Optimisation dans les télécommunications

RTS 1 – Oct. Nov. 2010

Hervé Rivano
(thx Katia Jaffrès-Runser)

Présentation du cours

- L'optimisation : pourquoi, qu'est-ce que c'est ?
- Quelques techniques générales classiques
- L'optimisation convexe
- La programmation linéaire
  - Théorie et pratique (math inside)
  - Génération de colonne
- Programmation linéaire en nombres entiers

Quelques problèmes d'optimisation

- Au niveau PHY:
  - Allocation de fréquences GSM
  - Allocation de longueurs d'onde WDM
- Au niveau MAC
  - Allocation des canaux en TDMA
- Au niveau Routage
  - Recherche du plus court chemin
  - Routage d'un (plusieurs) trafic(s) sur un réseau radio maillé
- Conception de réseau
  - Combinaison des problèmes ci-dessus
  - Coût énergétique, nombre de clients simultanés, tarification

Optimiser ?

- Espace des solutions D
  - Explicite / implicite

- On cherche à résoudre un problème du type

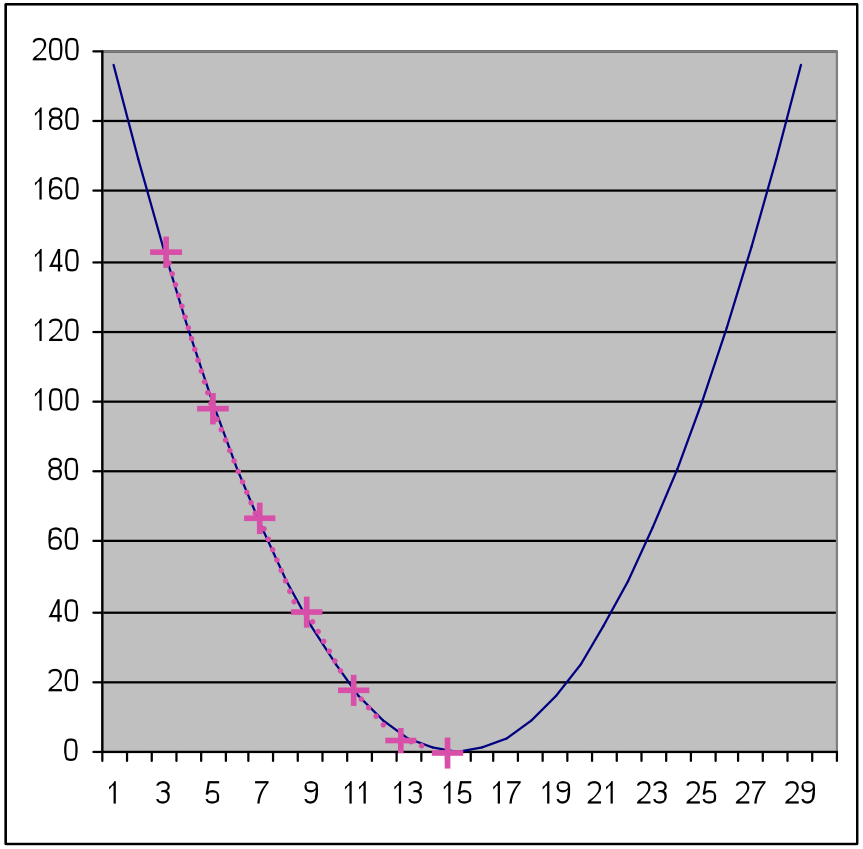
$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathcal{D}} f(\mathbf{x})$$

- f : fonction objective. En général f:D→R
- Et si on veut maximiser ?

