Algorithmes de recherche locale (2)

Résolution de Problèmes d'Optimisation
Master 1 informatique I2L / WeDSci

Sébastien Verel
verel@univ-littoral.fr
http://www-lisic.univ-littoral.fr/~verel

Université du Littoral Côte d'Opale
Laboratoire LISIC
Equipe OSMOSE

Plan

Optimization

Inputs

- Search space : Set of all feasible solutions,

$$\mathcal{X}$$

- Objective function : Quality criterium

$$f : \mathcal{X} \to \mathbb{R}$$

Goal

Find the best solution according to the criterium

$$x^{\star} = \text{argmax } f$$

But, sometime, the set of all best solutions, good approximation of the best solution, good 'robust' solution...

Contexte

Black box Scenario

We have only $\{(x_0, f(x_0)), (x_1, f(x_1)), ...\}$ given by an "oracle"
No information is either not available or needed on the definition of objective function

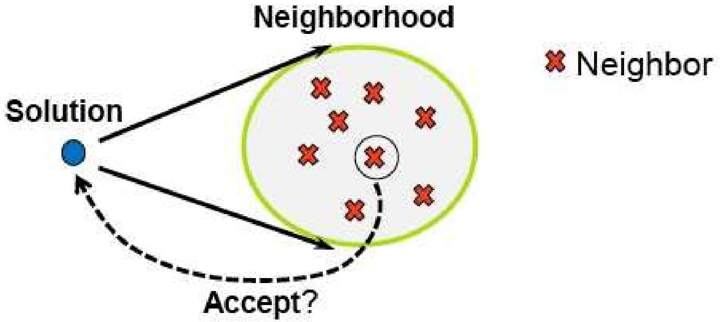
- Objective function given by a computation, or a simulation
- Objective function can be irregular, non differentiable, non continous, etc.

Typologie des problèmes

- Espace de recherche très large dont les variables sont discrètes (cas NP-complet) : optimisation combinatoire
- Espace de recherche dont les variables sont continues : optimisation numérique

Stochatic algorithms with unique solution (Local Search)

- $\mathcal{X}$ set of solutions (search space)
- $f : \mathcal{X} \to \mathbb{R}$ objective function
- $\mathcal{V}(x)$ set of neighbor's solutions of x

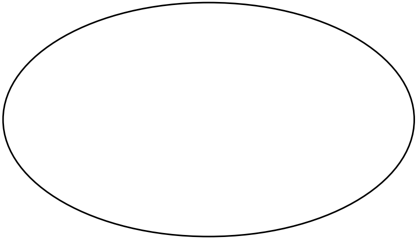


Idée derrière la stratégie locale

Pourquoi une stratégie locale de recherche basé sur un voisinage ?

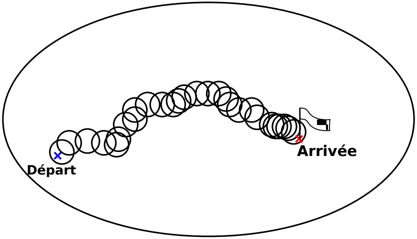
Idée derrière la stratégie locale

Pourquoi une stratégie locale de recherche basé sur un voisinage ?



Idée derrière la stratégie locale

Pourquoi une stratégie locale de recherche basé sur un voisinage ?

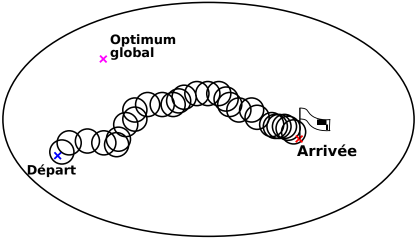


Réduire la résolution du problème global
à une suite de problèmes de petite taille

- Avantage : réduire la complexité

Idée derrière la stratégie locale

Pourquoi une stratégie locale de recherche basé sur un voisinage ?



