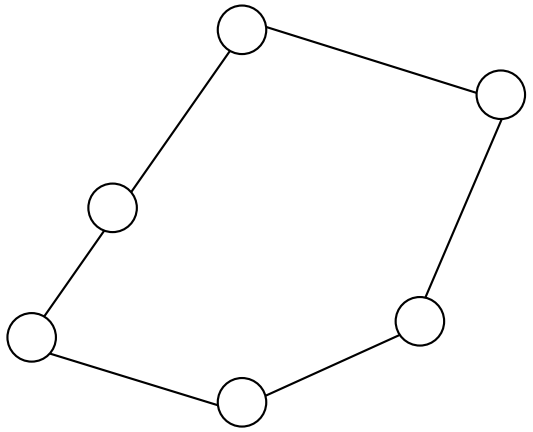
Optimisation combinatoire

- intéresse de nombreux chercheurs
- problèmes difficiles à résoudre de manière exacte
  - exemple : PVC

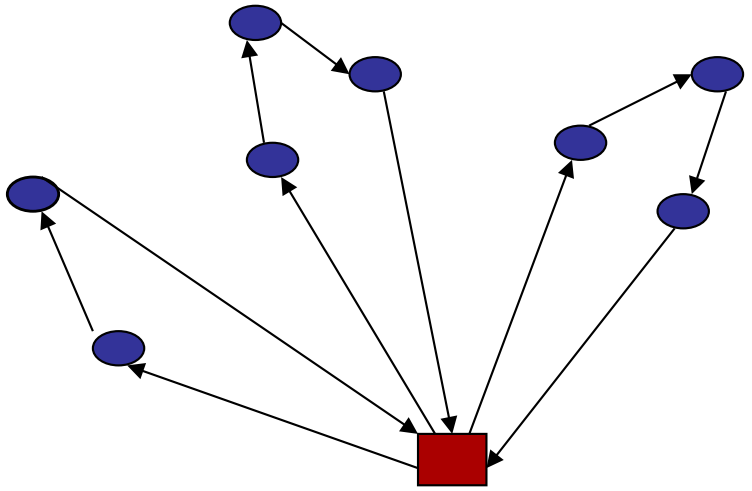


- méthodes traditionnelles (SÉP)
  - coûteuses en termes de temps de calcul
- solution
  - méthodes de résolution approximatives (heuristiques)

Heuristiques

- but
  - déterminer une approximation aussi près que possible de l'optimum d'un problème en un temps raisonnable

- 2 types
  - méthodes constructives
    - exemple : méthode du plus proche voisin pour le PVC
    - très rapide
    - qualité des solutions laisse à désirer

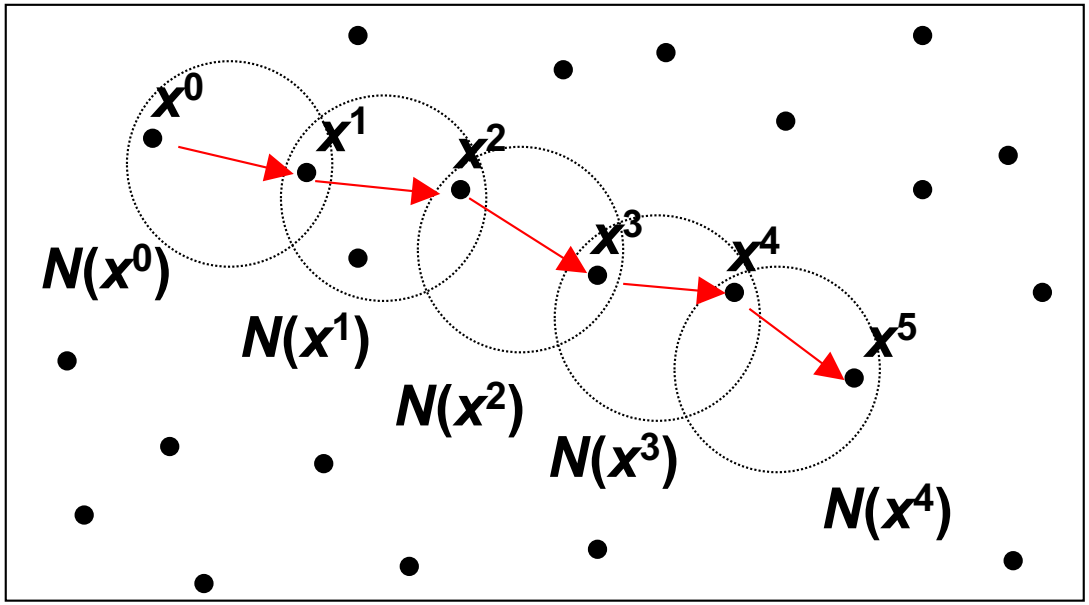


Heuristiques

- 2 types (suite)
  - méthodes de recherche locale

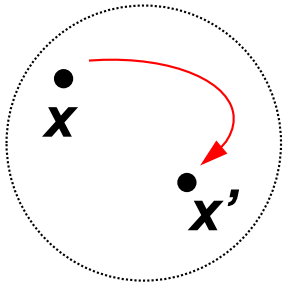
N(x) : ensemble de tous les voisins de x
x' : solution voisine de x

X : ensemble des solutions



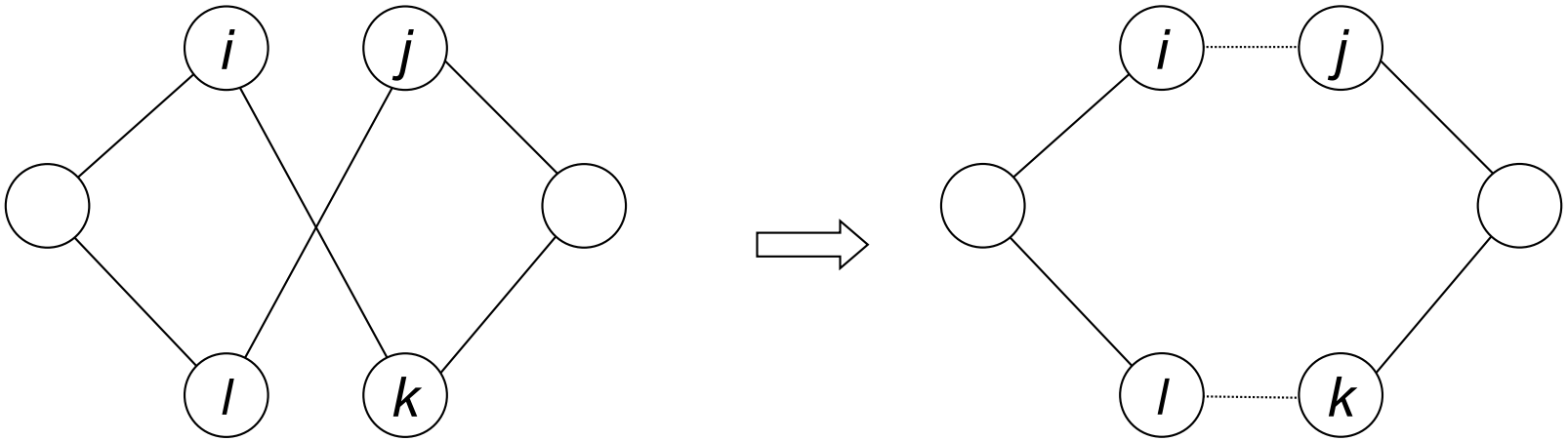
- passage d'une solution $\boldsymbol{x} \in \mathbf{X}$ à une solution $\boldsymbol{x'} \in \boldsymbol{N(x)}$