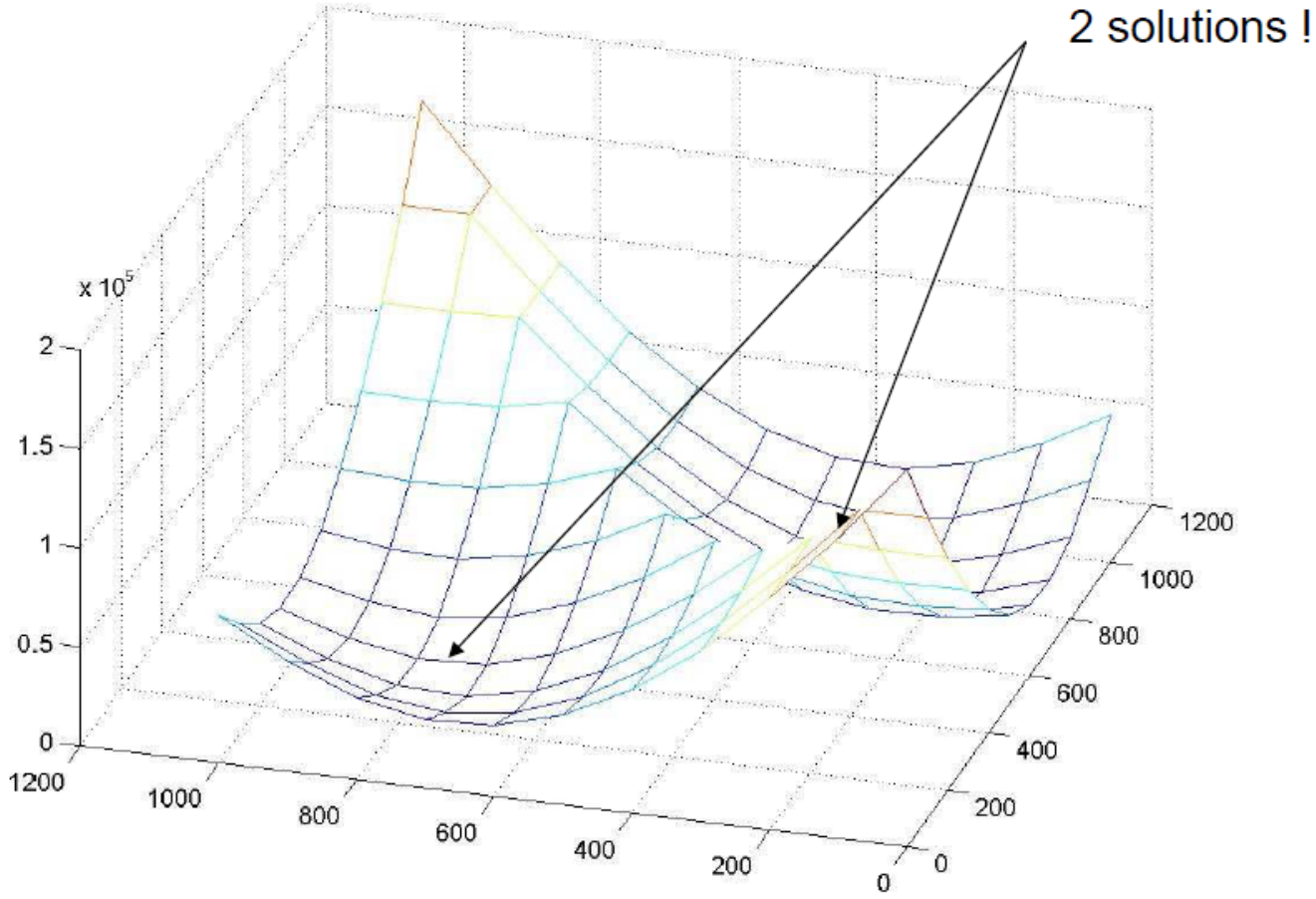
Difficulté d'un problème d'optimisation

Difficulté : formulation Continue.

