CHAPITRE 3: LES ALGORITHMES ÉVOLUTIONNISTES, LES ALGORITHMES GÉNÉTIQUES

COURS : MÉTAHEURISTIQUES ET ALGORITHMES ÉVOLUTIONNISTES

RÉALISÉ DR. S. SLATNIA

UNIVERSITÉ DE BISKRA
Master d'informatique
2020-2021

MÉTAHEURISTIQUES ET ALGORITHMES ÉVOLUTIONNISTES

- Les métaheuristiques recherche locale consistent fondamentalement à faire **évoluer** une **configuration de référence** (**configuration courante**)
  - ❖ En la remplaçant itérativement par une nouvelle configuration choisie dans son **voisinage**.
- Il existe une famille de métaheuristiques qui consiste à faire **évoluer** un **ensemble de solutions** (**population**) : ces techniques sont appelées les algorithmes évolutionnistes [S.Doncieux].

MÉTHODES BASÉES SUR LES POPULATIONS – MÉTHODES ÉVOLUTIVES

- Les méthodes basées sur des populations de solutions sont parfois appelées méthodes évolutives parce qu'elles font évoluer une population d'individus selon des règles bien précises.
- Ces méthodes alternent entre des périodes **d'adaptation** individuelle et des périodes de **coopération** durant lesquelles les individus peuvent échanger de l'information.

Méthode Évolutive

1. Générer une population initiale d'individus
2. Tant qu'aucun critère d'arrêt n'est satisfait faire
    3. Exécuter une procédure de coopération;
    4. Exécuter une procédure d'adaptation individuelle
5. Fin du tant que

EVOLUTION NATURELLE [S.DONCIEUX]

❑ Sélection naturelle (Darwin 1859)

“ Production d’une variation et tri par sélection et élimination ”

→ Surfécondité, héritage des différences individuelles et lutte pour la survie causée par des ressources limitées.

❑ Hérédité Mendelienne

Existence d’unités de transmission de l’hérédité : facteurs (deviendront les gènes) (Mendel 1865). Importance de la recombinaison.

❑ Génétique des populations

Concept de fitness (Haldane 1924) et de paysage de fitness (Wright 1932).

EVOLUTION NATURELLE : INSPIRATION POUR UN ALGORITHME D'APPRENTISSAGE? [S.DONCIEUX]

- **Principe algorithmique**

  Sélection naturelle néo-Darwinienne.

- **Espace de recherche**

  Génome, composé de séquences d'ADN contenant des gènes, unités de transmission d'un caractère. **Opérateurs : recombinaison et mutation**.

- **Critère maximisé**

  fitness, notion de paysage de fitness contenant pics et vallées.

ÉMERGENCE DE L'IDÉE [S.DONCIEUX]

- L'évolution naturelle **explore** un paysage de fitness avec des **extrema multiples** (Wright 1932) : conduit à la vue de l'évolution comme un processus d'optimisation.

- Tentatives d'utilisation dans le cadre d'une **procédure d'optimisation** (Box 1957).

- Années 60 : apparition d'ordinateurs pour la modélisation et la simulation.

LES ALGORITHMES ÉVOLUTIONNISTES

- Les algorithmes évolutionnistes sont déjà anciens et ils sont issus historiquement de trois «écoles»

  - **L.Fogel (1966), La programmation évolutive** (**EP** Evolutionary Programming),
  - **J.Rechenberg (1973), Les stratégies évolutives** (**ES** pour Evolution Strategies),
  - **J.Holland (1975), Les algorithmes génétiques** (**GA** pour Genetic Algorithm).

- Les algorithmes génétiques ont été introduits et étudiés notamment par Holland et Goldberg. Ils **s'inspirent de la théorie de l'évolution des espèces de Darwin** [S.Doncieux].

L'INSPIRATION DARWINIENNE [S.DONCIEUX]

- Selon la **génétique** et la **théorie de l'évolution (l'union=Néo-Darwinisme)**

  - **Croisement:** Un enfant hérite son patrimoine génétique pour moitié de sa mère et pour moitié de son père (**reproduction** sexuée).

  - **Mutations:** Les enfants ne sont pas identiques aux parents car des altérations des gènes peuvent se produire .

    - Parmi les mutations, certaines peuvent être favorables et d'autres défavorables.

    - Il naît beaucoup de descendants : mais seuls les individus les mieux adaptés pourront survivre et transmettre leurs gènes à leur tour à leur descendance.

- Dans ce modèle, on observe que

  - Le **hasard** joue un rôle moteur pour **produire** de **nouveaux individus différents** de leurs parents.

  - **Sélection naturelle:** effectue le tri entre les variations favorables et les autres.

PRINCIPE D'UN ALGORITHME ÉVOLUTIONNISTE

- Nécessite le **codage** de solutions potentielles au problème posé (appelées **individus**)
- Travaille en parallèle sur une **population** d'individus
- Évalue l'**éfficacité** de chaque individu (grâce à une fonction de **fitness**) et **sélectionne** les plus **performants**
- **Génération** de nouveaux individus à l'aide d'opérateurs d'**exploration** et d'**exploitation** de l'espace de recherche :
  - **Croisement (crossover)** : création d'un nouvel individu à partir de plusieurs
  - **Mutation** : modification aléatoire d'une partie de l'individu [S.Doncieux].

PRINCIPE D'UN ALGORITHME ÉVOLUTIONNISTE

- Les algorithmes génétiques (et les algorithmes évolutionnistes en général) s'inspirent de la **théorie de l'évolution**
- Pour résoudre un **problème d'optimisation** (avec un espace de recherche et une fonction d'évaluation), un algorithme génétique fait évoluer itérativement une population de configurations.
- A chaque itération (**génération**) :

- Des **opérateurs de variation** (**mutation** et **croisement**) permettent d'engendrer de nouvelles configurations (les **enfants**) à partir des configurations de la population courante (les **parents**).

- Les configurations obtenues peuvent être **bonnes** ou **non** selon la fonction d'évaluation. Un **opérateur de sélection** élimine les configurations les moins bonnes.

PRINCIPE D'UN ALGORITHME ÉVOLUTIONNISTE

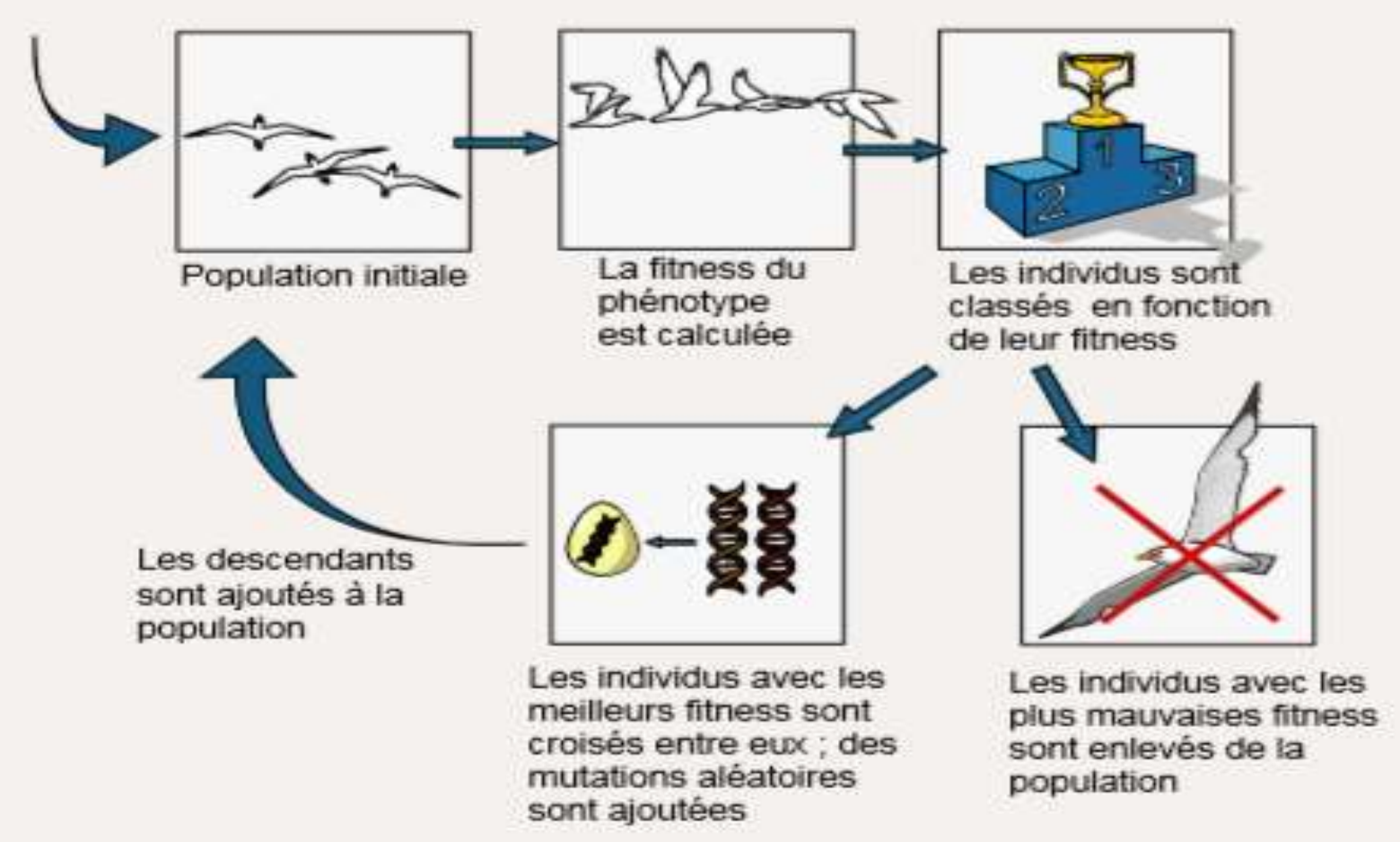
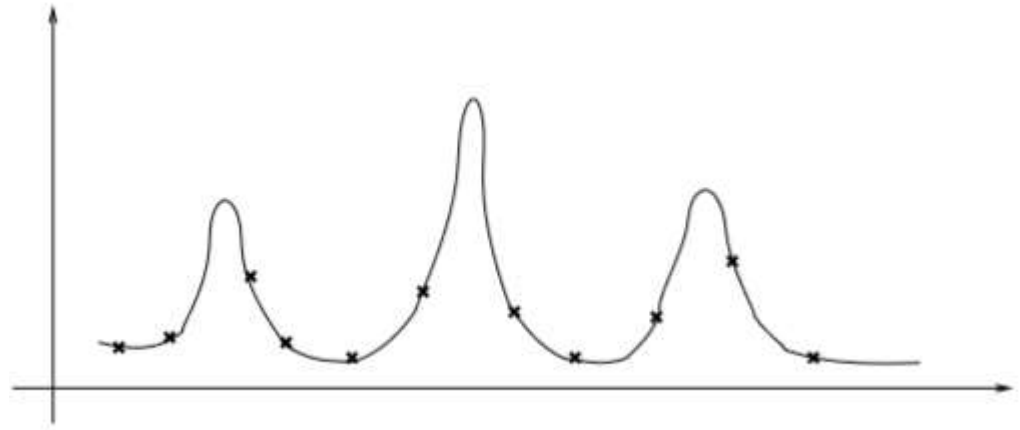
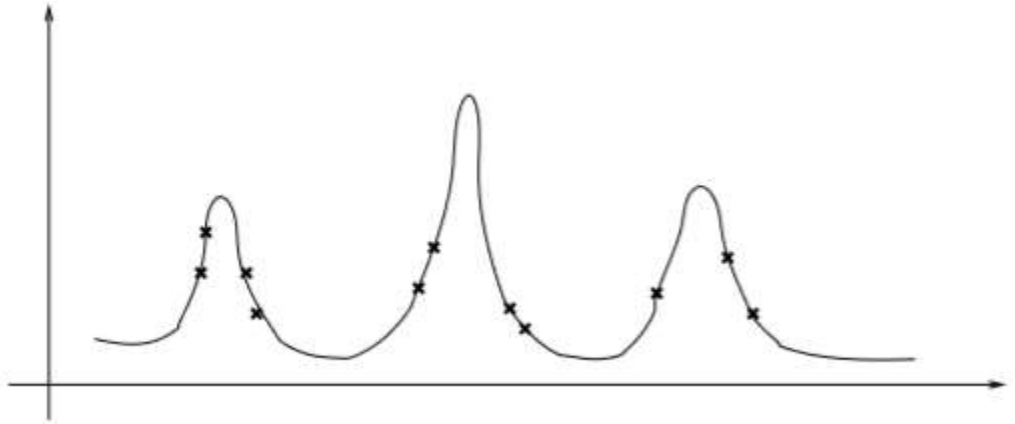


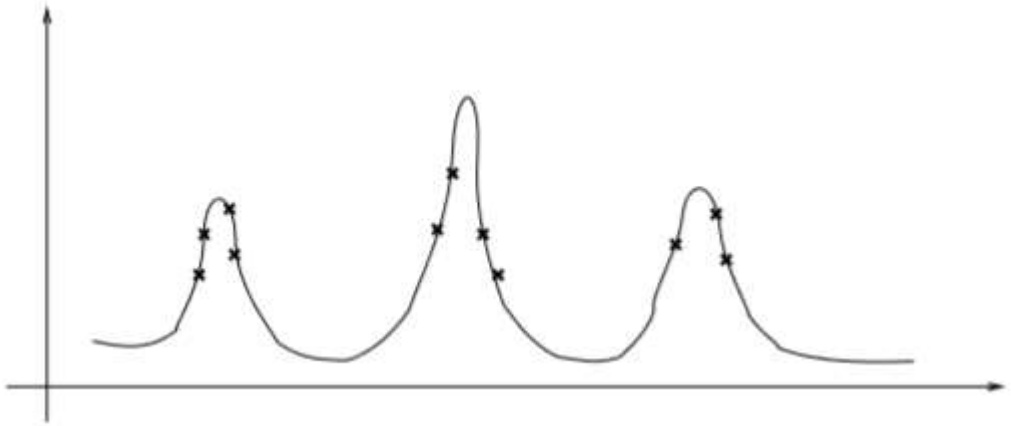
Figure. Principe de base des algorithmes évolutionnistes [S.Doncieux]

PRINCIPE D'UN ALGORITHME ÉVOLUTIONNISTE

[S.DONCIEUX]







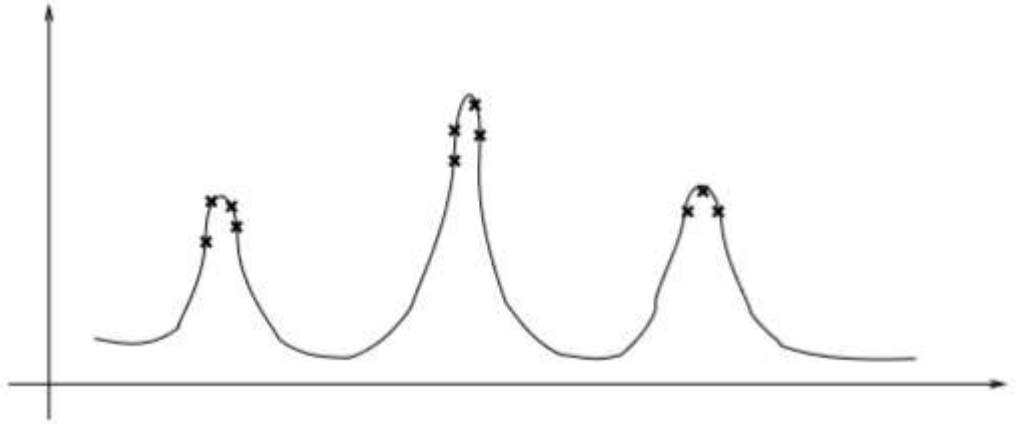


SCHÉMA D'UN ALGORITHME ÉVOLUTIONNISTE

- Un algorithme évolutionnaire engendre une **population initiale** P de μ individus, puis fait évoluer itérativement la population P (**générations**).