$F(x') = \min F(x''), x'' \in N(x)$



Heuristiques

- 2 types (suite)
  - méthodes de recherche locale
    - exemple : méthode d'échange 2-opt pour le PVC
      - retirer 2 arêtes (i, j) et (k, l) et les remplacer par les arêtes (i, k) et (j, l)

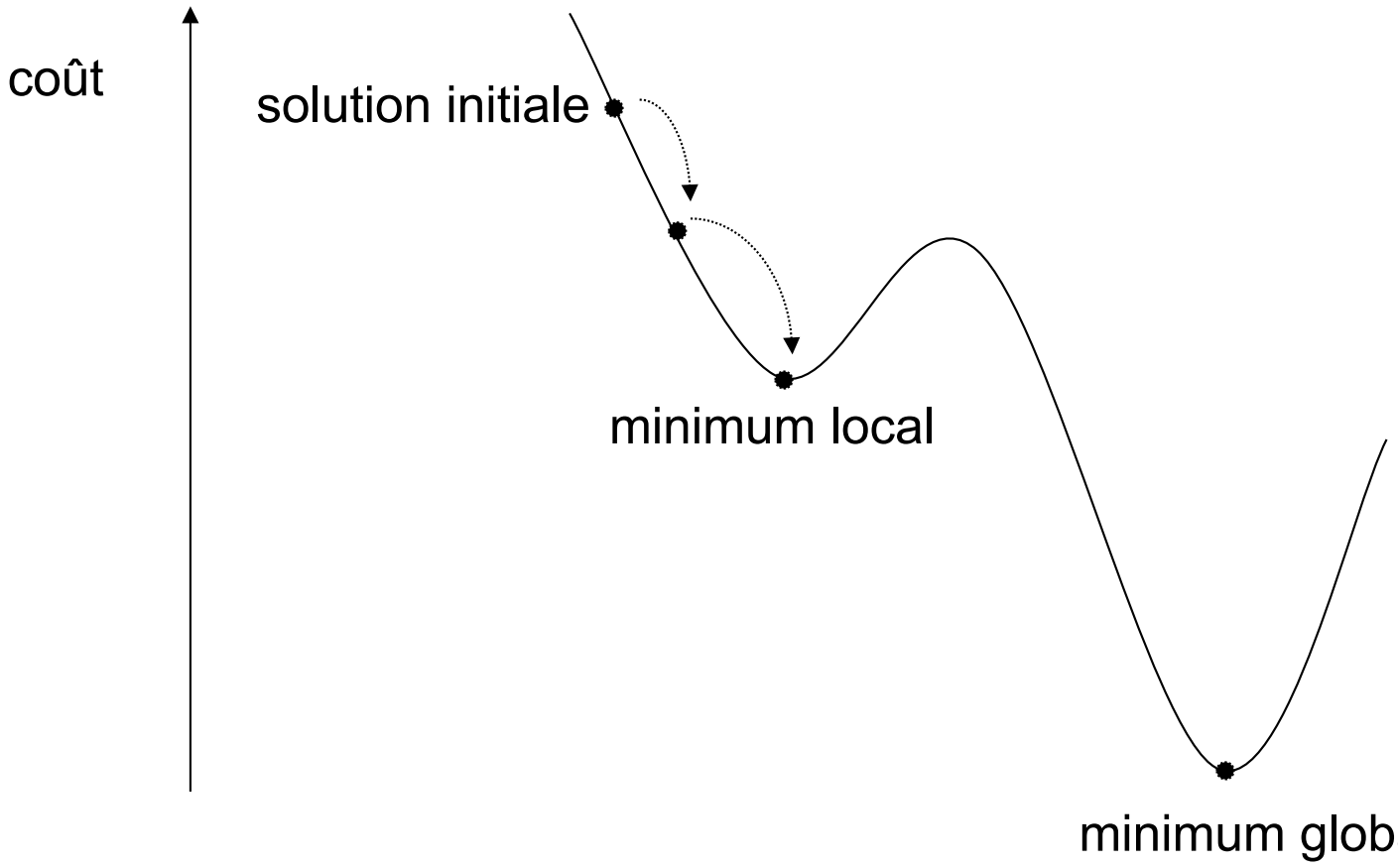


- plus puissantes que les méthodes constructives
- plus coûteuses en termes de ressources informatiques