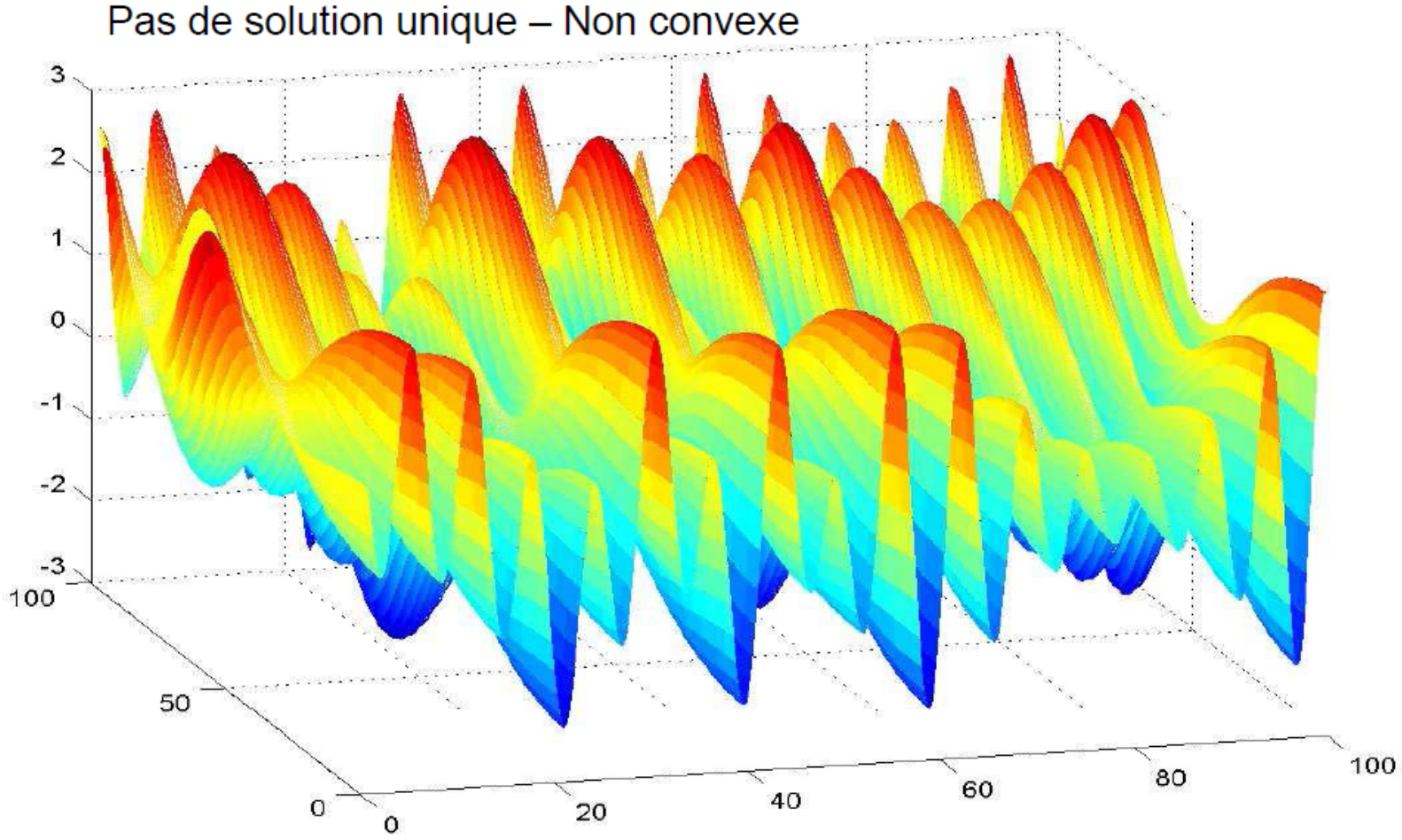
- Le nombre de solutions *n* est infini !
- La difficulté se trouve au niveau de la nature de la fonction d'évaluation :
  - Si *f* est convexe, une simple stratégie de descente converge.



Difficulté : Formulation continue



Difficulté : Formulation continue



Difficulté d'un problème

- Algorithme de recherche le plus évident : la recherche exhaustive
- Est-ce qu'une telle approche est toujours réalisable ?
    - … cela dépend du temps de calcul à sa disposition !

Difficulté d'un problème

- Quels facteurs influencent la durée de la résolution ?
  - → Le nombre de variables.
  - → La taille/forme du domaine de définition de chaque variable.
  - → La forme de la fonction d'évaluation.
  - → Le temps de calcul nécessaire à l'évaluation d'une solution.

Les principaux algorithmes de résolution.

Recherche linéaire

- Recherche linéaire :
  - Brique de base des autres méthodes continues.
  - Recherche itérative du minimum dans une seule direction :
    - A l'itération k, on cherche :

