ALGORITHMES ÉVOLUTIONNAIRES

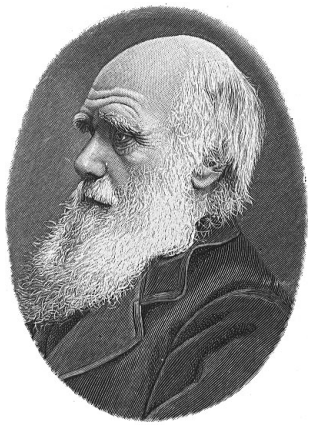
DARWINISME ARTIFICIEL

Evelyne LUTTON - **INRA AgroParisTech - Grignon**

http ://evelyne-lutton.fr/

DARWINISME, ÉVOLUTIONNISME.

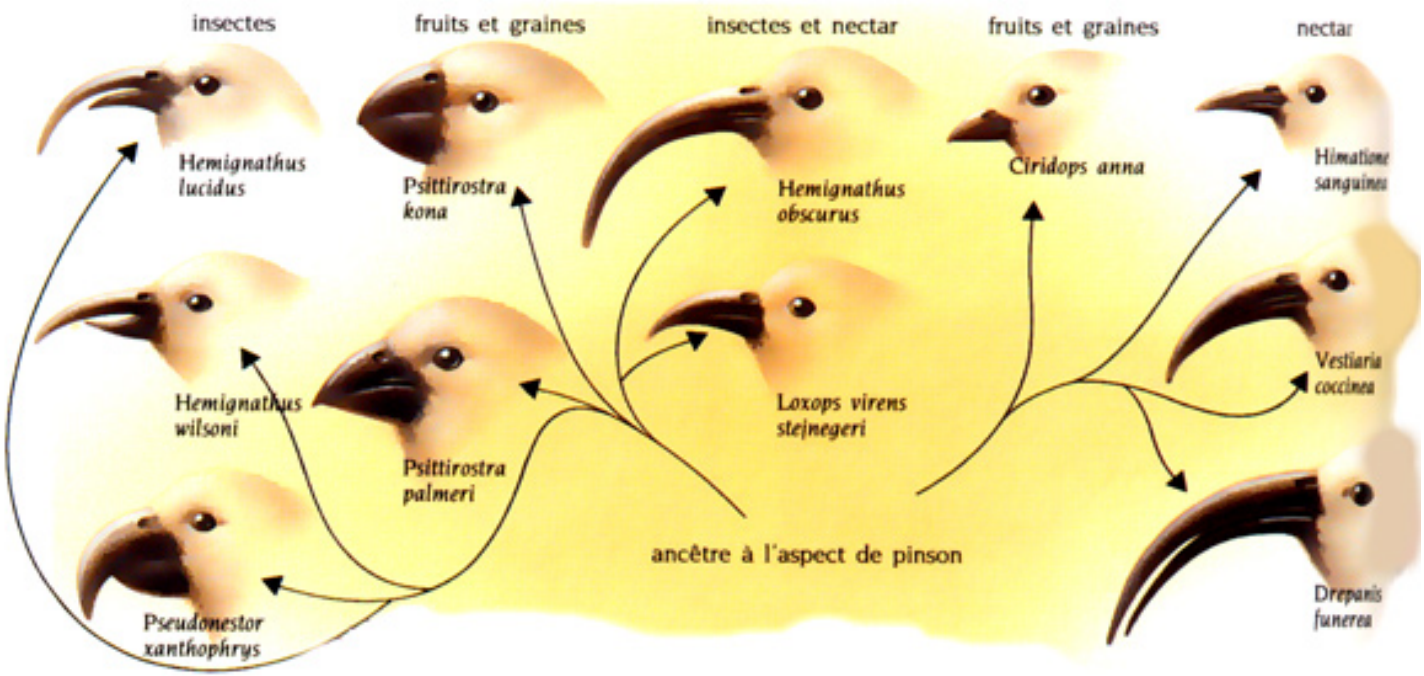
Charles Robert Darwin (1809-1882).



1831 - Voyage de 5 ans sur le HMS Beagle (îles Galapagos)

Ouvrage “Sur l’origine des espèces”, Novembre 1859.

Le pinson de Darwin



« On peut dire, par métaphore, que la sélection naturelle recherche, à chaque instant et dans le monde entier, les variations les plus légères ; elle repousse celles qui sont nuisibles, elle conserve et accumule celles qui sont utiles ; elle travaille en silence, insensiblement, partout et toujours, dès que l'occasion s'en présente, pour améliorer tous les êtres organisés relativement à leurs conditions d'existence organiques et inorganiques » (darwin, 1859).

Des mécanismes ultra-simples ...

– **Variations,** macroscopiques et microscopiques, au sein des espèces.

– **Lutte pour la vie.**

– **Sélection naturelle :** triomphe de la lignée qui possède une variation utile dans son environnement.

LE DARWINISME ARTIFICIEL

Utiliser des principes d'évolution organique en tant que technique d'optimisation globale.

Imiter les phénomènes d'apprentissage collectifs (adaptation) des populations naturelles.

Deux grands courants :

- **algorithmes génétiques et systèmes de classeurs :**
  John Holland, David Goldberg.

- **stratégies d'évolution et evolutionary computation :**
  Ingo Rechenberg, Hans-Paul Schwefel, Lawrence Fogel.

... Un ensemble de techniques regroupées sous un terme générique

Algorithmes Évolutionnaires (AE) = {

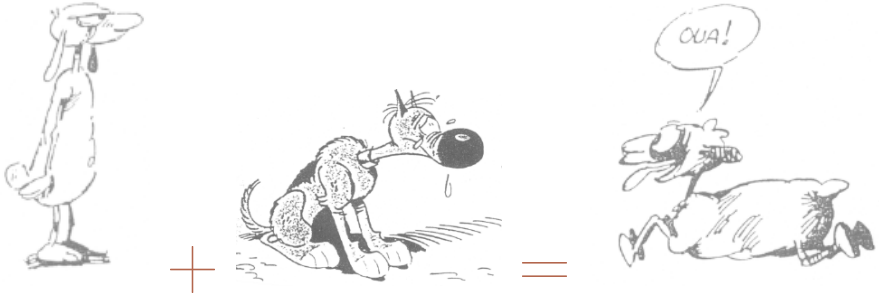
- Algorithmes Génétiques (AG),
- Stratégies d'Évolution (SE),
- Programmation Évolutionnaire (PE),
- Programmation Génétique (PG),
- Systèmes de Classeurs (CS).

INGRÉDIENTS

Population

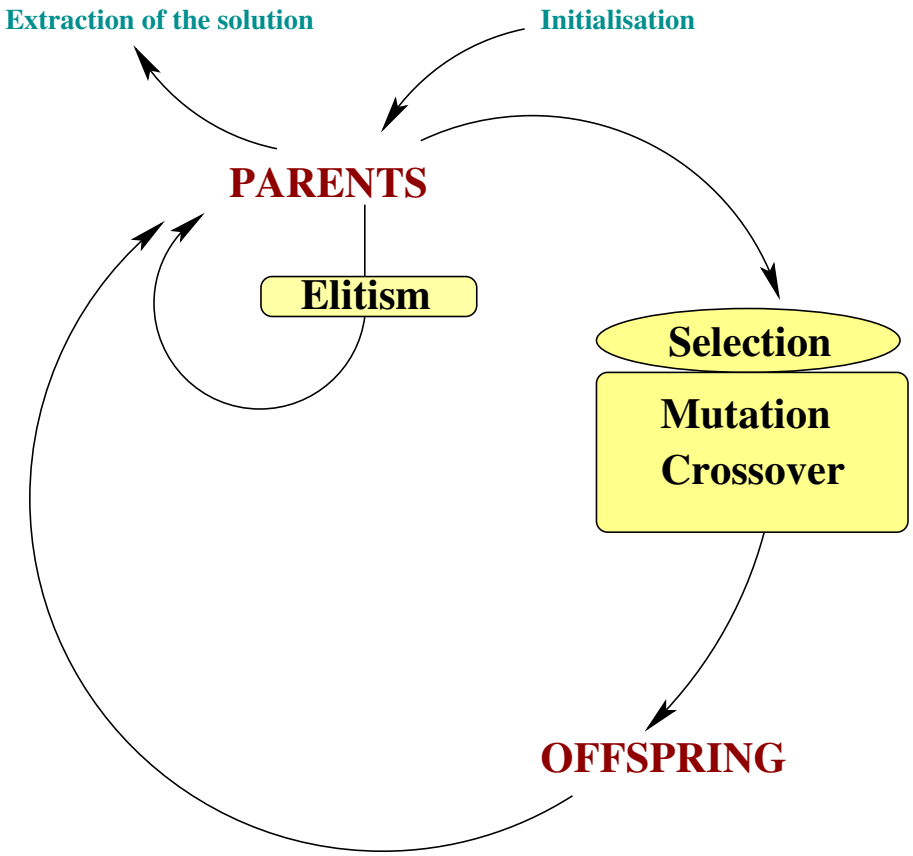
Sélection

Opérateurs génétiques



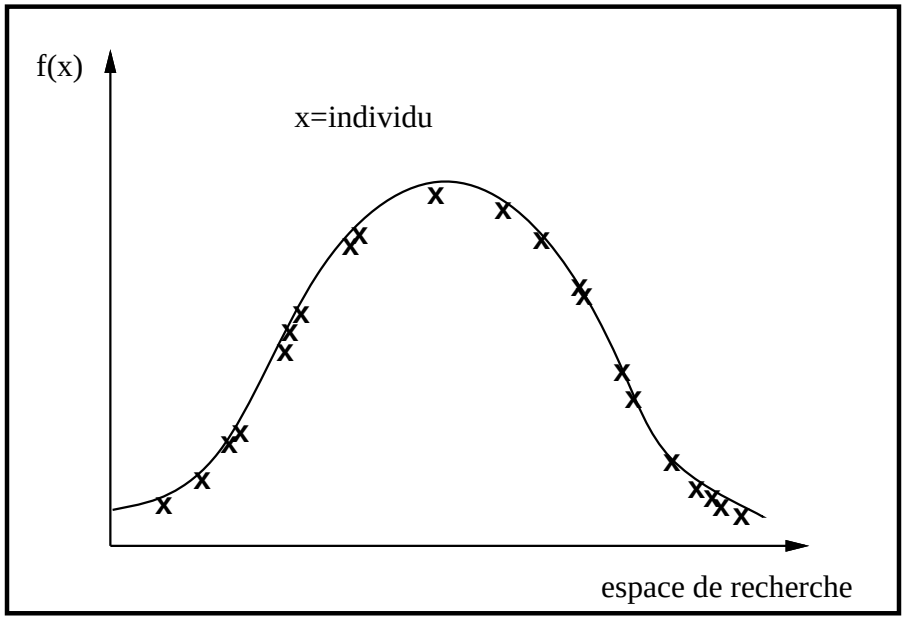
Évolution simulée

La boucle évolutionnaire

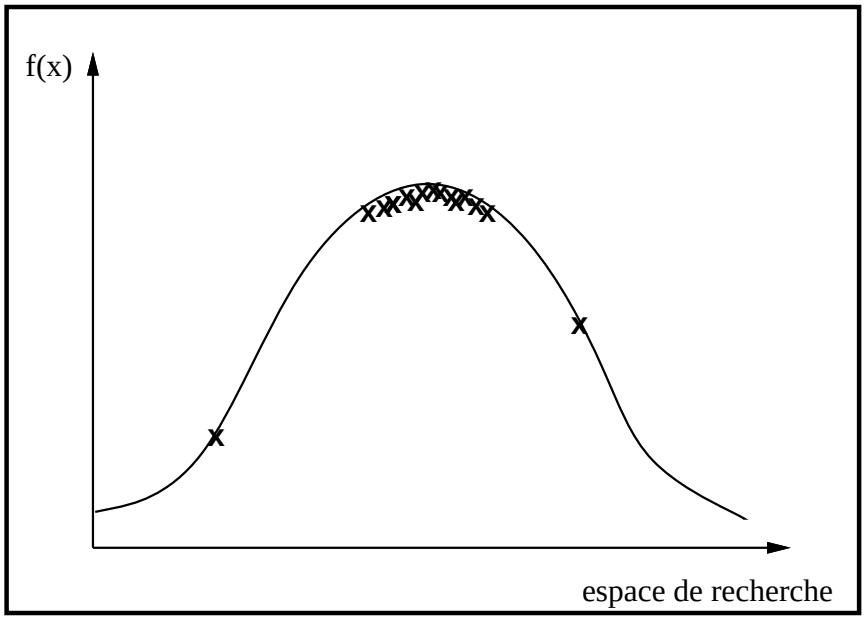


OPTIMISER UNE “ADAPTATION À L’ENVIRONNEMENT”

Solutions = individus d’une population.



Population initale aléatoire



Population évoluée

Un peu de vocabulaire ...

Algorithme évolutionnaire	Méthode d'optimisation
individu	solution : vecteur
population	ensemble de solutions
chromosome	codage de la solution (binaire par exemple)
croisement ou recombinaison	opération sur deux codes
mutation	opération sur un code
environnement	espace de recherche
degré d'adaptation à l'environnement "fitness"	valeur de la fonction d'évaluation
évolution	maximisation de la fonction d'évaluation

LE MOTEUR ÉVOLUTIONNAIRE

Évaluation : estimer la qualité d'un individu.
⟶"fitness", "performance", "fonction d'évaluation", "adaptation à l'environnement."

Sélection : trier les meilleurs individus.
⟶ tirages aléatoires biaisés (roulette wheel), sélection sur le rang, par tournoi.

Reproduction : appliquer les opérateurs génétiques, *croisements* et *mutations*, avec des probabilités p_c et p_m.

⟶ Lié à la représentation de l'espace de recherche :

Solutions ⟷ chromosomes
Phénotypes ⟷ génotypes

Remplacement : fabriquer la génération suivante.
⟶ élitisme, pourcentage de renouvellement de la population, stratégies $(\mu + \lambda)$ ou (μ, λ).

INITIALISATION / ARRÊT DU PROCESSUS

Initialisation :

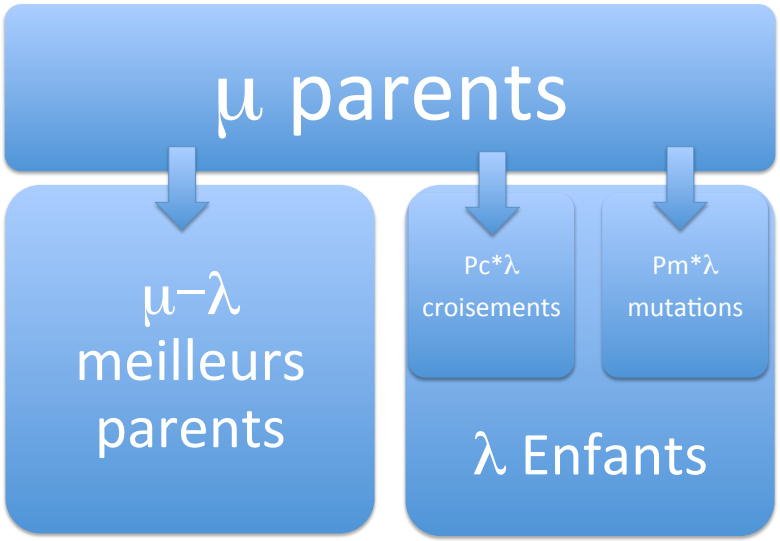
échantillonnage l'espace de recherche (aléatoire, régulier),
introduction de solutions initiales,
restrictions de la recherche.