Heuristiques

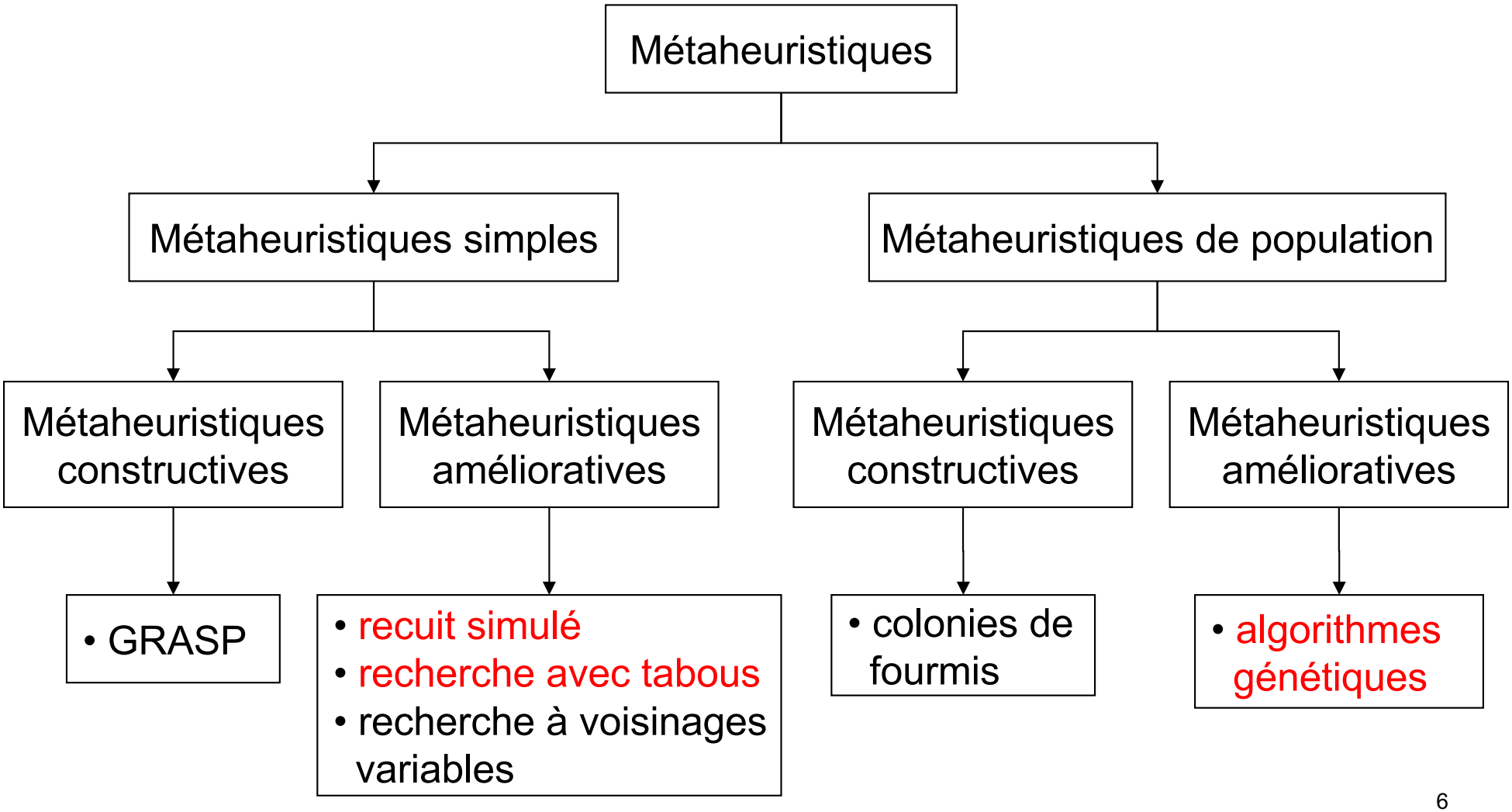
- carence des méthodes de recherche locale
  - incapables de progresser au-delà du premier optimum local rencontré

Problème de minimisation



Métaheuristiques

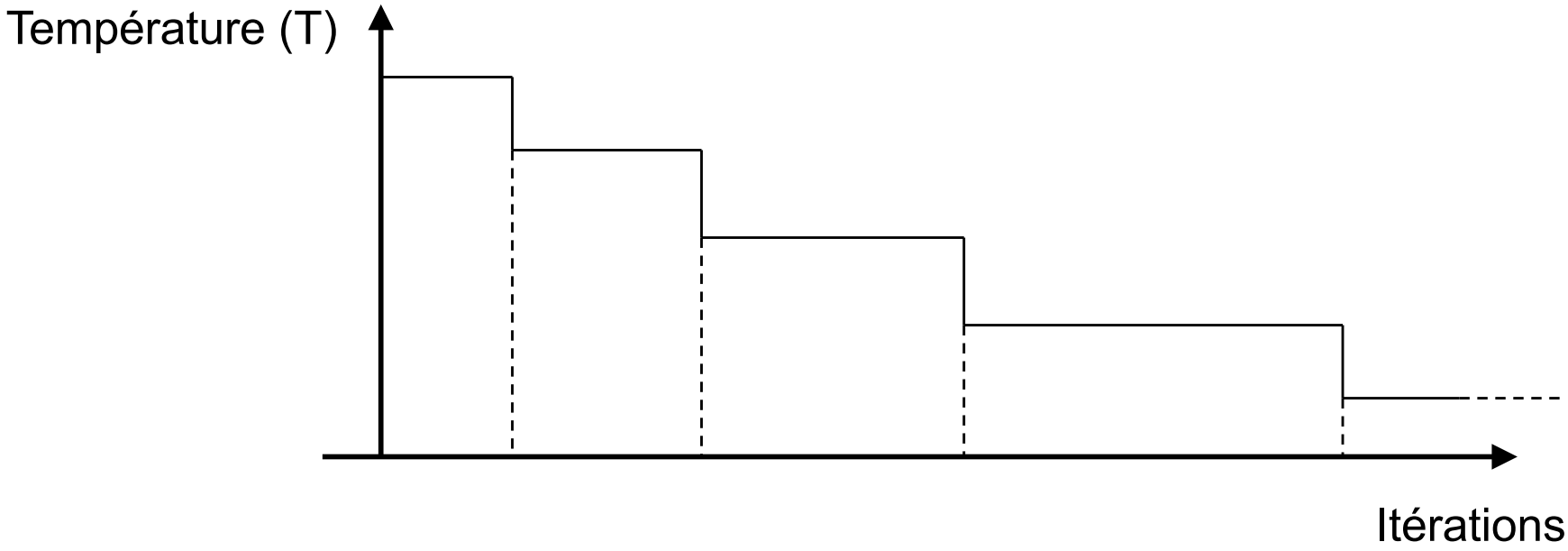
- heuristiques qui permettent de surmonter l'obstacle de l'optimalité locale



RECUIT SIMULÉ

1. Recuit simulé (RS)

- Metropolis et al. (1953)
- phénomènes de recuit thermodynamique
- schéma de refroidissement



Recuit simulé (RS)

- Pour un palier donné, la probabilité d'accepter un nouvel état est donnée par

$$p(\lambda E) = \exp\left(\frac{-\lambda E}{kT}\right)$$

où λE augmentation d'énergie

T température

k constante

– Si $\lambda E \leq 0$, le nouvel état du système est accepté

– Si $\lambda E > 0$, le nouvel état est accepté avec probabilité $p(\lambda E)$

Recuit simulé (RS)

- Analogies entre la simulation thermodynamique et l'optimisation combinatoire

Simulation thermodynamique	Optimisation combinatoire
états du système	solutions réalisables
énergie	fonction objectif
changement d'état	solution voisine
température	paramètre
état stable	minimum local

Recuit simulé (RS)

ÉCOLE POLYTECHNIQUE MONTRÉAL

Algorithme de RS

1. Solution initiale x de coût $F(x)$
2. Tirer x' au hasard dans $N(x)$
   - Si $F(x') \leq F(x)$, accepter x' comme nouvelle solution initiale et retourner à l'étape 1.
   - Sinon, accepter x' avec probabilité $p(x')$

$$p(x') = \exp\left(\frac{-[F(x') - F(x)]}{T}\right)$$

   - Tirer au hasard un nombre r dans l'intervalle [0,1]
     - Si $r \leq p(x')$, accepter x' comme nouvelle solution initiale et retourner à l'étape 1.
     - Sinon, répéter l'étape 2 avec un nouveau x' tiré dans $N(x)$.