$$x_{k+1} = x_k + \lambda_k d$$

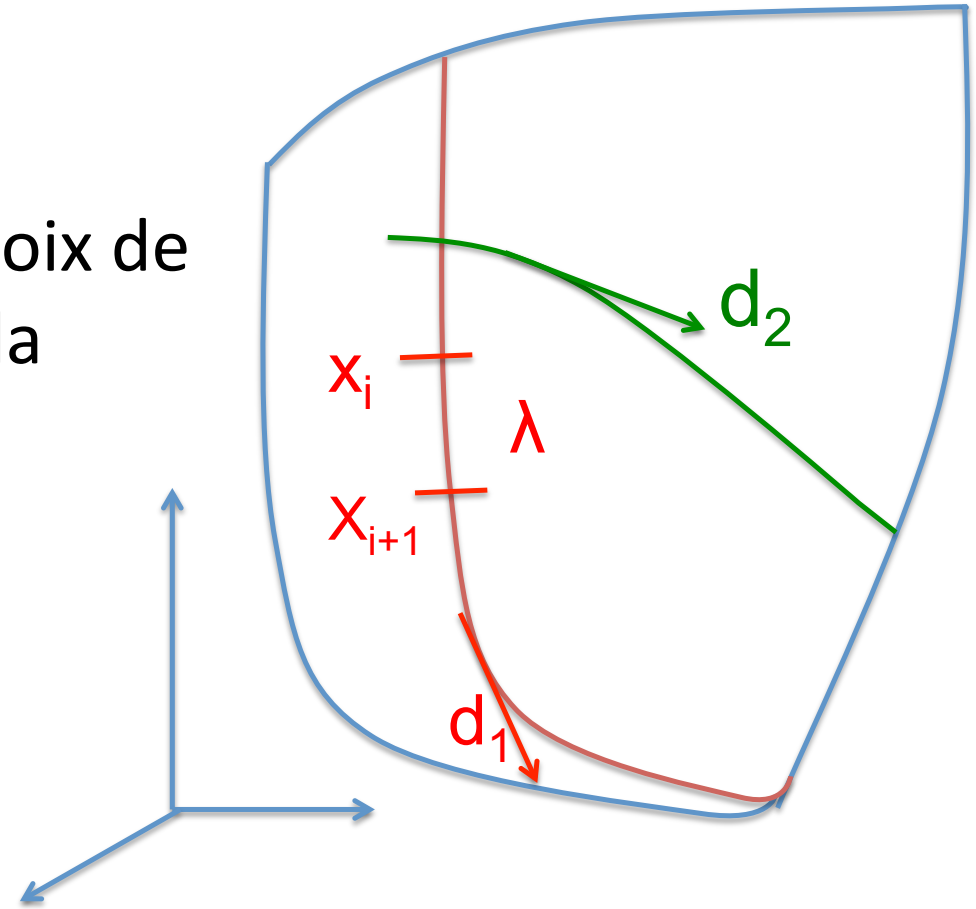
Pas de recherche

Direction

    - Tel que : $f(x_{k+1}) \leq f(x_k)$

Recherche linéaire

- Deux recherches linéaires
- Performance dépend du choix de d et de lambda



Algorithmes basés sur la dérivée.

- Méthode de Newton :
  - Résolution de g(x) = 0, $g:\Re^n \rightarrow \Re^n$
    - à partir d'une solution de départ $x_0 \in \Re^n$.
  - A chaque itération, on calcule le point x_{k+1} :