- A chaque génération :

1) **Sélection pour la reproduction**: on choisit dans la population courante P les λ individus qui deviendront les parents (mating pool P').

2) **Opérateurs de variation**: on applique le croisement et la mutation aux individus de P' et on obtient une population P" de λ enfants.

3) **Évaluation:** on évalue la performance des éléments de P" (enfants).

4) **Sélection pour la survie** : on choisit parmi éléments de P(population courante) et de P" (enfants) les μ individus qui constitueront la population P de la génération suivante.

PLUSIEURS TYPES D'ALGORITHMES ÉVOLUTIONNISTES

- Le schéma d'algorithme évolutionnistes peut s'«instancier» de différentes manières.
- Des cas particuliers sont :
  - L'algorithme génétique générationnel
  - L'algorithme génétique stationnaire (steady state)
  - L'algorithme (μ, λ)-ES (stratégie d'évolution)
  - L'algorithme (μ+ λ)-ES (stratégie d'évolution)
  - L'algorithme EP(programmation évolutionnaire)

CARACTÉRISTIQUES DES AEs

Caractéristiques communes des AE

- Génération aléatoire d'un ensemble de solutions
- Sélection de certaines solutions
- Génération de nouvelles solutions

CARACTÉRISTIQUES DES AEs

Différences entre AEs

1. Types d'individus
   - Quel espace de recherche ? → **codage**
2. Type d'évolution

   Indiquer comment décider de la survie des individus et comment choisir les nouveaux individus qui vont entrer dans la population

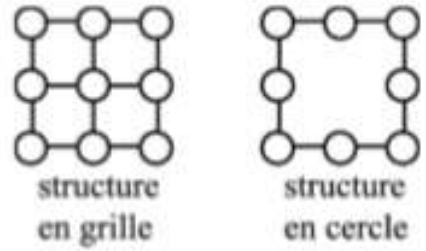
   - Comment l'explorer ?
     - Mutation (lié au codage)
     - Croisement (lié au codage)
     - Population & algorithme de sélection

CARACTÉRISTIQUES DES AEs

Différences entre AEs

3. Structure de voisinage

   À chaque individu on associe un sous-ensemble d'autres individus avec lesquels il peut échanger de l'information (pop struct et pop N struct)



4. Sources d'information

   Le nombre d'individus qui coopèrent pour créer un nouvel individu est souvent égal à 2

5. Irréalisabilité

   Un individu est un objet défini avec des règles bien précises.

6. Intensification

7. Diversification

CARACTÉRISTIQUES DES AEs

Codage : Solutions candidates représentées de différentes façons :

- chaînes binaires (algorithmes génétiques)
- vecteur de flottant (stratégie d'évolution)
- arbre (programmation génétique)
- quelconque (graphe, par exemple : programmation évolutionniste)

→ élément clé : pouvoir définir les opérateurs suivants :

- ✓ génération aléatoire
- ✓ mutation
- ✓ croisement

→ Propriétés à prendre en compte pour le lien génotype-phénotype :

- ✓ Redondance
- ✓ Gradualité
- ✓ Localité

ÉVOLUTION DE LA DIVERSITÉ

- La population tend à devenir de plus en plus homogène au cours de la recherche
  - On parle de **diminution de la diversité** (ou **perte de la diversité**).
- Quand la population est constituée d'individus très semblables, la diversité est épuisée. Il est alors généralement inutile de continuer la recherche.
  - L'épuisement de la diversité peut constituer un critère d'arrêt de l'algorithme.
- Quand la diversité est épuisée et que l'algorithme n'a pas obtenu des solutions jugées suffisamment bonnes, on parle de **convergence prématurée.**

LES ALGORITHMES GÉNÉTIQUES (AGs) [S.AMÉDÉE ET ALL]

HISTORIQUE (1) [S.AMÉDÉE ET ALL]

- **Charles Darwin (1960)** : Publie son livre (L'origine des espèces au moyen de la sélection naturelle ou la lutte pour l'existence dans la nature).

  - Théorie de l'évolution des espèces (pr de reproduction).
    - La loi de croissance et de reproduction.
    - La loi d'hérédité qu'implique quasiment la loi de reproduction
    - La loi de variabilité, résultant des condition d'existence.
    - La loi de multiplication des espèces qui amène la lutte pour l'existence et qui a pour conséquence la sélection naturelle.

- **Mendel (1966)** : publie l'article (**recombinaison des gênes**)

HISTORIQUE (2) [S.AMÉDÉE ET ALL]

- **John Holland (1975)** : - (1960) étudie les systèmes évolutifs
  - (1975) il introduit **le premier modèle formel des algorithmes génétiques** (the canonical genetic algorithm AGC) dans son livre Adaptation in Natural and Artificial Systems.
    - Utilisent **le croisement, la mutation et la reproduction d'individus**
- **Goldberg (1989)** : (vulgarisation des AGs), Ajouta à la théorie des algorithmes génétiques les idées suivantes :
  - Un individu est lié à un **environnement** par son code **d'ADN**.
  - Une solution est liée à un **problème** par son indice de **qualité**.
- **Koza (1992)** : Programmation génétique. Permet de trouver le code informatique optimal pour résoudre un problème.

HISTORIQUE (3) [S.AMÉDÉE ET ALL]

- Il s'agit de simuler l'évolution d'une population d'individus divers (généralement tirée aléatoirement au départ)
- On applique différents opérateurs (recombinaisons, mutations)
- Que l'on soumet à une sélection, à chaque génération.

❖ Si la **sélection** s'opère à partir de la **fonction d'adaptation**, alors la **population tend à <u>s'améliorer</u>** [S.Amédée et all].

❑ Un tel algorithme ne nécessite aucune connaissance du pbm
=> Une boîte noire comportant des **entrées** (les variables) et des **sorties** (les fonctions objectif).

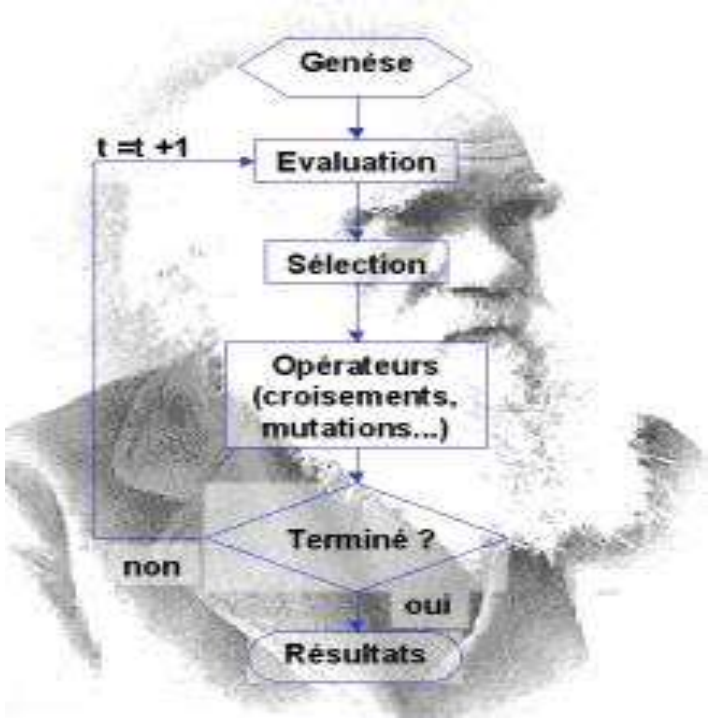


Figure. Organigramme d'un algorithme évolutionnaire

PRINCIPE D'UN AG (1)

AGs Recherche le ou les extrema d'une fonction définie sur un espace de données [S.Amédée et all].

■ Modélisation :

1. **Un principe de codage de l'élément de population:** associe à chacun des points de l'espace d'état une structure de données.

   ❖ (Bon) La qualité du codage des données conditionne le succès des AGs.

2. **Un mécanisme de génération de la population initiale:** Ce mécanisme doit être capable de produire une population d'individus non homogène qui servira de base pour les générations futures.

   ❖ Le choix de la population initiale est important car il peut rendre plus ou moins rapide la convergence vers l'optimum global.

   ❖ Dans le cas où l'on ne connaît rien du problème à résoudre, il est essentiel que la population initiale soit répartie sur tout le domaine de recherche.

PRINCIPE D'UN AG (2)

3. **Une fonction à optimiser:** Retourne une valeur appelée fitness ou fonction d'évaluation de l'individu.

■ Evolution :

4. **Des opérateurs permettant de diversifier la population au cours des générations et d'explorer l'espace d'état:**
   - ❖ **Opérateur de croisement** recompose les gènes (exploitation) d'inds existant dans pop.
   - ❖ **Opérateur de mutation** a pour but de garantir l'exploration de l'espace d'états.
5. **Des paramètres de dimensionnement:**
   - ✓ Taille de la population,
   - ✓ Nombre total de générations ou critère d'arrêt,
   - ✓ Probabilités d'application des opérateurs de croisement et de mutation.

FONCTIONNEMENT DES AGS

- Les AGs fournissent des solutions aux problèmes n'ayant pas de solutions calculables en temps raisonnable de façon analytique ou algorithmique [S.Amédée et all].
- Aucune connaissance de la manière dont résoudre le problème n'est requise,
- Des milliers de solutions (génotypes) plus ou moins bonnes sont crées au hasard puis sont soumises à un procédé d'évaluation de la pertinence de la solution mimant l'évolution des espèces : les plus "adaptés",
  - Les solutions au problème qui sont les plus optimales survivent, la population évolue par générations successives en croisant les meilleures solutions entre elles et en les faisant muter,
  - Puis en relançant ce procédé un certain nombre de fois afin d'essayer de tendre vers la solution optimale.

APPLICATION GÉNÉRALE (MODÈLE CANONIQUE)

- Le mécanisme **d'évolution** et de **sélection** est **indépendant du problème** à résoudre : seules changent trois fonctions [S.Amédée et all] :

1. La fonction qui s'occupe de représenter le problème en codant chaque information caractérisant une solution possible selon un codage bien particulier,
   - Chaque information représente alors un gène et toutes les valeurs que peuvent prendre cette caractéristique représentent les allèles possibles pour ce gène, et en concaténant tous ces gènes pour obtenir un chromosome qui lui représente une solution dans son intégralité.
2. La fonction inverse qui à partir d'un chromosome permet d'obtenir une solution par décodage du génôme.
3. La fonction qui évalue l'adaptation d'une solution à un problème, sa pertinence.

DOMAINES D'UTILISATION

- **Optimisation** : optimisation de fonctions, planifiacation, etc ...
- **Apprentissage** : classification, prédiction, robotique, etc ...
- **Programmation automatique** : programmes LISP, automates cellulaires, etc ...
- **Etude du vivant, du monde réel** : marchés économiques, comportements sociaux, systèmes immunitaires, etc

CARACTÉRISTIQUES DES AGS [J.ROUTIER ET ALL]

- Les **principales différences** des algorithmes génétiques par rapport aux autres paradigmes (énumératives ou basées sur gradient) de recherche sont les suivantes :

1. On utilise un codage des paramètres du problème : on représente toutes les caractéristiques d'une solution par un ensemble de gènes,
2. On traite une population de solutions : pas sur une unique situation, cela introduit donc du parallèlisme.

   ↳ éviter le piège d'un minimum local

3. L'évaluation de l'optimalité du système n'est pas dépendante du domaine.
4. On utilise des règles de transition probabilistes : il n'y a pas d'énumération de l'espace de recherche.

   ↳ pas déterministes

ALGORITHMES GÉNÉTIQUES

- Un AG générique à la forme suivante [J.C.Routier et all] :

initialiser la population (générer aléatoirement une population de *N* chromosomes *x*)
calculer le degré d'adaptation *f(x)* de chaque individu
Tant que non fini ou non convergence
    reproduction des parents
        sélectionner 2 individus à la fois
        appliquer les opérateurs génétiques
    calculer le degré d'adaptation *f(x)* de chaque enfant
    sélectionner les survivants parmi les parents et les enfants
fin Tant que

ALGORITHMES GÉNÉTIQUES

- **Algorithme (cas de la minimisation de la fonction f)** [S.Doncieux]

```
ALGORITHMEGENETIQUE()
1     t ←0
2     INITIALISER(Pt)
3     for i ←0 to N
4     do
5         EVALUER(Pt(i))
6     while critere d'arret non remplis
7     do
8         Q ←∅
9     for i ←0 to N
10    do
11        x ← RECOMBINER(SELECT_REPRO(Pt))
12        x ← MUTER(x)
13        EVALUER(x)
14        Q ←x ∪Q
15        Pt+1 ← SELECT_REMPLACE(Pt,Q)
16        t ←t + 1
```

CRITÈRE DE CONVERGENCE

- Le critère de convergence peut être de nature diverse:
  - Un taux minimum qu'on désire atteindre d'adaptation de la population au problème,
  - Un certain temps de calcul à ne pas dépasser,
  - Une combinaison de ces deux points.

EVALUATION DE L'ADAPTATION (1)

[J.C.Routier et all]

- Dépend du problème.
- Exemples possibles :

➢ comparaison d 'images	nombre de pixels semblables
➢ contrôle d 'un robot	nombre de chocs contre les murs
➢ classification	nombre d 'exemples bien classés
➢ vie artificielle	quantité moyenne de nourriture ingérée dans une simulation
➢ régression de fonction	somme ou variance des erreurs sur un jeu d 'exemples

EVALUATION DE L'ADAPTATION (2)

- **Fonctions d'adaptation usuelles** [J.C.Routier et all] **:**

❖ Somme des erreurs absolues entre les valeurs calculées et valeur attendues en sortie, pour chacun des *fitness cases* :

$$\varphi = \sum_{i=1}^{n} |P(e_i) - s_i|$$

❖ Somme des carrés des écarts entre valeur calculée et valeur attendue (« squared error ») :

$$\varphi = \sum_{i=1}^{n} (P(e_i) - s_i)^2$$

❖ Souvent on normalise la fonction d'adaptation :

$$f_a = \frac{1}{1 + f_s}$$

f_s est nul dans le cas idéal, >0 autrement

LE CODAGE (1) [S.AMÉDÉE ET ALL]

- Chaque paramètre d'une solution est assimilé à un gène, toutes les valeurs qu'il peut prendre sont les allèles de ce gène, on doit trouver une manière de coder chaque allèle différent de façon unique.
- Un chromosome est une suite de gène, on peut par exemple choisir de regrouper les paramètres similaires dans un même chromosome et chaque gène sera repérable par sa position.

