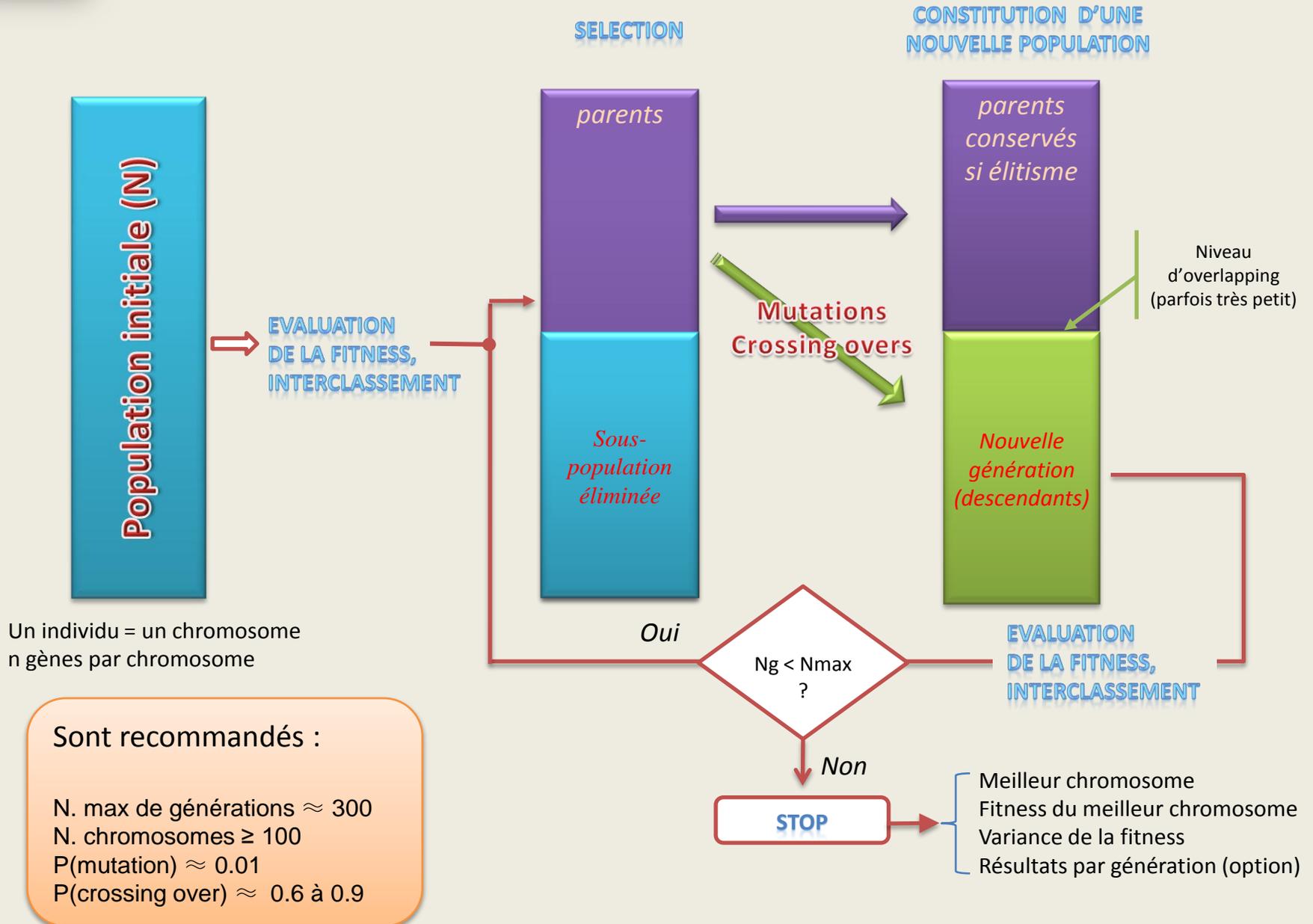


Principe général de fonctionnement des algorithmes génétiques



- Les Algorithmes Génétiques (GA) font partie de la programmation évolutive (*evolutionary programming*¹), c'est-à-dire bio-inspirée.
- Ils forment une classe d'heuristiques² permettant d'adresser un nombre considérable de problèmes dans tous les domaines.
- Ils sont utiles lorsque l'obtention de solutions exactes est impossible ou bien inatteignables en temps raisonnable.
- Leur complexité est en $O(nLn(n))$, n étant le nombre de variables à estimer

Par comparaison le problème du voyageur de commerce (dit NP-difficile) est en $O(n!)$ pour un algorithme déterministe (exploration de toutes les solutions !)