$$x_{k+1} = x_k - g'(x_k)^{-1}.g(x_k)$$

  - Ici, le pas de recherche est inversement proportionnel au gradient

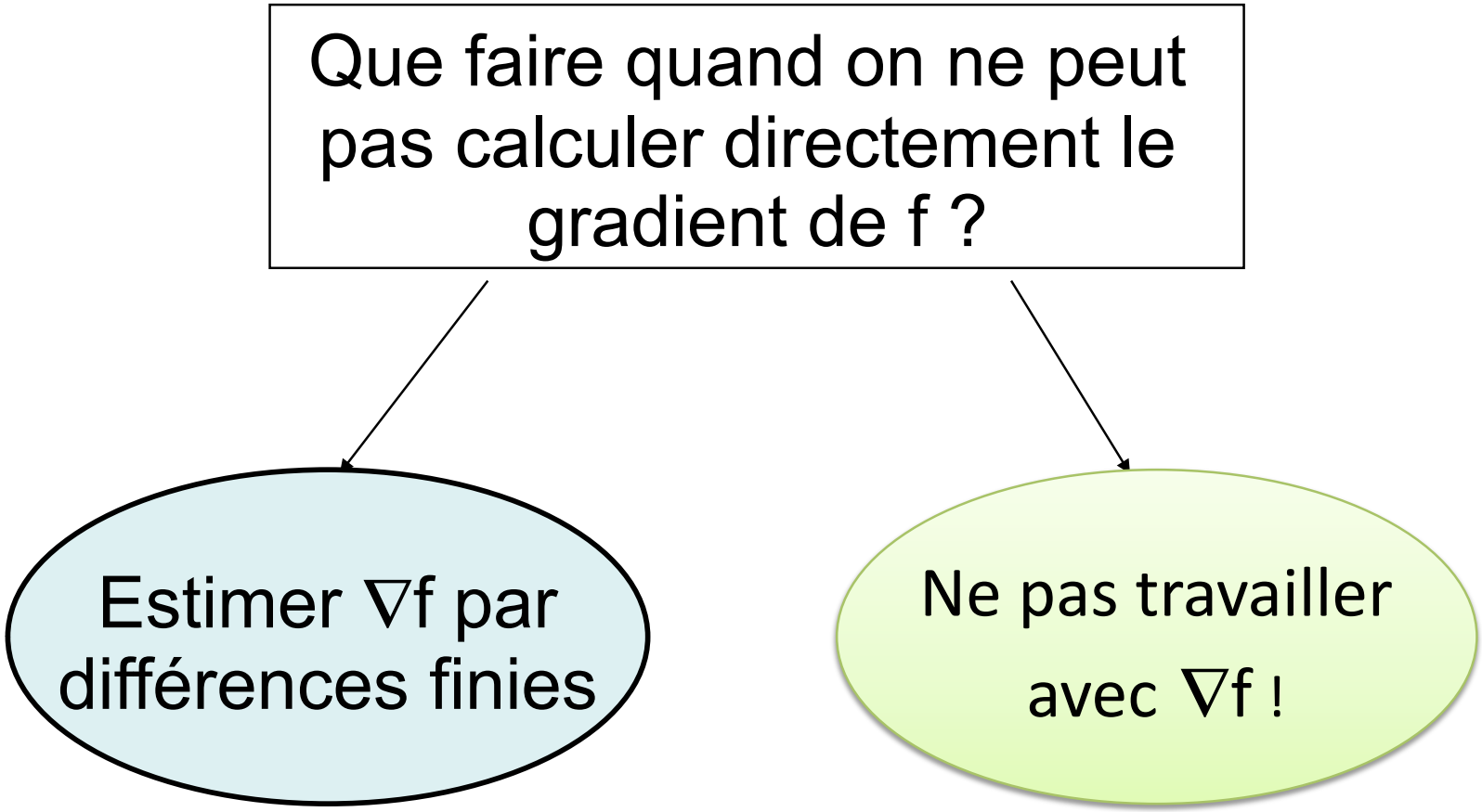
Algorithmes basés sur la dérivée.

- Méthode de Newton pour minimiser f(x) :
  - On utilise le même principe en posant
    $$g(x) = \nabla f(x).$$
  - On cherche alors le point où $\nabla f(x)=0$:

$$x_{k+1} = x_k - \nabla^2 f(x_k)^{-1}.\nabla f(x_k)$$

  - Converge rapidement si x_0 est proche de x*, sinon, c'est très long…..

1- Algorithmes d'optimisation continue



Algorithmes de recherche directe

- Méthodes de Recherche Directe.
- C'est l'observation d'un jeu de solutions qui implique à chaque itération
  - Une comparaison du jeu de solutions avec la meilleure existante.
  - Une stratégie de choix du prochain jeu de solution.
- Pas de calcul de gradient.

Recherche locale

- Stratégie triviale ! A chaque itération :
  - On teste les proches voisins dans l'espace discret des solutions
  - On choisit le meilleur.
- Notion de voisinage
  - 2 solutions s1 et s2 sont « voisines » si s2 est obtenue par une « modification élémentaire » de s1
- Converge forcément vers un minimum local !
- Comment trouver l'optimum global ?

Les Métaheuristiques.

Une Métaheuristique est un ensemble de concepts qui peuvent être utilisés pour définir des heuristiques (algorithmes) qui peuvent être appliquées à un grand ensemble de problèmes différents.

- On les utilise en général quand les problèmes sont difficiles.

Les Métaheuristiques.

- C'est un schéma générique d'algorithme qui peut être adapté à différents problèmes d'optimisation.