Réduire la résolution du problème global
à une suite de problèmes de petite taille

- Avantage : réduire la complexité
- Risque : ne pas aboutir à une solution optimale

Recherche Locale Aléatoire (marche aléatoire)

Heuristique d'**exploration** maximale

Recherche locale aléatoire
Marche aléatoire

Choisir solution initiale $x \in \mathcal{X}$
Evaluer x avec f
repeat
  choisir $x' \in \mathcal{V}(x)$ aléatoirement
  Evaluer x' avec f
  $x \leftarrow x'$
until Nbr d'éval. $\leq$ maxNbEval

- Algorithme inutilisable en pratique
- Algorithme de comparaison
- Opérateur local de base de nombreuses métaheuristiques

Hill-Climber (HC)

Heuristique d'**exploitation** maximale.

Hill Climber (best-improvement)

Choisir solution initiale $x \in \mathcal{X}$
Evaluer s avec f
repeat
  Choisir $x' \in \mathcal{V}(x)$ telle que $f(x')$ est maximale
  if x' strictement meilleur que x **then**
    $x \leftarrow x'$
  end if
until x optimum local

- Algorithme de comparaison
- Opérateur local de base de métaheuristique

Hill-Climber (HC)

Quel est l'inconvénient majeur du Hill-Climbing ?

Optimum local et global

Optimum local

Etant donné $(\mathcal{X}, f, \mathcal{V})$, f à maximiser.

$x^\star$ est un optimum local ssi pour tout $x \in \mathcal{V}(x^\star)$, $f(x) \leq f(x^\star)$

Optimum local strict

Etant donné $(\mathcal{X}, f, \mathcal{V})$, f à maximiser

$x^\star$ est un optimum local strict ssi pour tout $x \in \mathcal{V}(x^\star)$, $f(x) < f(x^\star)$

Optimum global

Etant donné $(\mathcal{X}, f, \mathcal{V})$, f à maximiser.

$x^\star$ est un optimum global ssi pour tout $x \in \mathcal{X}$, $f(x) \leq f(x^\star)$

Bassin d'attraction

Etant donné $(\mathcal{X}, f, \mathcal{V})$, f à maximiser.

Définition

Une solution x appartient au bassin d'attraction de l'optimum local $x^\star$ si et seulement si il existe une suite de solutions voisines améliorante de x à $x^\star$.
Ce type de marche est appelé *marche adaptative*.

Définition formelle

$x^\star$ un optimum local et $B(x^\star)$ le bassin d'attraction de $x^\star$.

$x \in B(x^\star)$ si et seulement si il existe une suite de solutions $x_0, x_1, \ldots x_\ell$ telles que $x_0 = x$ et $x_\ell = x^\star$ et pour tout $i \in \{0, \ldots \ell - 1\}$, $x_{i+1} \in \mathcal{V}(x_i)$, et $f(x_i) < f(x_{i+1})$.

Remarque : ℓ est la longueur de la marche adaptative, qui est un indicateur de difficulté du problème d'optimisation.

Hill-Climber (HC)

Peut-on imaginer des situations où ce n'est qu'un inconvénient relatif ?

Optimum local

Exercice

- Trouver les maxima locaux (strict et non strict)

0	7	12	18	14	9	9	14	16	14
0	4	8	11	10	7	9	14	17	16
0	4	6	7	7	5	5	8	10	10
0	7	10	11	10	7	4	2	3	4
0	9	13	14	12	9	5	2	1	1
0	10	13	14	12	8	5	3	3	4
0	9	11	11	9	6	6	7	7	7
0	6	7	7	6	6	8	8	10	12
0	5	7	6	4	3	4	4	13	16
0	4	6	6	4	2	0	4	14	16

Uncontrainst Quadratic Binary Problem (UBQP)

Exercice

Le problème Uncontrainst Quadratic Binary (UBQP) est un problème d'optimisation combinatoire NP-difficile définit par :

$$\forall x \in \{0,1\}^n, \quad f(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} q_{ij} x_i x_j$$

- Créer une classe `UBQPEvalFunc` qui dérive de `EvalFunc` qui permet de calculer la fonction f. Les instances définissant la matrice (q_{ij}) sont disponibles sur le site de QAPlib.
- Comparer les performances des recherches locales Hill-Climber best-improvement et first-improvement sur ce problème UBQP.