Arrêt, extraction des solutions :

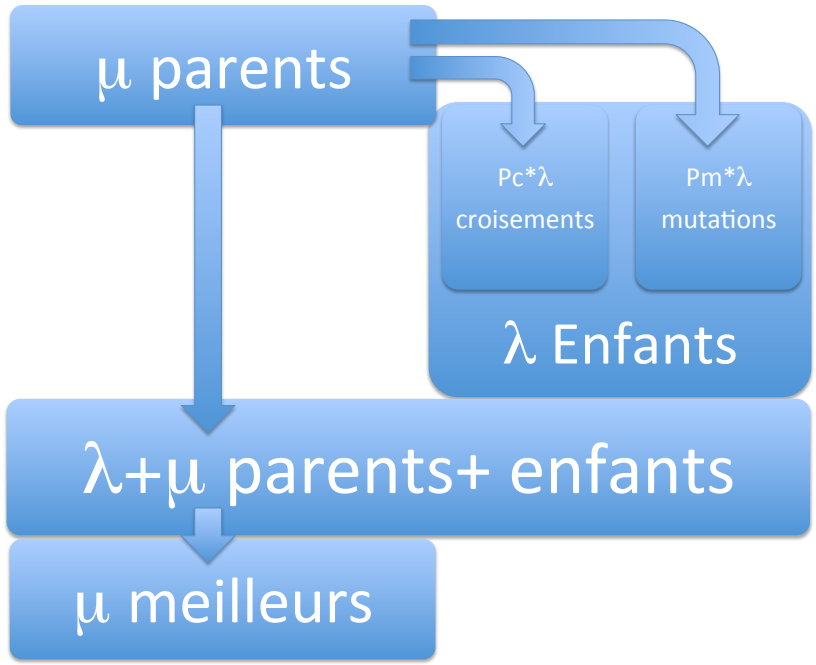
le meilleur individu de la dernière génération !

TAUX DE RENOUVELLEMENT FIXE : (μ, λ)

Conserver la mémoire des bonnes solutions et renouveler la population.



TAUX DE RENOUVELLEMENT VARIABLE : $(\mu + \lambda)$



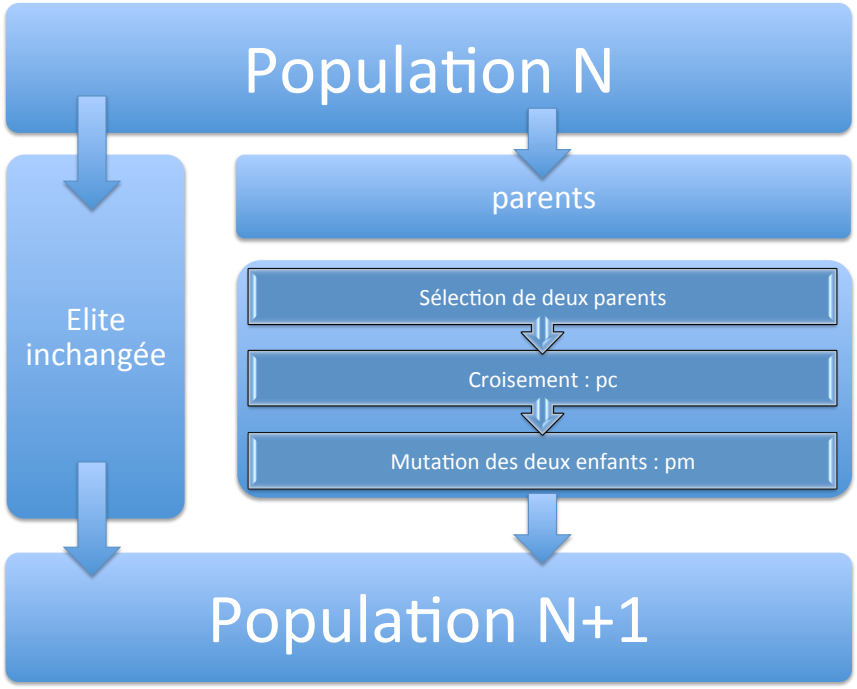
L'ENCHAÎNEMENT DES OPÉRATIONS : en parallèle ou en cascade

Les ES-(μ, λ) et $(\mu + \lambda)$ appliquent les opérateurs en parallèle : $p_c + p_m = 1$

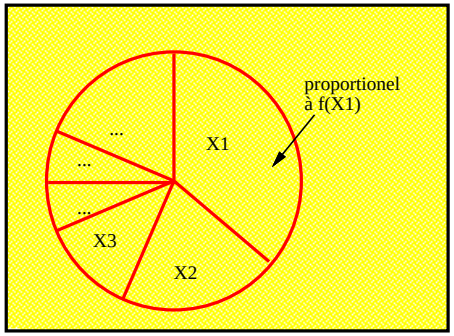
Les AG appliquent les opérateurs en cascade :

$$p_c + p_m < 1$$

- croisés et mutés : $p_m * p_c$
- croisés et non mutés : $(1 - p_m) * p_c$
- mutés et non croisés : $p_m * (1 - p_c)$
- inchangés : $(1 - p_m) * (1 - p_c)$



LE RÔLE DE LA PRESSION SÉLECTIVE



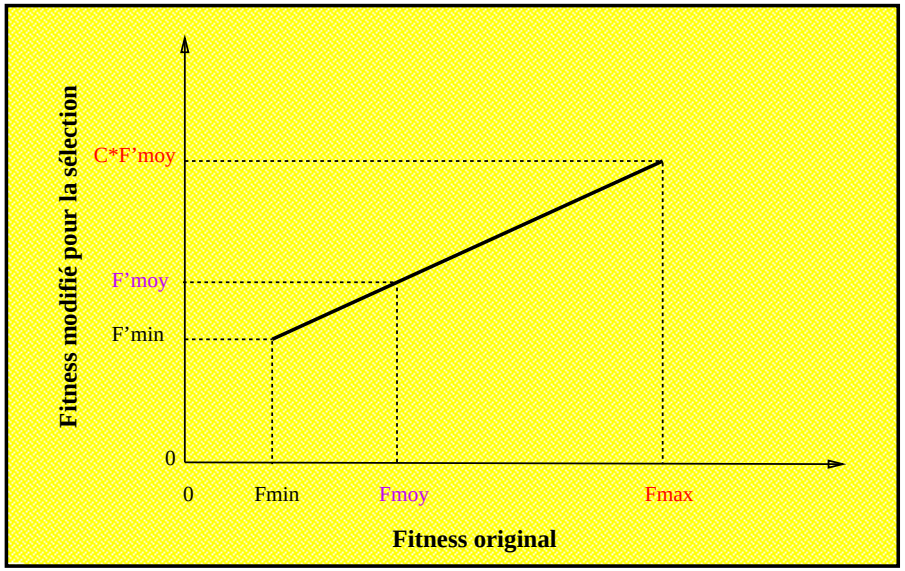
$$P(x) = \frac{f(x)}{\sum_{x \in Population} f(x)}$$

Ni trop, ni trop peu !

$\longrightarrow$ Scaling

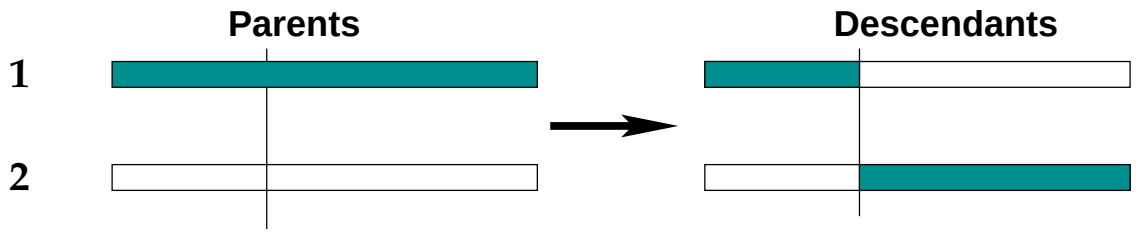
$\longrightarrow$ Ranking

$\longrightarrow$ Tournoi

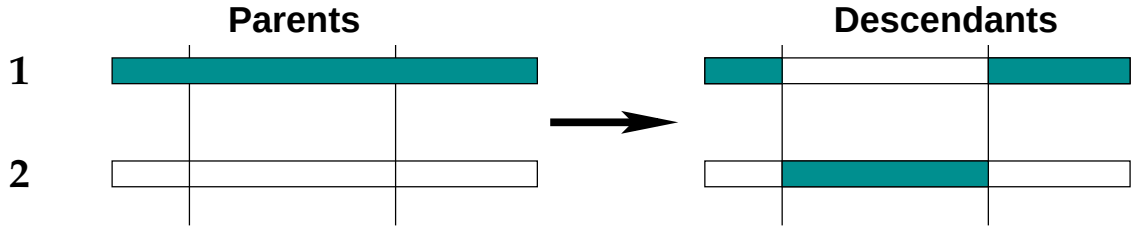


Chaque individu est représenté par une chaîne (binaire) de longueur fixe.

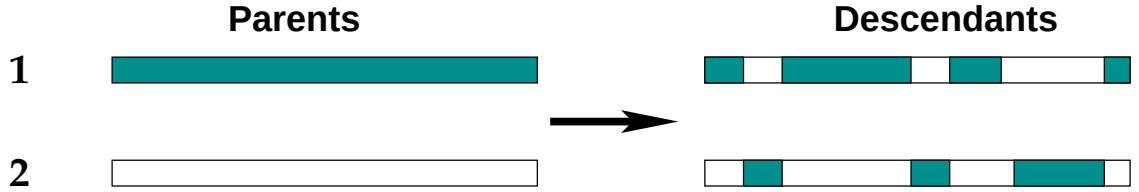
Croisement a 1 point



Croisement à 2 points

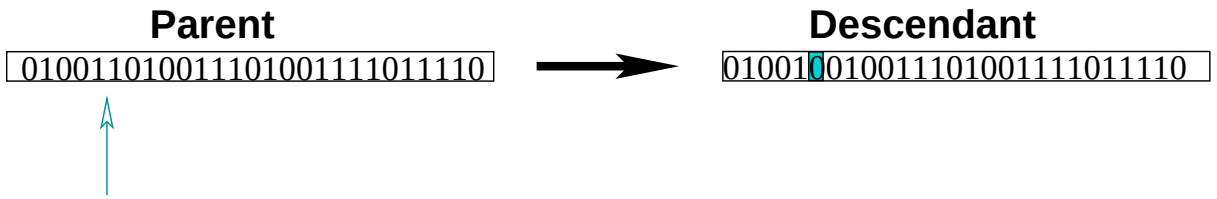


Croisement uniforme



Mutation discrète

Une petite perturbation du génôme :



p_m est usuellement très faible et fixée.

Convergence prouvée si $p_m(k)$ décroît à chaque génération k en respectant :

$$p_m(k) \geq \frac{1}{2} * k^{-\frac{1}{M*L}}$$

M est la taille de la population et L est la longueur des chromosomes.

EXEMPLE : ONEMAX

– **Chomosomes binaires**, de longueur $lmax$ 0101110010100111

– **fitness** = nombre de “1” dans la chaîne

optimum : 1111111111111111

– **population initiale** : P chromosomes binaires aléatoires.

Run d'un AG canonique sur Onemax 32

Gen : 0 Max : 21.000000 Min : 11.000000 Moy : 15.760000 Cmax :2.0
Gen : 1 Max : 23.000000 Min : 11.000000 Moy : 17.180000 Cmax :1.9
Gen : 2 Max : 22.000000 Min : 13.000000 Moy : 18.340000 Cmax :1.6
Gen : 3 Max : 24.000000 Min : 15.000000 Moy : 19.460000 Cmax :2.0
Gen : 4 Max : 25.000000 Min : 16.000000 Moy : 20.380000 Cmax :2.0
Gen : 5 Max : 26.000000 Min : 17.000000 Moy : 21.320000 Cmax :2.0
Gen : 6 Max : 27.000000 Min : 20.000000 Moy : 23.300000 Cmax :2.0
Gen : 7 Max : 28.000000 Min : 22.000000 Moy : 24.220000 Cmax :2.0
Gen : 8 Max : 28.000000 Min : 21.000000 Moy : 24.820000 Cmax :1.8
Gen : 9 Max : 29.000000 Min : 23.000000 Moy : 25.720000 Cmax :2.0
Gen :10 Max : 30.000000 Min : 22.000000 Moy : 26.260000 Cmax :1.8
Gen :11 Max : 30.000000 Min : 23.000000 Moy : 27.000000 Cmax :1.7
Gen :12 Max : 32.000000 Min : 23.000000 Moy : 27.560000 Cmax :1.9
Gen :13 Max : 32.000000 Min : 24.000000 Moy : 27.840000 Cmax :2.0
Gen :14 Max : 32.000000 Min : 25.000000 Moy : 28.260000 Cmax :2.0
Gen :15 Max : 32.000000 Min : 25.000000 Moy : 28.840000 Cmax :1.8
Gen :16 Max : 32.000000 Min : 25.000000 Moy : 29.540000 Cmax :1.5
Gen :17 Max : 32.000000 Min : 27.000000 Moy : 30.040000 Cmax :1.6
Gen :18 Max : 32.000000 Min : 27.000000 Moy : 30.480000 Cmax :1.4
Gen :19 Max : 32.000000 Min : 29.000000 Moy : 30.700000 Cmax :1.7
Gen :20 Max : 32.000000 Min : 30.000000 Moy : 30.980000 Cmax :2.0
Gen :21 Max : 32.000000 Min : 30.000000 Moy : 31.680000 Cmax :1.1
Gen :22 Max : 32.000000 Min : 31.000000 Moy : 31.920000 Cmax :1.0
Gen :23 Max : 32.000000 Min : 32.000000 Moy : 32.000000 Cmax :1.0
Gen :24 Max : 32.000000 Min : 32.000000 Moy : 32.000000 Cmax :1.0
Gen :25 Max : 32.000000 Min : 32.000000 Moy : 32.000000 Cmax :1.0
Gen :26 Max : 32.000000 Min : 32.000000 Moy : 32.000000 Cmax :1.0

LA REPRÉSENTATION CONTINUE : STRATÉGIES D'ÉVOLUTION

La recherche se fait dans $\mathbb{R}^n$

Croisement barycentrique : $\forall i \in 1,..,n, \quad x'_i = \alpha x_i + (1-\alpha) y_i$

α, choisi par tirage unifome dans $[0,1]$ ou $[-\epsilon, 1+\epsilon]$

Mutation Gaussienne : $\forall i \in 1,...,n, \quad x'_i = x_i + N(0,\sigma)$

deux paramètres p_m et σ.

Mutation Log-normale auto-adaptative : **σ est intégré au code génétique : (x,σ)**

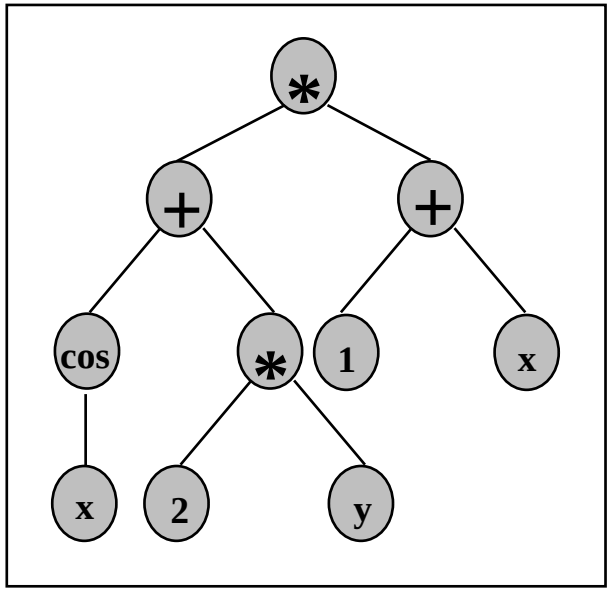
$$\forall i \in 1,...,n, \quad \sigma'_i = \sigma_i exp(N(0,\tau))$$
$$x'_i = x_i + N(0,\sigma'_i)$$

LA REPRÉSENTATION FONCTIONNELLE : PROGRAMMATION GÉNÉTIQUE

Créer des programmes sans programmer ! !