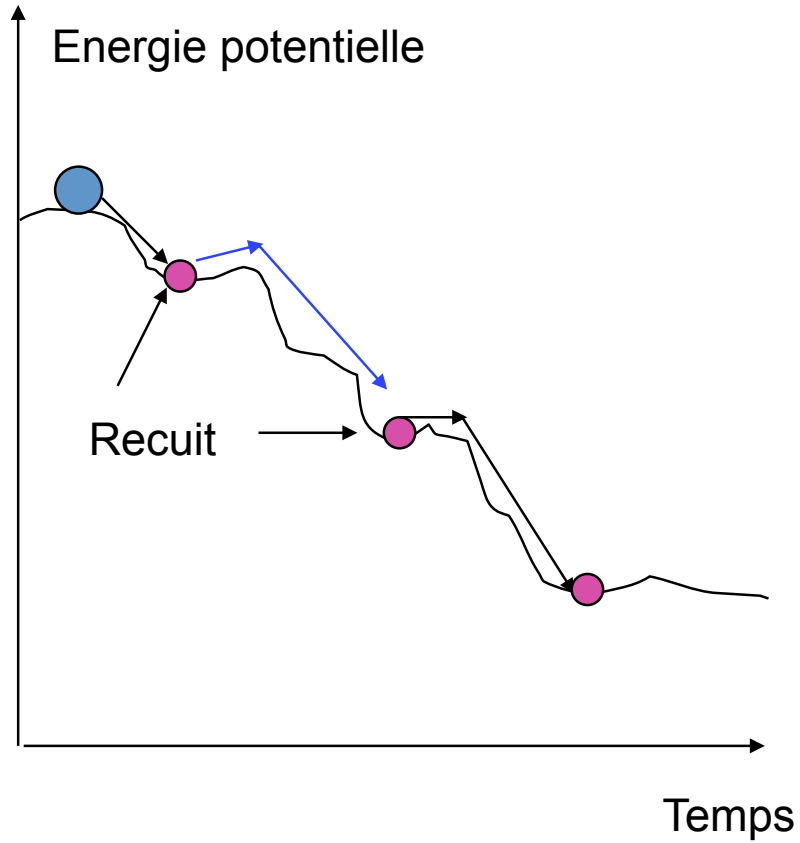
- Mêlent recherche globale et locale :
  - Parcours améliorant des voisinages
  - Dégradation de la solution acceptée

Les Métaheuristiques

- Elles comprennent principalement les algorithmes :
  - Recuit Simulé,
  - Recherche Tabou,
  - Génétique,
  - Colonies de fourmis.
- …. Et d'autres !

Le Recuit Simulé.

- Inspiré du procédé de recuit des métaux :
  - Pour arriver à un état stable (optimal) après fusion, un métal est refroidit lentement.
  - Au cours du refroidissement, il est recuit par intermittence.
  - Ceci lui permet d'arriver à un état d'énergie minimal stable, optimal.



Le Recuit Simulé.