<i>NB VILLES</i>	<i>NB POSSIBILITES</i>	<i>TEMPS DE CALCUL</i>
5	120	120µs
10	181440	0.18 ms
15	43 MILLIARDS	12 h
20	60 E +15	1928 ans
25	310 E + 21	9,8 Milliards A.

¹ Ne pas confondre avec la programmation dynamique

² Une heuristique est une méthode de calcul qui fournit rapidement (en temps polynomial) une solution réalisable, pas nécessairement optimale, pour un problème d'optimisation NP-difficile.

Encodage des gènes

Encodage binaire

Chromosome A	101100101100101011100101
Chromosome B	111111100000110000011111

Tout nombre réel peut être encodé en une chaîne de 0 et 1 (bit string) par une opération d'encodage (pour crossing-overs et mutations) puis décodée pour évaluer la fitness de la fonction.

Encodage permutation

Chromosome A	1 5 3 2 6 4 7 9 8
Chromosome B	8 5 6 7 2 3 1 4 9

Très spécifique aux problèmes d'ordonnement. On cherche un ordre optimal de la séquence (par exemple des trajets entre villes, de tâches à effectuer...)

Encodage par valeurs

Chromosome A	1.2324 5.3243 0.4556 2.3293 2.4545
Chromosome B	ABDJEIFJDHDIERJFDLDFLFEGT
Chromosome C	(back), (back), (right), (forward), (left)

Nécessite des opérateurs de cross over et mutation particuliers. Par exemple on additionne (ou soustrait) une petite valeur au réel pour provoquer une mutation, on effectue $D = 1.5P1 - 0.5P2$ pour les cross over (croisement linéaire de wright).

Exemple de l'encodage binaire de valeurs réelles

- Soit d le nombre de décimales désiré (la précision).
- Soient x_{max} et x_{min} les bornes supérieures et inférieures de l'intervalle de variation pour x
- Soit n la taille minimale de la chaîne binaire.
- n est alors le plus petit entier tel que : $|x_{max} - x_{min}| \cdot 10^d \leq 2^n$ (1)

Ainsi, pour $x \in [-1.28, 1.28]$, si l'on veut une précision de 2 décimales, il faudra prendre $n = 8$ pour le codage. En effet, comme : $(1.28 + 1.28) \cdot 10^2 = 256$ et que $2^8 = 256$,
On en déduit que $n = 8$ est le plus petit entier qui satisfait la relation (1).

- Quand $n = 8$ et $x \in [-1.28, 1.28]$, la chaîne binaire 00000000 correspondra alors à la borne inférieure $x_{min} = -1.28$ alors que 11111111 représentera la borne supérieure $x_{max} = +1.28$.
- Les valeurs intermédiaires sont linéairement réparties entre les deux bornes. De ce fait, on peut associer une valeur de x à n'importe quelle chaîne de longueur $n = 8$, grâce à la formule suivante :

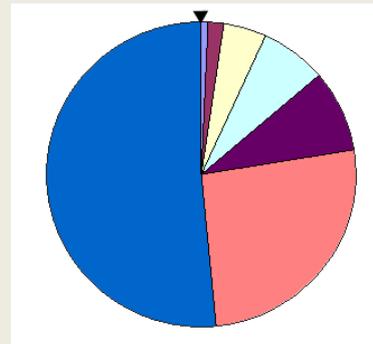
$$(2^n - 1) \cdot (x - x_{min}) / (x_{max} - x_{min})$$

SELECTION – Roulette wheel

Séquence	Valeur	U(x)	% de chance de reproduction	% cumulés
00011010	0.1015625	0.364990	$0.364990 / 2.595947 = 0.14$	0.14
11011110	0.8671875	0.460693	$0.460693 / 2.595947 = 0.18$	$0.14 + 0.18 = 0.32$
10111010	0.7265625	0.794678	$0.794678 / 2.595947 = 0.31$	$0.32 + 0.31 = 0.63$
01101100	0.4218750	0.975586	$0.975586 / 2.595947 = 0.37$	$0.63 + 0.37 = 1.00$
TOTAL =		2.595947		

On tire un nombre $p \in [0,1]$. Soit $p = 0.55$, dans la colonne 5 on trouve que le chromosome N°3 est sélectionné.