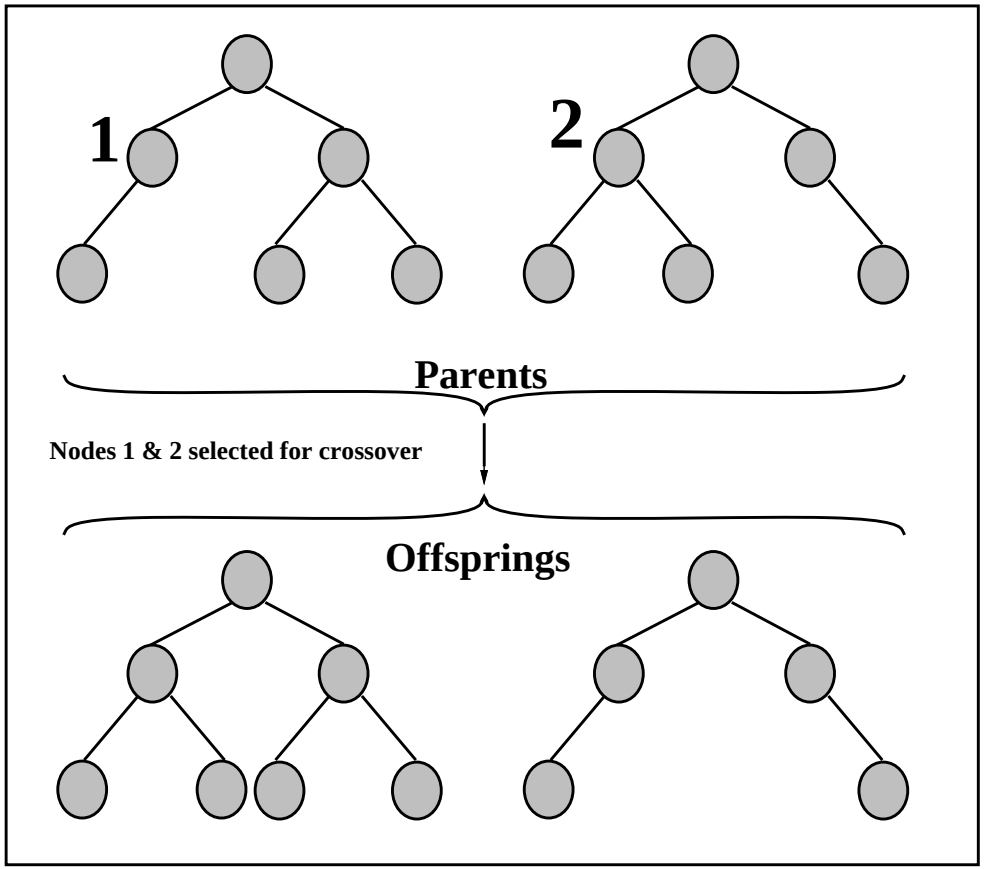
John Koza

– Stuctures de longueurs variables.
– Représentation par arbre.

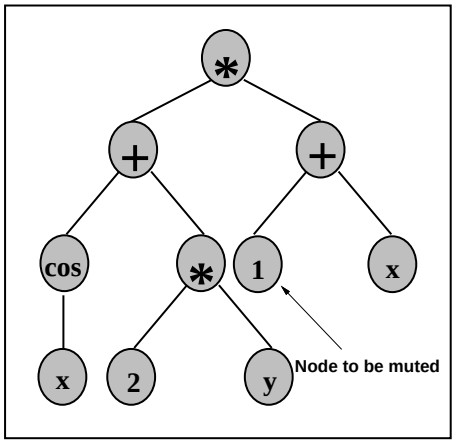


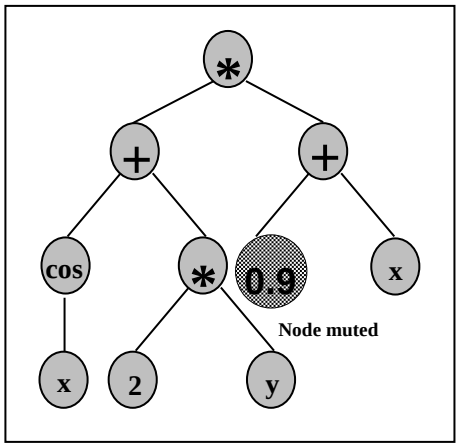
La fonction $((\cos(x) + 2 * y) * (1 + x))$.

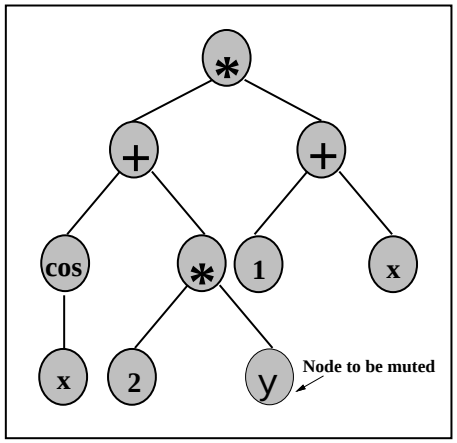
Croisement PG

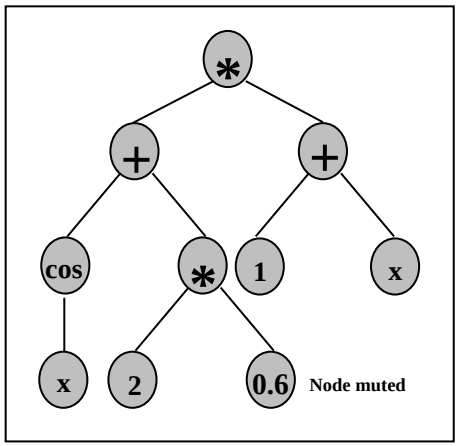


Mutations PG









RÉSUMÉ

- Les AE fonctionnent sur une population de solutions (phénotype / génotype).

- La fonction à optimiser est utilisée pour donner un classement (sélection) : pas de continuité ni de dérivabilité.

- La convergence se traduit par une concentration de la population autour de l'optimum global.

- La vitesse de convergence et la qualité de la solution finale dépendent :
  de la représentation (codage) des individus,
  de la forme de la fonction à optimiser,
  des paramètres de l'AE.

- Les temps de calculs en version séquencielle peuvent être longs.

Who's who

ALGORITHMES GÉNÉTIQUES

(David Goldberg)

- chromosomes = chaînes de caractères,
- croisement à 1 site, à plusieurs sites, uniformes,
- stratégies de remplacement élitistes.

STRATÉGIES D'ÉVOLUTION

(Hans-Paul Schwefel)

- chromosomes $\in \mathbf{R}^n$,
- mutation Gaussienne, croisements barycentriques,
- stratégies de remplacement (μ, λ), $(\mu + \lambda)$.

EVOLUTIONARY COMPUTATION

(Lawrence et David Fogel)

Who's who

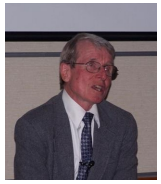
GENETIC PROGRAMING

(John Koza)

– chromosomes de taille variable (arbres).

CLASSIFIER SYSTEMS

(John Holland, David Goldberg)

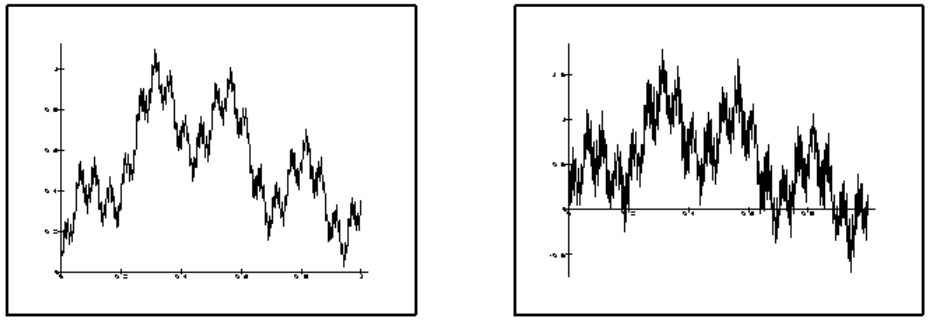


– chromosomes = règles d'un système expert.

Quand utiliser un AE ?

– **cas linéaire, convexe : trop lent !**

– **cas plus complexe : exemple des fonctions fractales**



Fonctions de Weierstrass de dimension 1.5 et 1.7

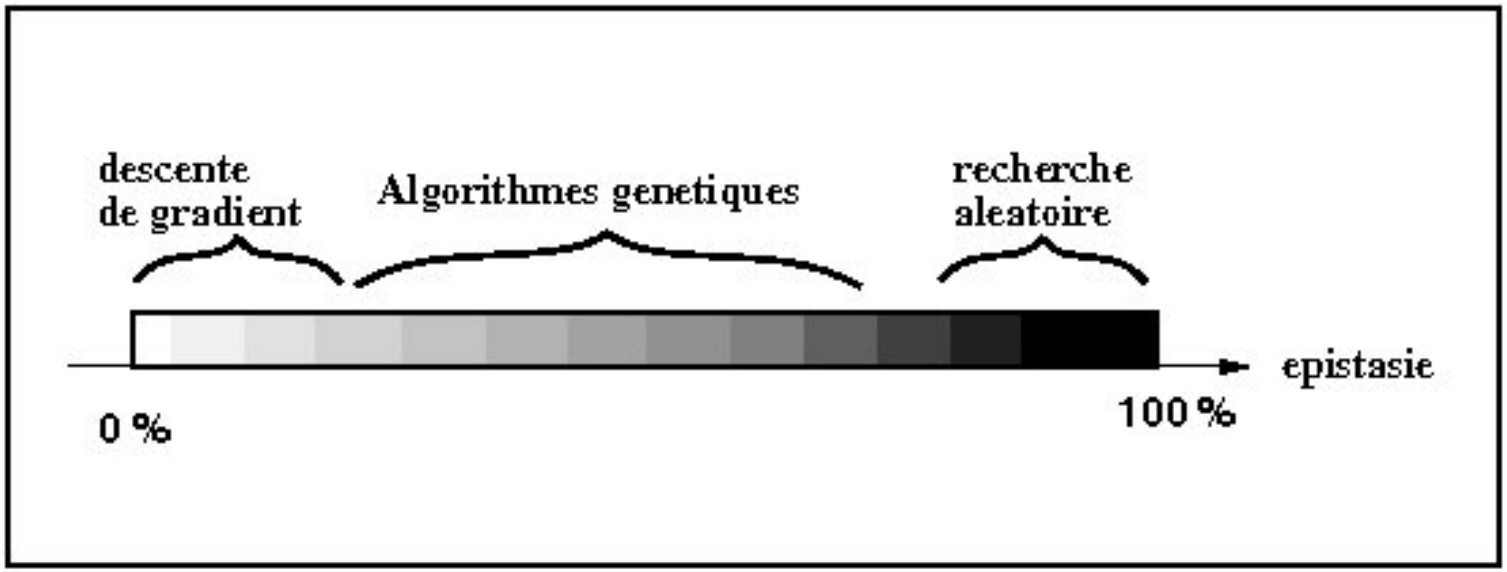
– **cas pseudo-aléatoire : exemple d'une fonction en cryptographie**

$D_p\{x^4 \ mod \ p, a\}$ **avec** a **et** p **des entiers codés sur 500 bits,** $a < p$

$$D_p\{x, y\} = Min\{(x - y) \ mod \ p, (y - x) \ mod \ p\}$$

Recommandations pratiques

Recherche d'un optimum global dans un environnement complexe et multidimensionnel

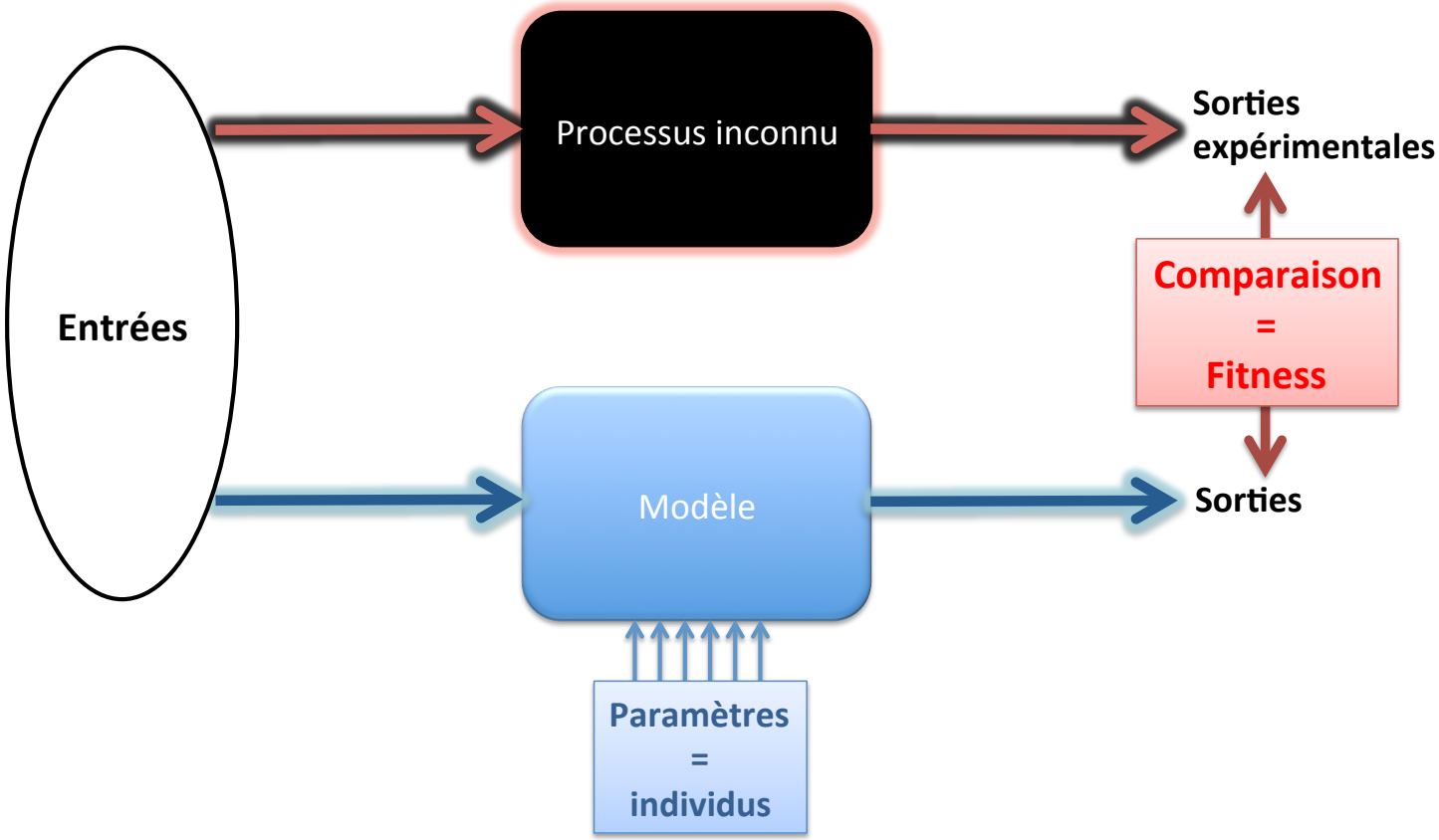


Adaptation à un environnement variable $\longrightarrow$ IA, contrôle, commande de processus.