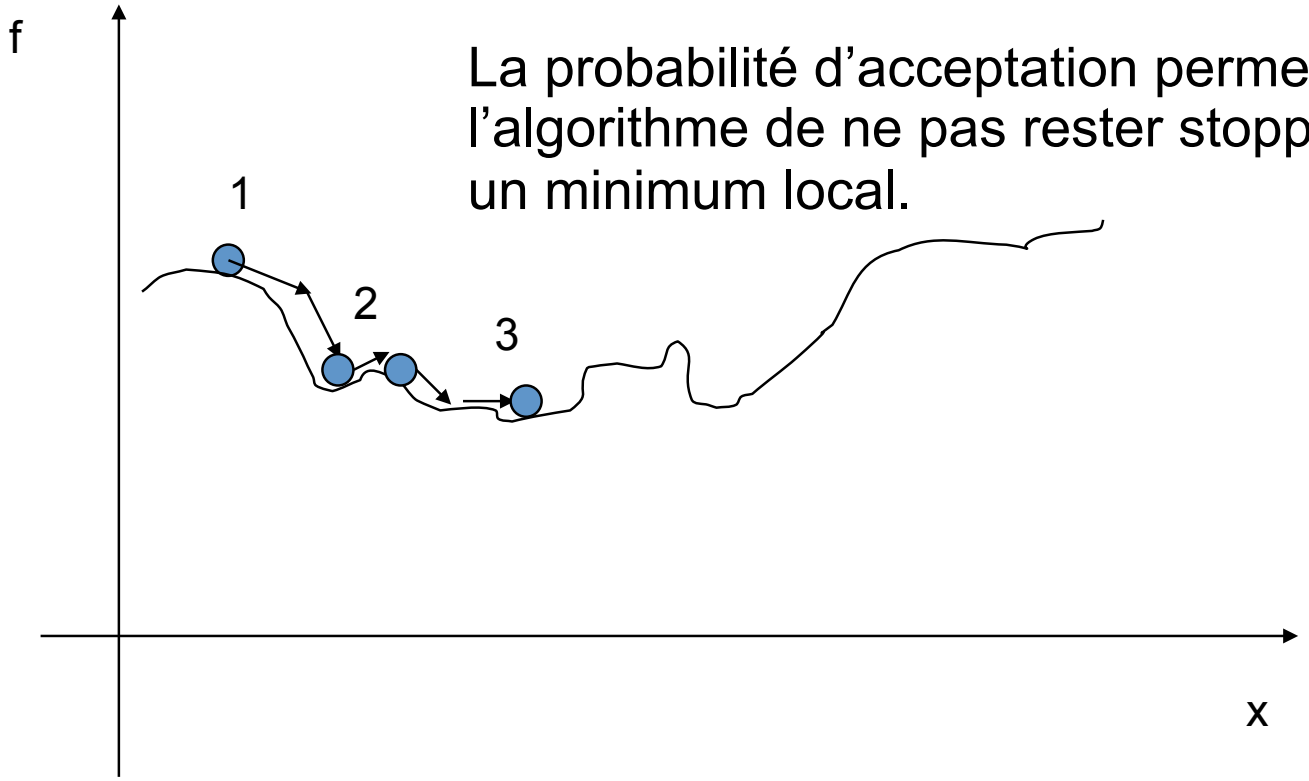
- Par analogie, on définit l'énergie initiale du processus d'optimisation E et la température initiale T.
- A chaque itération, on modifie localement la solution et on analyse le changement d'énergie.
  - Si ΔE est négatif, on accepte la solution,
  - Si ΔE est positif, on accepte la solution avec une probabilité donnée par :

$$p = \exp\left(-\Delta E / T\right)$$

Le Recuit Simulé.

- Après n itérations, on fait décroître T.
- T élevée → grande probabilité d'accepter une transition qui dégrade f.
- T est faible → converge vers un minimum local
- Paramètres :
  - Température initiale
  - schéma de refroidissement de T (n et loi)
  - Critère d'arrêt

