
- Arrêt: premier optimum local trouvé (il n'y a pas de solution voisine meilleure que la solution courante)

Algorithmes d'optimisation locale

- **Amélioration itérative**: à chaque itération, sélectionner dans le voisinage une solution meilleure que la solution courante

- **Descente la plus rapide**: à chaque itération, sélectionner la meilleure solution du voisinage

Hill-Climber ou Steepest descent

1. Initialisation aléatoire s appartenant à S
2. Choisir le voisin s' le plus performant de $N(s)$

   pour tout s_1 de $N(s)$ / $f(s_1) \leq f(s')$
3. Aller à l'étape 2 si une amélioration est possible

Remarque:

- S'arrête sur optimum local
- On peut choisir le premier plus performant au lieu *du* performant

Algorithmes de recherche locale

- Questions fondamentales :
  - ✓ Définition du voisinage
  - ✓ Stratégie de recherche dans le voisinage
  - ✓ Complexité de chaque itération:
  - ✓ Dépend du nombre de solutions dans le voisinage
  - ✓ Dépend du calcul efficace du coût de chaque solution voisine

- Différents aspects de l'espace de recherche peuvent influencer la performance d'un algorithme de recherche locale

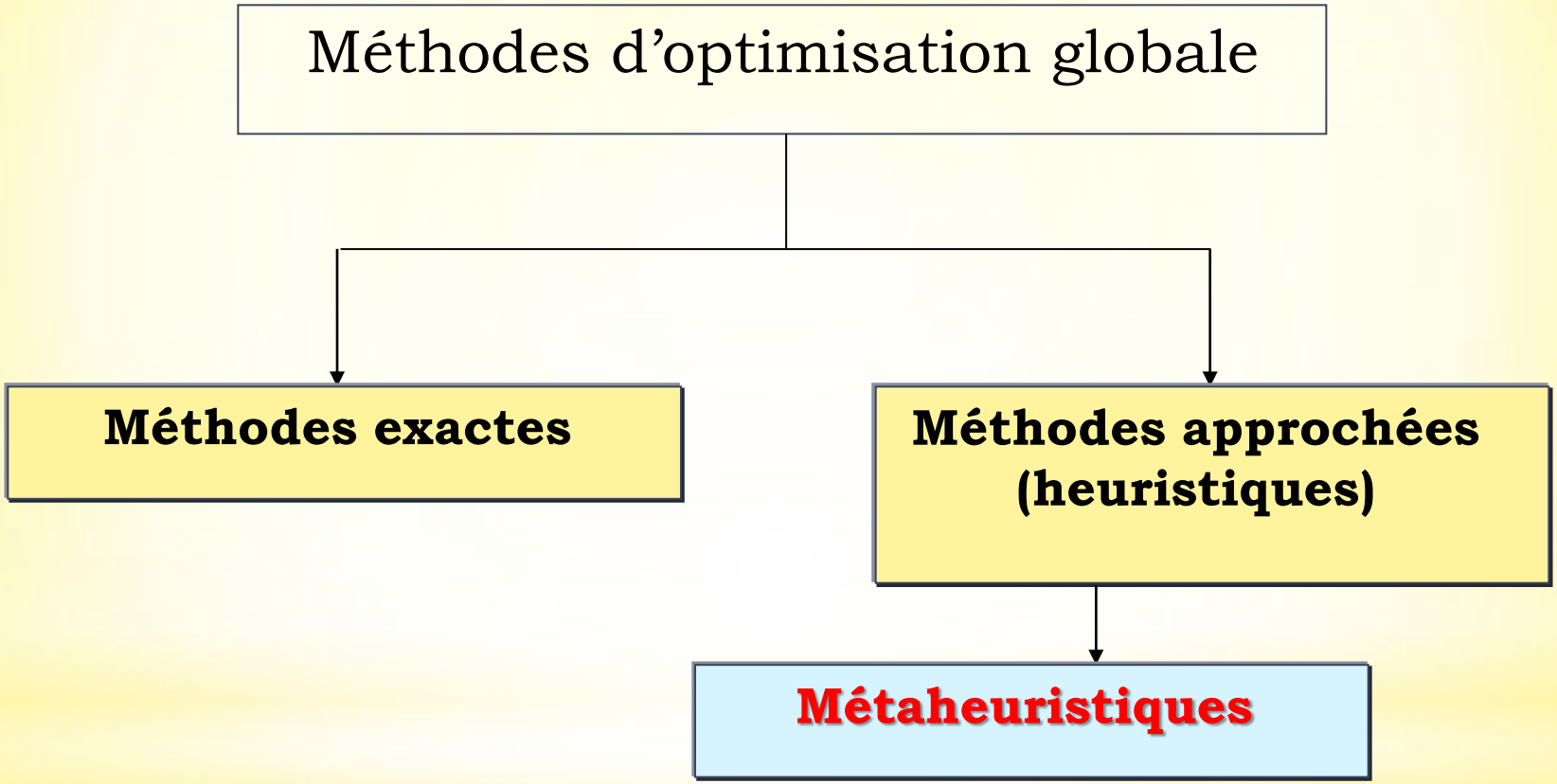
- Connexité: il doit y avoir un chemin entre chaque paire de solutions de l'espace de recherche

- Distance entre deux solutions: nombre de sommets visités sur le chemin le plus court entre elles.