DES DOMAINES D'APPLICATION EXTRÊMEMENT VARIÉS

- *théorie des graphes*
- *théorie des jeux*
- *robotique*
- *modélisation de données 3D*
- *traitement d'images*
- *synthèse d'images*
- *traitement de la parole*
- *applications militaires*
- *économie*
- *fiabilité*
- *test de logiciel*
- *linguistique*
- *science des matériaux, mécanique*
- *géophysique*
- *thermodynamique*
- *écologie*
- *génétique*
- *médecine*

Résolution de problèmes inverses par approche en “boîte noire”



Construire des applications

– Identifier l'espace de recherche et les contraintes du problème.

– Choisir un codage efficace (gérer la redondance).

– Construire une fonction de fitness économique.

– Bien régler les opérateurs, stratégies, paramètres.

Comment tester son algorithme ?

Identifier tous les paramètres de contrôle

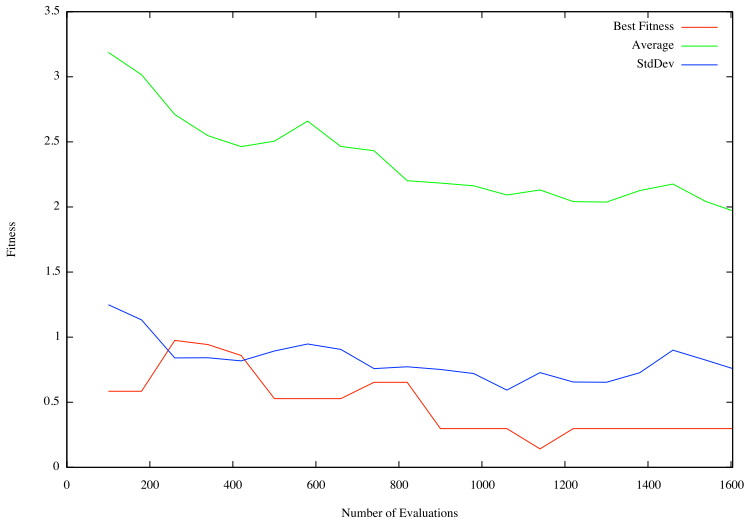
– taille de population, λ, μ,
– probabilités d'application des opérateurs,
– rayons de mutation σ,
– nombre de générations.

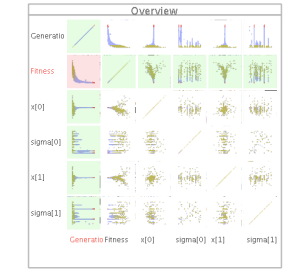
Faire un plan de tests

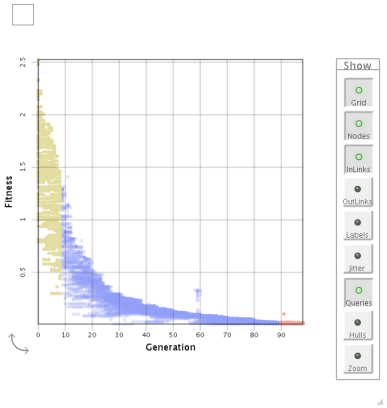
– échantillonner chaque variable,
– ne changer qu'un paramètre à la fois,
– tester toutes les combinaisons,
– faire des répétitions : un AE est stochastique ! !

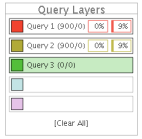
Analyser le comportement de l'algorithme

– valeurs moyennes et variances du meilleur fitness,
– surveiller la perte de diversité ! ! !

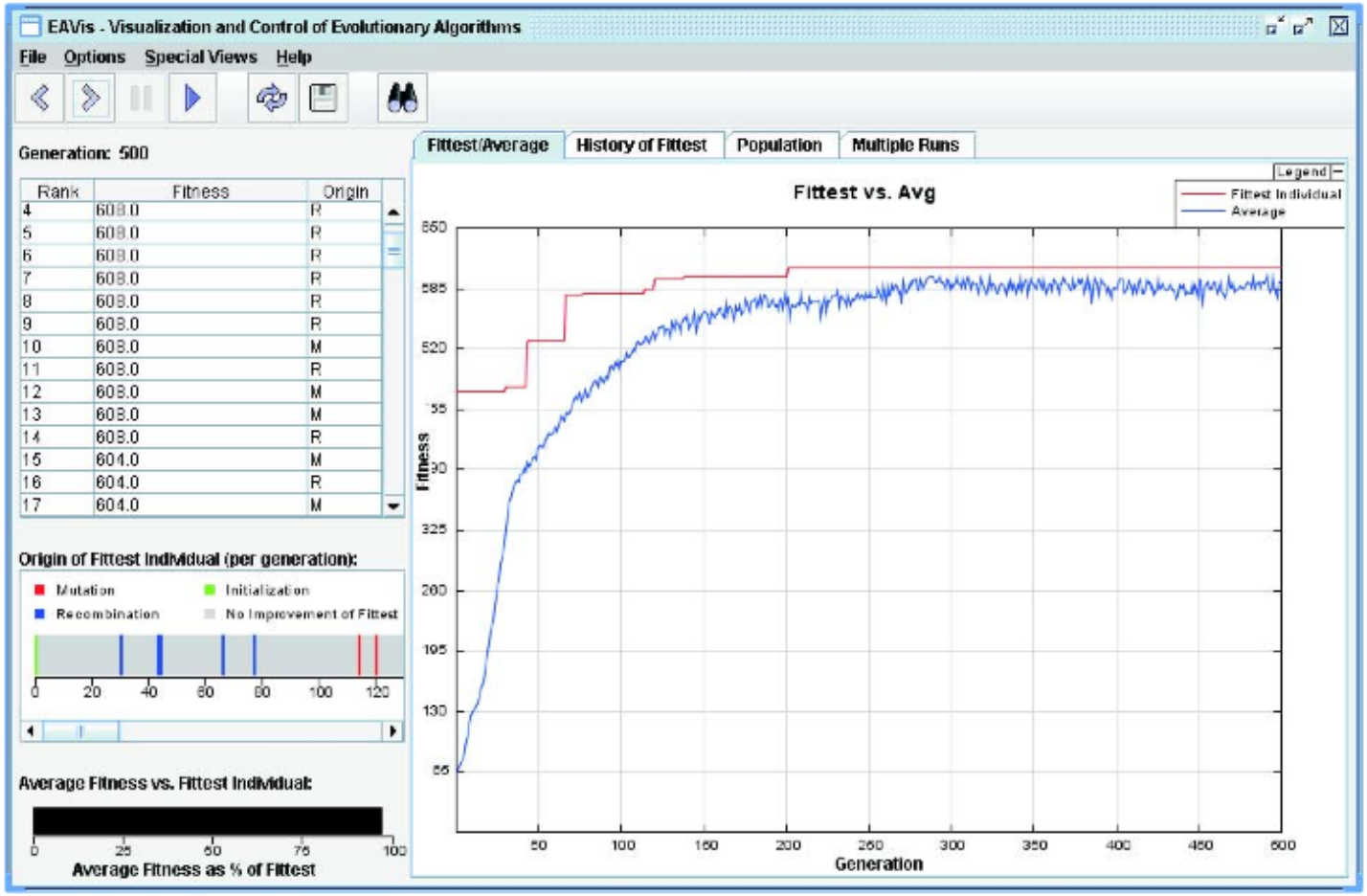








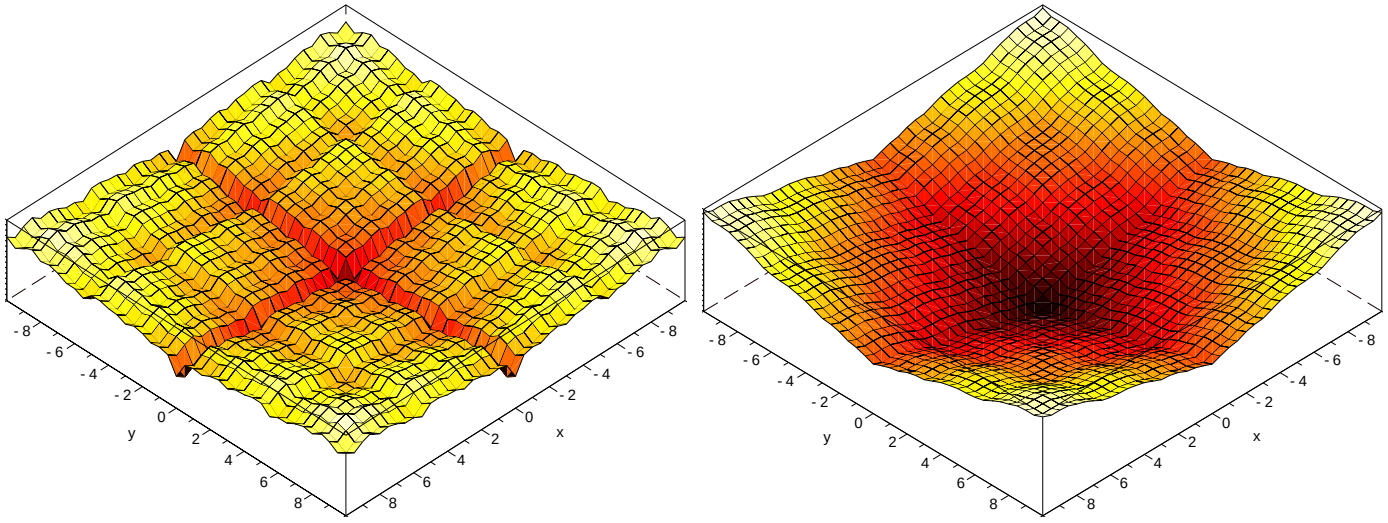
Visualiser le comportement de l'algorithme



Exemple de fonctions-test

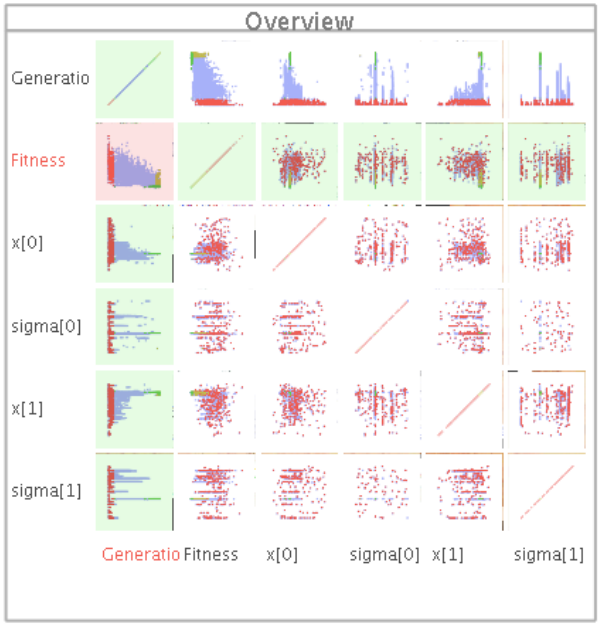
Les fonctions de Weierstrass en 2D :

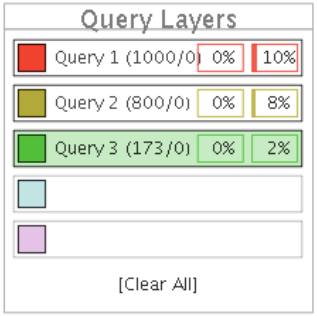
$$f(x,y) = \sum_{n=-\infty}^{+\infty} 2^{-nH}(1-\cos 2^n x) + \sum_{n=-\infty}^{+\infty} 2^{-nH}(1-\cos 2^n y)$$

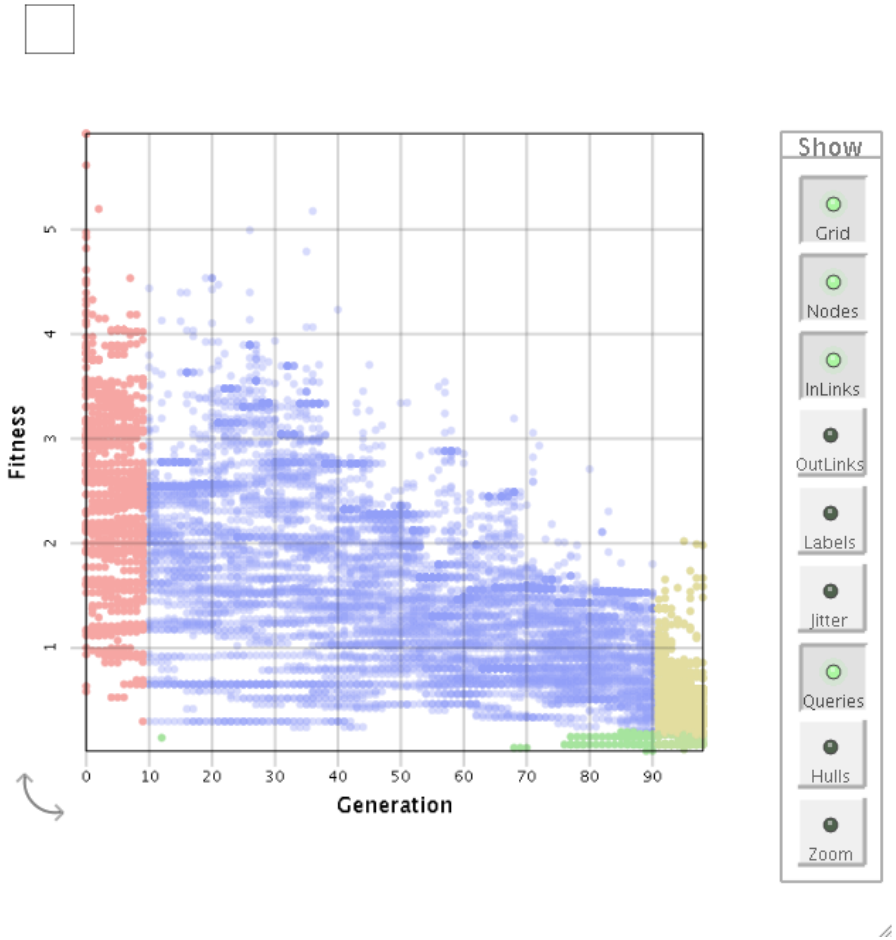


Exposants de Hölder $H = 0.2$ (très irrégulier) **et $H = 0.9$ (plus régulier).**