Le Recuit Simulé

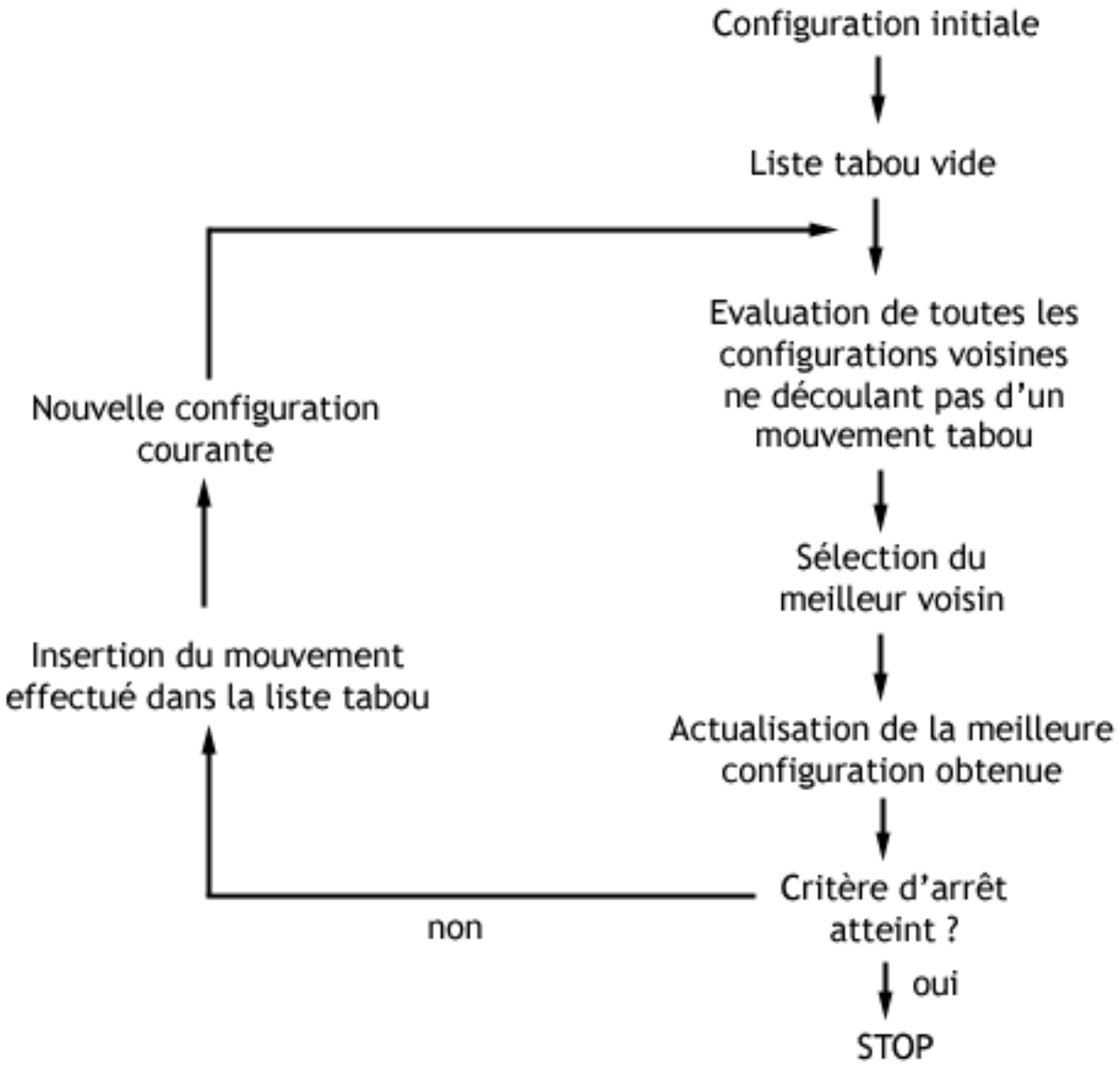


La probabilité d'acceptation permet à l'algorithme de ne pas rester stoppé dans un minimum local.

La Recherche Tabou

- Se base sur une recherche locale
  - On prend la meilleure solution du voisinage
  - Même si elle est moins bonne que l'actuelle
- On garde en mémoire les choix déjà fait
  - Stocke le chemin parcouru dans une liste tabou
  - On ne refait pas les mêmes choix
  - Taille de la liste bornée

La Recherche Tabou



La Recherche Tabou.

- Paramètres :
  - taille de la liste tabou
  - critère d'arrêt
- Définition du voisinage critique

Algorithmes Evolutionnaires.

- C'est la famille qui comprend les algorithmes génétiques.
- Inspiré par l'Evolution des Espèces dans la nature.
- Basée sur la sélection naturelle des individus les meilleurs et le croisement des individus.

Algorithmes Evolutionnaires.

- Macro-Algorithme :
  - On créé un premier ensemble de solutions : la Population Initiale.
  - On évalue chaque individu de cette population.
  - On répète :
    - Sélection des meilleurs Individus.
    - Lancement des opérateurs génétiques pour créer une nouvelles population.
    - Evaluation de la nouvelle population.
  - Jusqu'à ce que le critère soit atteint.

Algorithmes Evolutionnaires.

- Opérateurs génétiques :
  - Mutation des Individus
  - Croisement des Individus
- Type d'algorithmes utilisés très souvent.
- Difficile à paramétrer

Colonies de Fourmis.

- Inspiré par la quête de nourriture d'une colonie de fourmis et de leur comportement social.
- Une fourmi : peu de mémoire !
- Pour palier à cela, leur société est organisée autour du travail collaboratif.
- Communication par des stimuli :
  - Dépôt d'une phéromone par la fourmi pour signaler le chemin vers une source de nourriture.
  - Cette phéromone est volatile !

Colonies de fourmis.

- L'algorithme créé une population de fourmis qui va se déplacer dans l'ensemble de solution représenté sous la forme d'un graphe.
- A chaque déplacement, on collecte plus d'information sur f.
- Chaque fourmi choisi son chemin aléatoirement mais la probabilité de choix est pondérée par la concentration de phéromone.
- Plus la concentration est forte, plus la fourmi aura tendance à choisir son chemin.

Colonies de Fourmis.

- Pour appliquer cette métaheuristique à une problème, on doit :
  - Représenter le problème comme un problème de recherche sur un graphe.
  - Un retour positif sur le choix d'un chemin doit être défini (phéromone).
  - Une stratégie gloutonne doit être défini pour construire la solution au fur et à mesure.