- Diamètre: distance entre les deux solutions les plus eloignées (diamètres réduits!)

- Difficultés:
  - ✓ Arrêt prématuré sur le premier optimum local
  - ✓ Sensibles à la solution de départ
  - ✓ Sensibles au voisinage choisi
  - ✓ Sensible à la stratégie de recherche
  - ✓ Nombre d'itérations

- Extensions pour réduire les difficultés des algorithmes de recherche locale:
  - Réduction des voisinages: considérer un sous-ensemble du voisinage
  - Accepter parfois des solutions qui n'améliorent pas la solution courante, de façon à pouvoir sortir d'un optimum local.
  - Multi-départ: appliquer l'algorithme de recherche locale à partir de plusieurs solutions de départ
  - Multi-voisinages dynamiques: changer de définition de voisinage après avoir obtenu un optimum local
  - → **Meta-heuristiques**


Méthodes d'optimisation globale
Méthodes exactes
Méthodes approchées (heuristiques)
Métaheuristiques

Heuristiques

- Du grec heuriskein : trouver/découvrir (heureka)
- Une heuristique est plutôt une méthode qui cherche (stratégie)… puisqu'on ne peut garantir le résultat

- Définition : une heuristique est une méthode qui cherche de bonne solution (proche de l'optimalité)

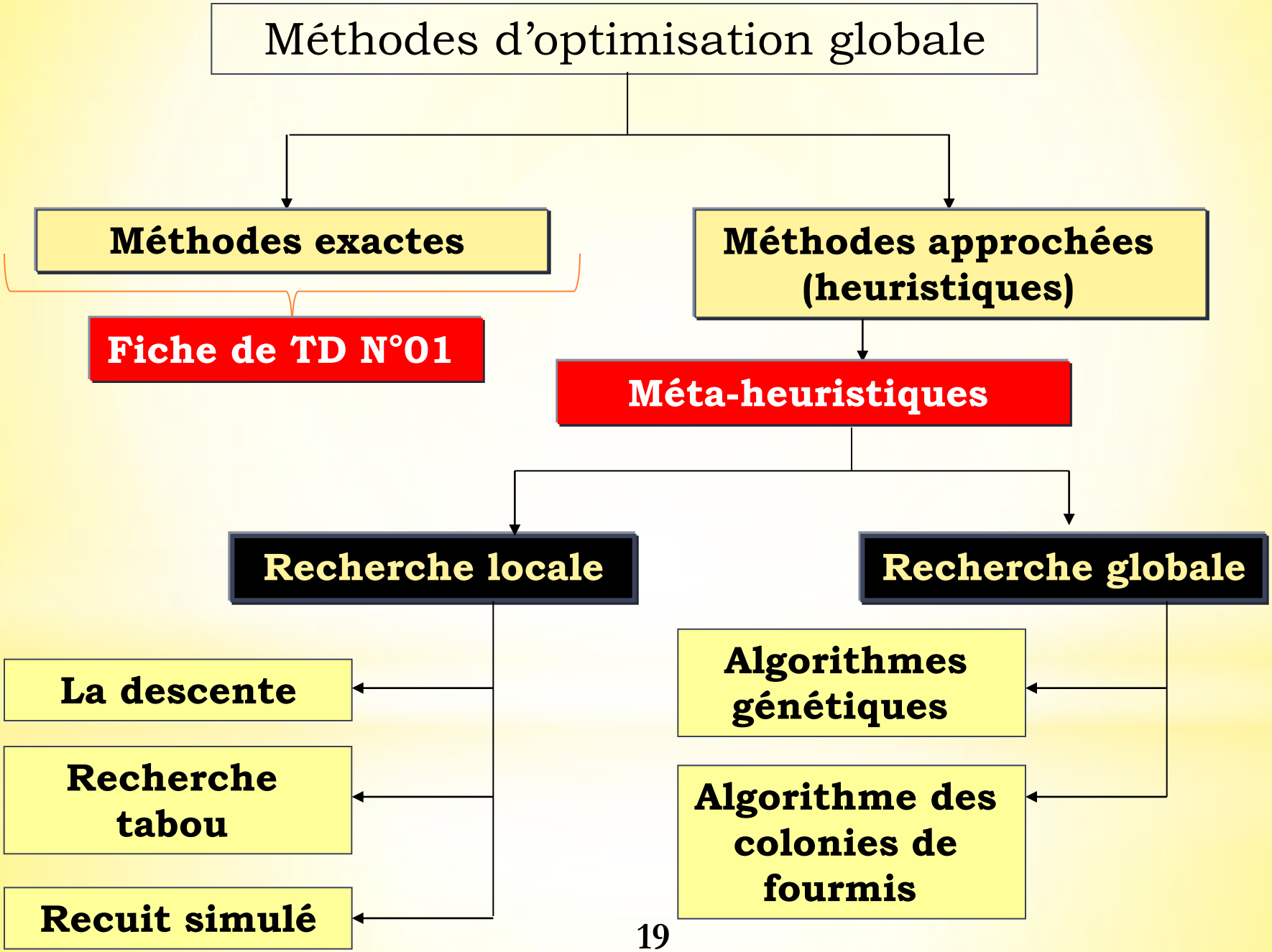
Remarques :