LE CODAGE (2)

- Chaque individu est représenté par un ensemble de chromosomes, et une population est un ensemble d'individus

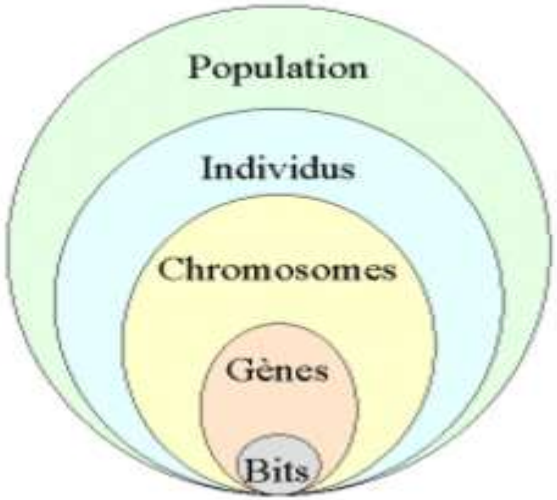


Figure. les cinq niveaux d'organisation d'un algorithme génétique

TYPE DE CODAGE : CODAGE BINAIRE

- Trois principaux types de codage utilisables [S.Amédée et all] :

1. **Le codage binaire** : (le plus utilisé)
   - Chaque gène dispose du même alphabet binaire {0, 1}
   - Les chromosomes qui sont des suites de gènes sont représentés par des tableaux de gènes et les individus de notre espace de recherche sont représentés par des tableaux de chromosomes.
   - Ce cas peut être généralisé à tout alphabet allélique n-aire permettant un codage plus intuitif,

   exemple. pour le problème du voyageur de commerce on peut préférer utiliser l'alphabet allélique {c1, c2, c3, ..., cn} où ci représente la ville de numéro i.

TYPE DE CODAGE : CODAGE RÉEL

2. **Le codage réel** : cela peut-être utile notamment dans le cas où l'on recherche le maximum d'une fonction réelle.

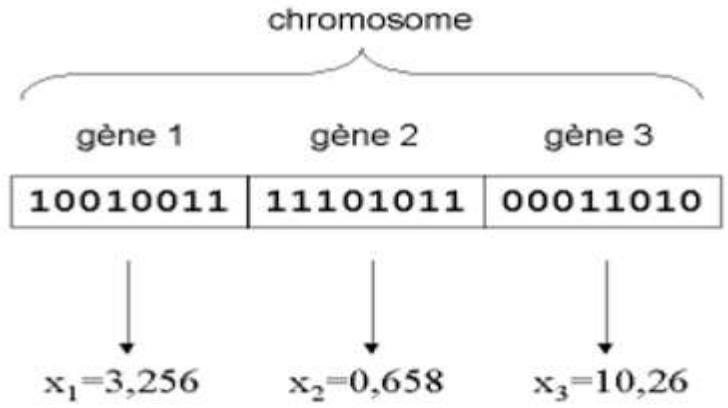


Figure. Codage des variables réelles

- Dans le cas d'un codage binaire on utilise souvent la "distance de Hamming" comme mesure de la dissimilarité entre deux éléments de population, cette mesure compte les différences de bits de même rang de ces deux sequences.
- Le codage binaire commence à montrer ses limites (deux éléments voisins en terme de distance de Hamming ne codent pas nécessairement deux éléments proches dans l'espace de recherche).

TYPE DE CODAGE : CODAGE DE GRAY

3. Le codage de Gray

Le codage de Gray est un codage qui a comme propriété que entre un élément n et un élément **(n+1)**, voisin dans l'espace de recherche, un seul bit diffère.

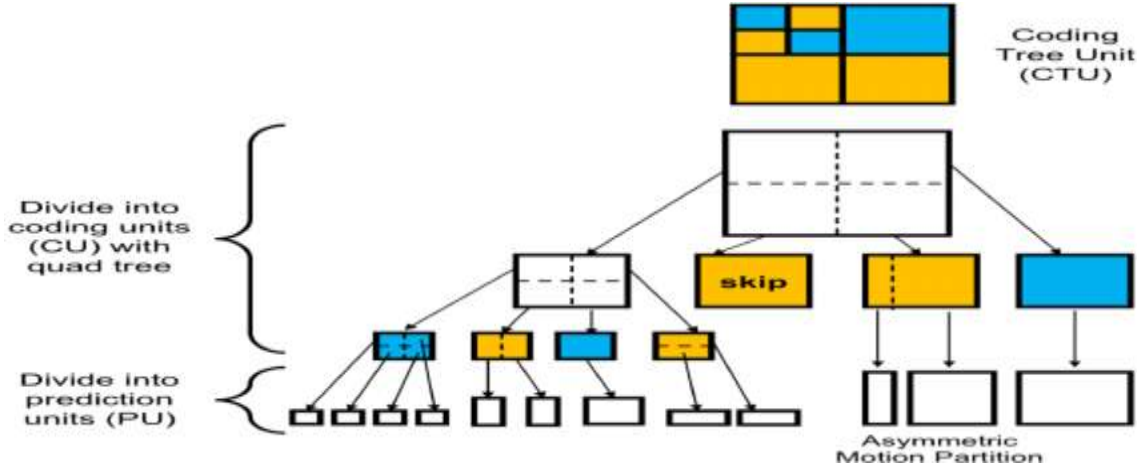
Decimal Values	Natural Binary Code	Gray Code	Gray Code values
0	000	000	0
1	001	001	1
2	010	011	3
3	011	010	2
4	100	110	6
5	101	111	7
6	110	101	5
7	111	100	4

TYPE DE CODAGE : CODAGE SOUS FORME D'ARBRE

4. Le codage sous forme d'arbre

Ce codage utilise une structure arborescente avec une racine de laquelle peuvent être issus un ou plusieurs fils. Un de leurs avantages est qu'ils peuvent être utilisés dans le cas de problèmes où les solutions n'ont pas une taille finie.

- Les arbres résultants sont souvent difficiles à analyser et que l'on peut se retrouver avec des arbres (solutions) dont la taille sera importante (taille quelconque peuvent être formés).



LA SÉLECTION

- Quels seront les individus de P qui vont être survivre, de se reproduire ou de mourir dans la nouvelle population P‘ [S.Amédée et all] .

  ↳ Reliée à son efficacité relative au sein de la population.

- Cinq types de méthodes de sélection différentes :

  1. **La méthode de la "loterie biaisée" (roulette wheel) de GoldBerg,**
  2. **La méthode du rang**
  3. **La méthode "élitiste"**
  4. **La sélection par tournois,**
  5. **La sélection universelle stochastique.**

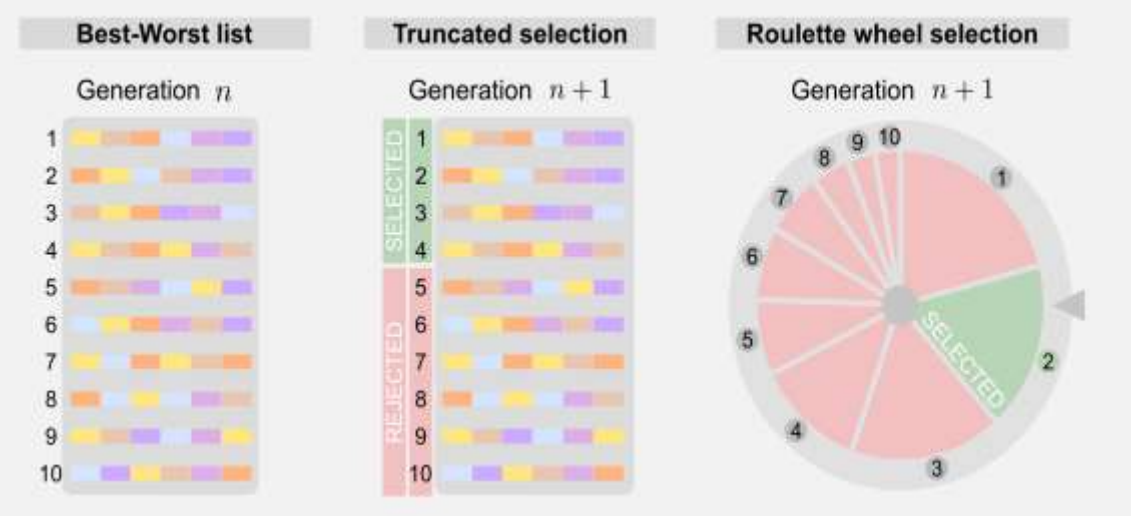
MÉTHODE DE LA ROULETTE (1)

- Pour chaque chromosome *i* on calcule son **degré d'adaptation** f_i et on pose :

$$p_i = \frac{f_i}{\sum_i f_i} \times 100$$

- On crée une roulette biaisée où chaque *i* occupe une portion p_i (secteur d'une roue).
- L'angle du secteur étant proportionnel à la **qualité** de l'individu qu'il représente.
- Pour déterminer **les descendants d'une génération de taille *n*, il faut *n* tirages**

  ↳ **Vous tournez la roue et vous obtenez un individu.**



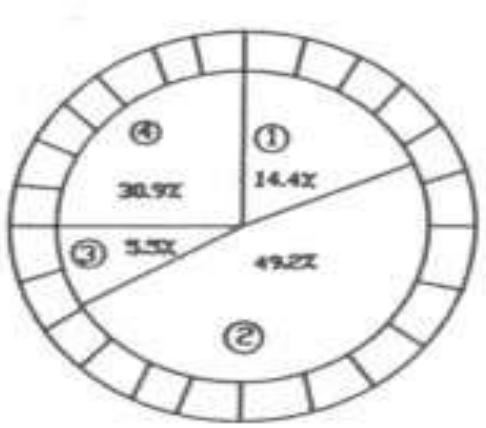
MÉTHODE DE LA ROULETTE (2)

- **Les tirages des individus sont ainsi pondérés par leur qualité.**

  ↳ les **meilleurs** individus ont plus de chance d'être croisés et de **participer** à l'amélioration de notre population.

exemple : maximiser $f(x) = x^2$ avec n= 4 pris dans [0, 31]

i	chaîne	f_i	% total
1	**01101**	169	14.4
2	**11000**	576	49.2
3	**01000**	64	5.5
4	**10011**	361	30.9
Total		1170	100

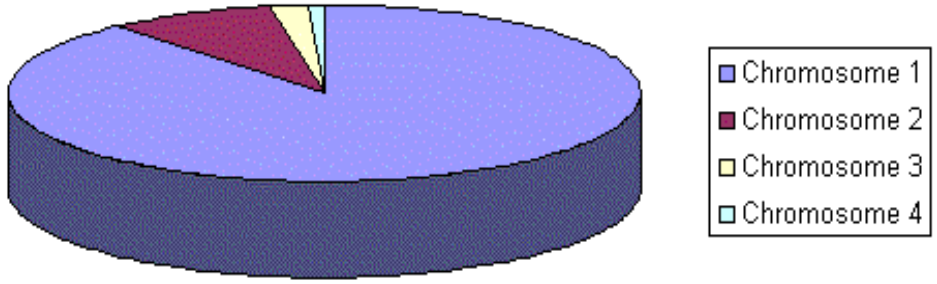


- Lors d'un tirage la chaîne 1 occupe 14.4% de la roue de loterie, il y a une probabilité de 0.144 que l'on obtienne une copie de cet individu [J.C.Routier et all] .

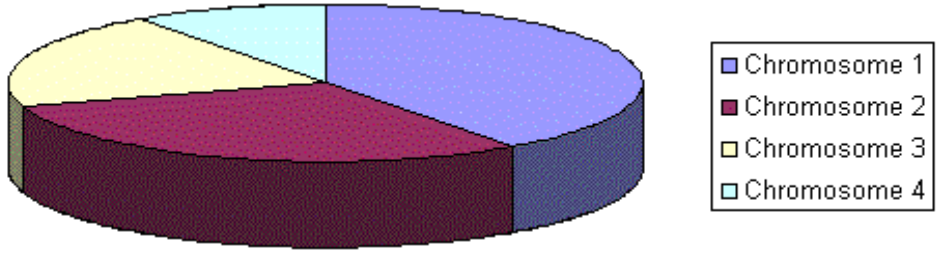
MÉTHODE DU RANG (1)

- Les parents sont d`abord triés par ordre d'adaptation (valeur de *f*), en commençant par les moins performants :
  - Le plus faible a un score de 1,
  - Le deuxième plus faible a un score de 2
  - Le troisième plus faible a un score de 3

  Etc.



- On divise chaque score par la somme de tous les scores:



- On procède ensuite comme pour la roulette ; la méthode donne plus de chance aux chromosomes les plus faibles d'être repêchés [J.C.Routier et all] .

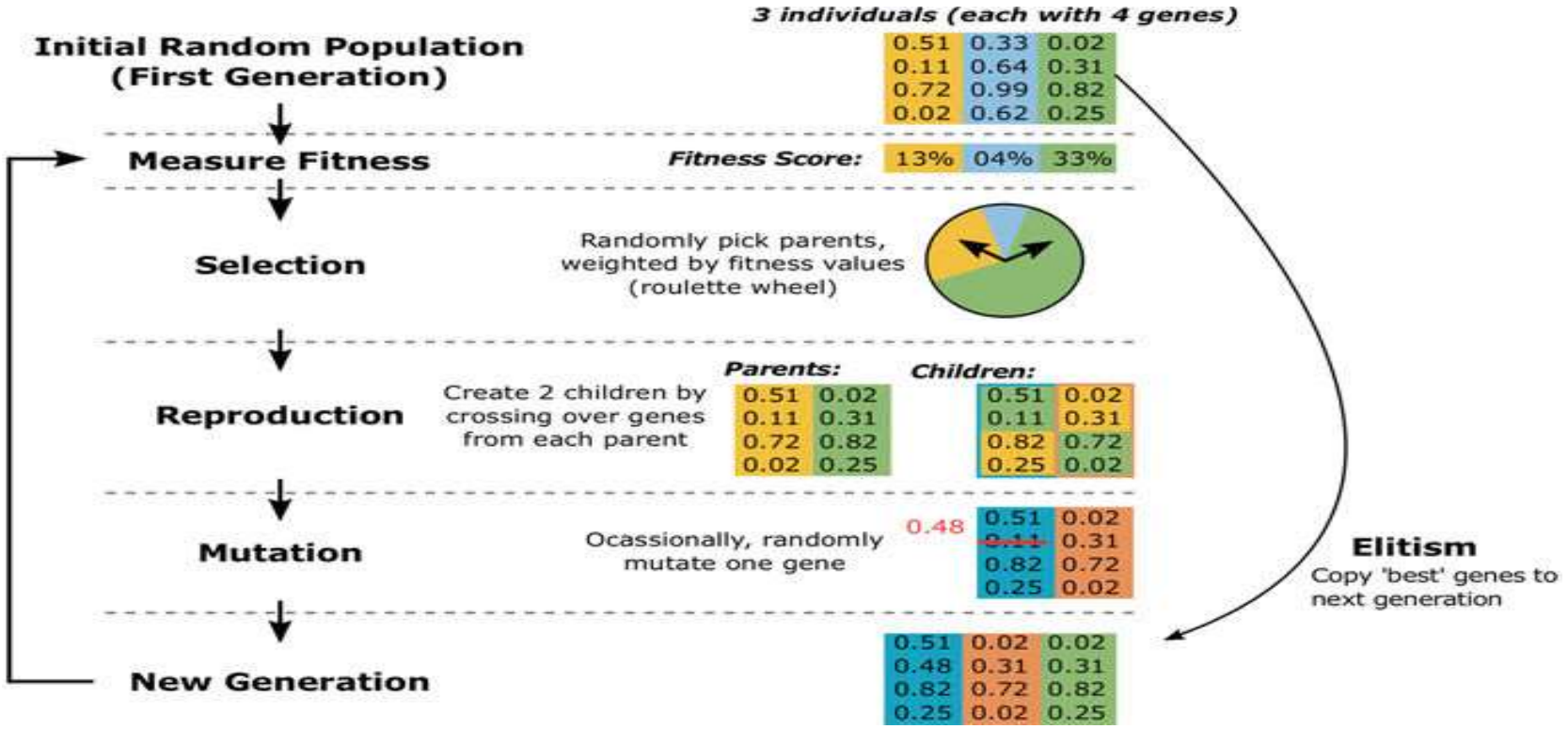
MÉTHODE DU RANG (2)

- **Tableau.** Exemples de sélection par rang pour 6 chromosomes. Avec Cette sélection, tous les individus ont une chance pour se reproduire.