Fonction de Weierstrass 2D d'exposant de Hölder 0.2

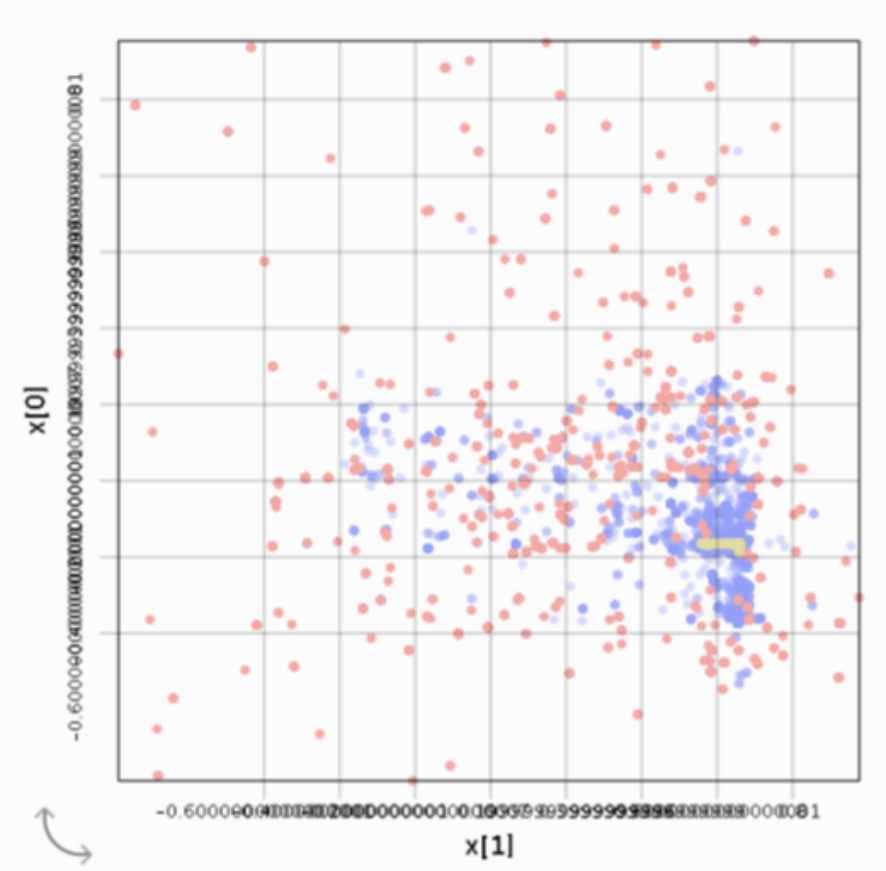


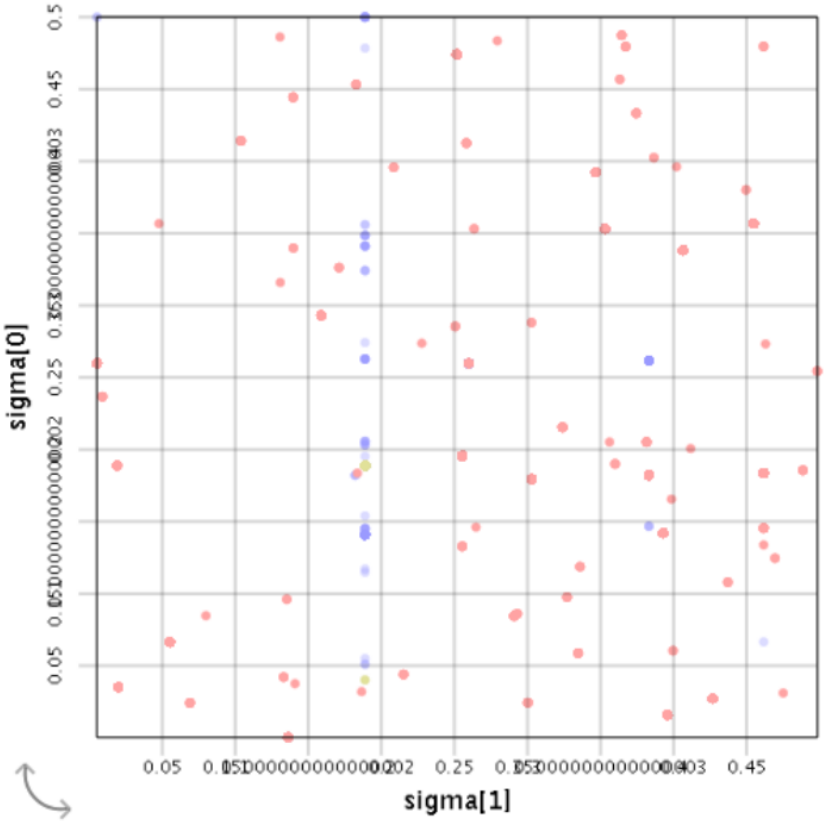




Les 10 premières générations, les 10 dernières, zone de meilleur fitness.

Fonction de Weierstrass 2D d'exposant de Hölder $H = 0.2$

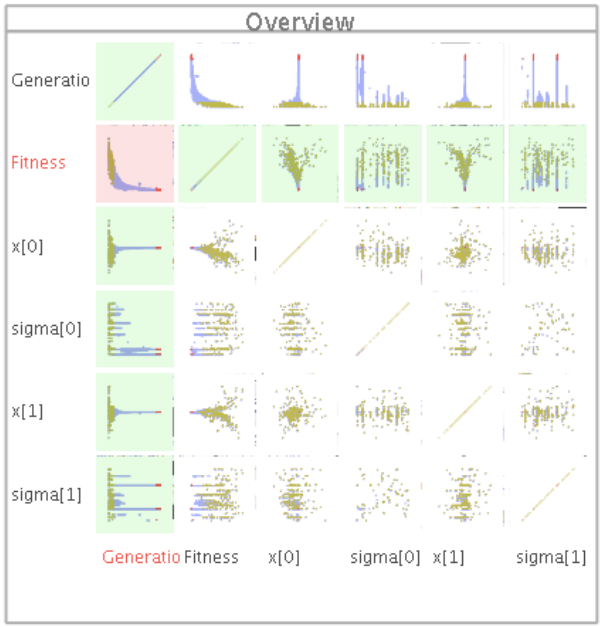


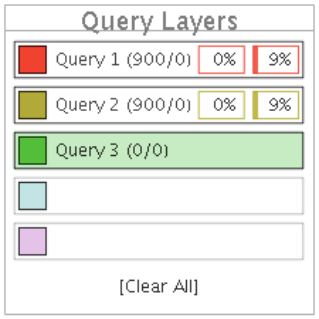


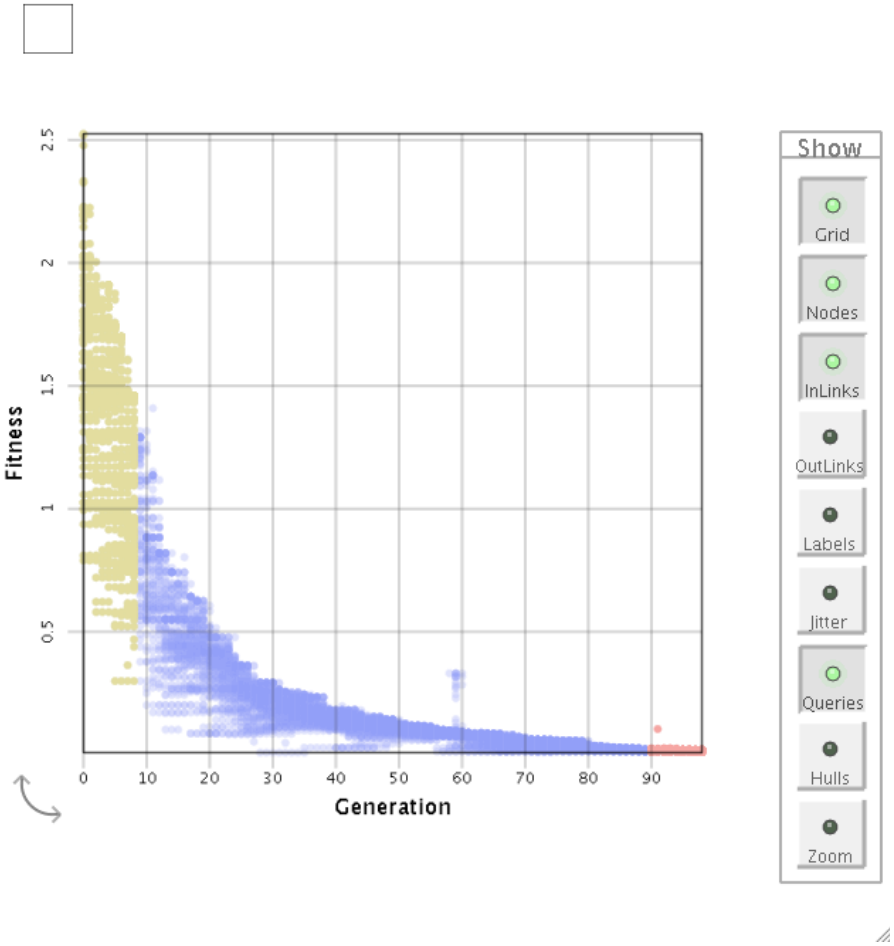
Generations 0 à 10,

generations 90 à 100.

2D Weierstrass function of Hölder exponent 0.9







Premières générations, dernières générations.

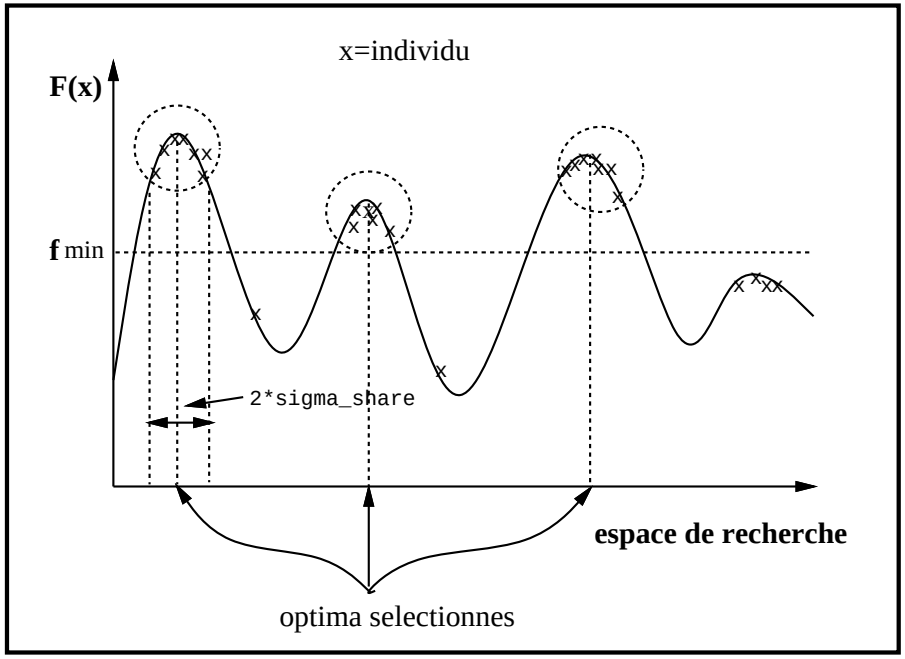
ALGORITHMES ÉVOLUTIONNAIRES "NON STANDARDS"

Niches écologiques : **détection de plusieurs optima.**

Optimisation multi-objectif : **échantillonnage du front de Pareto.**

Coévolution.

Pour “coloniser” plusieurs optima



– par maintien de la diversité génétique
(Caviccio, De Jong, Mauldin),

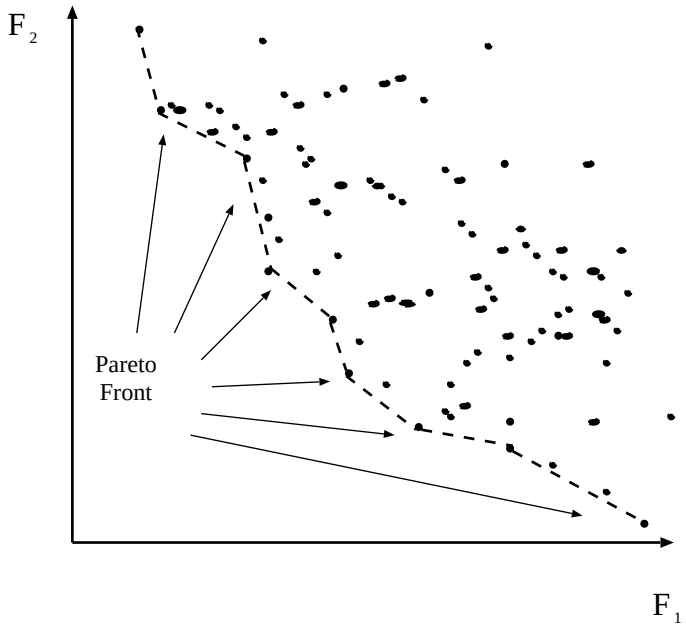
– par partage des ressources
(Goldberg, Richardson),

$$\text{Fitness'}(x) = \frac{\text{Fitness}(x)}{\sum_{x' \in Vois(x)} Sh(d(x,x'))}$$

$$Sh(d) = \begin{cases} 1 - (\frac{d}{\sigma_{share}})^{\alpha} & \text{si } d < \sigma_{share} \\ 0 & \text{sinon} \end{cases}$$

– par restriction des croisements.

– Front de Pareto ≡ Meilleur compromis
– sélection spécifique + nichage
– En une fois, un EA produit un échantillonnage du front de Pareto.



Front de Pareto = ensemble des solutions non dominées

ALGORITHMES ÉVOLUTIONNAIRES PARISIENS

Principe : La solution est représentée par plusieurs individus dans la population.

⟶ Ce n'est plus exactement une optimisation.

:-(Plus délicat a mettre en œuvre.

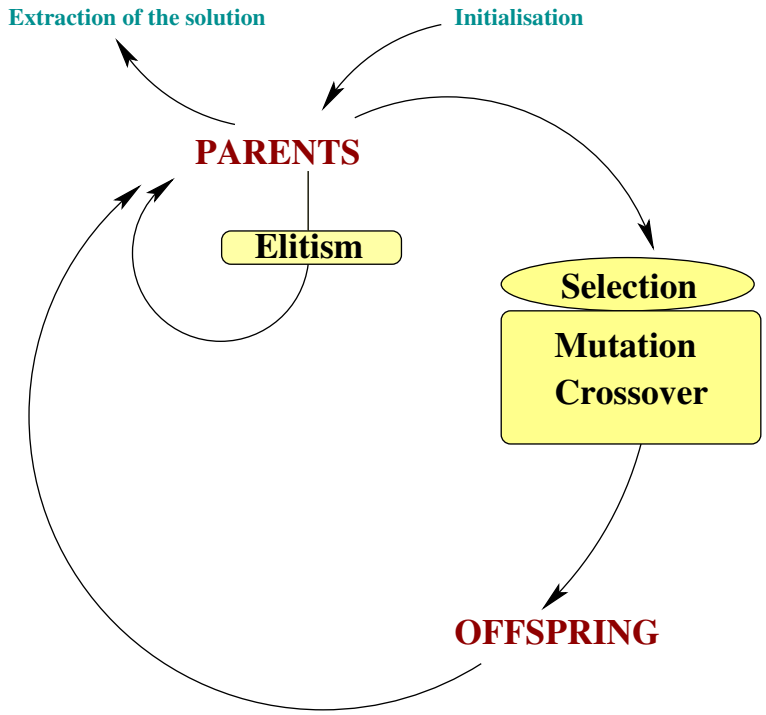
:-) Gain de temps d'exécution important.

:-) Meilleure utilisation des informations a priori.

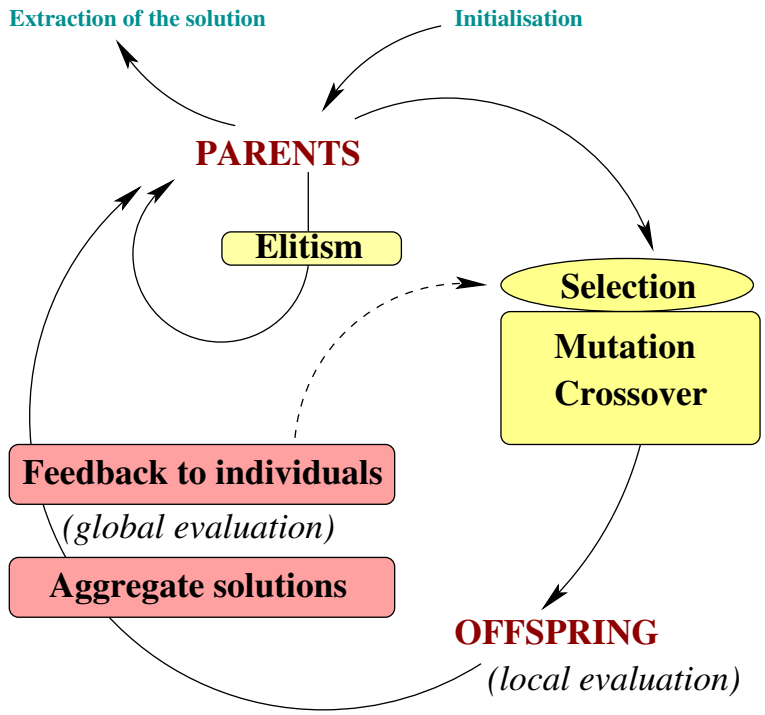
Utilisation :

- Résolution du problème inverse pour les IFS.
- Art et design.
- Text retrieval.
- Vision stéréo.
- Tomographie 3D.
- Modélisation de processus de maturation de fromage.