Recuit simulé (RS)

ÉCOLE POLYTECHNIQUE MONTRÉAL

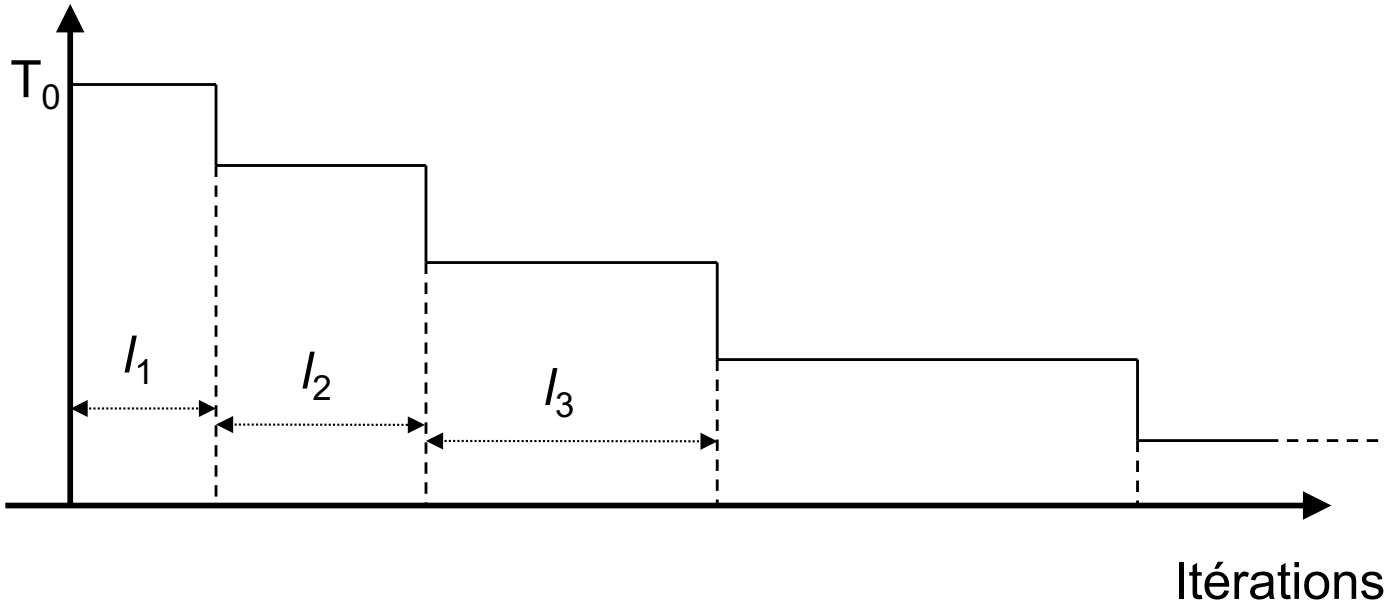
Différence entre une méthode de recherche locale et le RS

- Recherche locale
  - solution courante $x \in X$
  - structure du voisinage N(x)
    - types de transformations qui permettent de passer de x à $x' \in$ N(x)

- RS
  - exploration de X n'est pas déterministe
    - N(x) n'est pas entièrement évalué pour trouver le meilleur x'
    - x' est tiré au hasard dans N(x) et on applique la règle de Metropolis et al.

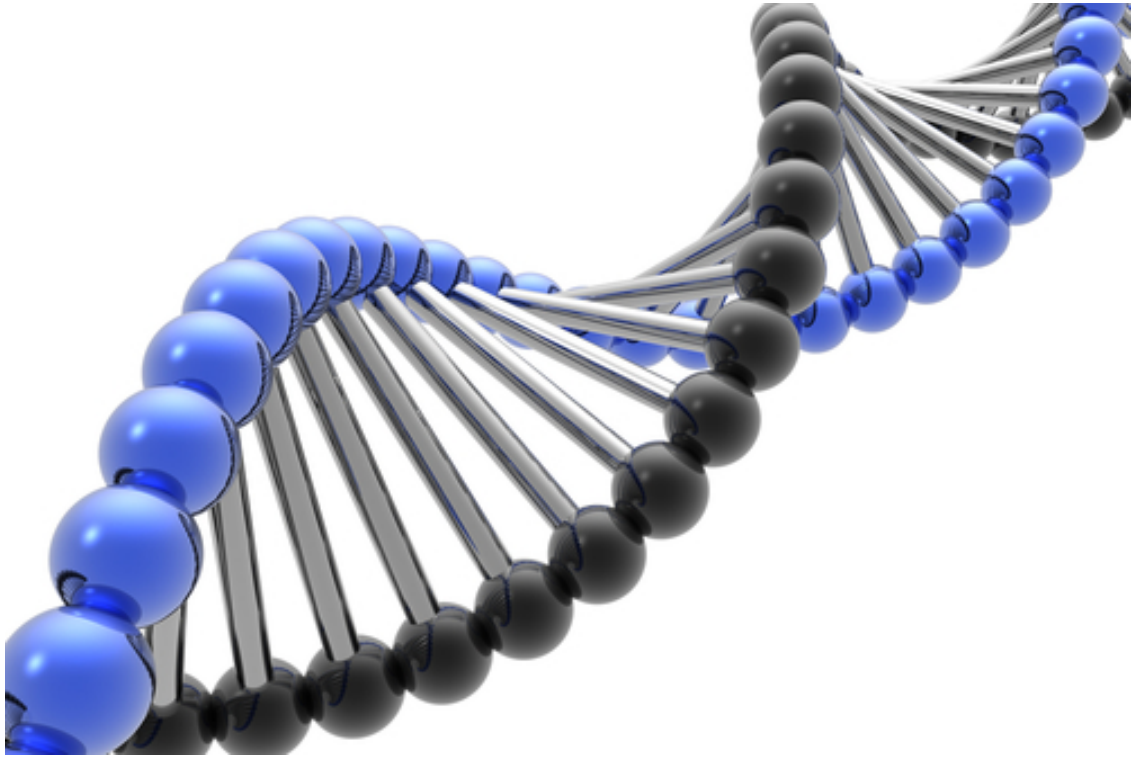
Recuit simulé (RS)

- Remarques
  - RS n'est pas bloqué par un optimum local
    - on peut accepter *x'* même si F(*x'*) > F(*x*)
  - Paramètres
    - longueur des paliers (l_1, l_2, l_3,…), température initiale (T_0)
    - critère d'arrêt (nombre d'itérations, stagnation)