- ✓ Temps de calcul raisonnable
- ✓ Sans garantir faisabilité ou l'optimalité.
- ✓ Très large succès : des techniques performantes de résolution

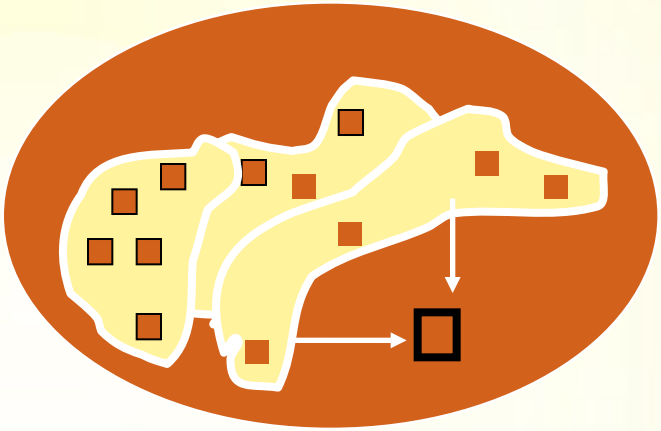
Évaluation des Heuristiques

- Le problème n'est pas tellement de générer une solution, mais de connaître sa qualité

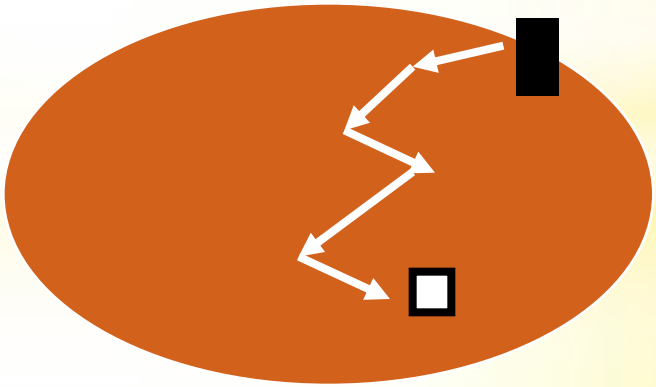
- Évaluation en moyenne
- Évaluation en meilleur solution obtenue
- Évaluation du compromis entre qualité/coût


Méthodes d'optimisation globale
Méthodes exactes
Méthodes approchées (heuristiques)
Fiche de TD N°01
Méta-heuristiques
Recherche locale
Recherche globale
La descente
Recherche tabou
Recuit simulé
Algorithmes génétiques
Algorithme des colonies de fourmis