Metaheuristics

Random search / Hill Climbing

Algorithm 1 Random walk

Choose randomly initial solution $x \in \mathcal{X}$
repeat
  Choose $x' \in \mathcal{V}(x)$ randomly
  $x \leftarrow x'$
until ...

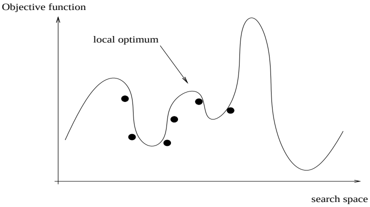
Algorithm 2 Hill-climbing

Choose randomly initial solution $x \in \mathcal{X}$
repeat
  Choose $x' \in \mathcal{V}(x)$ such as $f(x')$ is maximal
  if $f(x')$ is better than $f(x)$ **then**
    $x \leftarrow x'$
  end if
until x local optimum

Metaheuristics

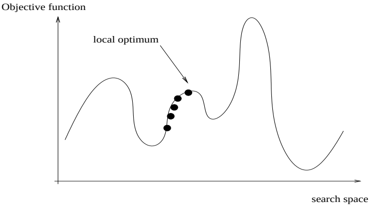
Random search / Hill Climbing

Random walk



maximal exploration, diversification

Hill-climbing



maximal exploitation, intensification

Tradeoff between Exploration / Exploitation

Escape from local optima, etc.
$\Longrightarrow$ simulated annealing, tabu search, Iterated Local Search, Variable Neighborhood Search

Recuit Simulé (Simulated Annealing)

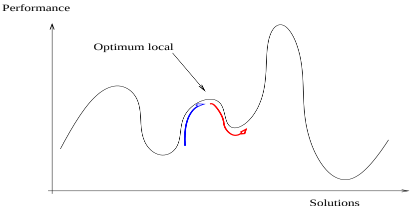
Utilisé depuis les années 80,

- Metropolis (1953) simulation du refroidissement de matériaux (Thermodynamique)
- Kirkpatrick, Gelatt, Vecchi (IBM 1983) utilisation pour la résolution de problème d'optimisation.

But : échapper aux optima locaux
Principe : probabilité non nulle de sélection d'une solution voisine dégradée

Recuit Simulé : analogie

Système physique	Problème d'optimisation
Energie	fonction objectif
Etats du système	solution
États de basse énergie	bonne solution
Température	paramètre de contrôle



Recuit Simulé
(ici pour un problème de maximisation)

Choisir solution initiale $x \in \mathcal{X}$ et temperature initiale T
repeat
  choisir aléatoirement $x' \in \mathcal{V}(x)$, $\Delta = f(x') - f(x)$
  if $\Delta \geqslant 0$ **then**
    $x \leftarrow x'$
  else
    u nombre aléatoire de $[0, 1]$
    if $u < e^{\frac{\Delta}{T}}$ **then**
      $x \leftarrow x'$
    end if
  end if
  update temperature T
until Critère d'arrêt vérifié

Recuit Simulé : remarques

Si $\Delta \leqslant 0$ alors la probabilité $\exp(\frac{\Delta}{T})$ est proche de 0 lorsque :

- la différence $|\Delta = f(x') - f(x)|$ est grande
- la temperature est petite

Conséquences :

- lorsque température grande (début de la recherche) :
  $\longrightarrow$ recherche aléatoire
- lorsque température petite (fin de la recherche) :
  $\longrightarrow$ Hill-Climbing

Recuit Simulé : température initiale T_0

Première manière

Selon le choix d'un expert
ou tester plusieurs valeurs T_0

Deuxième manière

Meilleur valeur parmi un ensemble possible I_T

Troisième manière : taux d'acceptation initial

Evaluer $\Delta_0 = f(x_0^{'}) - f(x_0)$:

- Choisir p (grand si possible) solutions aléatoires initiales x_0 et une solution voisine $x_0^{'}$
- calculer la moyenne de Δ_0 sur l'échantillon

Température initiale T_0 telle que $\tau_0 = e^{\frac{\Delta_0}{T_0}}$ désiré :
qualité “médiocre” ($\tau_0 = 0.50$) : démarrage à haute température
qualité “bonne” ($\tau_0 = 0.20$) : démarrage à basse température