Recuit simulé (RS)

- Remarques
  - seule métaheuristique pour laquelle il existe une preuve de convergence à l'optimum
    - sous certaines hypothèses
      - si T → 0 en même temps que le nombre d'itérations → ∞
  - a suscité beaucoup d'intérêt fin 80' et début 90' à cause de la "convergence garantie"
  - robuste et relativement efficace si structure de voisinage bien choisie
  - facile à mettre en œuvre



ALGORITHMES GÉNÉTIQUES

2. Algorithmes génétiques (AG)

- Fogel et al. (1966)

- inspirés des processus de sélection naturelle et évolution des espèces (Darwin)

- principes de base
  - sélection
  - reproduction
  - mutation

- différents du RS et de la recherche avec tabous
  - AG explorent X en analysant un groupe de solutions (population) et pas une solution unique

Algorithmes génétiques (AG)

ÉCOLE POLYTECHNIQUE MONTRÉAL

- idée de base
  - combiner des solutions (parents) pour générer de nouvelles solutions (enfants)
    - avec les bonnes caractéristiques des parents
    - exempts de leurs mauvaises caractéristiques

- processus itératif
  - une génération = une itération
  - à chaque génération, la population courante est transformée à l'aide d'opérateurs génétiques
    - sélection
      - choix aléatoire des paires de parents qui se reproduiront
    - reproduction
      - 2 parents sont combinés → 2 enfants
    - mutation
      - altération aléatoire des enfants (faible probabilité)
      - enfants mutés = nouvelle population

Algorithmes génétiques (AG)

ÉCOLE POLYTECHNIQUE MONTRÉAL

Algorithme AG

1. Trouver une population initiale

2. Tant que le nombre maximum d'itérations n'est pas atteint, faire
   - sélection des parents
   - reproduction
   - mutation

   et on garde en mémoire la meilleure solution rencontrée.

Algorithmes génétiques (AG)

ÉCOLE POLYTECHNIQUE MONTRÉAL

Illustration des opérateurs

(optimisation d'une fonction, n variables 0-1)

- sélection
  - 2 solutions → 2 vecteurs de variables binaires

    A = (0 0 1 1 0 1 1 0 1 0)

    B = (1 0 0 1 1 0 1 0 0 1)
- reproduction (crossover à un point)
  - une coupure choisie aléatoirement

Algorithmes génétiques (AG)

Illustration des opérateurs

(optimisation d'une fonction, n variables 0-1)

- sélection
  - 2 solutions → 2 vecteurs de variables binaires

    A = (0 0 1 1 | 0 1 1 0 1 0)

    B = (1 0 0 1 | 1 0 1 0 0 1)
- reproduction (crossover à un point)
  - une coupure choisie aléatoirement

Algorithmes génétiques (AG)

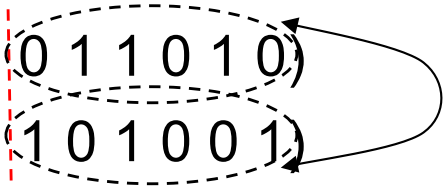
Illustration des opérateurs

(optimisation d'une fonction, n variables 0-1)

- sélection
  - 2 solutions → 2 vecteurs de variables binaires

A = (0 0 1 1 0 1 1 0 1 0)

B = (1 0 0 1 1 0 1 0 0 1)



- reproduction (crossover à un point)
  - une coupure choisie aléatoirement

    enfant 1 = (0 0 1 1 1 0 1 0 0 1)

    enfant 2 = (1 0 0 1 0 1 1 0 1 0)

- mutation
  - inverser la valeur d'une variable tirée au hasard

    Exemple : enfant 1 : aucun changement

    enfant 2 : inverser la 3e variable

    (1 0 **1** 1 0 1 1 0 1 0)

Algorithmes génétiques (AG)

Illustration des opérateurs

(optimisation d'une fonction, n variables 0-1)

- sélection
  - 2 solutions → 2 vecteurs de variables binaires

    A = (0 0 | 1 1 0 1 1 | 0 1 0)

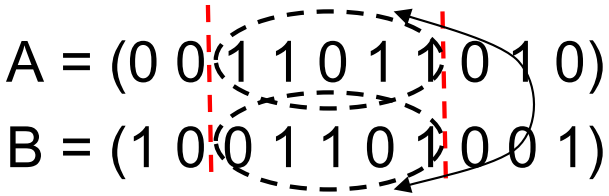
    B = (1 0 | 0 1 1 0 1 | 0 0 1)
- reproduction (crossover à 2 points)
  - 2 coupures choisies aléatoirement

Algorithmes génétiques (AG)

Illustration des opérateurs

(optimisation d'une fonction, n variables 0-1)

- sélection
  - 2 solutions → 2 vecteurs de variables binaires



- reproduction (crossover à 2 points)
  - 2 coupures choisies aléatoirement

    enfant 1 = (0 0 0 1 1 0 1 0 1 0)

    enfant 2 = (1 0 1 1 0 1 1 0 0 1)

Algorithmes génétiques (AG)

- À l'origine…
  - milieu très dogmatique (intelligence artificielle)
  - Solutions toujours représentées en vecteurs booléens (0 1 0 1 1 …)
  - toujours 2 enfants
  - crossovers standards (aléatoires) → 1 points, 2 points, …
  - mutation toujours présente
  - résultats plutôt médiocres si on suit aveuglément le dogme
  - coûteux en temps de calcul (population)
  - problèmes de convergence prématurée vers des populations où tous les individus sont identiques

Algorithmes génétiques (AG)

- Difficultés avec AG standards
  - représenter la solution en vecteurs booléens
    - pas approprié pour le PVC
    - exemple : tournée sur 8 villes
      - comment définir un crossover ayant du sens ?
      - comment les combiner ? crossover 1 point ?