Approche globale évolutionaire

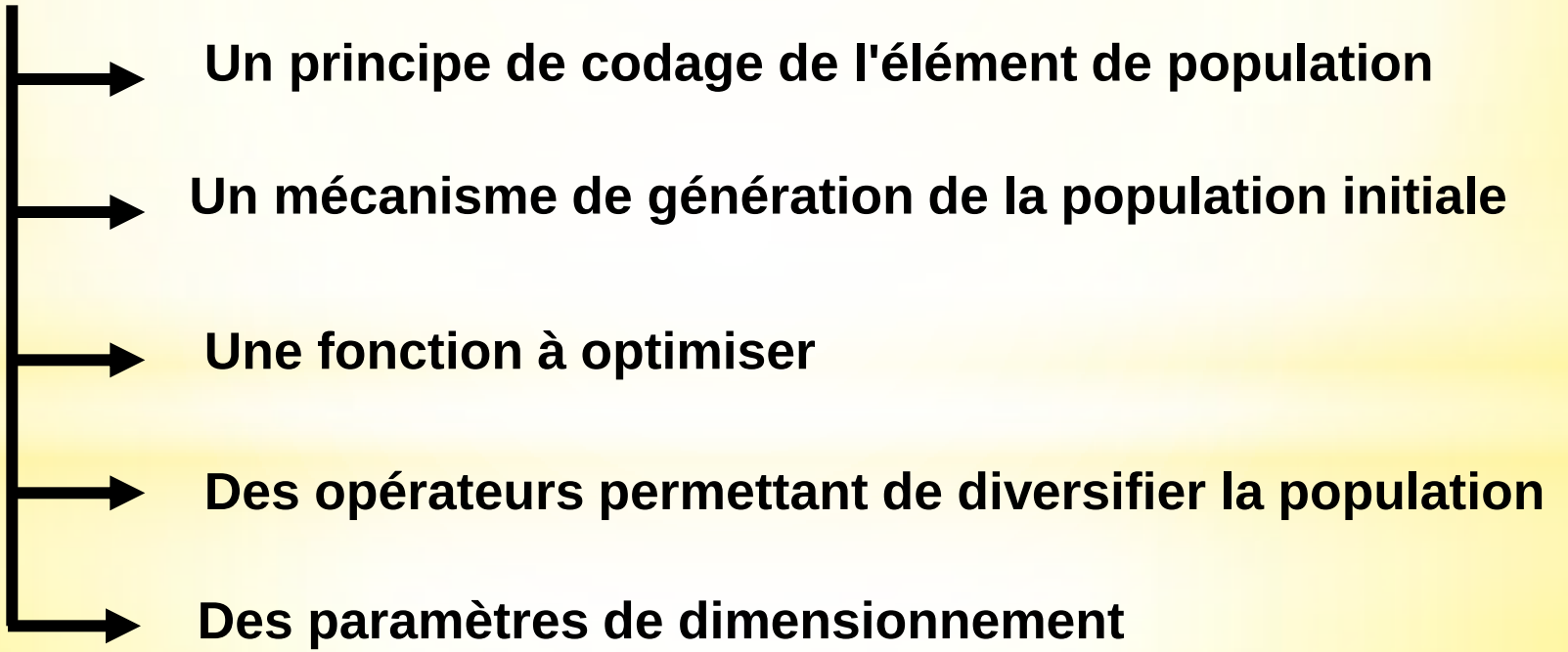


Approche locale itérative

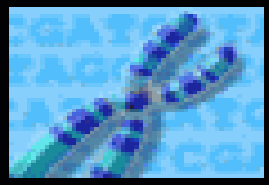


Algorithmes génétiques

- **C’est un algorithme d'optimisation s'appuyant sur des techniques dérivées de la génétique et de l'évolution naturelle.**
- **Pour l'utiliser, on doit disposer des cinq éléments :**



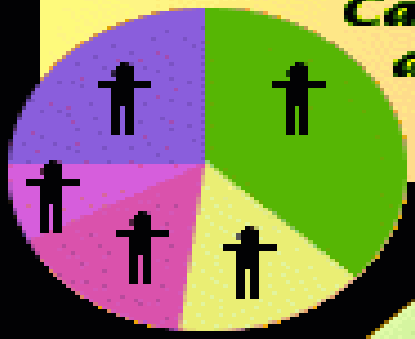
PROCESSUS de SELECTION NATURELLE



Génération de la population initiale

Calcul des adaptations
F( ) = ?