Chromosomes	1	2	3	4	5	6	total
Probabilités initiales	89%	5%	1%	4%	3%	2%	100%
Rang	6	5	1	4	3	2	21
Probabilités finales	29%	24%	5.%	19%	14%	9%	

MÉTHODE ÉLITISTE (1)

- Cette méthode consiste à sélectionner les n individus dont on a besoin pour la nouvelle génération **P'** en prenant les **n meilleurs** individus de la population **P** après l'avoir **triée** de manière **décroissante** selon la fitness de ses individus [S.Amédée et all].



MÉTHODE ÉLITISTE (2)

- Permettre une convergence (plus) rapide des solutions

  ↳ Sensible à la présence **d'optimas locaux**

- Il est inutile de préciser que cette méthode est encore pire que celle de la loterie biaisée dans le sens où elle aménera à une **convergence prématurée** encore **plus rapidement** et surtout de manière encore plus sûre que la méthode de sélection de la loterie biaisée;

- La **pression** de la sélection est **trop forte**, **la variance nulle** et **la diversité inexistante**, du moins le peu de **diversité** qu'il pourrait y avoir ne résultera pas de la sélection mais plutôt du **croisement** et des **mutations**.

  ↳ **Il faut opter pour une autre méthode de sélection.**

MÉTHODE DU TOURNOI (1)

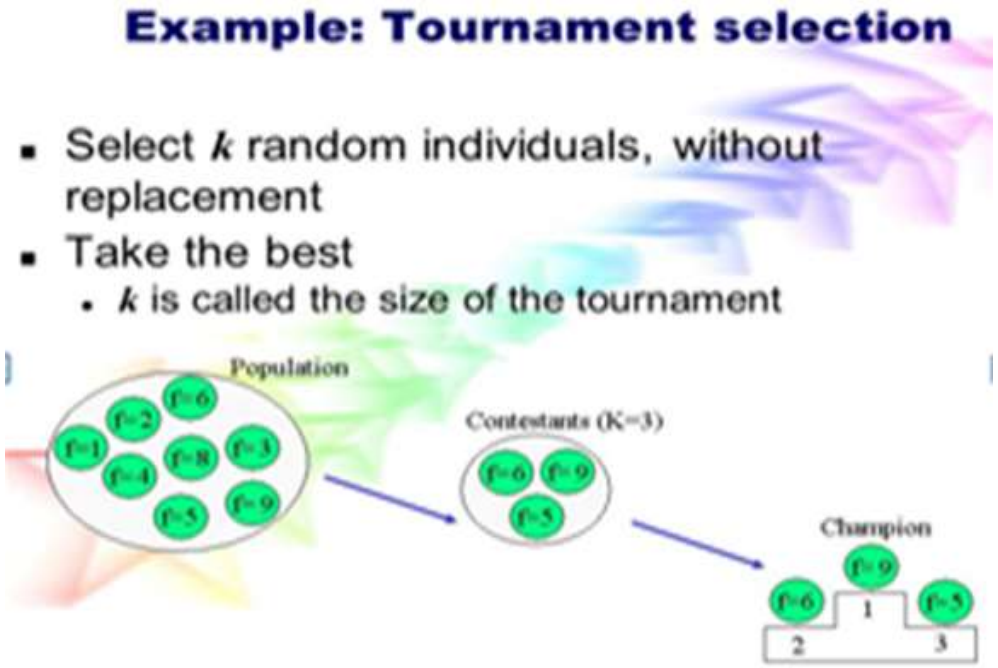
- Partant de la population de ***n*** chromosomes, on **forme** **n paires** au hasard et on détermine le vainqueur dans chacune par sa valeur de ***f*** [J.C.Routier et all] .

- Dans les paramètres de l'**AG**, on détermine une probabilité de victoire du plus chromosome le plus fort, représentant sa chance d'être sélectionné par la suite. Cette probabilité doit être grande (entre 70% et 100%). A partir des n paires, on sélectionne ainsi n individus pour la reproduction.

- **NB.** Autre méthode, la sélection universelle stochastique

MÉTHODE DU TOURNOI (2)

- Le principe de la sélection par tournoi augmente les chances pour les individus de piètre qualité de participer à l'amélioration de la population.
- Un tournoi consiste en une rencontre entre plusieurs individus pris au hasard dans la population.
  - ↳ Le vainqueur du tournoi est l'individu de meilleure qualité.
- Nous pouvons choisir de ne conserver que le vainqueur comme nous pouvons choisir de conserver les 2 meilleurs individus ou les 3 meilleurs
- Selon que nous souhaitons créer beaucoup de tournois,
  - ❖ Créer des tournois avec beaucoup de participants
  - ❖ Mettre en avant ceux qui gagnent les tournois haut la main.
- Un même individu peut être participé à plusieurs tournois.



LE CROISEMENT : CROISEMENT À 1 POINT

Espérance d'amélioration de nouvelles générations [J.C.Routier et all].

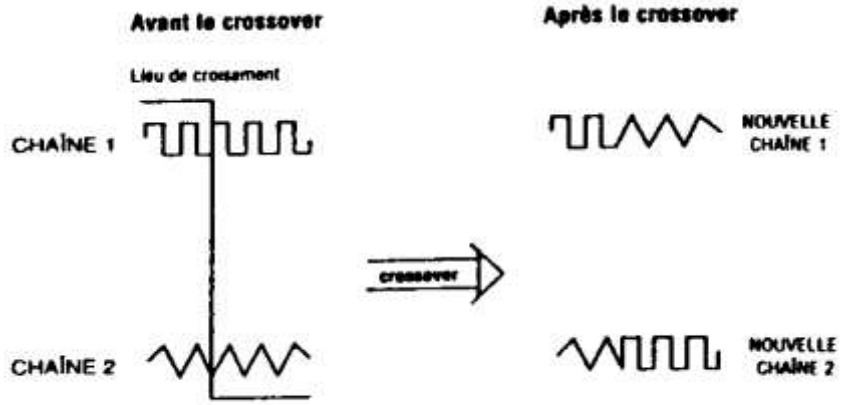
- **Croisement à 1 point**

  2 chromosomes de taille l .

  On choisit aléatoirement un entier k entre 1 et l-1.

  k représente le point de croisement des deux chaînes

On crée 2 nouveaux individus en échangeant les caractères des chaînes initiales compris entre k+1 et l .



exemple : l=5 et k= 3

C_1 = **0 1 1 | 0 1**
C_2 = **1 1 0 | 0 0**

$C_{1,2}$ = **0 1 1 | 0 0**
$C_{2,1}$ = **1 1 0 | 0 1**

LE CROISEMENT : CROISEMENT À 2 POINTS

❖Croisement à 2 points

2 chromosomes de taille l.

On considère qu'ils forment chacun un anneau fermé et on choisit aléatoirement 2 entiers k_1 *et* k_2 compris entre 1 et l-1.

k_1 *et* k_2 représentent les points de croisement des deux chaînes

On crée 2 nouveaux individus en **échangeant les caractères des chaînes initiales compris entre k_1+1 et k_2**

exemple : l=10, k_1= 1 *et* k_2=8

C_1 = **0 | 1 1 1 0 0 1 1 | 1 1**
C_2 = **1 | 1 1 0 0 0 0 0 | 0 0**

$C_{2,1}$ = **0 | 1 1 0 0 0 0 0 | 1 1**
$C_{1,2}$ = **1 | 1 1 1 0 0 1 1 | 0 0**

LE CROISEMENT : CROISEMENT UNIFORME

❖ Croisement uniforme

Définir de manière aléatoire un "masque" (une chaîne de bits de même longueur que les chromosomes des parents) sur lesquels il sera appliqué.

```
Mask:        0110011000      (Randomly generated)
Parents:     1010001110      0011010010

Offspring:   0011001010      1010010110
```

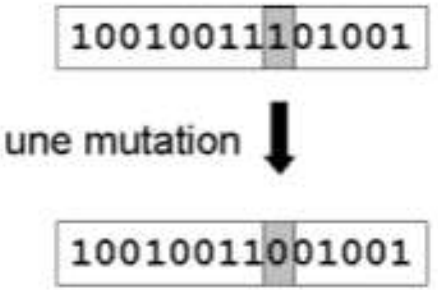
LA MUTATION

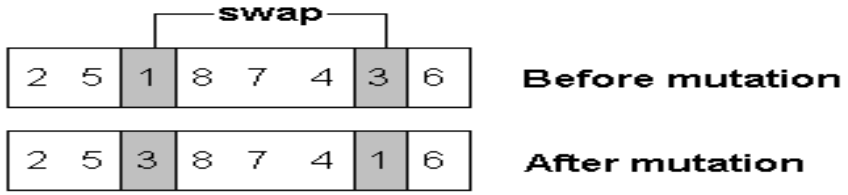
Permet de sortir de minima locaux [J.C.Routier et all] .

- Il s'agit de la modification aléatoire de la valeur d'un caractère de la chaîne.

- Le taux de mutation (probabilité **Pm**) est généralement choisi très faible (≈ **0.001**), comprise **[0.01, 0.001]**
  - pour chaque caractère des descendants, probabilité de **1/1000** qu'il mute
  - On peut aussi prendre **Pm = 1 / lg** où **lg** est la longueur de la chaîne de bits codant notre chromosome [S.Amédée et all].
- Pour un codage binaire, elle consiste simplement à changer un 0 en un 1.

❖ Si la mutation joue un rôle secondaire (dû au taux faible), elle permet **l'exploration de dimensions** (éventuellement utiles).





GRANDS AVANTAGES [S.AMÉDÉE ET ALL]

- Il **garantit** la **diversité** de la population,
- La mutation permet **d'atteindre** la **propriété d'ergodicité**.
  - **L'ergodicité** est une propriété garantissant que chaque point de l'espace de recherche puisse être atteint.
- Il permet **d'éviter** un phénomène connu sous le nom de **dérive génétique** (certains gènes favorisés par le hasard se répandent au perte des autres et sont ainsi présents au même endroit sur tout les chromosomes.)
- Il permet de **limiter les risques** d'une **convergence prématurée** causée par exemple par une méthode de sélection élitiste imposant à la population une pression sélective trop forte.

LE REMPLACEMENT

- Réintroduire les descendants obtenus par application successive des opérateurs de sélection, de croisement et de mutation (la population P') dans la population de leurs parents (la population P).

❖ **Le remplacement stationnaire** : les enfants remplacent automatiquement les parents sans tenir compte de leurs performances respectives,

↳ Le nombre d'individus de la population constant (initialiser la population initiale avec un nombre suffisant d'individus).

❖ **Le remplacement élitiste** : on garde au moins l'individu possédant les meilleures performances d'une génération à la suivante

↳ Un nouvel individu (enfant) prend place au sein de la population que s'il remplit le critère d'être plus performant que le moins performant des individus de la population précédente.

↳ Les enfants d'une génération ne remplaceront pas nécessairement leurs parents comme dans le remplacement stationnaire et par la même la taille de la population n'est pas figée au cours du temps.

↳ Ce type de stratégie améliore les performances des algorithmes évolutionnaire dans certains cas.

↳ présente aussi un désavantage en augmentant le taux de convergence prématuré.

INITIALISATION DES PARAMÈTRES [J.C.ROUTIER ET ALL]

- **Croisement**
  - La probabilité varie de 0 à 100% :
    - 0 % (pas de croisement) => clones parfaits
    - 100% => pas de clones
- **Mutation**
  - Probabilité variant de 0% à 100%
  - Probabilité normalement faible (~1/1000)
- **Taille de la population**
  - **Une petite** taille **limite l'exploration de l'espace**
  - **Une grande** taille **réduit la vitesse de convergence**

optimiser $f(x)=x^2$ pour x entre 0 et 31 population de n= 4 individus

Choix du codage du paramètre : x codé en binaire sur l=5 caractères.

un individu $\in \{0,1\}^5$

degré d'adaptation : on peut utiliser f directement dans ce cas.

génération initiale et pré-calculs

n°	chaîne	x	$f(x)$	p_i	attendus $n \times p_i$
1	**0 1 1 0 1**	13	169	0,14	0,58
2	**1 1 0 0 0**	24	576	0,49	1,97
3	**0 1 0 0 0**	8	64	0,06	0,22
4	**1 0 0 1 1**	19	361	0,31	1,23
Total			1170	1	4
moyenne			293		
max			576		

Croisement

individu	position croisement	obtenu	x	$f(x)$
0 1 1 0 | 1	4	**0 1 1 0 | 0**	12	144
1 1 0 0 | 0	4	**1 1 0 0 | 1**	25	625
1 1 | 0 0 0	2	**1 1 | 0 1 1**	27	729
1 0 | 0 1 1	2	**1 0 | 0 0 0**	16	256
Total				1754
Moyenne				439
Max				729

Mutation

On essaie une mutation de 0,001 et rien n'est modifié.

Il faut maintenant sélectionner les survivants et recommencer

ÉTAPES DE DÉVELOPPEMENT D'UN ALGORITHME GÉNÉTIQUE

1. Spécification du problème, définition des contraintes et des critères d'optimalité
2. Encodage du domaine du problème sous forme de chromosome
3. Définition de la fonction d'adaptabilité pour évaluer la performance du chromosome
4. Définition des opérateurs génétiques
5. Application de l'algorithme et réglage fin subséquents des paramètres [J.C.Routier et all].

CARACTÉRISTIQUES DE LA DIFFÉRENCE

- **Types d'Individus** : en général il s'agit de solutions
- **Type d'évolution** : remplacement générationnel avec une population de taille constante
- **Structure de voisinage** : population possiblement structurée
- **Sources d'information** : deux parents
- **Irréalisabilité** : l'opérateur de croisement évite la génération de solutions non admissibles
- **Intensification** : aucune
- **Diversification** : mutation qui est une procédure de bruitage

AVANTAGES ET INCONVÉNIENTS

- Les algorithmes évolutionnaires constituent une approche originale : il ne s'agit pas de trouver une solution analytique exacte, ou une bonne approximation numérique,
- Trouver des solutions satisfaisant au mieux à différents critères, souvent contradictoires.
- S'ils ne permettent pas de trouver à coup sûr la solution optimale de l'espace de recherche, du moins peut-on constater que les solutions fournies sont généralement meilleures que celles obtenues par des méthodes plus classiques, pour un même temps de calcul.