$$x_1 = (1\ 2\ 3\ 4\ 5\ 6\ 7\ 8)$$

$$x_2 = (1\ 5\ 7\ 4\ 3\ 8\ 6\ 2)$$

      - 2 enfants:

$$x_A = (1\ 2\ 3\ 4\ 3\ 8\ 6\ 2)$$

$$x_B = (1\ 5\ 7\ 4\ 5\ 6\ 7\ 8)$$

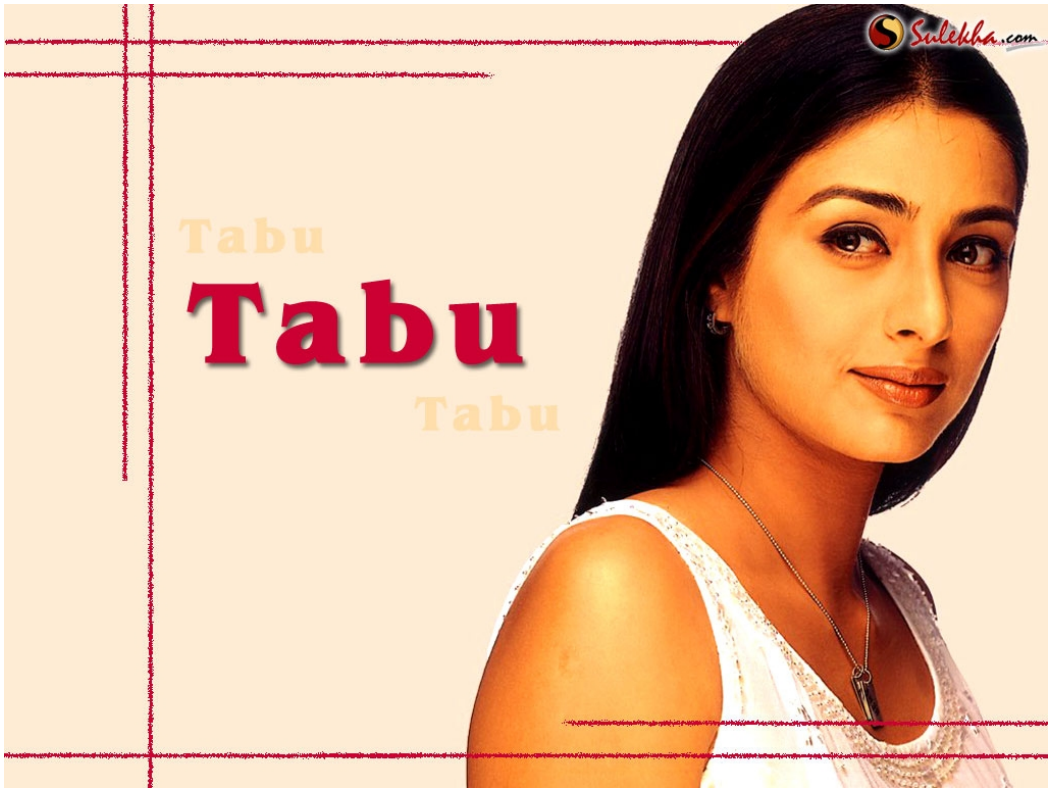
      - problème : x_A et x_B ne sont pas des solutions admissibles

    - définir un bon crossover = bien modéliser le problème
      - difficile, ne peut pas être laissé au hasard

Algorithmes génétiques (AG)

- application directe
  - rien ne guide la recherche vers des solutions valables
    - les enfants sont souvent de très mauvaises solutions car combinaison aléatoire de 2 solutions

- solutions
  - intensification : ajouter une méthode de recherche locale à la mutation
    - pour améliorer la solution au lieu de la perturber aléatoirement
  - introduire des concepts tirés d'autres approches efficaces
    - exemple : recherche avec tabous
  - bien modéliser le problème
  - s'éloigner du dogme
  - approches hybrides



RECHERCHE AVEC TABOUS

3. Recherche avec tabous (RT)

- métaheuristique qui contrôle et guide une heuristique interne de recherche locale adaptée au problème à résoudre pour lui permettre de transcender les optima locaux

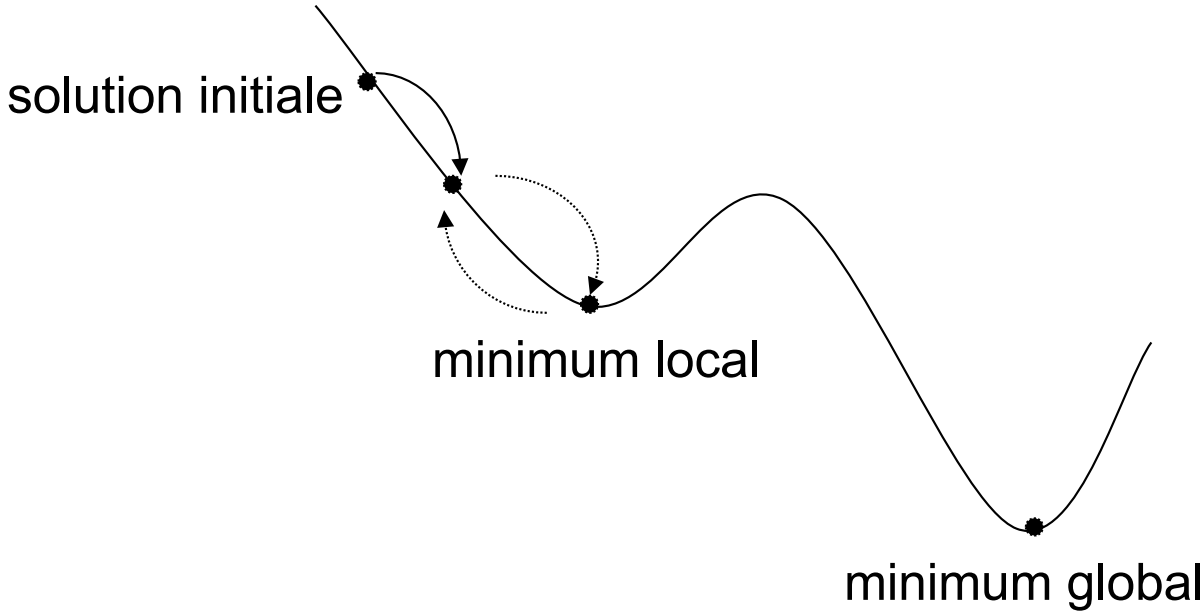
- inspirée de l'intelligence artificielle (mémoire)

- Glover (1986)

- Indépendamment, Hansen (1986)

RT : principes de base

- poursuivre la recherche même s'il y a une dégradation de la valeur de la fonction-objectif
  - risque de cycler



- solution :
  - mémoriser le cheminement récent de la recherche
  - interdire le retour vers des solutions déjà visitées (tabous)

RT : espace des solutions et voisinage

- RT est une généralisation des méthodes de recherche locale traditionnelles

1. Définir l'espace des solutions X dans lequel s'effectuera la recherche. Exemples :

   PVC → ensemble de toutes les tournées

   localisation simple → ensemble de toutes les combinaisons de localisation

2. Définir les transformations locales permettant de modifier la solution courante *x* pour en obtenir une autre différente *x'*.

   x' : solution voisine de *x*

   N(*x*) :
   - ensemble de tous les voisins de *x* obtenus suite à l'application d'une de ces transformations
   - voisinage de *x*
   - c'est un sous-ensemble de X