Calcul de la probabilité de sélection de chaque individu



si fin

Résultats

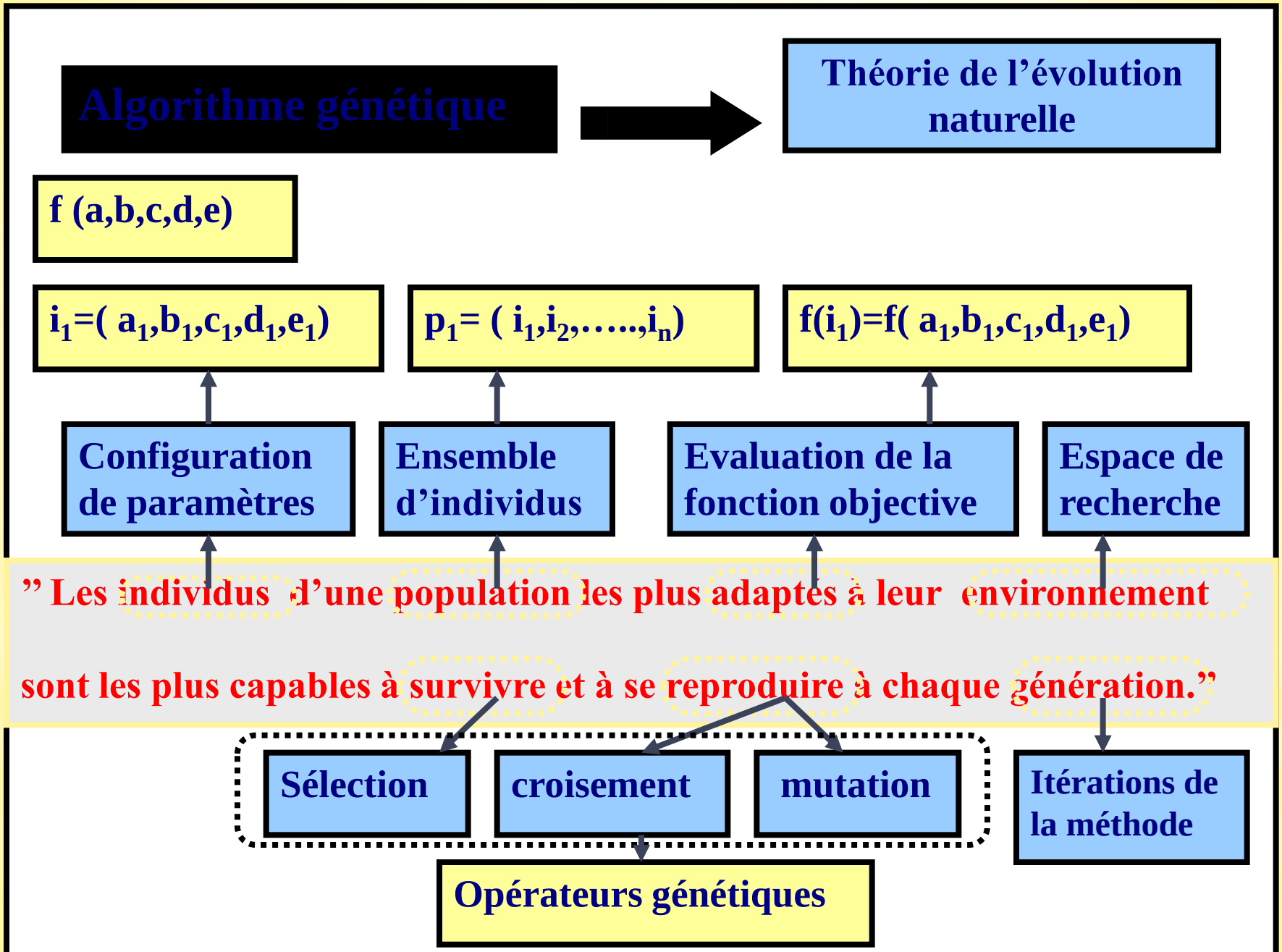
Elimination des plus mauvais individus

Calcul d'une nouvelle génération par reproduction :
- **Sélection des parents**
- **Reproduction**
  (cross-over, mutation)

La notion de voisinage est remplacée par l'application d'opérateurs de "mutation" et de "croisement" (cross-over).

```
Algorithme :
Choisir la taille t de la population
Générer la population initiale P      // i.e. un ensemble de solutions
WHILE {le critère d'arrêt n'est pas satisfait}
                // recombinaisons par mutation et croisement
     Reproduction des individus de P

                // sélection probabiliste de t individus
                // créés à l'étape précédente (plus ils sont "bons",
                // et plus la probabilité de les sélectionner
                // est forte
     Mise à jour de P
ENDWHILE
```


Algorithme génétique
Théorie de l'évolution naturelle
f (a,b,c,d,e)
$i_1=(a_1,b_1,c_1,d_1,e_1)$
$p_1= (i_1,i_2,\ldots..,i_n)$
$f(i_1)=f(a_1,b_1,c_1,d_1,e_1)$
Configuration de paramètres
Ensemble d'individus
Evaluation de la fonction objective
Espace de recherche
" Les individus d'une population les plus adaptés à leur environnement sont les plus capables à survivre et à se reproduire à chaque génération."
Sélection
croisement
mutation
Itérations de la méthode
Opérateurs génétiques

L’optimisation par colonies de fourmis

Ce type d'algorithme a été proposé (1992, par Marc Dorigo), il s'inspire des comportements collectif de dépôt et de suivie de traces

- **ACO (Ant Colony Optimization) : Comportement collectif de fourmis pour résoudre des problèmes combinatoires.**
- **Recherche d'un «meilleur» chemin.**
- **Communiquent entre elles, à travers la phéromone**

Dans la nature, les fourmis utilisent des phéromones pour marquer des informations dans leur environnement:

Elles utilisent ensuite leur capteur de phéromones pour déterminer le chemin le plus marqué en phéromones qu'elles suivront.

- **'Fourmis'**
  - Agents de recherche qui miment les comportement des fourmis réels.

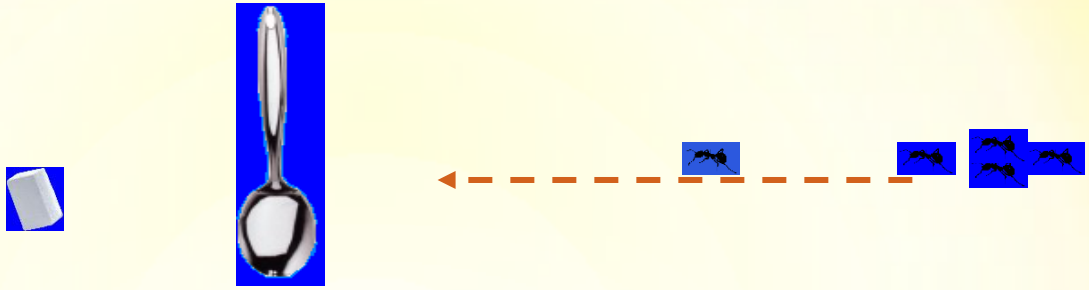
- **Problème posé par les éthologues :**
  - Comment des insectes qui ne voient pas trouvent le PCC entre leur nid et la nourriture???

- **Réponse :**
  - Utilisation du milieu pour communiquer entre les individus.

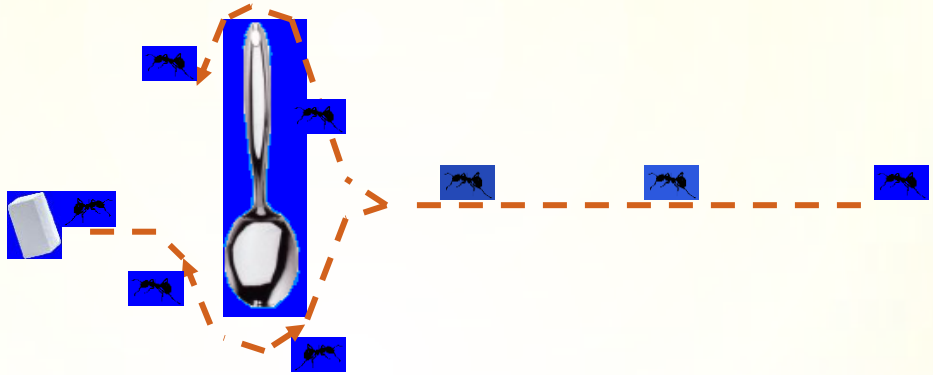
- **Le phéromone:**
  - Une fourmi qui se déplace, laisse une quantité variable de phéromone sur son chemin, qui est détectée par les prochaines fourmis, et qui détermine avec une grande probabilité leur chemin.