D'après J.M. Alliot (directeur du Centre International de Mathématiques et d'Informatique de Toulouse).



Roulette wheel, représentation schématique. Les aires sont proportionnelles aux fitness des chromosomes

SELECTION – par rang

Variante de la roulette. Les chromosomes sont classés par rang (1 à N) de fitness. L'aire de représentation de la roulette est proportionnelle au rang du chromosome

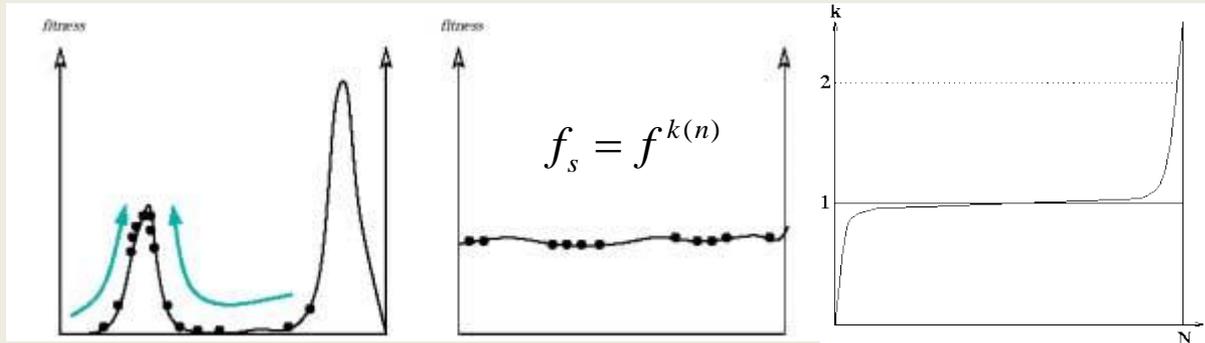
SELECTION – par tournoi

On tire au hasard plusieurs individus dans la population. Le meilleur est gardé, les individus médiocres ont donc quelques chances d'être conservés.

SELECTION – steady state « état stable »

- Ici, on sélectionne les n meilleurs chromosomes comme parents. Ils seront conservés (tous ou partie) si on a choisi un algorithme avec élitisme. Cette méthode est très utilisée car moins coûteuse en temps.
- On effectue un crossing over (selon une certaine probabilité) entre deux parents pour produire deux descendants. Les mutations sont appliquées (selon une certaine probabilité) aux parents pour produire des descendants.

Les pièges



Lorsque l'espace de recherche présente de très forts gradients,, il peut y avoir une convergence prématurée vers un maximum local (à gauche).

Le remède s'appelle le **scaling** : par une mise à échelle artificielle (linéaire ou exponentielle), on réduit l'amplitude de fitness entre chromosomes. La sélection se fait alors non plus sur la véritable fitness f (à gauche) mais sur son image après scaling : f_s (milieu). Les chromosomes les moins performants ne sont alors pas (peu) pénalisés et continuent de se reproduire, de la même façon que les meilleurs...

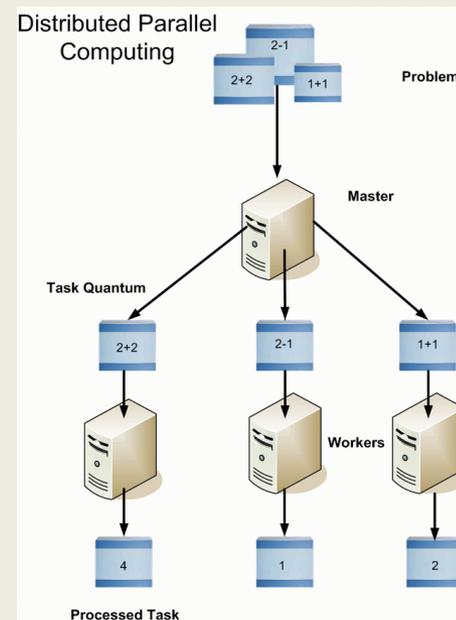
Cette méthode présente l'inconvénient de gêner la convergence en fin de processus, car l'on souhaiterait alors favoriser les meilleurs. Pour éviter cela, le scaling exponentiel est préférable, car l'on peut faire varier le paramètre k au cours des générations (à droite), avec une formule du type :

$$k(n) = \left(\tan\left(\frac{n\pi}{2(N+1)}\right) \right)^{0.1} \quad n = n^\circ \text{ de génération, } N = n^\circ \text{ de l'ultime génération}$$