THÉORÈME DES SCHÉMAS

THÉORÈME DES SCHÉMAS [J.C.ROUTIER ET ALL]

❖ On appelle schéma un motif de chaîne dans lequel le joker * (‘ # ’) remplace indifféremment un **1** ou un **0**.

Les schémas	**1**00*** et **1*******	sont présents dans les individus	**1**00*** et **1******* et **101000**

➢ L'algorithme a pour conséquence la **conservation** au cours des générations des **schémas** les **mieux adaptés**.

On comprend bien que :

Le schéma **1**1*** a **moins de chance** d'être transmis que le schéma ****11***, même si ils apparaissent dans le même individu **(10110).**

Donc :

Les informations représentés par des portions de chaînes (gènes) seront plus facilement **transmises**
et les "**meilleures**" portions seront peu à peu sélectionnées et assemblées au sein d'un même (ou de quelques) individu(s).

→ Les AG exploirent en parallèle **$O(n^3)$** schémas (parallélisme implicite).

Le **théorème des schémas** [J.C.Routier et all] indique que :

Les schémas les mieux adaptés apparaissent avec une occurrence exponentiellement croissante au cours des générations.

si $m(S,t)$ est le nombre d'occurrences du schéma S à la génération t.

$$m(S,t+1) \geq m(S,t) \times \frac{f(S,t)}{\overline{f}} (1 - p_{opérateur} \times (\text{effet négatif}))$$

f=fitness

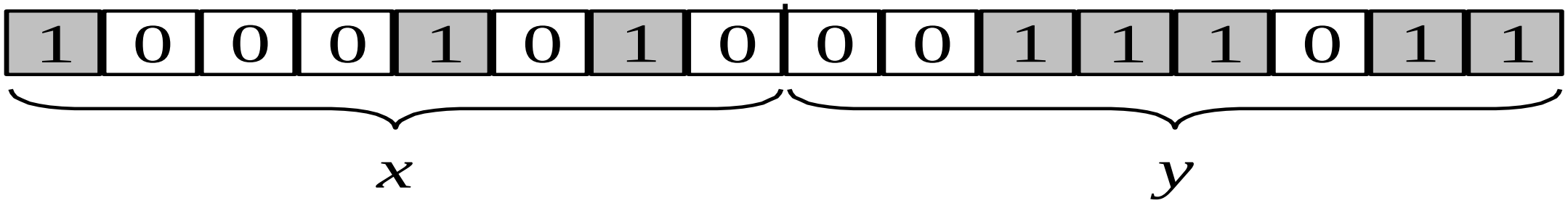
probabilité de survie due aux opérateurs

- On veut trouver le maximum de la fonction [J.C.Routier et all] :

$$f(x,y) = (1-x)^2 e^{-x^2-(y+1)^2} - (x - x^3 - y^3) e^{-x^2-y^2}$$

où *x* et *y* varient entre –3 et 3.

- La première étape consiste à représenter les variables du problème sous forme de chromosomes : *x* et *y* sont écrit sous forme de deux chaînes binaire concaténées de m bits chacune (m=8 dans l'exemple) :



- Ensuite, on fixe la taille de la population de chromosomes (N=6) et on génère une population initiale.
- L'adaptation de chaque chromosome est alors calculée en plusieurs étapes :

1. La chaîne de 16 bits est séparée en deux mots de 8 bits :

1	0	0	0	1	0	1	0	et	0	0	1	1	1	0	1	1

2. Les deux mots sont convertis en décimal :

$$(10001010)_2 = 1\times 2^7 + 0\times 2^6 + 0\times 2^5 + 0\times 2^4 + 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 0\times 2^0 = (138)_{10}$$

and

$$(00111011)_2 = 0\times 2^7 + 0\times 2^6 + 1\times 2^5 + 1\times 2^4 + 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 1\times 2^0 = (59)_{10}$$

3. Les résultats sont convertis de l'intervalle [0,255] à [-3,3]

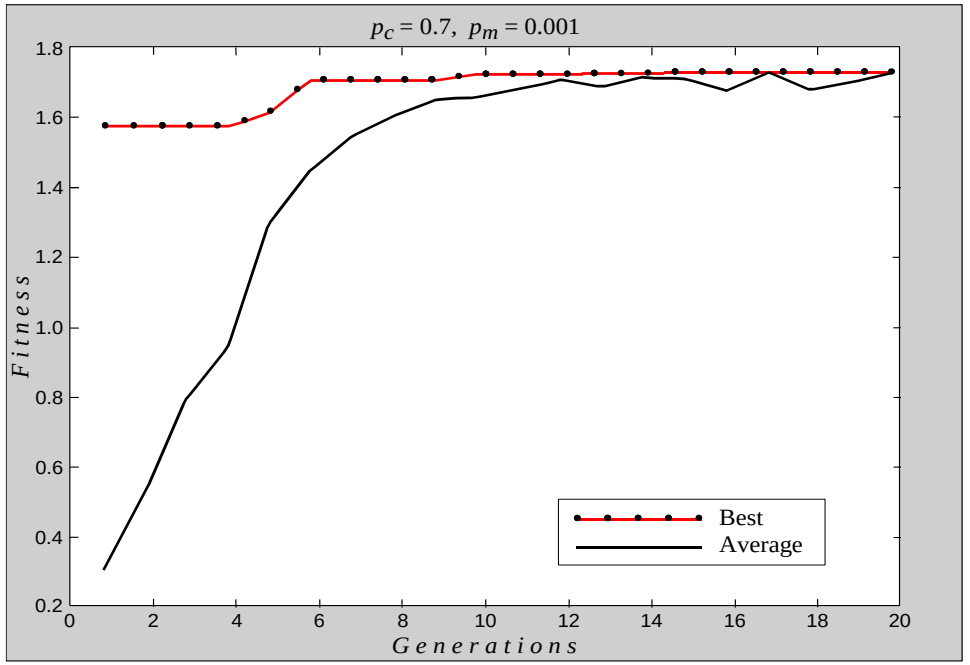
$$138 * \frac{6}{255} - 3 = 0.247 \qquad 59 * \frac{6}{255} - 3 = -1.611$$

4. L'adaptation du chromosome est alors donnée par

f (0.247,-1.611)

- On répète l'étape précédente pour chaque chromosome avant de passer aux transformations génétiques.
- Pour trouver le maximum de la fonction, on utilisera une probabilité de croisement de 0.7 et une probabilité de mutation de 0.001. Le nombre de générations est fixé à 100 (l'algorithme génétique créera au plus, 100 générations de chromosomes avant de s'arrêter).

❖ Courbe de performance pour 20 générations de 60 chromosomes



PROGRAMMATION GÉNÉTIQUE

PROGRAMMATION GÉNÉTIQUE [J.C.Routier et all]

- Algorithmes évolutionnaires plus récents (travaux de **John Koza** dans les années 90).
- PG parcourt l'espace des programmes à la recherche de celui hautement adapté à la solution d'un problème donné.
- Tout programme d'ordinateur est une séquence d'opérations (fonctions) appliquées à des valeurs (arguments); les différents langages de programmation diffèrent par les type d'instructions et d'opérations, en plus de constructions syntaxiques différentes.
- PG manipule les programmes à l'aide d'opérateurs génétiques, ils sont traités comme des données à transformer qui, une fois modifiées, deviennent de nouveaux programmes. Un langage bien adapté à ce genre de manipulations est **LISP**.

LISP 101 [J.C.ROUTIER ET ALL]

- Langage à structure de données de type orienté-symbole. Les structures de bases sont les **atomes** et les **listes**.
- Un atome est le plus petit élément indivisible dans la syntaxe de LISP (e.g. le nombre 21, le symbole X ou la chaîne de caractères "Ceci est une chaîne").
- Une liste est un objet composé d'atomes et/ou d'autres listes.
- Les listes de LISP sont écrites comme une collection ordonnée d'items entre une paire de parenthèses.
  Ainsi, la liste

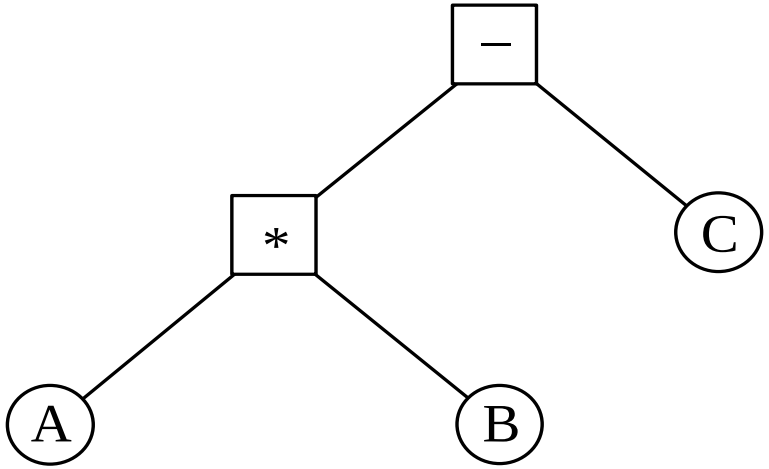
  (– (* A B) C)

  demande de soustraire deux arguments, la liste (*A B) et l'atome C. Mais d'abord, il faut multiplier les atomes A et B.

- Les atomes et les listes sont appelés des expressions symboliques ou « **S-expression** ».
- Toutes les données et tous les programmes sont des S-expressions, ce qui permet au langage de traiter les programmes comme des données.
- En particulier, les programmes en LISP peut s'auto-modifier ou générer de nouveaux programmes, ce qui rend le langage attirant pour la programmation génétique.

- Toute S-expression peut être représentée par un arbre.

  S-expression (– (*A B) C)



COMMENT APPLIQUER LA PROGRAMMATION GÉNÉTIQUE

- Avant d'appliquer la programmation génétique à un problème, il faut accomplir *cinq étapes préliminaires* :

1. Définir les terminaux
   Entrées du programme
2. Choisir les fonctions à utiliser
   Opérations arithmétiques, expressions de programme, sous programmes, fonctions mathématiques, etc.
3. Définir la fonction d'adaptation
   Souvent une fonction d'erreur
4. Choisir les paramètres d'exécution
   Même que pour GA
5. Choisir la méthode de sélection du meilleur résultat
   Généralement le meilleur programme à un instant donné

POPULATION INITIALE [J.C.ROUTIER ET ALL]

- On **fixe une profondeur maximale** pour les arbres.
- Création d'arbres aléatoires par 2 méthodes principales :
  - **« grow »** : chaque nœud est tiré dans l 'ensemble **{terminaux} + {fonctions}**

    ⇨ **les arbres sont de forme irrégulière**
  - **« full »** : on ne peut tirer un terminal que lorsque l'on est à la profondeur maximum

    ⇨ **les arbres équilibrés et « pleins »**
- Une synthèse, la méthode **« ramped half & half »** :
  - on va générer équitablement des arbres de profondeurs régulièrement :

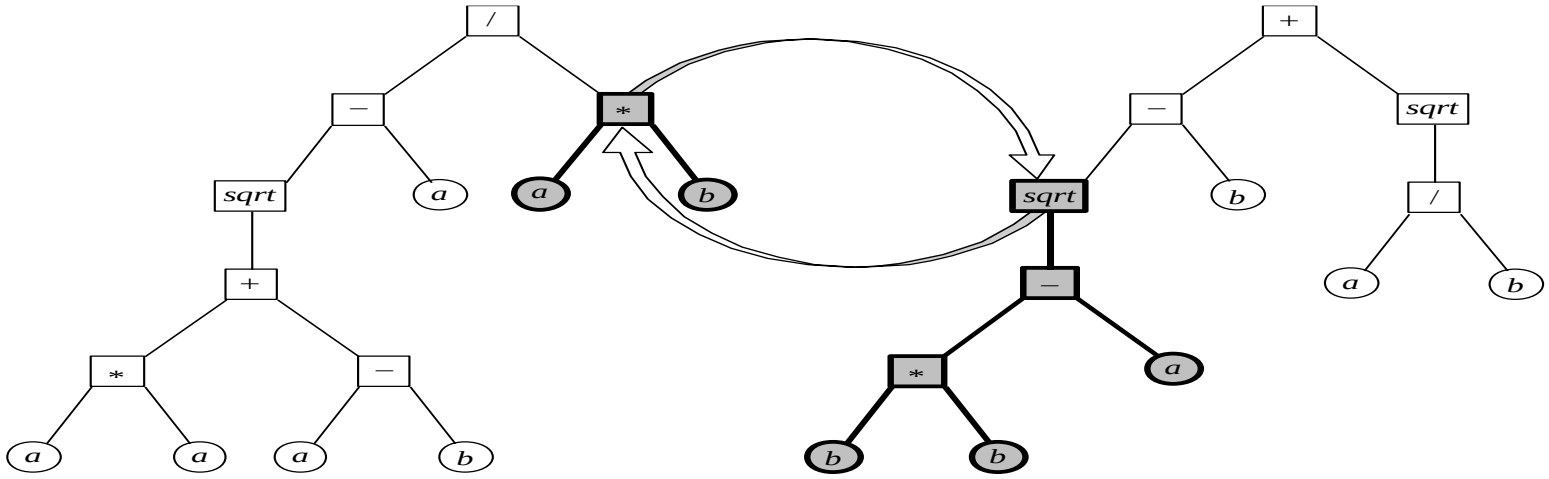
    2, 3, 4, …, maximum
  - à chaque profondeur, une moitié est générée par la méthode « full », l 'autre par la méthode « grow »

  ⇨ L 'objectif est d 'obtenir plus de **variabilité** dans la population (méthode préférentielle).