- **Forme de '*autocatalytic behavior*' :**
  - Plus des fourmis suivent le chemin plus le chemin devient attractif.

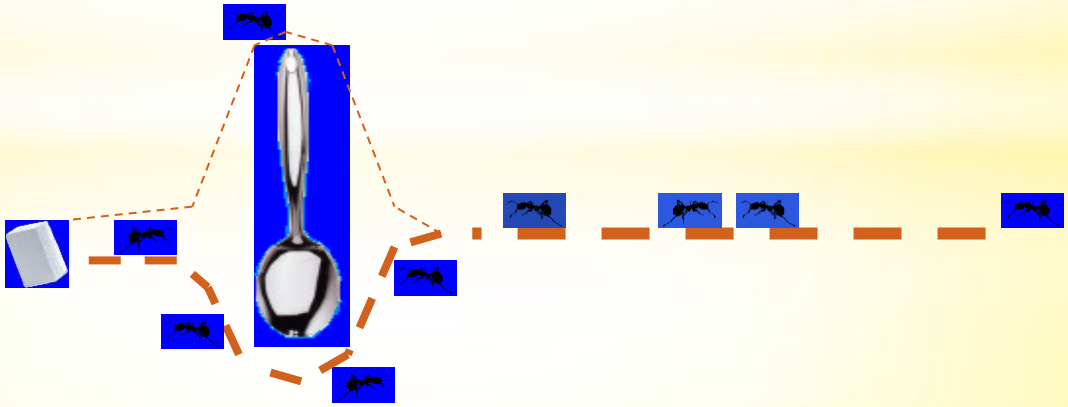
Détection de la nourriture



Aller vers la nourriture selon différentes chemin



Choix du plus cours chemin

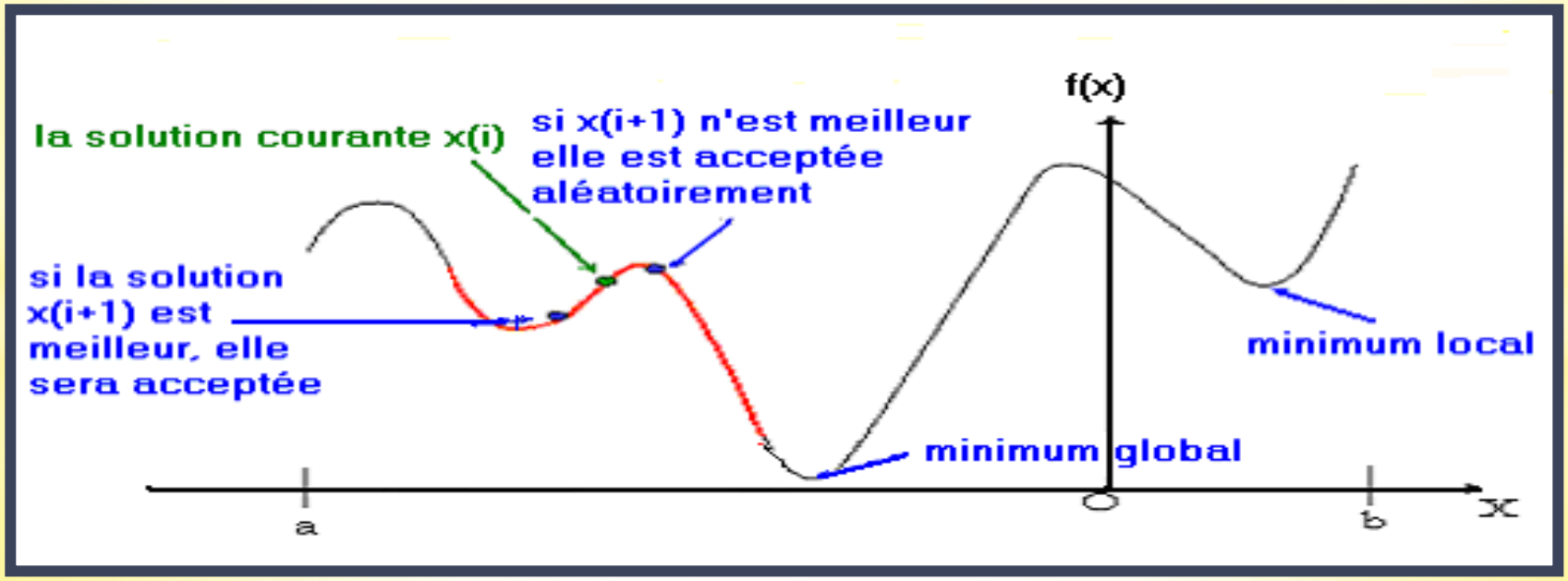


Recuit simulé

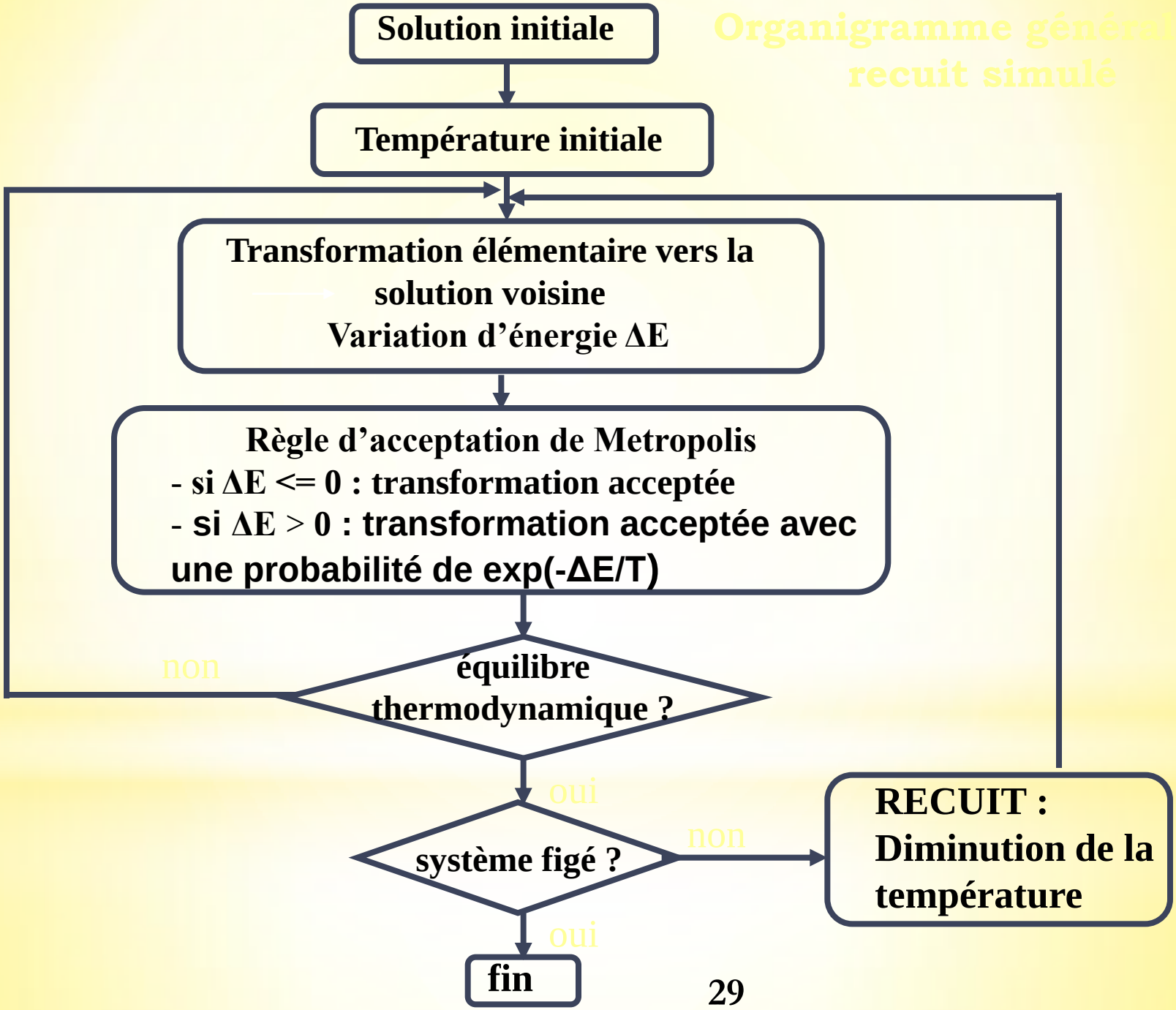
- **Algorithme qui s'inspire d'un mécanisme naturel pour trouver une solution optimale à un problème**

- **Recuit = processus physique de chauffage suivi d'un lent refroidissement pour obtenir une structure cristalline plus solide.**

- **Recuit simulé= processus qui baisse lentement sa température jusqu'à ce que le système se fige et que plus aucun changement ne soit observé.**



Organigramme général du recuit simulé



RECUIT SIMULE (recherche d'un minimum)

choisir une solution initiale *s*
choisir une température initiale $T_i > 0$
choisir une température finale $T_f > 0$ // $T_f < T_i$
choisir un nombre d'itération NB (à une température donnée)
choisir le coefficient de diminution de la température $\Phi \in [0,1[$

```
 T <- T_{i}$
WHILE { T_f < T}
FOR {k = 1 to NB}
S' <- voisin aléatoire de s
Δ <- f(s') - f(s)$
IF {Δ ≤ 0}
      s <- s'
ELSE
      s <- s' avec la probabilité e^{-Δ/T}
ENDIF
ENDFOR
T <- T*Φ
ENDWHILE
```

Recherche tabou

Introduite par Glover 1986

Choix du meilleur voisin (même s'il se révèle mois bon)

Utilisation d'une liste des voisins visités pour éviter de boucler

La méthode consiste à se déplacer d'une solution vers une autre par observation du voisinage de la solution de départ et à définir des transformations tabous que l'on garde en mémoire.