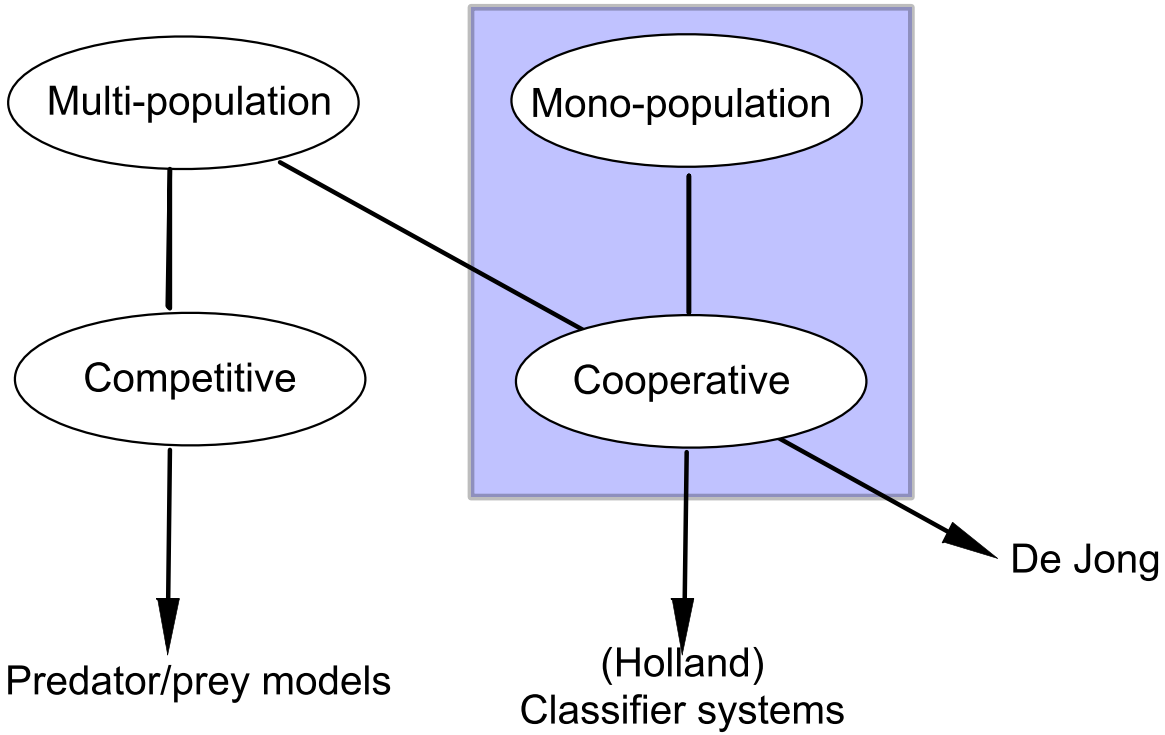
Approche évolutionnaire Parisienne : coopération-coévolution



Classique



Parisien

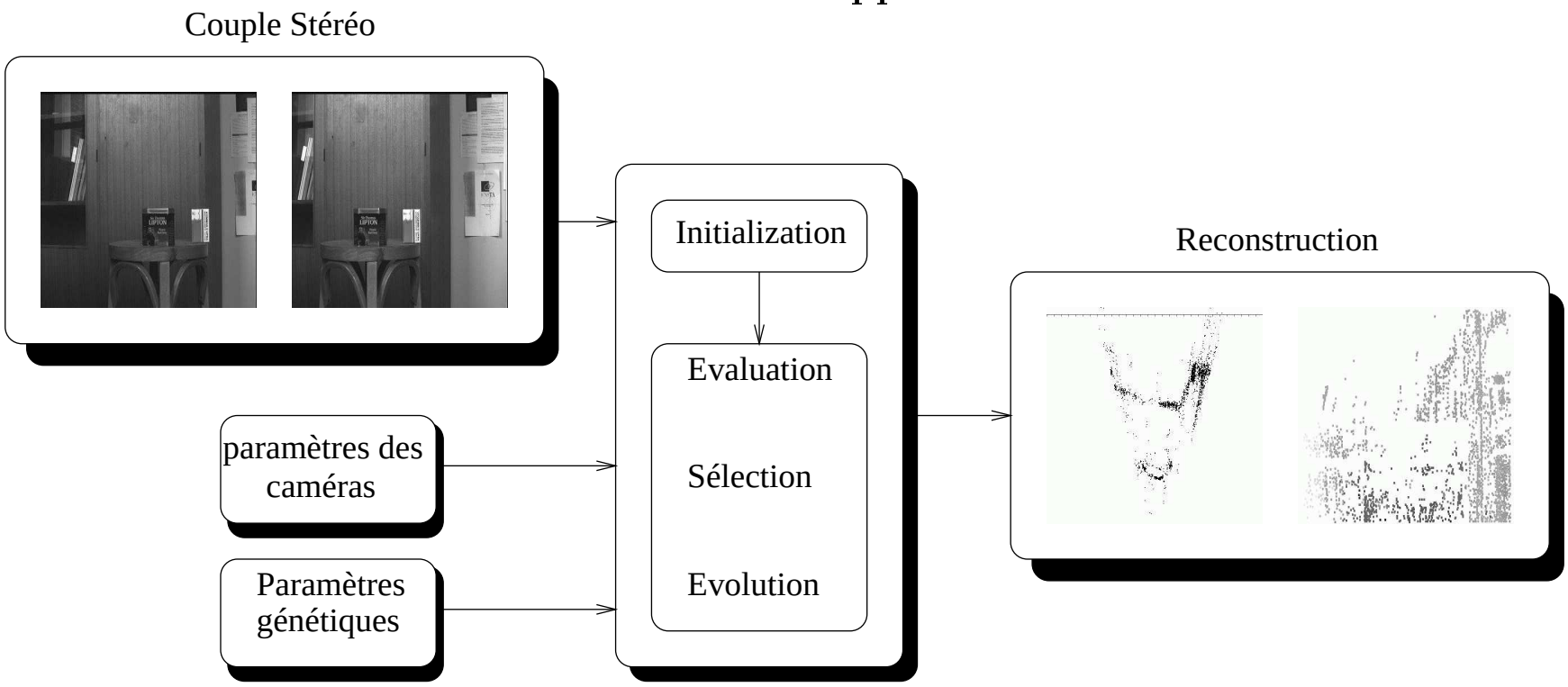


Parisian approach

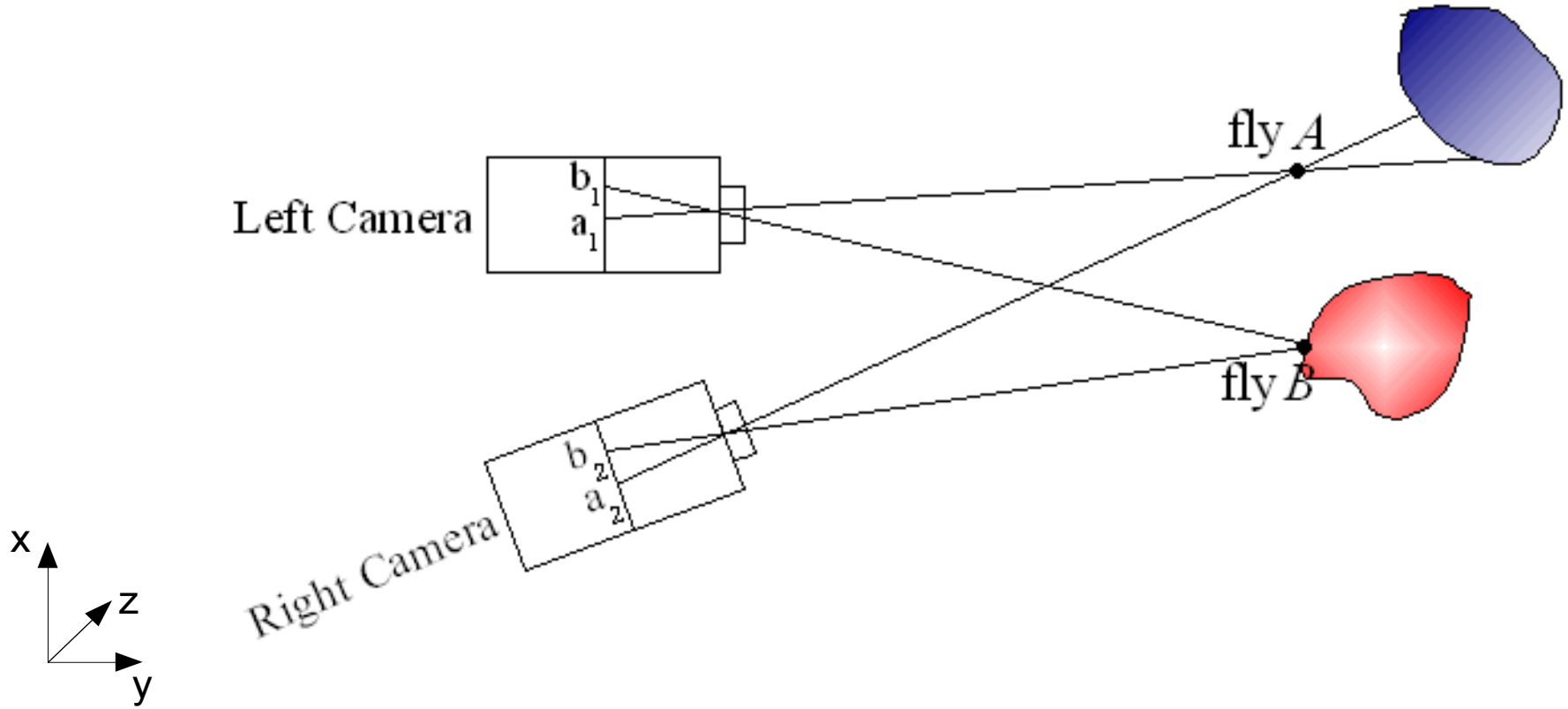
ALGORITHME DES MOUCHES

Vision stéréo pour la robotique :

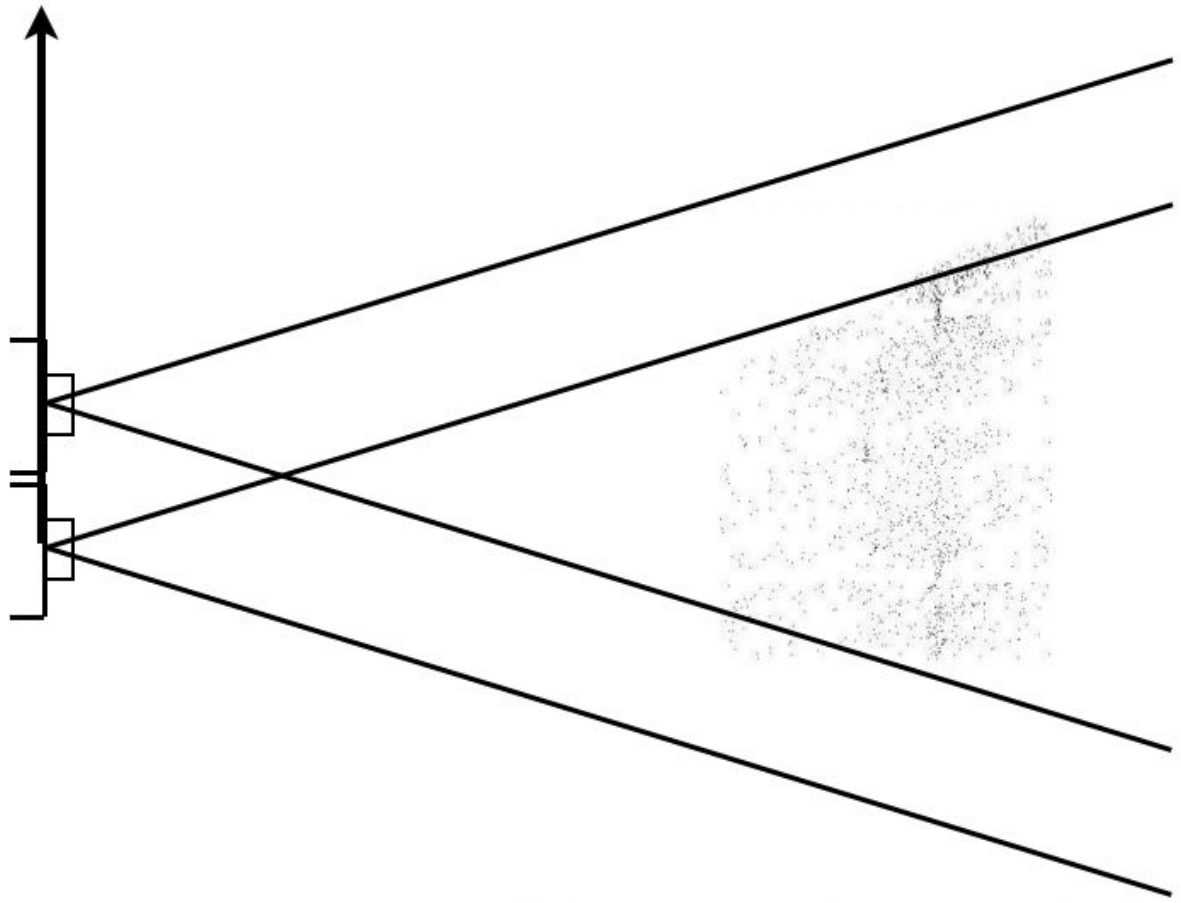
- **Vers un traitement en temps réel des séquences stéréo.**
- **Approche Parisienne.**