Recuit Simulé : décroissance de "température"

Pallier

Evolution par pallier :

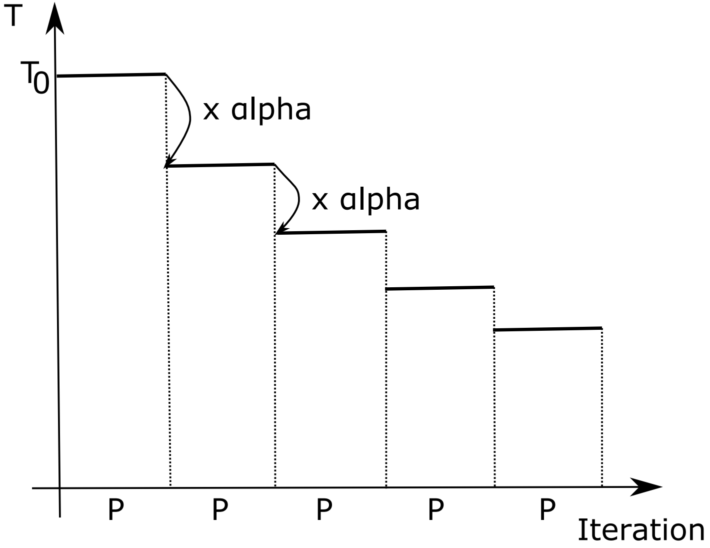
- Pendant P tentatives (itérations), la température est constante
- Changement tous les P itérations

Variation géométrique

Décroissance suivant une loi géométrique entre deux paliers k et $k+1$: $T_{k+1} = \alpha T_k$

souvent $0.8 \leq \alpha < 1.0$, et très souvent $\alpha \geqslant 0.95$

Recuit Simulé : décroissance de "température"



Recuit Simulé : Critère d'arrêt

Plusieurs possibilités :

- Le budget (temps de calcul, énergie) alloué est épuisé
- La qualité obtenue est "bonne", suffisante
- Plus d'amélioration observée, et espérée

Recuit Simulé : Remarques

- Toutes ces indications ne sont pas universelles :
  L'analyse du problème et l'expérience de concepteur permettent de les adapter
- Recherche des meilleurs paramètres est un sujet de recherche (apprentissage des meilleurs paramètres...)

Premières Applications : dans le placement de circuits électroniques

Recuit Simulé : Bibliographie

E. Aarts, J. Korst : "Simulated Annealing and Boltzmann machine" John Wiley, New-York 1989
P. Siarry : "La méthode du recuit simulé : théorie et application" ESPCI - IDSET , 10 rue Vauquelin, Paris 1989

Recherche Tabou (Tabu Search)

Introduite par Glover en 1986 :
"Future paths for Integer Programming and Links to Artificial Intelligence", Computers and Operations Research, 5 :533-549, 1986.

But : échapper aux optima locaux
Principe : Introduction d'une notion de mémoire dans la stratégie d'exploration

Interdiction de reprendre des solutions déjà (ou récemment) rencontrées

Recherche Tabou (Tabu Search)

Choisir solution initiale $x \in \mathcal{X}$
Initialiser Tabou M
repeat
  choisir $x' \in \mathcal{V}(x)$ telle que :
    x' est la meilleure solution voisine non taboue, ou
    x' est la meilleure solution voisine taboue avec critAsp(x')

  $x \leftarrow x'$
  update Tabou M
until Critère d'arrêt vérifié

Recherche Tabou : mémoire des tabous

Les tabous sont souvent des mouvements tabous pendant une durée

exemple : problème maxsat avec $n = 6$
$M = (0, 3, 0, 0, 0, 0)$
le deuxième bit ne peut être modifié pendant 3 itérations.
$M = (1, 2, 0, 0, 2, 5)$
seuls bits non tabou 3 et 4

Lorsqu'un mouvement est effectué :
interdiction pendant k itérations

Exercice Tabou

Exercice

Exécuter un Tabou sur un problème MAX-SAT.

Recherche Tabou : mémoire des tabous

Lorsqu'un mouvement est effectué :
interdiction pendant k itérations
Si k trop faible, tabou peu efficace
Si k trop grand, les solutions sont "à flanc de coteau".