Une transformation tabou est une transformation que l'on s'interdit d'appliquer à la solution courante.

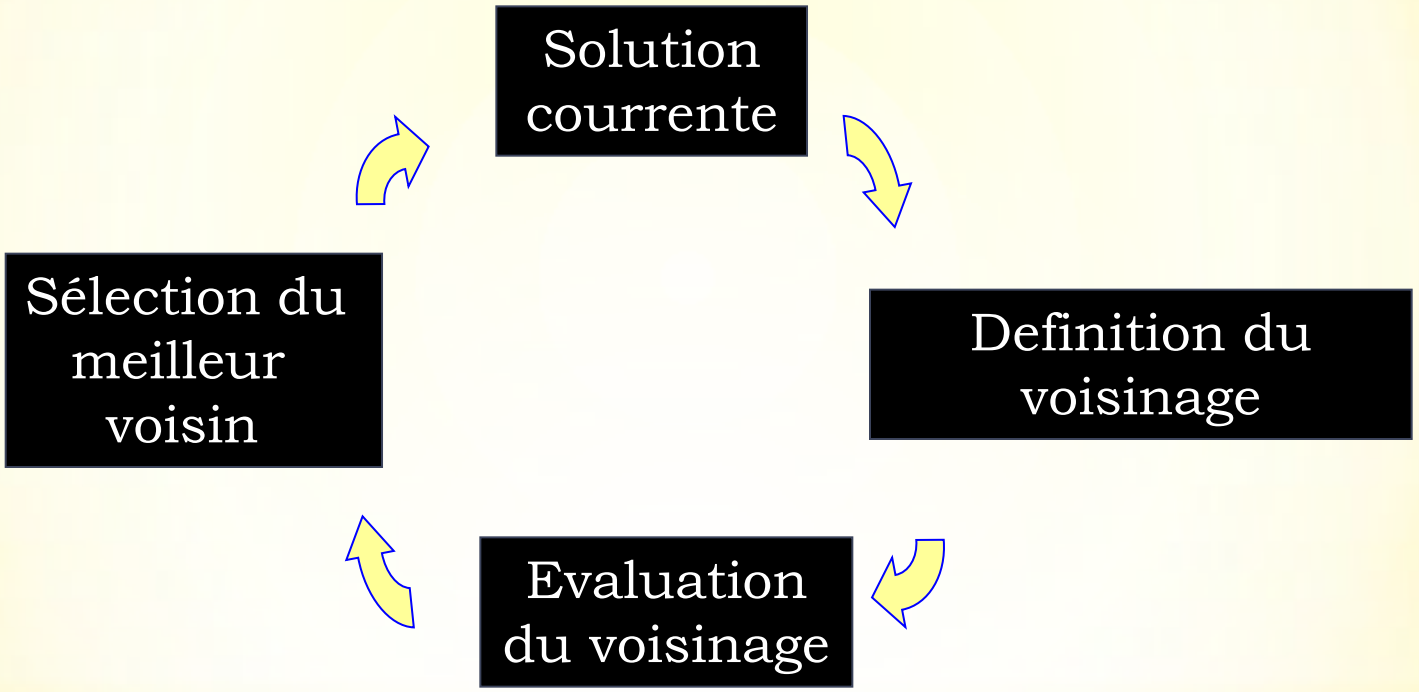
Recherche Tabou (TS)

- A chaque itération, « le moins mauvais » voisin est choisit
- Pour éviter les cycles, c'est à dire la répétition infinie d'une séquence de mouvements, les L derniers mouvements sont considérés comme interdits, L étant la taille de la liste tabou
- À chaque itération, le mouvement effectué est donc le moins mauvais mouvement non tabou

LA RECHERCHE TABOU

- Garder des traces du passé pour mieux s'orienter dans le future Utilisation d'une (petite) mémoire (la liste tabou) pour éviter de tomber dans un optimum local et/ou pour éviter de boucler
- La gestion de la liste tabou est de type FIFO (First In First Out)}

```
choisir la taille de la liste tabou L_TABOU
choisir un nombre d'itération NB
choisir une solution initiale s
meilleure_evaluation <- f(s), $meilleure_solution <- s
change <- TRUE
WHILE {change = TRUE}
change <- FALSE
FOR {k = 1 to NB}
choisir le voisin s' de s minimisant f et ∉ L_TABOU
IF {f(s') < meilleure_evaluation}
      meilleure_solution <- s'
      meilleure_evaluation <- f(s')
      change = TRUE
ENDIF
mettre à jour la liste tabou, i.e. ajouter s' à L_TABOU
s <- s'
ENDFOR
ENDWHILE
```



Une itération dans la recherche tabou

Conclusion

- ✓ Les méta-heuristiques présentent de très bons résultats sur certains types de problèmes (complexes)
- ✓ Algorithmes faciles à mettre en œuvre
- ✓ Il faut faire les bons choix de paramétrage
- ✓ Solution non garantie
- ✓ Tendance : hybridation des méta-heuristiques