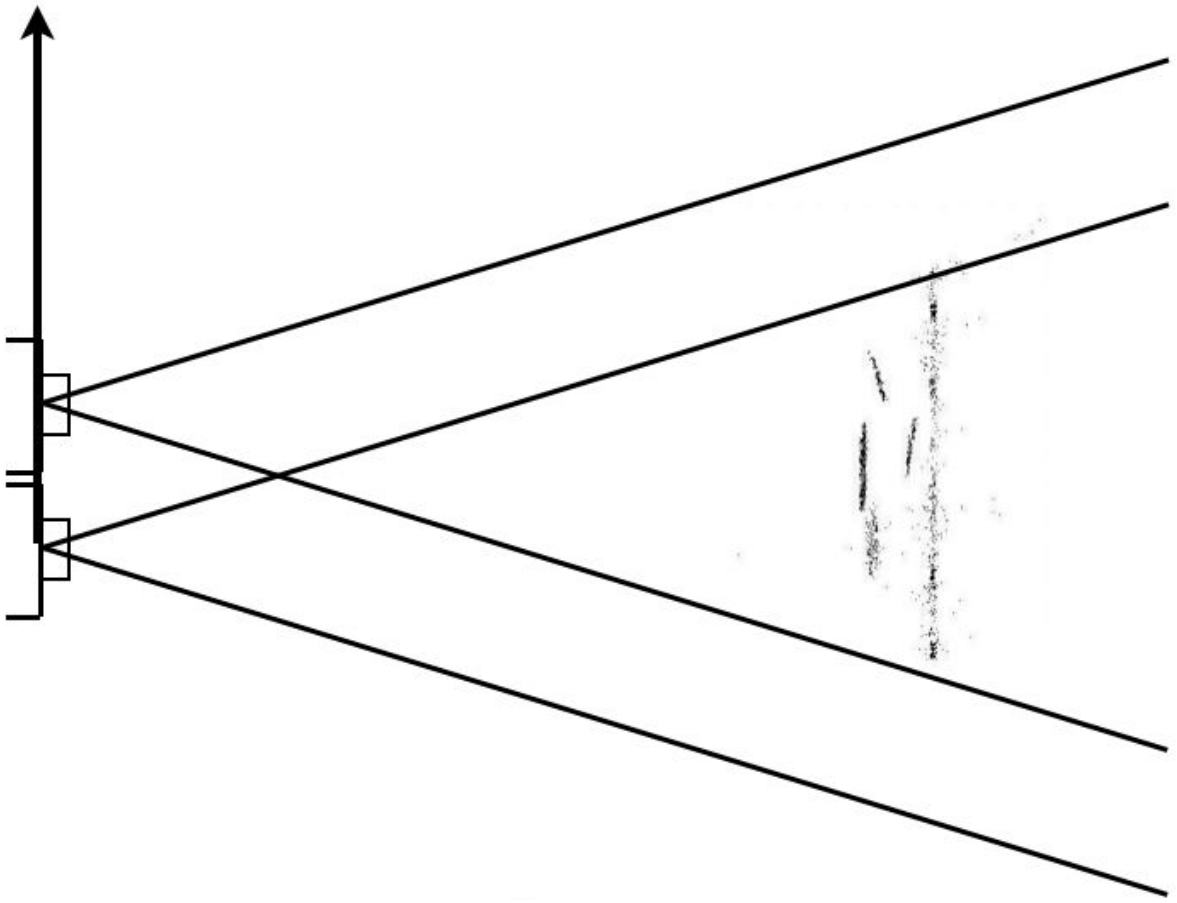
ALGORITHME DES MOUCHES



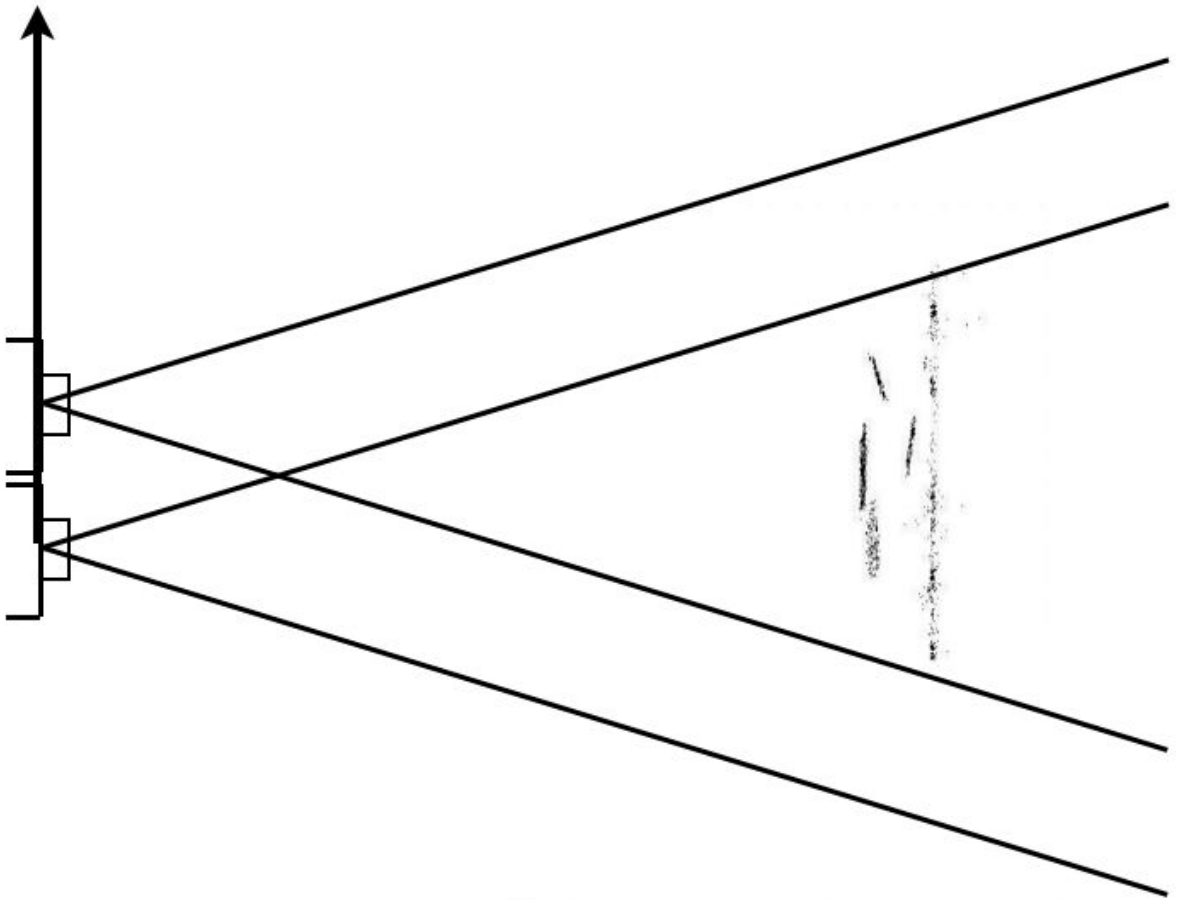
The Flies algorithm



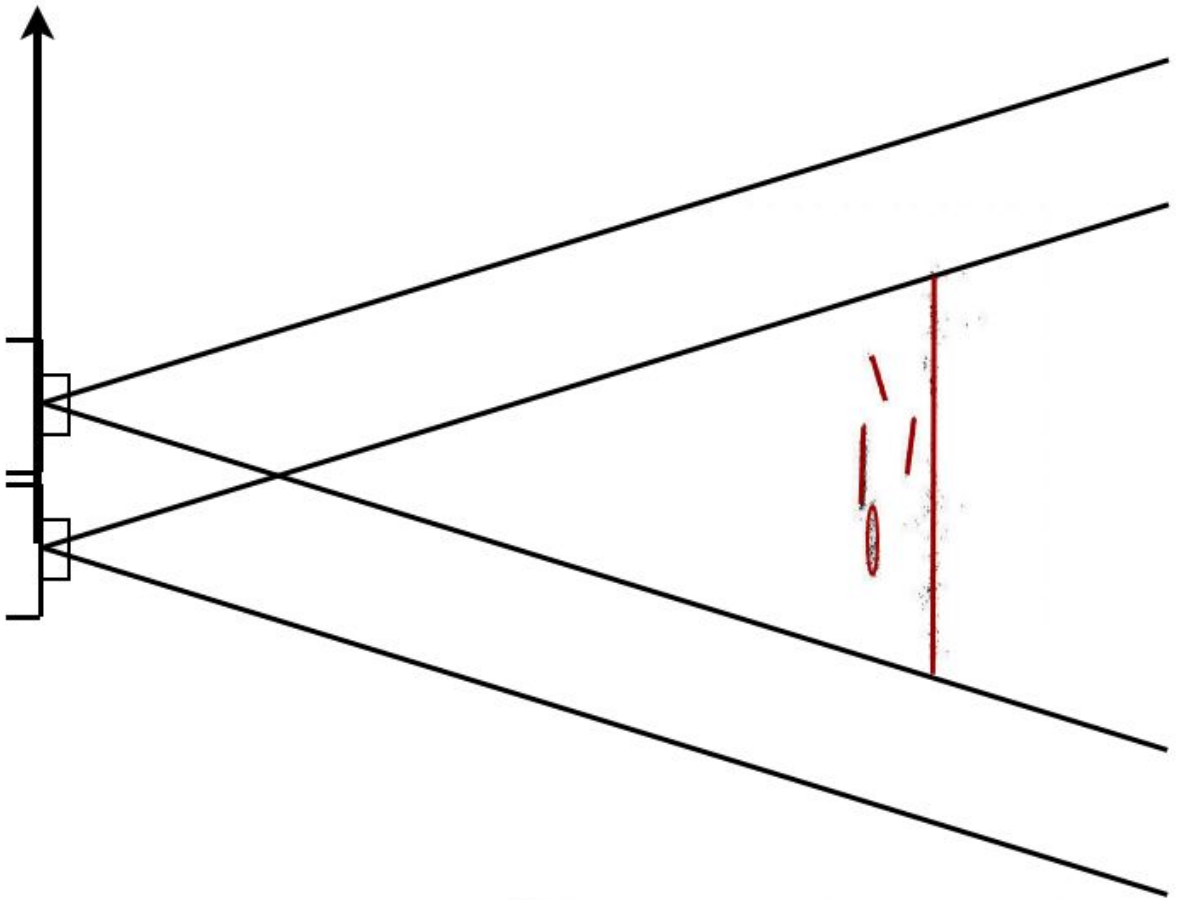
The Flies algorithm



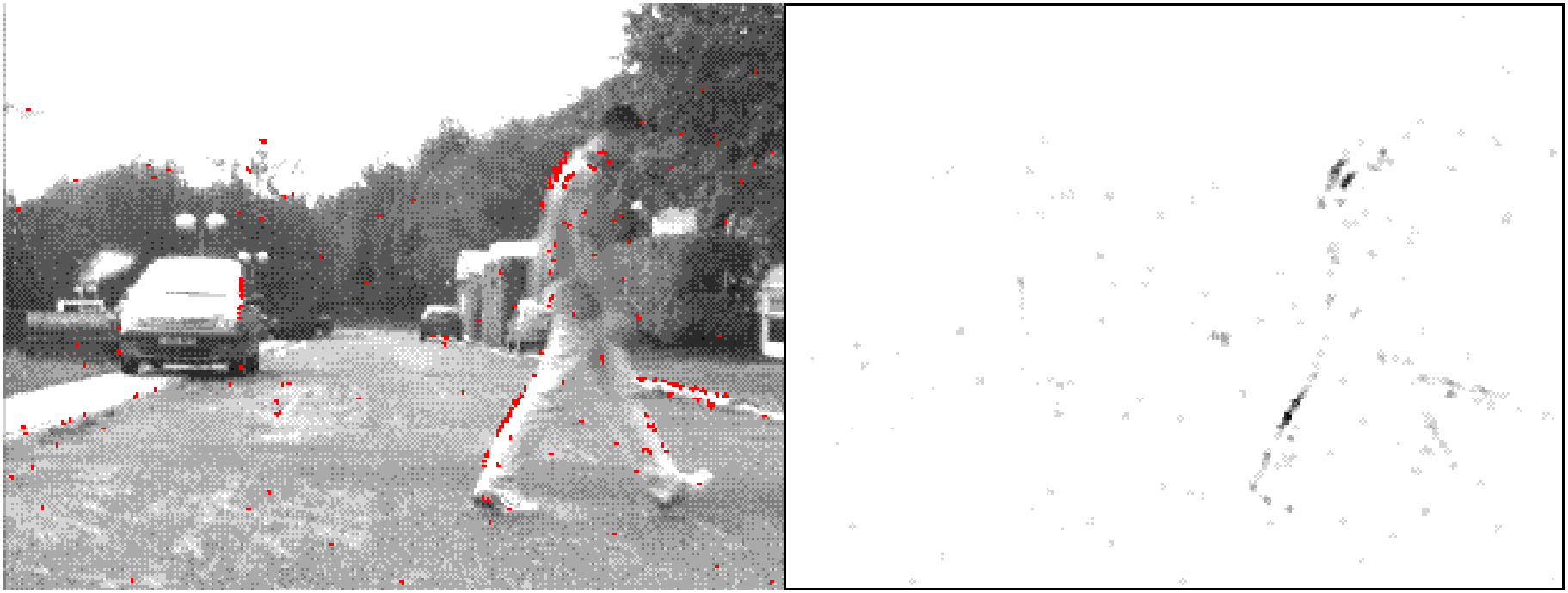
The Flies algorithm



The Flies algorithm



Application pour la détection d'obstacles



ALGORITHMES ÉVOLUTIONNAIRES INTERACTIFS

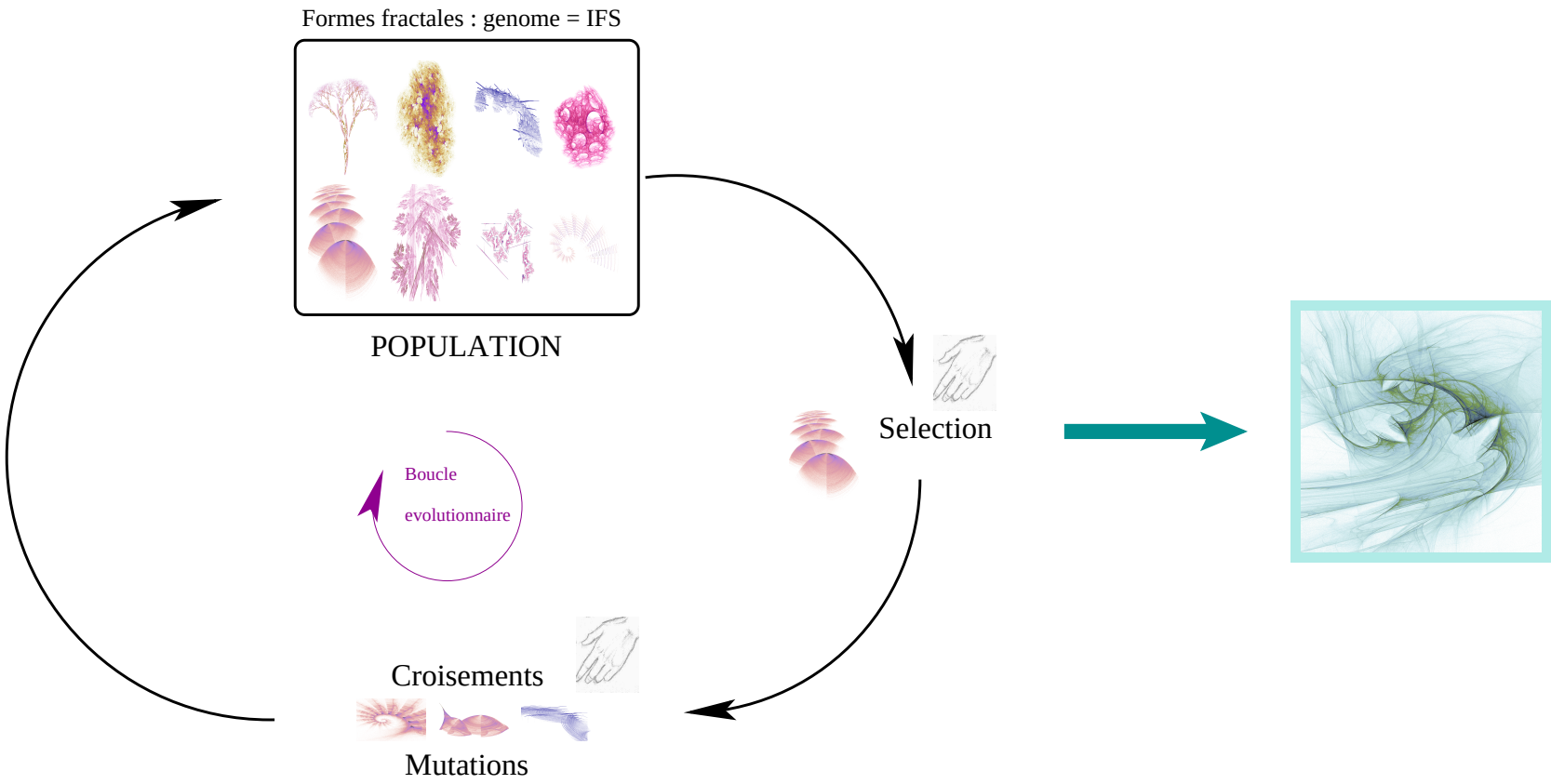
Principe : l'humain dans la boucle évolutionnaire.