L'ADAPTATION ET LA SÉLECTION [J.C.ROUTIER ET ALL]

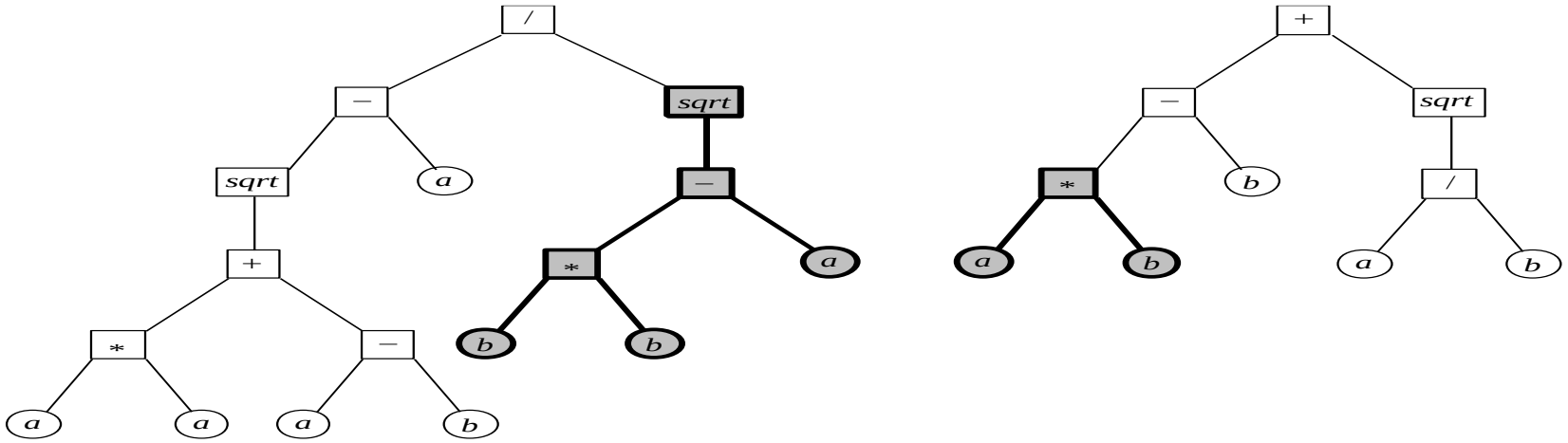
- Le choix de la fonction d'adaptation se fait de manière similaire aux AGs
- On retrouve aussi les méthodes de sélection utilisées dans les AGs :
  - sélection proportionnelle au degré d'adaptation, avec normalisation éventuelle (« scaling »)
  - sélection basé sur le rang de l 'individu dans la population (« ranking »)
  - sélection par tournoi : la plus courante, car rapide et facilement parallélisable

❖ OPÉRATEURS GÉNÉTIQUES : CROISEMENT [J.C.Routier et all]



(/ (− (sqrt (+ (∗ a a) (− a b))) a) (∗ a b))

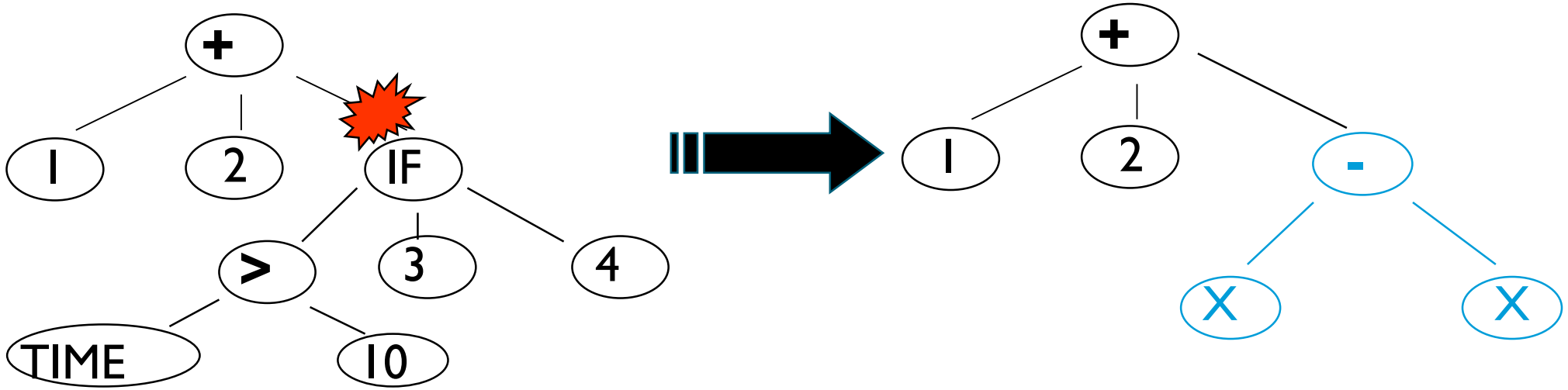
(+ (− (sqrt (− (∗ b b) a)) b) (sqrt (/ a b)))



(/ (− (sqrt (+ (∗ a a) (− a b))) a) (sqrt (− (∗ b b) a)))

(+ (−(∗ a b) b) (sqrt (/ a b)))

❖ OPÉRATEURS GÉNÉTIQUES : MUTATION [J.C.Routier et all]



- Destruction d 'un sous-arbre
- Remplacement par un sous-arbre aléatoire, créé comme lors de la génération de la population initiale.

NOTE SUR LES OPÉRATEURS GÉNÉTIQUES

- Le croisement ou la mutation sont susceptibles de transformer n'importe quel sous-arbre argument d'une fonction.

  ⇨ Les fonctions doivent être capables d 'accepter toutes sortes de valeurs en argument, et il est préférable qu 'elle aient toutes le même type de valeur de retour (*propriété de clôture*)

  Exemple : remplacer la division standard par la division « protégée » qui renvoie 0 ou un grand entier en cas de division par 0.

EXEMPLE [J.C.Routier et all]

$$c = \sqrt{a^2 + b^2}$$

- On veut trouver un programme qui calcule la fonction

- On dispose des **10** cas d'adaptation suivants, choisis au hasard parmi les domaines des variables **a** et **b** :

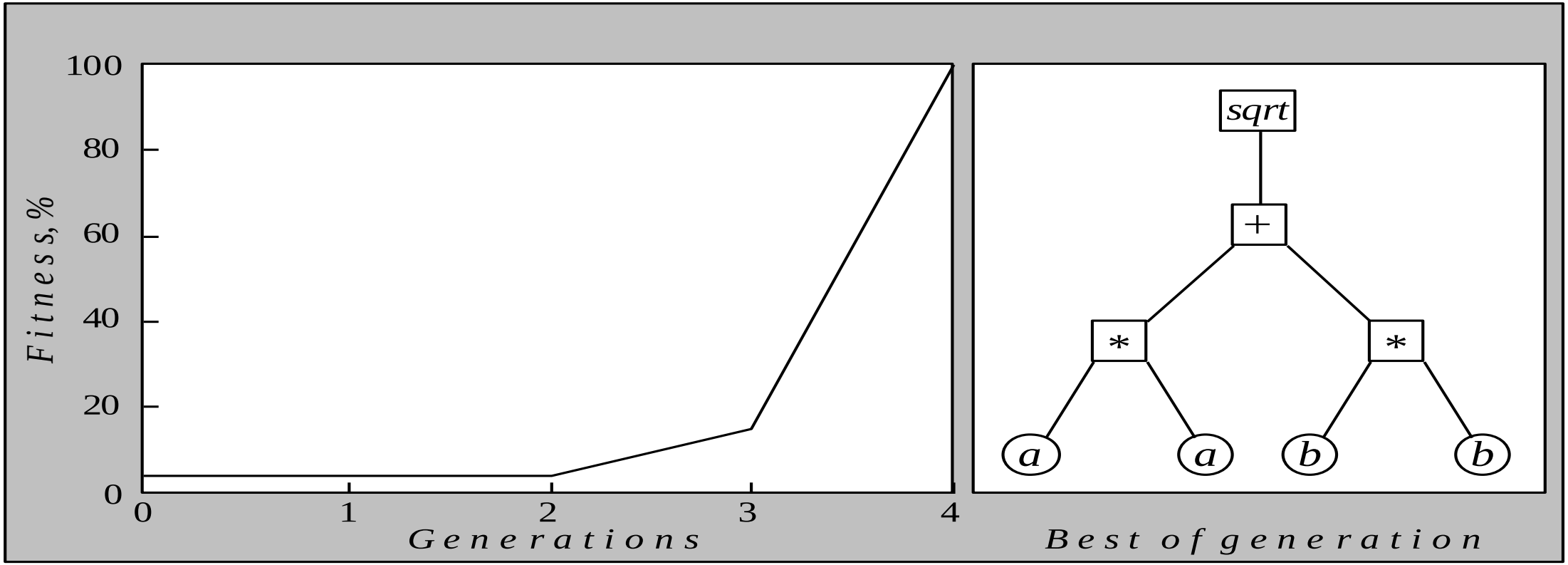
a	b	c	a	b	c
3	5	5.830952	12	10	15.620499
8	14	16.124515	21	6	21.840330
18	2	18.110770	7	4	8.062258
32	11	33.837849	16	24	28.844410
4	3	5.000000	2	9	9.219545

LES 5 ÉTAPES PRÉLIMINAIRES

1. Identification des terminaux : a et b
2. Choix des fonctions à utiliser : +, -, *, / et sqrt
3. Définition de la fonction d'adaptabilité : somme des erreurs quadratiques sur tous les cas entre le résultat calculé et celui donné par le tableau précédent
4. Paramètres : Taille de la population et nombre de générations
5. Méthode de sélection du meilleur programme : le meilleur à chaque génération.

Une fois ces étapes franchies, on génère au hasard une population initiale de programmes candidats et on part le cycle de reproduction on appliquant les opérations de croisement, mutation et clonage à chaque génération

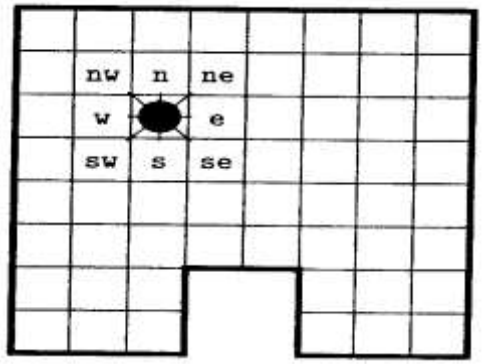
HISTORIQUE DE LA MEILLEURE S-EXPRESSION



- L'algorithme converge vers la bonne solution en 5 itérations [J.C.Routier et all]

EXEMPLE PLUS ÉVOLUÉ [J.C.ROUTIER ET ALL]

- le Problème [J.C.Routier et all] :

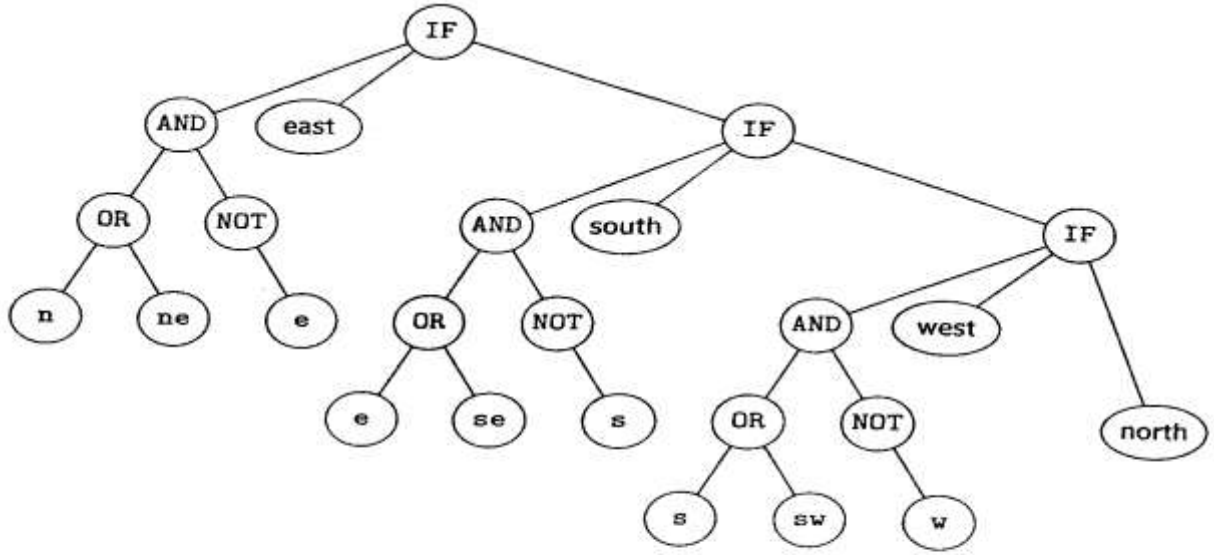


Un robot doit suivre le contour indiqué;
trouver la suite d'instructions à suivre.

Les fonctions autorisées :

`if/3`
`and/2 or/2 not/1`
`north south east west` *(déplacements)*
`s se sw e ne nw n w` *(tests d'obstacles ≈ capteurs)*

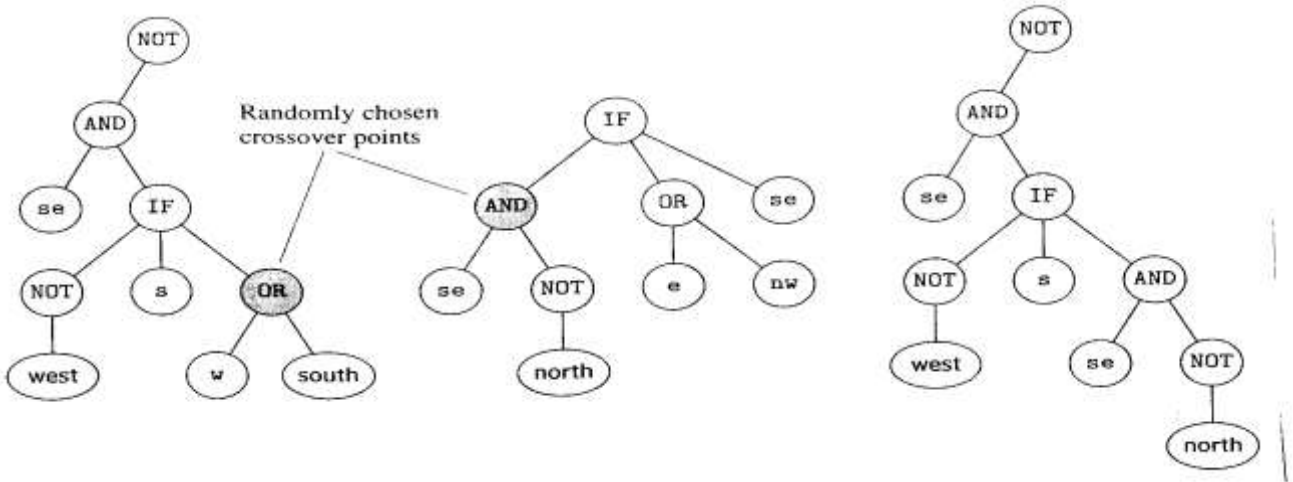
- un programme solution

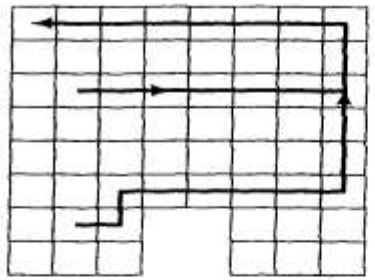



```
(IF (AND (OR (n) (ne)) (NOT (e)))
    (east)
    (IF (AND (OR (e) (se)) (NOT (s)))
        (south)
        (IF (AND (OR (s) (sw)) (NOT (w)))
            (west)
            (north))))
```

- population = 5000 individus (initialement aléatoire)
- sélection par
  - tournoi (10%)
  - croisement (90 %)
- Adaptation : on teste le programme sur 10 essais
  valeur = nbre cas du contour explorées (max = 320)

opération de croisement





```
(AND (NOT (NOT (IF (IF (NOT (nw))
                       (IF (e)(north) (east))
                       (IF (west)(0) (south))
                   (OR (IF (nw)(ne)(w))
                       (NOT (sw)))
                   (NOT (NOT (north))))))
     (IF (OR (NOT (AND (IF (sw)(north)(ne))
                       (AND (south)(1))))
             (OR (OR (NOT (s))
                     (OR (e)(e)))
                 (AND (IF (west)(ne)(se))
                      (IF (1) (e)(e)))))
         (OR (NOT (AND (NOT (ne))(IF (east)(s)(n))))
             (OR (NOT (IF (nw)(east)(s)))
                 (AND (IF (w)(sw)(1))
                      (OR (sw)(nw)))))
         (OR (NOT (IF (OR (n)(w))
                      (OR (0)(se))
                      (OR (1)(east))))
             (OR (AND (OR (1)(ne))
                      (AND (nw)(east)))
                 (IF (NOT (west))
                     (AND (west)(east))
                     (IF (1)(north)(w)))))))
```

Figure 4.5

The Most Fit Individual in Generation 0

AVANTAGES DE LA PG SUR LES AGs [J.C.ROUTIER ET ALL]

- Approches évolutionnaires similaires, mais PG n'est pas restreint à des chromosomes de longueur fixée à priori.
- Les éléments des expressions peuvent être de complexités différentes en comparaison des éléments dans les chaînes utilisées dans les AG.
- Comme dans AG, la représentation du problème en termes de chromosomes n'est pas évidente et peut mener à des fausses solutions si mal faite.