Avantages

- Ils ont un grand pouvoir d'exploration en largeur de l'espace de solution.
- Ils peuvent traiter un problème multi-objectifs (on veut optimiser une fonction selon plusieurs critères)
- Ils peuvent fournir plusieurs solutions dans le cas de problèmes multimodaux.
- Ils peuvent converger même quand le problème comporte des éléments stochastiques
- Ils sont facilement parallélisables pour un gain de temps calcul = on distribue la population en n sous-populations sur autant de machines (distributed computing) ou cœurs d'une machine unique (parallel computing). Un mécanisme de rendez-vous reconstitue l'ensemble de la population avant la génération suivante (Message Parsing Interface: MPI, OpenMPI).

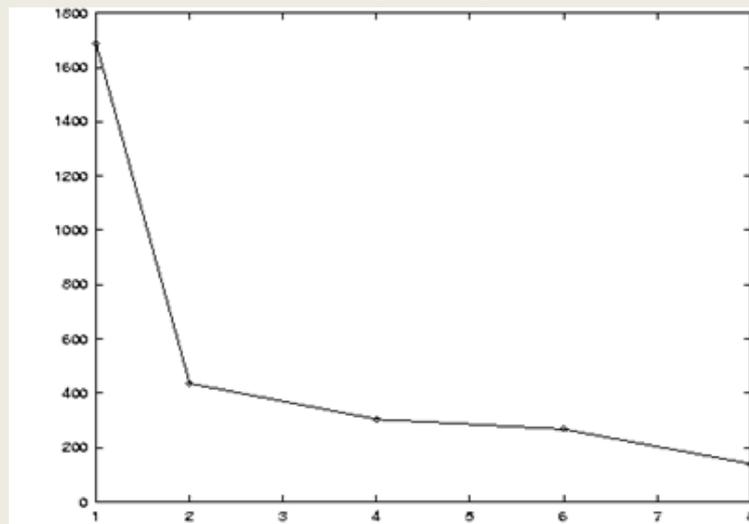
Principe du parallélisme par distribution



Exemple de code R permettant de transformer des données (transformData()) en utilisant les 6 cœurs d'une machine de bureau :

```
library(foreach) # 2 packages
library(doSNOW)
cl <- makeCluster(6, type="SOCK") # on utilise 6 cœurs
registerDoSNOW(cl) uID <- unique(ID)
foreach(i=icount(length(uID)) %dopar% {
transformData(dat[dat$ID==uID[i,]]) }
stopCluster(cl)
```

Évolution du temps de calcul en fonction du
nombre de machines (cœurs)



Limites et difficultés

- Ils sont (beaucoup) moins efficaces (temps et résultats) qu'un algorithme déterministe vis-à-vis d'un problème donné.
- Il ne garantissent en aucun cas que la solution obtenue est la meilleure, ni l'unique.
- Les nombreux paramètres qui les contrôlent sont délicats à régler (probabilités, scaling, élitisme, overlap...), et il n'existe pas de règle définitive à ce sujet.
- Le choix de codage est arbitraire. Cela peut influencer grandement la vitesse de convergence.
- Ils peuvent avoir des difficultés à converger localement.
- Afin de garantir la robustesse des algorithmes évolutifs, le calcul d'un très grand nombre de fitness (parfois de l'ordre de plusieurs centaines de milliers).
- Ils peuvent éprouver des difficultés à gérer des contraintes nombreuses et complexes.

Quelques références

Calcul parallèle

Scilab : <http://ats.cs.ut.ee/u/kt/hw/scilabpvm/>

R : <http://cran.fhcrc.org/web/views/HighPerformanceComputing.html>

Algorithmes génétiques

Scilab: https://help.scilab.org/docs/5.4.0/fr_FR/section_3aae18570efc509132b58dad425a00af.html

R : <http://cran.r-project.org/web/packages/GA/index.html>

Voir aussi dans la page : <http://sites.unice.fr/coquillard/links.html>