- On interroge l'utilisateur régulièrement (fitness interne/fitness externe).
- Il peut faire des "optimisations locales", contrôler le processus à plusieurs niveaux.
- Permet d'optimiser des paramètres "subjectifs" difficiles à modéliser autrement.

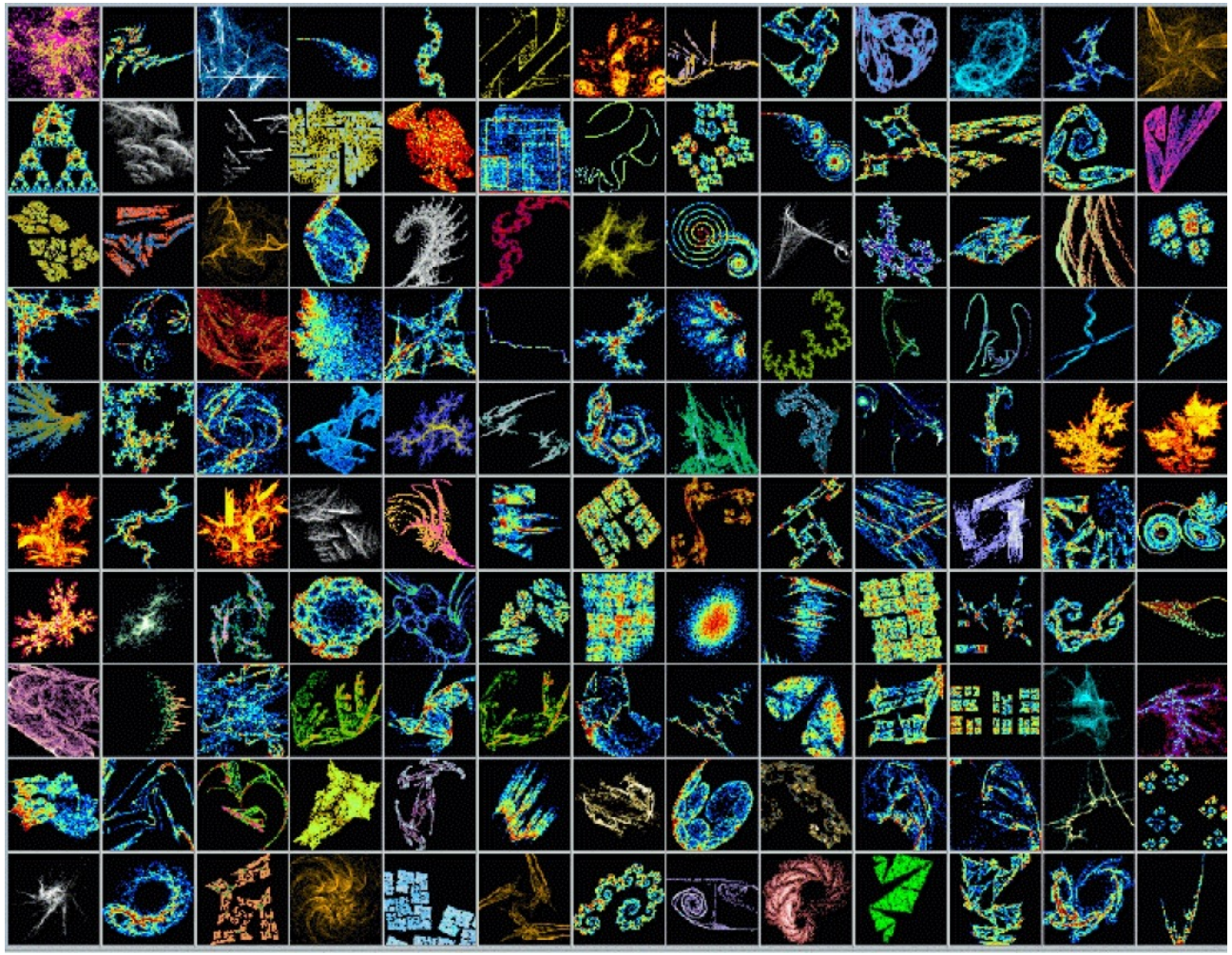
Usage :

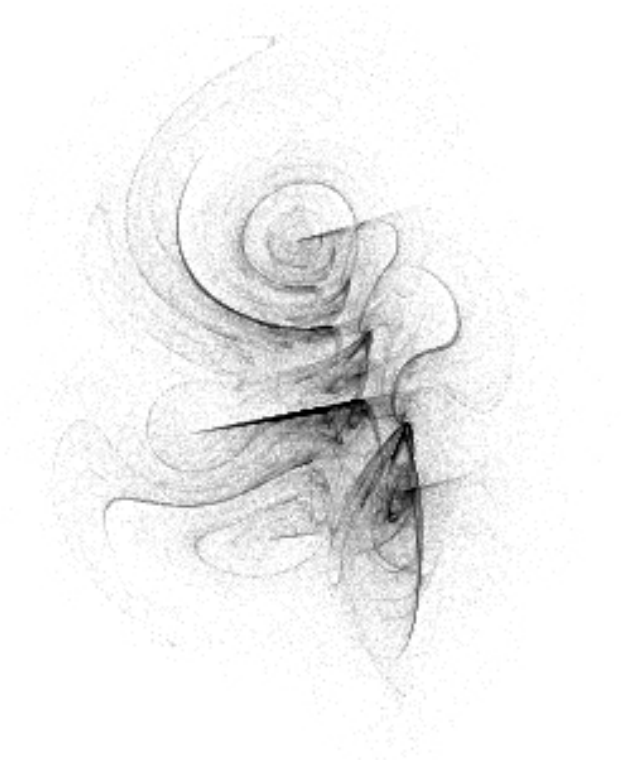
- art et design (ArtiE-Fract et l'atelier des fractales),
- text retrieval (Novartis Pharma),
- e-learning (Paraschool).

L'évolution interactive de formes fractales

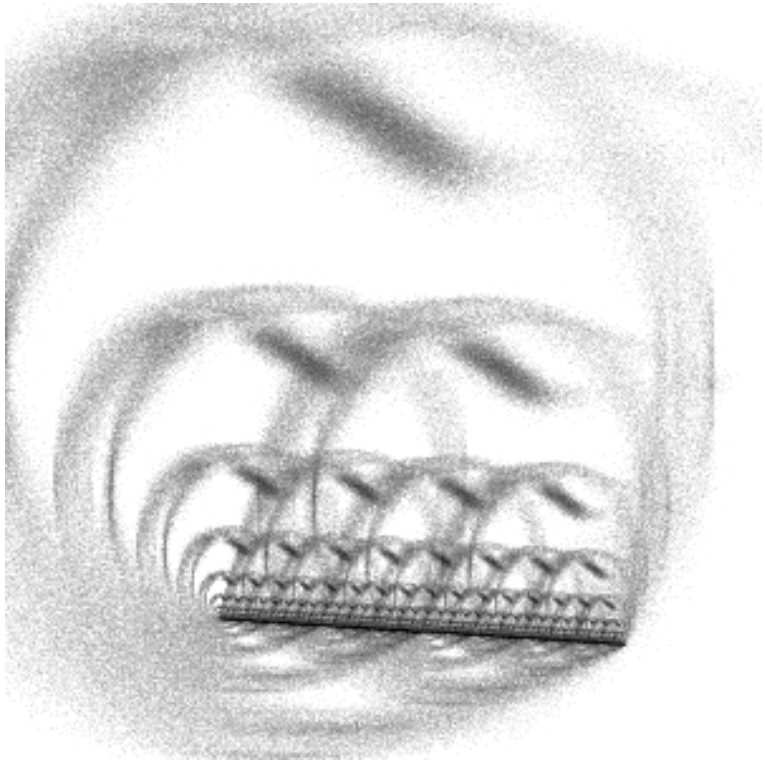


ArtiE-Fract





DANCING WOMAN ***(Danseuse)***



WATER WHEEL ***(Roue d'eau)***

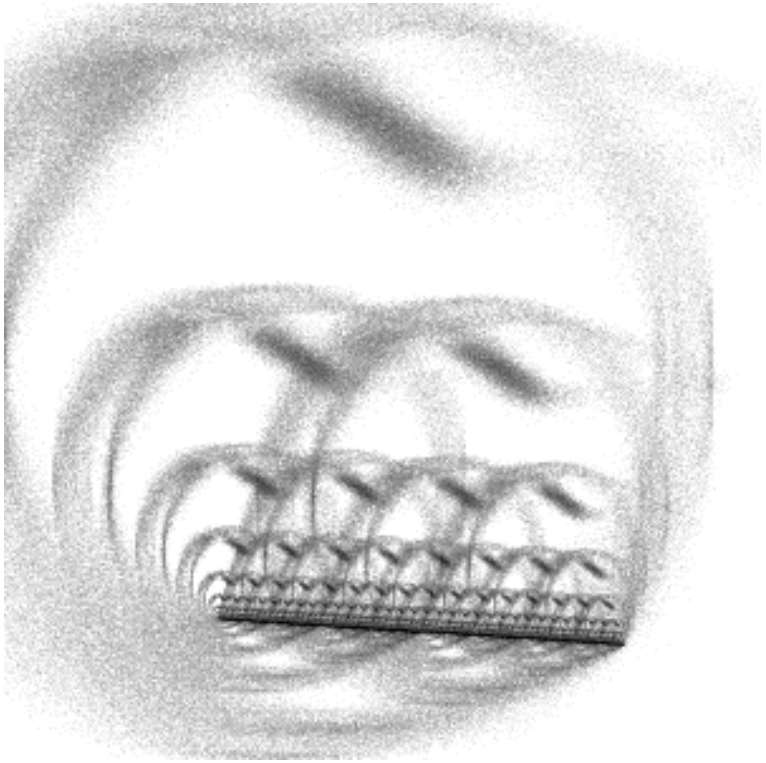
Emmanuel Cayla 2002

HARSH HORIZON ***(Horizon violent)***

FUNNY CIRCLES ***(Drôles de cercles)***

Emmanuel Cayla 2002

FOX ***(Renard)***



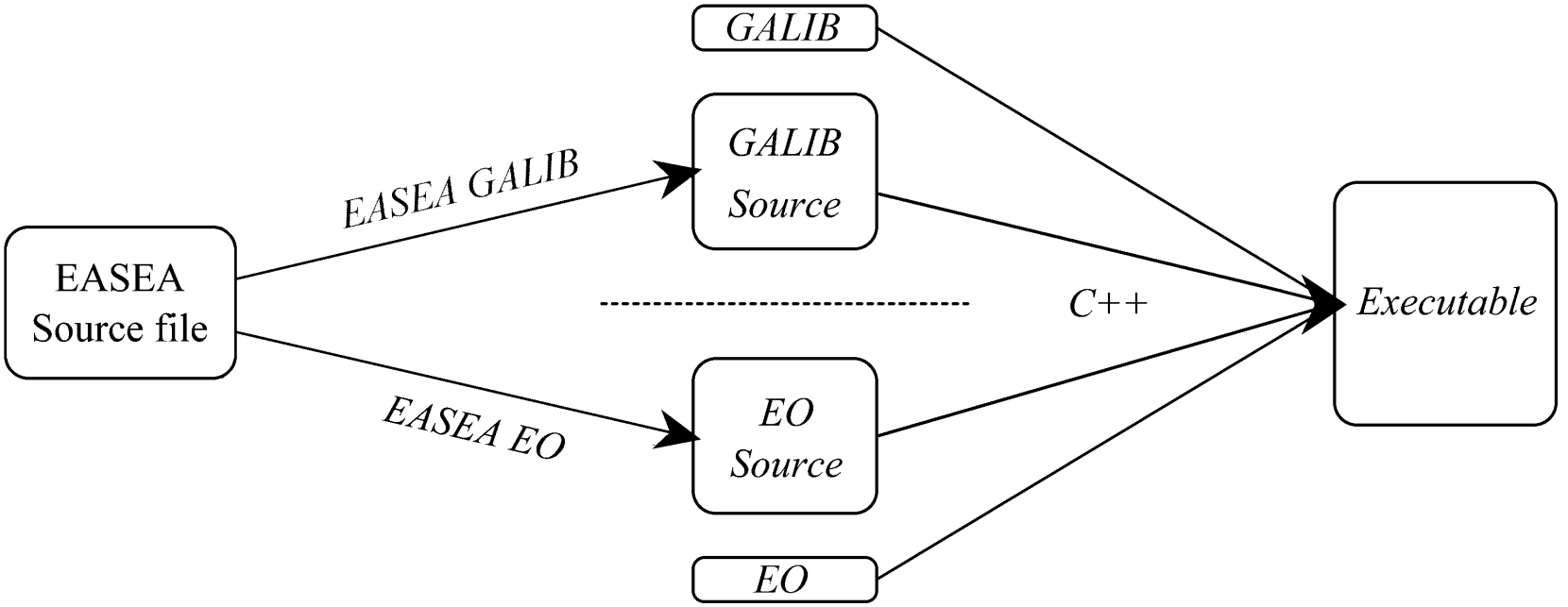
WATER WHEEL ***(Roue d'eau)***

Emmanuel Cayla 2002

EASEA : EAsy Specification for Evolutionary Algorithms

Langage de spécification d'Algorithmes Évolutionnaires.

http ://apis.saclay.inria.fr/evo-lab/EVO-easea-eng.html



```
/****** onemax.ez ******/

\User declarations :
#define SIZE 10
float pMutPerGene=0.1;

inline void swap(bool& a, bool& b)
  {bool c=a; a=b; b=c;}
\end

\User classes :
GenomeClass { bool x[SIZE]; }
\end

\GenomeClass::initialiser : // "initializer" is also accepted
  for (int i=0;i<SIZE;i++) Genome.x[i]=tossCoin(.5)?1:0;
\end

\GenomeClass::crossover :
  int CrossoverPosition=random(0,SIZE);

  for(int i=0;i<CrossoverPosition+1;i++)
    swap(child1.x[i],child2.x[i]);
\end

\GenomeClass::mutator : // Must return the number of mutations
  int NbMut=0;
  for (int i=0;i<SIZE;i++)
    if (tossCoin(pMutPerGene)){
       NbMut++;
       Genome.x[i]=Genome.x[i]?0:1;
    }
return NbMut;
\end

\GenomeClass::evaluator : // Returns the score
  int Score=0;
  for (int i=0; i<SIZE;i++)
    Score+=(int)Genome.x[i];
  return Score;
\end

\Default run parameters :        // Please let the parameters appear
                                 // in this order
  Number of generations : 15     // NB_GEN
  Mutation probability : 1       // MUT_PROB
  Crossover probability : 1      // XOVER_PROB
  Population size : 30           // POP_SIZE

  Offspring population size : 12 // 40%
  Replacement strategy : Plus    //  Comma, SteadyState, Generational
  Discarding operator : Worst    //  Best, Tournament, Parent, Random
  Evaluator goal : Maximise      // Minimise
  Elitism : On                   // Off
\end
```