MULTI-OBJECTIF

OPTIMISATION MULTI-OBJECTIF [S.DONCIEUX]

Maximiser/minimiser $f_m(x)$, $m = 1,2,...,M$

sujet à $g_j(x) \geq 0$, $j = 1,2,...,J$

$h_k(x) = 0$, $k = 1,2,...,K$

$x_i^L \leq x_i \leq x_i^U$, $i = 1,2,...,n$

- Une solution est un vecteur **x** de **n** variables de décision : $\mathbf{x} = (x_1,x_2,...,x_n)^T$
- Chaque variable $\mathbf{x_i}$ est contrainte d'appartenir à un intervalle $[x_i^{(L)}, x_i^U]$: défini un espace de décision.
- Les termes $\mathbf{h_k(x)}$ et $\mathbf{g_j(x)}$ sont des fonctions de contraintes.
- Une solution **x** qui ne satisfait pas les **J+K** contraintes et les **2N** limites est dite infaisable alors qu'une solution les **satisfaisant** est une solution **faisable**.

OPTIMISATION MULTI-OBJECTIF

Les approches heuristiques peuvent être classées en trois catégories [C.D. Flipo] :

- **Approches transformant le problème en un ou plusieurs problème(s) mono-objectif(s)**

  Ces approches consistent à transformer un problème donné en un ou plusieurs problèmes mono-objectifs.

  - Nécessitent une connaissance du problème et ne donne au final qu'une seule solution.

- **Approches Non Pareto**

  Ces approches transforment le problème initial.

  - Elles traitent les objectifs indépendamment tout le long de la recherche.
  - Elles ont du mal à trouver les solutions de compromis parce qu'elle traite les objectifs indépendamment.

- **Approches Pareto**

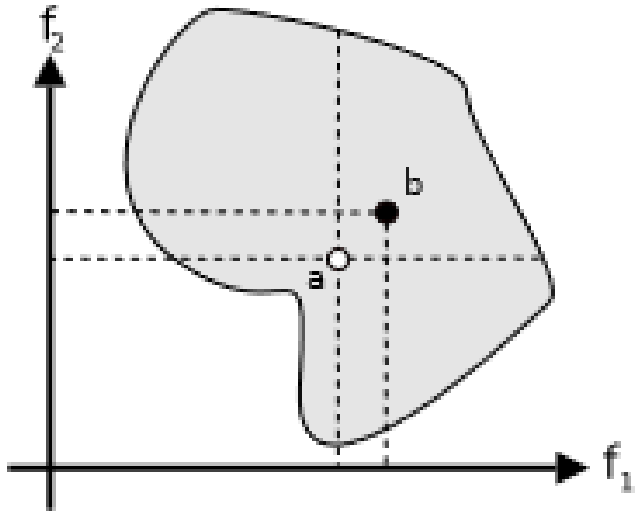
  Ces approches utilisent la notion de dominance pour comparer les solutions entre elles par rapport à tous les objectifs en même temps.

  - Une seule résolution permet d'approximer l'ensemble de la frontière Pareto.

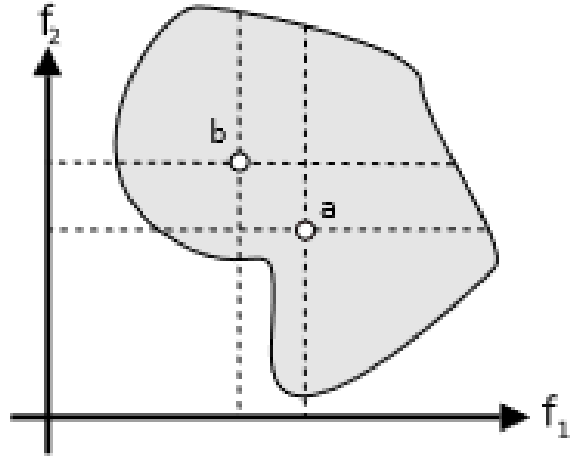
RELATION DE DOMINATION (1)

Définition [S.Doncieux]

- Une solution $\mathbf{x}^{(1)}$ **domine** $\mathbf{x}^{(2)}$ si :
  - $\mathbf{x}^{(1)}$ **n'est pas pire que** $x^{(2)}$ sur tous les objectifs
  - $\mathbf{x}^{(1)}$ est **strictement meilleur** que $\mathbf{x}^{(2)}$ sur au moins un objectif



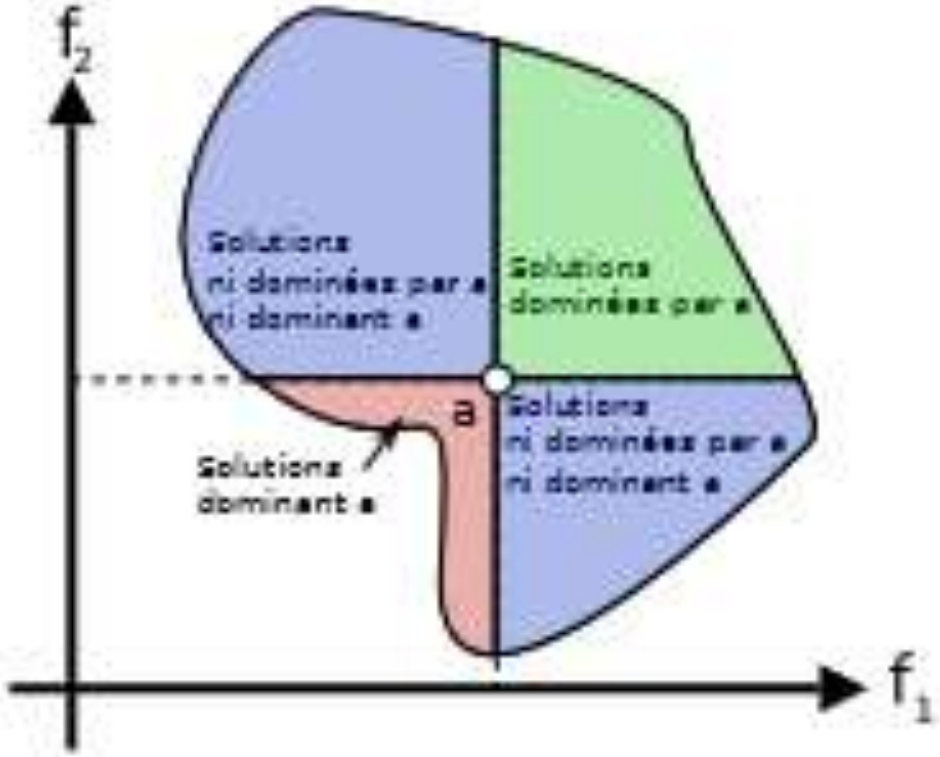
a domine *b*



a ne domine pas *b* et *b* ne domine pas *a*

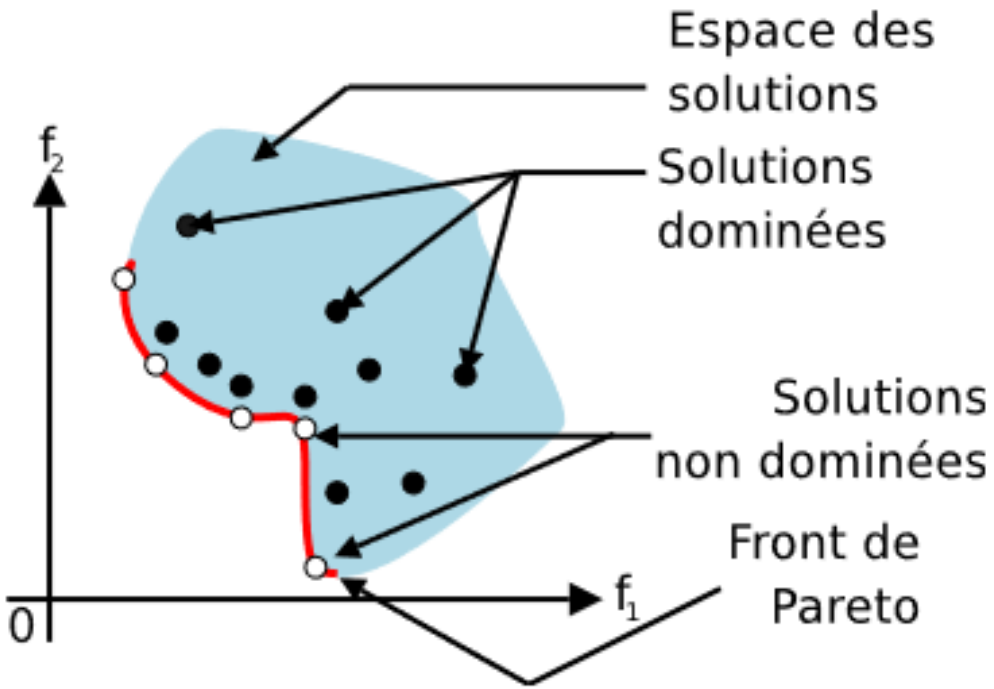
RELATION DE DOMINATION (2)

- Découpage de l'espace des solutions relativement à la relation de domination sur a [S.Doncieux].



RELATION DE DOMINATION (3)

- L'ensemble des solutions **non-dominées** est appelé l'ensemble des **solutions globalement Pareto-optimales** (ou **front de Pareto**) [S.Doncieux].



MÉTHODES CLASSIQUES [S.DONCIEUX]

- Somme pondérée
- Méthode de l' ε -contrainte
- Méthode de la métrique pondérée

❖ **Avantage**

certaines permettent de trouver chaque point Pareto-optimal

❖ **Inconvénients**

✓ Nécessite de déterminer des paramètres a priori : peut impliquer des itérations ou une optimisation préalable

✓ Un seul point trouvé à la fois

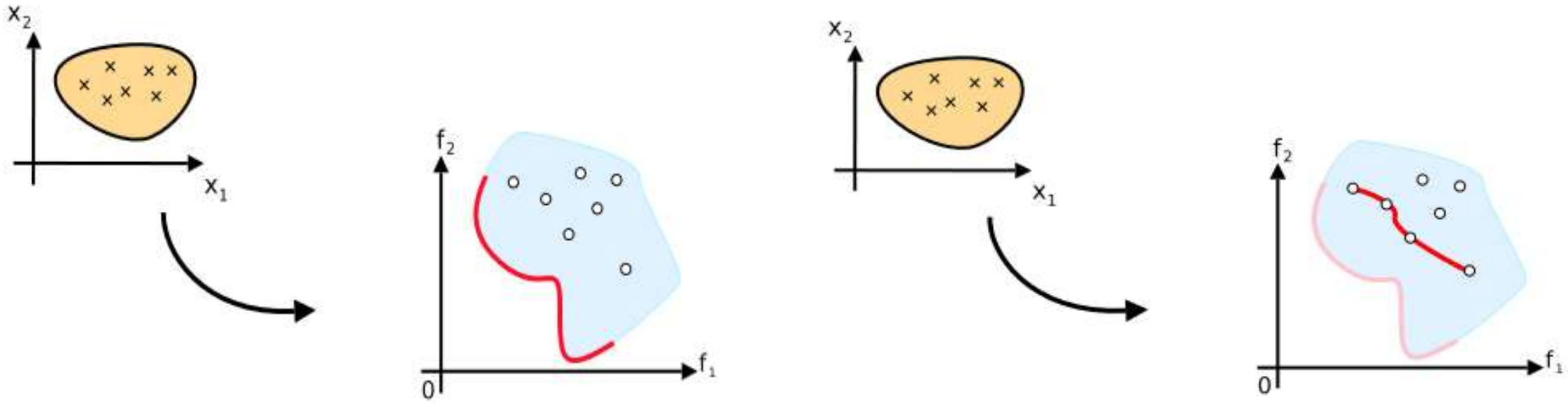
ALGORITHMES ÉVOLUTIONNISTES MULTI-OBJECTIFS (AEv-MOs)

- Les algorithmes génèrent une population de solutions candidates.

  → peuvent-ils converger directement vers une approximation du front de Pareto?

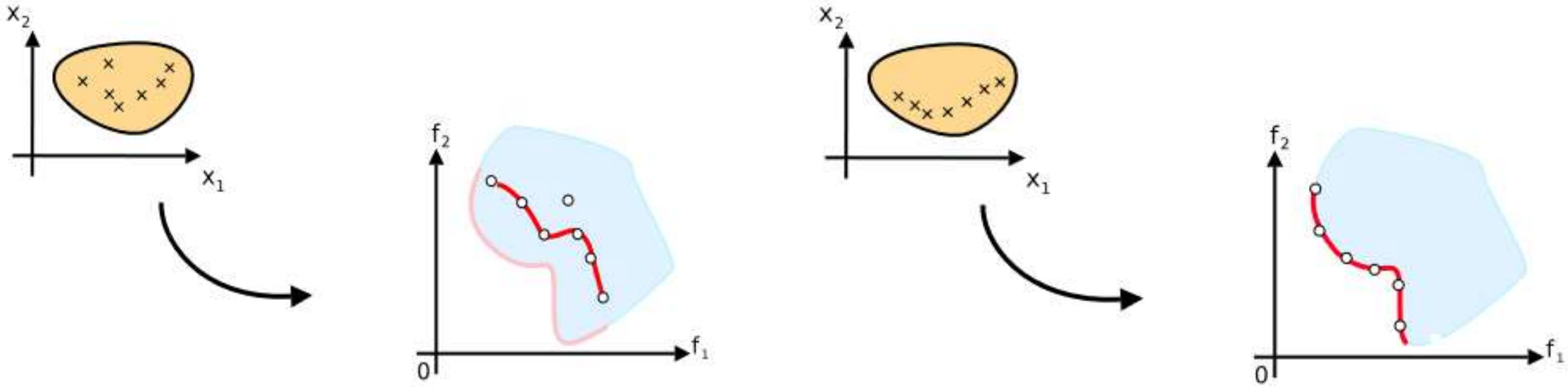
- **Avantages potentiels** [S.Doncieux]
  - Connaissance directe du front et de ses caractéristiques
  - Pas besoin de définir la zone intéressante du front
  - Possibilité de choisir a posteriori les solutions les plus intéressantes

AEv-MOs (1)



Graphe des solutions d'un Algorithmes Évolutionnistes Multi-Objectifs (AEv-MOs) [S.Doncieux].