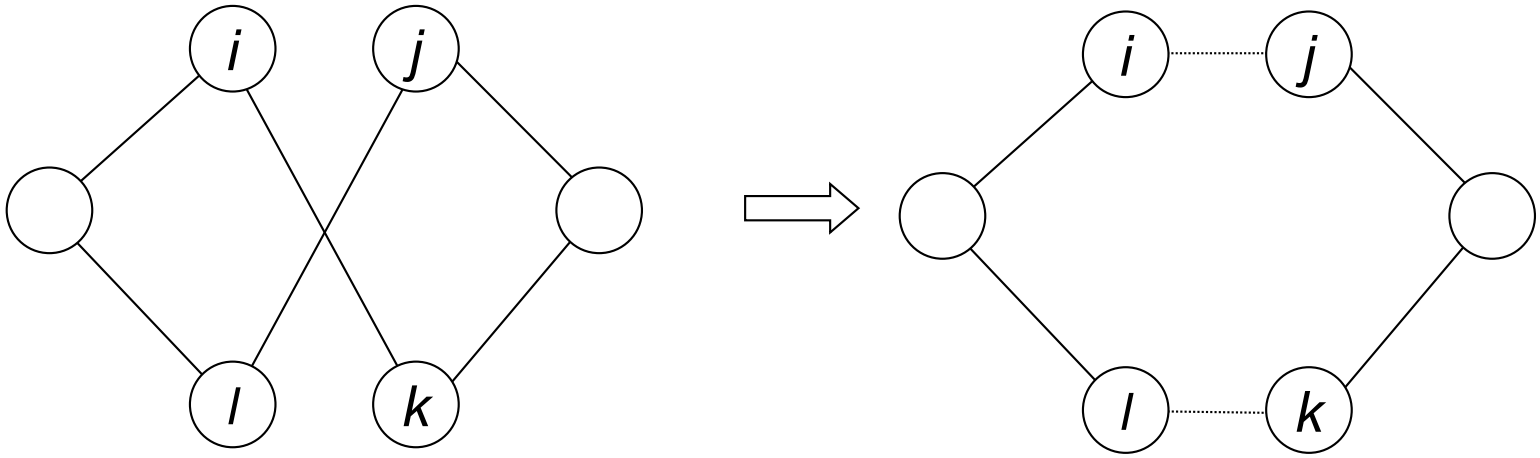
RT : mécanismes de tabous

- mémoire à court terme du processus de recherche

- possibilités :
  - liste des *t* dernières solutions visitées
    - peu utilisé
    - assez coûteux
  - liste des *t* dernières transformations effectuées et on interdit la transformation inverse
    - type de tabou le plus fréquent
  - liste des caractéristiques des *t* dernières solutions ou transformations
    - moins fréquent
    - peut être très efficace

RT : exemples de tabous

- PVC avec voisinage de type 2-opt
  - 2-opt constitue l'heuristique interne de l'algorithme de RT
  - transformations : retirer les arêtes (i, j) et (k, l) et les remplacer par (i, k) et (j, l)



  - tabous possibles :
    - interdire les tournées elles-mêmes
    - interdire la transformation inverse

      $[(i, k), (j, l)] \rightarrow [(i, j), (k, l)]$ interdit
    - interdire toute transformation impliquant (i, k) ou (j, l)

RT : critère d'aspiration

- but des tabous : empêcher de cycler
  - mais ils peuvent être trop forts :
    - solutions attrayantes sont interdites, et ce, même s'il n'y a pas de risque de cycler
    - peuvent faire stagner la recherche

- solution : critère d'aspiration (ou fonction d'aspiration)
  - mécanisme inverse qui permet de révoquer le statut TABOU d'une transformation si
    - elle donne une solution intéressante
    - elle n'introduit pas de risque de cycler

- possibilités :
  - révoquer le tabou si cela améliore la meilleure solution rencontrée
  - règle absolue : si pas de risque de cycler, ignorer le tabou

RT : algorithme générique

Soit le problème de minimiser F(x) pour $x \in X$

Posons

x	solution courante
x^*	meilleure solution rencontrée
T	liste des tabous
N(x)	voisinage de x

1. Initialiser

i. Choisir une solution intiale $x_0 \in X$.

ii. Poser $x^* = x_0$, $x = x_0$ et T = $\varnothing$.

2. Tant que critère d'arrêt pas satisfait, faire

i. Déterminer $x' \in \underline{N}(x)$ tel que F(x') = min F(x''), $x'' \in \underline{N}(x)$. Poser $x = x'$.

ii. Si F(x) < F(x^*), alors poser $x^* = x$.

iii. Mettre à jour T.

RT : critères d'arrêt

- critères d'arrêt les plus courants :
  - dès que la solution optimale est obtenue (si connue) ou une solution ayant une valeur prédéterminée
  - après un certain nombre d'itérations (ou un certain temps de calcul)
  - après un certain nombre d'itérations sans amélioration de la meilleure solution x^*

RT probabiliste

- Dans la version standard de la RT
  - la fonction F doit être évaluée pour chaque voisin de la solution courante *x*
    - peut être très coûteux

- solution
  - considérer un sous-ensemble N'(*x*) $\subset$ N(*x*) choisi aléatoirement et choisir la prochaine solution dans N'(*x*)

- avantages
  - plus rapide
  - échantillon aléatoire : propriété anti-cyclage qui complète la liste des tabous
    - on peut utiliser des listes plus courtes

- désavantages
  - on peut manquer de bonnes solutions, voir même l'optimum

RT : intensification

- usage plus poussé de la mémoire
- idée
  - intensifier l'effort de recherche dans une région de X qui paraît prometteuse
- comment ?
  - on arrête périodiquement le processus de recherche pour effectuer une phase d'intensification d'une durée limitée
- basée sur des concepts de mémoire à moyen terme
  - notions de fréquence d'apparition de certaines caractéristiques
- possibilités
  - fixer les bonnes caractéristiques des solutions
  - biaiser la fonction F pour favoriser certains types de solutions
  - augmenter la taille de l'échantillon (RT probabiliste)
  - changer l'heuristique interne pour une heuristique plus puissante

RT : diversification

- lorsque le processus de recherche tend à rester dans une portion restreinte de X

- But : rediriger le processus de recherche vers d'autres régions de X, peu ou pas explorées jusque là

- Basée sur des concepts de mémoire à long terme
  - notions de fréquence d'apparition de certaines caractéristiques

- possibilités
  - redémarrages
    - effectuer certaines transformations peu fréquentes et redémarrer le processus normal à partir de cette nouvelle solution
  - diversification continue
    - pénaliser transformations/caractéristiques fréquentes
    - favoriser transformations/caractéristiques rares

RT : gestion dynamique des tabous

- listes traditionnelles
  - circulaires (FIFO)
  - de longueur fixe

- désavantages
  - listes de longueur fixe n'empêche pas toujours cycles
  - bonne longueur : pas facile à déterminer