$\longrightarrow$ Stratégie de diversification

Robust Tabu Search [Taillard, 91]

$k =$ valeur aléatoire entre $[k_{min}, k_{max}]$ (à chaque interdiction)
avec k_{min}, k_{max} proportionnel à la dimension n.

Recherche Tabou : Critère d'aspiration

critAsp(x') : Enlever le caractère tabou d'une solution

critAsp(x') $=$ *true* ssi x' est la meilleure solution jamais rencontrée : $f(x') > f(x_{\text{best known}})$

Recherche Tabou : Bibliographie

Glover *et al* : "Tabu Search" Kluwer Academic Publishers, 1997

Iterated Local Search

Principe

Une fois la solution courante dans un optimum local,
Perturbation (grande modification) de la solution courante
pour initier une nouvelle recherche locale à partir de celle-ci.

Iterated Local Search (ILS)

Algorithme

Choisir solution initiale $x \in \mathcal{X}$
$x \leftarrow \text{localSearch}(x)$
repeat
  $x' \leftarrow \textit{perturbation}(x)$
  $x' \leftarrow \text{localSearch}(x')$
  Si accept(x, x') Alors
    $x \leftarrow x'$
  FinSi
until Critère d'arrêt vérifié

Iterated Local Search : paramètres

• localSearch : Très souvent, localSearch est un Hill-Climber

• Perturbation : le principe consiste à modifier aléatoirement un certain nombre de variables.
Par ex. : changement de k bits aléatoirement.
k est alors appelé la "perturbation strength"

• accept : Souvent le critère d'acceptation consiste à sélectionner si la solution est meilleure.
Mais d'autres choix sont possible (critère de "recuit", etc.)

Remarque : Tous ces paramètres peuvent être changer selon l'instance du problème à résoudre...

Travaux pratiques

Comparer les performances du recuit simulé et l'Iterated local search sur le problème knapsack

Variable Neighborhood Search (VNS)

[Mladenovic, N., Hansen, P. (1997).]

On suppose avoir un ensemble fini de voisinages $\{V_1, \ldots, V_p\}$

Choisir solution initiale $x \in \mathcal{X}$
$i \leftarrow 1$
repeat
  $x' \leftarrow \text{localSearch}_{V_i}(x)$
  if accept(x, x') **then**
    $x \leftarrow x'$
    $i \leftarrow 1$
  else
    $i \leftarrow i + 1$
    if $i > p$ **then**
      arrêt $\leftarrow$ True
    endIf
  endIf
until Critère d'arrêt vérifié

Variable Neighborhood Search (VNS) : Paramètres

Paramètres

- Voisinages :

Ensemble de voisinages souvent de plus en plus grand.
Par ex. $V_k(x) = \{\text{dist}(x', x) \leqslant k \mid x' \in \mathcal{X}\}$

- Ordre des voisinages :

Par défaut, selon l'ordre "naturel" $1, \ldots, p$
Ou selon une méthode de sélection (score du voisinage), par apprentissage par renforcement, etc.

Conclusion

- Problèmes d'optimisation combinatoire fréquents dans l'industrie (et fondamentaux en informatique théorique)
- Métaheuristiques de recherche locale :
  $\longrightarrow$ Recherche aléatoire
  $\longrightarrow$ Hill-climbing Best-improvment ou first-ascent
  $\longrightarrow$ Recuit simulé
  $\longrightarrow$ Recherche tabou
  $\longrightarrow$ Iterated Local Search, Variable Neighborhood Search
- Paysage de Fitness : métaphore qui permet
  $\longrightarrow$ l'analyse du lien entre métaheurisque et problème
  $\longrightarrow$ imaginer ou "apprendre" de nouvelles metaheurisques,
- Propriété principale :
  $\longrightarrow$ rugosité : optima locaux, structure de corrélation
  $\longrightarrow$ neutralité : réseau de neutralité

Travaux pratiques

Comparer la longueur des marches adaptatives sur le problème knapsack selon le type de fonction de pénalité (linéaire et quadratique par exemple)