AEv-MOs (2)



Graphe des solutions d'un Algorithmes Évolutionnistes Multi-Objectifs (AEv-MOs) [S.Doncieux].

AEv-MOs (3)

Algorithmes présentés :

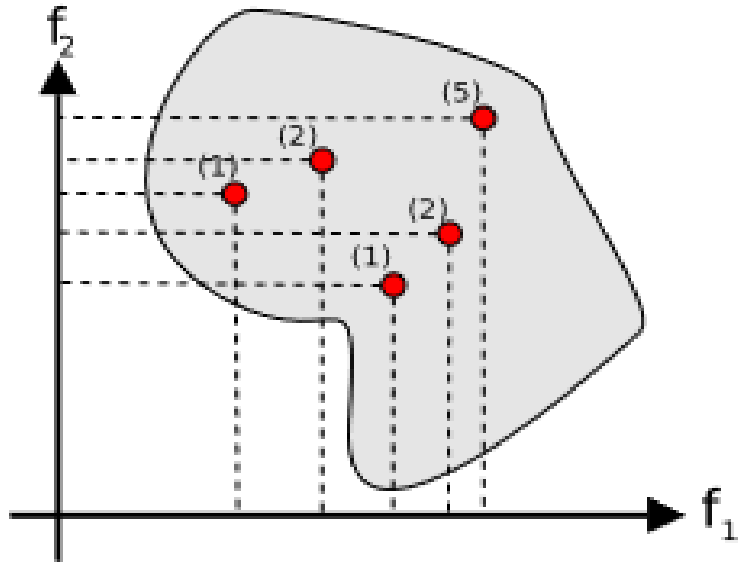
- Algorithme simple : MOGA
- Algorithme simple :VEGA
- Algorithmes les plus efficaces actuellement : NSGA-II, ε-MOEA, NSGA-III

MULTI-OBJECTIVE GENETIC ALGORITHM (MOGA)

- Algorithme similaire à un AG classique [S.Doncieux].
- Différence : façon de calculer la fitness.

- Calcul du rang de chaque individu :

r(i) = 1+n(i)

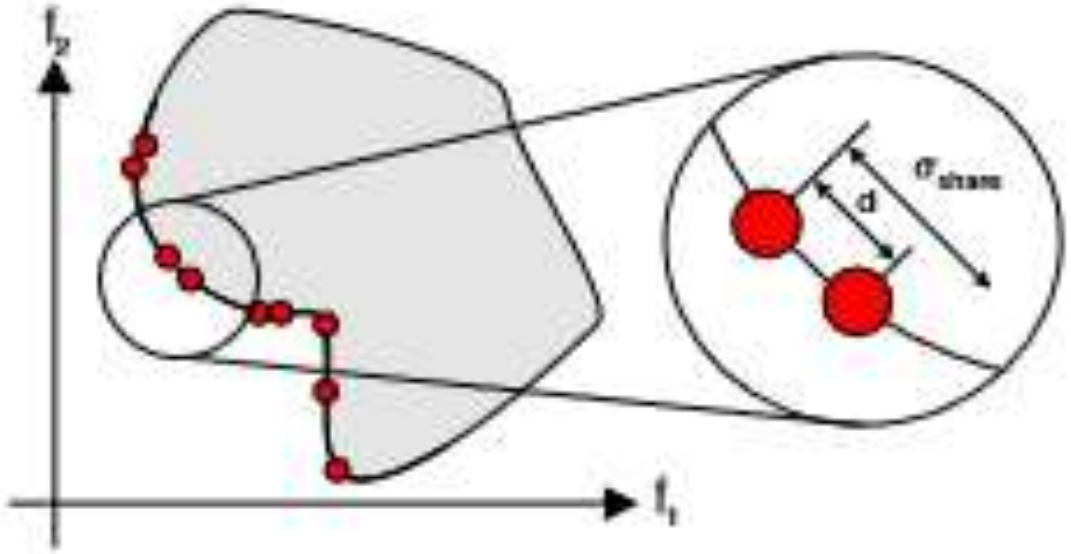


MULTI-OBJECTIVE GENETIC ALGORITHM (MOGA)

- Application d'un coefficient de "**sharing**" pour assurer une bonne couverture du front de pareto [S.Doncieux].

f(i) = r(i) + σshare(i)

0 < σshare(i) < 1



NSGA-II

- Elitist Non-Dominated Sorting Genetic Algorithm (NSGA-II) [S.Doncieux].

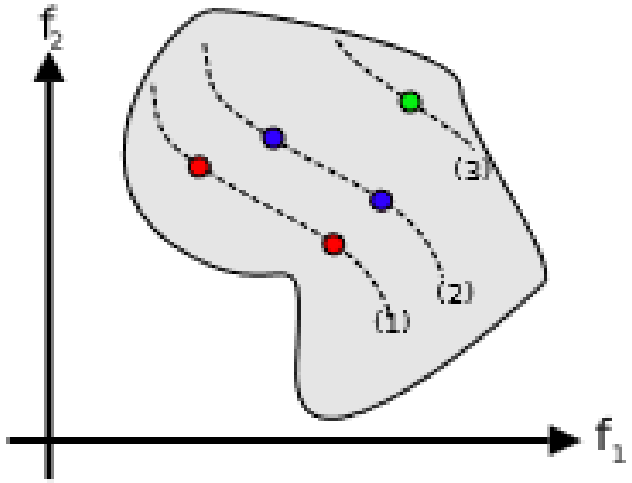
Principe :

- Algorithme élitiste : garde les meilleurs individus parmi les parents et les enfants
- Calcul des différents fronts d'individus non-dominés
- Remplissage de la nouvelle population :
  - ✓ ajout des fronts dans l'ordre croissant de leur rang
  - ✓ lorsqu'un front ne peut être ajouté en entier : prise en compte d'un coefficient de sharing
  - ✓ suppression des autres individus

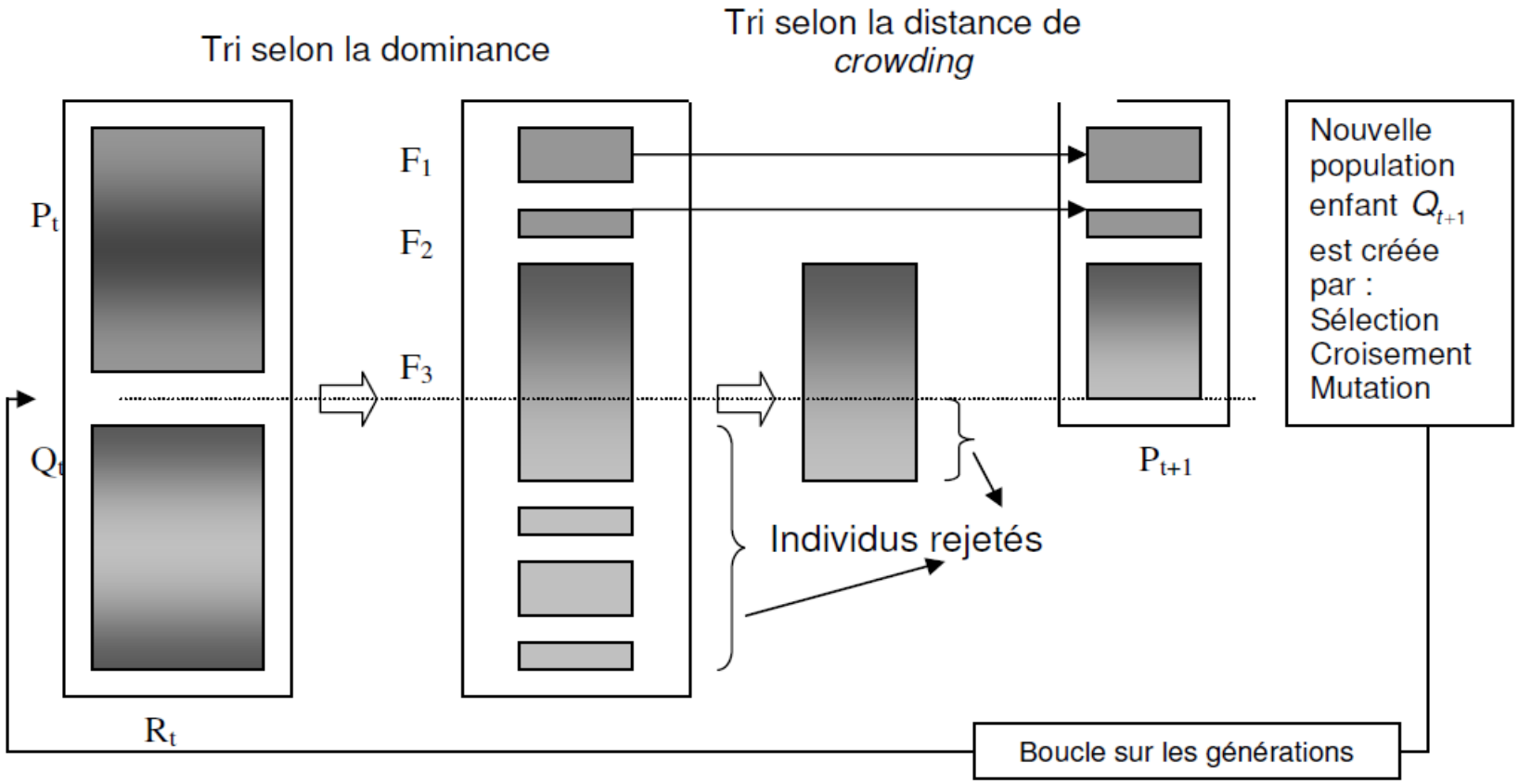
NSGA-II

Calcul des différents fronts :

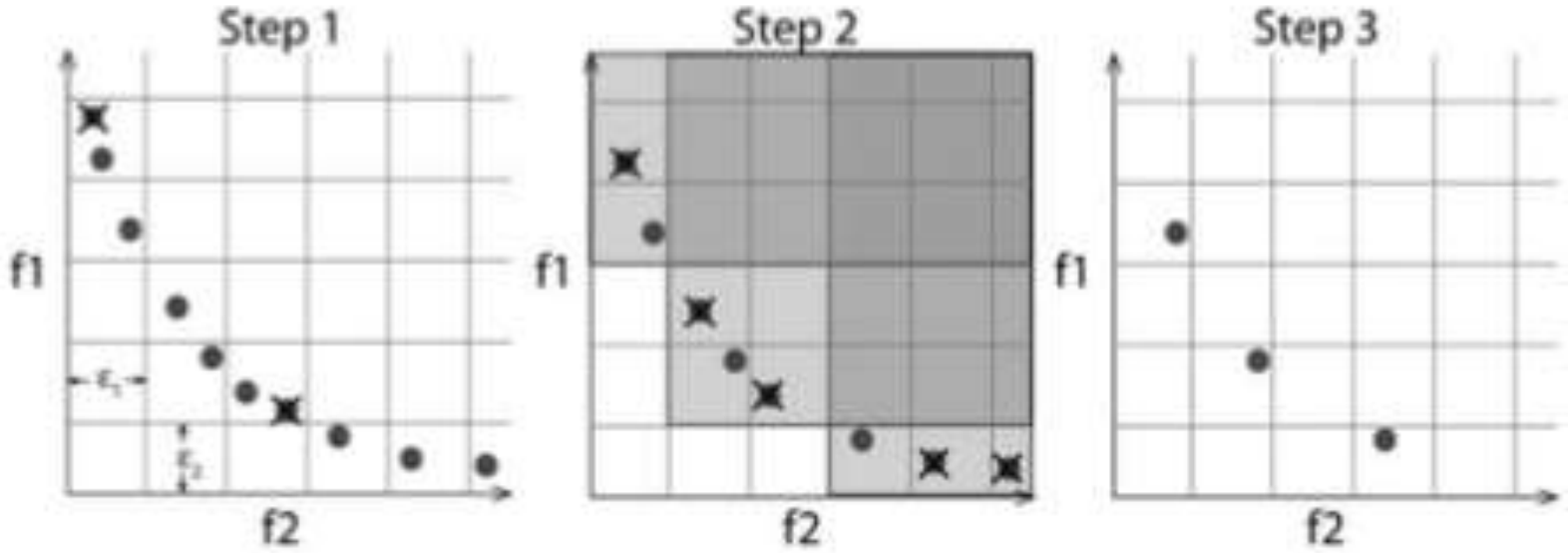
1. front de Pareto : premier front
2. suppression temporaire des individus de ce front de la population
3. calcul du nouveau front d'individus non-dominés : deuxième front
4. suppression temporaire des individus de ce front
5. réitération jusqu'à ce que tous les individus aient été assignés à un front



NSGA-II


Tri selon la dominance
Tri selon la distance de *crowding*
P_t
Q_t
R_t
F_1
F_2
F_3
P_{t+1}
Individus rejetés
Nouvelle population enfant Q_{t+1} est créée par :
Sélection
Croisement
Mutation
Boucle sur les générations

E -DOMINATION



E-MOEA

- Algorithme "steady-state" : un seul individu est généré à la fois pour être éventuellement inclus dans la population.
- co-existence d'une population et d'une archive des individus ε-dominants

E-MOEA

Principe :

- Initialisation de la population et de l'archive sur la base des individus ε-dominants
- Création d'un nouvel individu :
  - ❖ choix aléatoire d'un individu de l'archive
  - ❖ tirage aléatoire de 2 individus de la population et choix du dominant (ou aléatoire si pas possible)
  - ❖ croisement
- Réinsertion du nouvel individu :
  - ✓ dans la population : s'il domine tout ou n'est pas dominé
  - ✓ dans l'archive : s'il ε-domine ou appartient à un nouvel ε-hypercube

ANALYSE MULTI-OBJECTIF

Optimisation : Pourquoi faire?

- utilisation en ingénierie : optimisation de processus industriels en vue de gains de productivité
- utilisation en science :
  - aide à la conception de modèles (optimisation de paramètres)
  - aide à l'étude de modèles complexes paramétrés

Les solutions Pareto-optimales sont (généralement) nombreuses, optimales et arbitrairement proches :

- elles donnent un retour sur ce qui est possible
- analyser les liens entre critères et paramètres pour comprendre le système étudié

ANALYSE MULTI-OBJECTIF

Principe

1. Choisir quelques objectifs antagonistes (<3)
2. Définir l'espace de recherche
3. Trouver une approximation du front de Pareto
4. Analyser les résultats

DANS QUELS CAS UTILISER DES AE?

- Problèmes multi-objectifs
- La fonction à optimiser ne dispose pas des propriétés mathématiques permettant d'utiliser d'autres algorithmes
- Existence d'extrema locaux multiples
- conception de structures...

RÉFÉRENCES

- Les algorithmes évolutionnistes, INF6953
- Stéphane Doncieux, Algorithmes évolutionnistes, ISIR, UPMC.
- Souquet Amédée et Radet Francois-Gérard, algorithmes genetiques, 2004
- Jean-Christophe Routier et Michael Negnevitsky, Algorithmes et programmation génétiques,
- Clarisse Dhaenens-Flipo, Optimisation Combinatoire Multi-Objectif : Apport des Méthodes Coopératives et Contribution à l'Extraction de Connaissances. (2005). Université des Sciences et Technologies de Lille. Thèse d'habilitation pour la direction des recherches de l'U.S.T.L.