- nouvelles méthodes de gestion des tabous
  - faire varier la longueur des listes en cours d'exploration
  - étiquettes tabou aléatoires
    - durée aléatoire tirée entre $[T_{min}, T_{max}]$ à chaque transformation
  - liste fixe mais on tire aléatoirement la longueur active de la liste indiquant quels tabous seront maintenus

RT : évaluation du voisinage

- dans certains problèmes, la fonction F est très difficile à évaluer (ou très coûteuse)
  - évaluation devient prohibitive

- solution
  - utiliser une approximation de F pour évaluer le voisinage de *x*
  - on ne calculera la vraie valeur de F que pour une transformation choisie ou petit sous-ensemble de candidats prometteurs

- si la plupart des voisins d'une solution ont la même valeur de F, comment en choisir un ?
  - utiliser une fonction objectif auxiliaire qui mesurera un autre attribut désirable d'une solution

RT : modélisation efficace et gestion des contraintes

- **contraintes du problème**
  - pas toujours possible de tenir compte de toutes les contraintes lors de la définition de X et de N(x)
    - voisinages difficiles à construire, recherche irrégulière

      → mauvais résultats

- **solution : relaxer certaines contraintes**
  - voisinages plus simples → recherche plus efficace
  - ajouter à la fonction F un terme qui permet de pénaliser la violation de la contrainte

$$F(x) + \lambda(\text{violation de la contrainte si X})$$

où λ : poids de la pénalité

  - choix de λ
    - souvent ajusté automatiquement par l'algorithme
      - augmenté si beaucoup de violations
      - diminué si peu (ou pas) de violations

RT : conclusion

- RT est une approche qui repose sur des concepts simples à la base

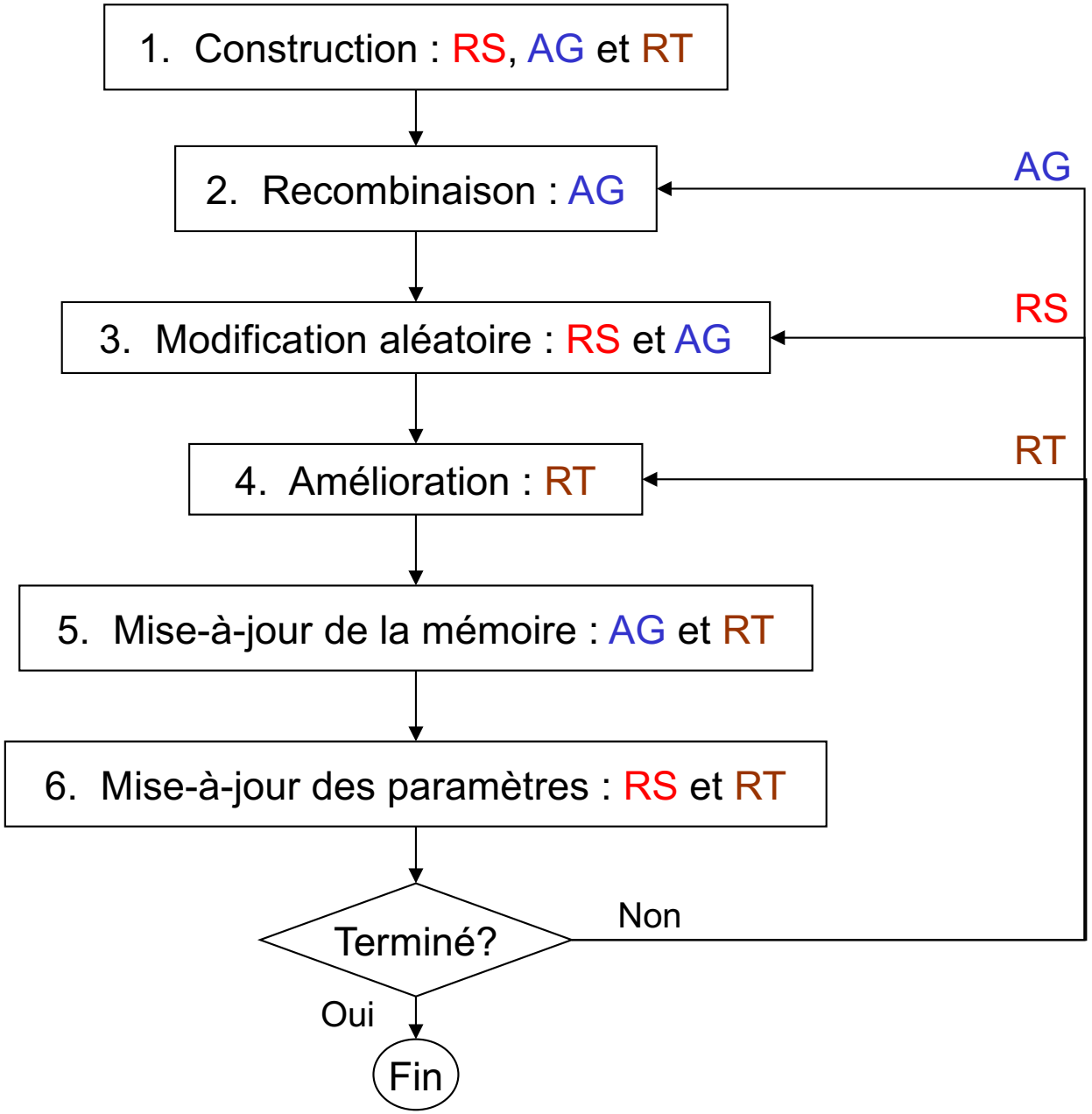
- développer un algorithme de RT efficace n'est pas trivial
  - quantité de composantes
  - flexibilité disponible pour les adapter

- éléments critiques
  - calibration des paramètres
  - modélisation du problème
    - choix de X
    - choix de N(x) = choix de l'heuristique interne

RT : applications

- problèmes de graphes
- PTV
- ordonnancement
- finance
- optimisation de fonctions continues
- gestion de portefeuille
- programmation mixte, dynamique, stochastique
- optimisation non-linéaire globale
- etc.

Résumé : Vue d'ensemble de RS, AG et RT



Résumé : Vue d'ensemble de RS, AG et RT

ÉCOLE POLYTECHNIQUE MONTRÉAL

	RS	AG	RT
Construction	solution initiale *x*	population initiale	solution initiale *x*
Recombinaison	–	reproduction	–
Modification aléatoire	*x'* tiré au hasard dans N(*x*) et accepté avec probabilité p(*x'*)	sélection et mutation	RT probabiliste
Amélioration	–	recherche locale pour optimiser les enfants	recherche locale pour améliorer *x*
Mise-à-jour de la mémoire	–	remplacement d'une génération	liste tabou
Mise-à-jour des paramètres	paramètre T, longueur des paliers	–	liste des tabous, longueur des listes, poids λ