UNIVERSITÉ
DE GENÈVE

FACULTÉ DES SCIENCES
Département d'informatique

MASTER OF COMPUTER SCIENCE

MÉTAHEURISTIQUES POUR L'OPTIMISATION
HOMEWORK ASSIGNMENT REPORT

Series 7 : Genetic Programming Using NGENES

Author :
Mohammad Oday DARWICH

Professors :
M. Bastien CHOPARD
Mme. Aziza MERZOUKI
M. Gregor CHLIAMOVITCH

12 December 2015

Introduction

La PG (Programmation Génétique) se classe dans la grande famille des algorithmes évolutionnaires qui s'inspire du paradigme de l'évolution darwinienne afin de faire évoluer des populations de solutions potentielles.

Cette population va subir les différentes étapes d'un processus évolutif simulé :

— sélection des individus les mieux adaptés à leur environnement (dans le cadre de la programmation génétique, cela correspond aux programmes répondant le mieux au problème posé)

— croisement des individus, dans le cadre de la programmation génétique, ce processus correspond à de "subtiles" échanges de codes entre deux programmes.

— mutation aléatoire des individus

et on veut évoluer un programme afin qu'il résolve un problème donné de façon optimale.

La partie la plus délicate dans les techniques évolutionnaires est l'écriture d'une fonction d'adaptation. Toujours dans le cadre du **mimétisme** du processus darwinien, seuls les individus les mieux adaptés à leur environnement survivent. Il devient donc nécessaire d'écrire de manière informatique une telle fonction *fitness* qui évaluera les individus (les programmes) afin de leur donner une note qualifiant leur degré d'adaptation.

L'autre écueil, plus spécifique à la programmation génétique consiste à sélectionner les briques de base qui permettront à créer des programmes. Si mon ensemble de brique de base est trop restreint, je ne pourrai peut-être jamais obtenir une solution au problème, par si cet ensemble est trop vaste, le processus algorithmique risque de ne jamais pouvoir converger.

Historiquement, les langages utilisés pour la programmation génétique étaient les langages fonctionnels comme le LISP ou Scheme.

Maintenant, on utilise des langages plus "classiques" comme le langage C, Python, Java, mais également le langage machine pour des raisons de rapidité.

0.1 Attente d'un tel programme

— produise un output adéquat pour tous les input possibles.

— Pour vérifier qu'un programme est bon, on pourrait le coupler à un simulateur et ajuster le programme pour minimiser le nombre d'erreur

— On peut voir cela comme un problème d'apprentissage ou aussi comme un problème d'optimisation : minimiser la différence entre les outputs produits et les outputs spécifiés.

— Le plus souvent cela consiste à trouver un programme qui produit des valeurs de sorties données à partir de valeurs d'entrées données

— Les données choisies sont définies comme l'ensemble d'apprentissage, ou l'ensemble test.

— Le but est alors que ce programme élaboré sur ce test-set soit capable de généralisation, c'est à dire qu'il aura capté la relation sous-jacente et qu'il pourra trouver l'output pour des inputs nouveaux.

1 Comment faire ?

1.1 Codeage par une S-expression

Koza a proposé de coder un programme avec un langage fonctionnel, à la Lisp, construit à base d'instructions appelée S-expressions par exemple comme dans ce TP on a

$$-2(\frac{sin(x)}{y}) \tag{1}$$

qui signifie

```
2 NEG X SIN Y DIV MUL
```

Si on remplace une opérande d'une S-expression par une autre S-expression, on obtient en général une formule valide, on peut alors définir des opérateurs d'évolution génétique qui mutent ou croisent les S-expressions

1.2 Function set et Terminal set

Comme on voit dans le code python, `functionSet et TerminalSet` ici une petite description.
Formellement l'espace de recherche des programmes génétiques est spécifié par un ensemble F de fonctions et un ensemble T de terminaux. où

— F contient les opérations possible dans cet espace

— T contient les variables et constantes possibles

For example F = `And, Or, NOT` avec des terminaux qui pourraient inclure des variables Booléennes et des constantes : T = b1, b2, b3, . . . , True, False.

Note Supp. : on peut définir des opérateurs IF dans le `function-set`
Par exemple : `(IF A B C D)` est une fonction à 4 arguments.
Si `A == B` alors `C` est retourné. Sinon D est retourné.

1.3 Propriété de clôture

- Chaque fonction doit pouvoir accepter comme arguments toutes les valeurs renvoyées par les autres fonctions, ainsi que toute valeur appartenante à T.
- La division, les opérateurs unaires, etc doivent être augmentés pour accepter n'importe quelle paire d'arguments.
- L'espace des programmes possibles est formé par toutes les composition de fonctions possibles à partir des éléments de F et de T.
- Cet espace est potentiellement de taille infinie et la longueur des individus est arbitraire.

1.4 Crossover

L'opération de croisement se fait en sélectionnant un lien au hasard dans l'arbre représentant chaque parent et en échangeant les sous-arbres correspondants, comme c'est présenter dans la figure ci dessous, et les point de croisement sont d'ordinaire choisis avec probabilité non-uniforme afin de favoriser les noeuds internes par rapport aux feuilles et les parties basses de l'arbre

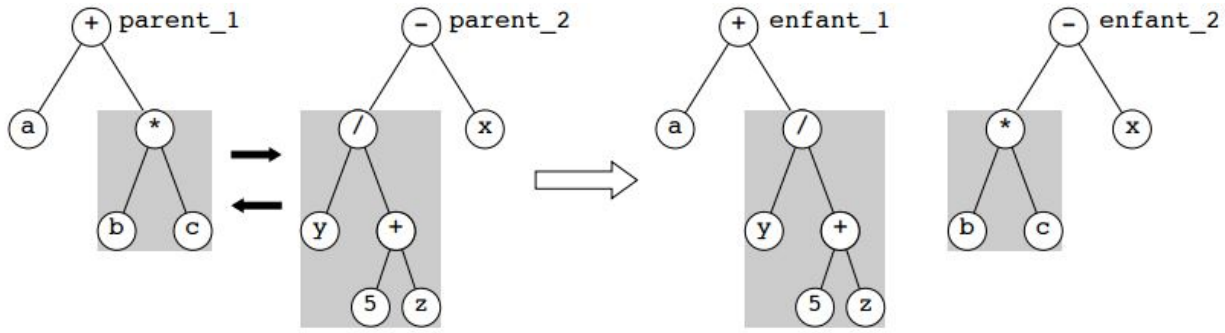

FIGURE 1 – Crossover KOZA

Note : quand je parle de l'arbre, ça fait référence au codage par arbre qui est équivalent à une S-expression mais plus lisible(vue dans le cours)

1.5 Mutation

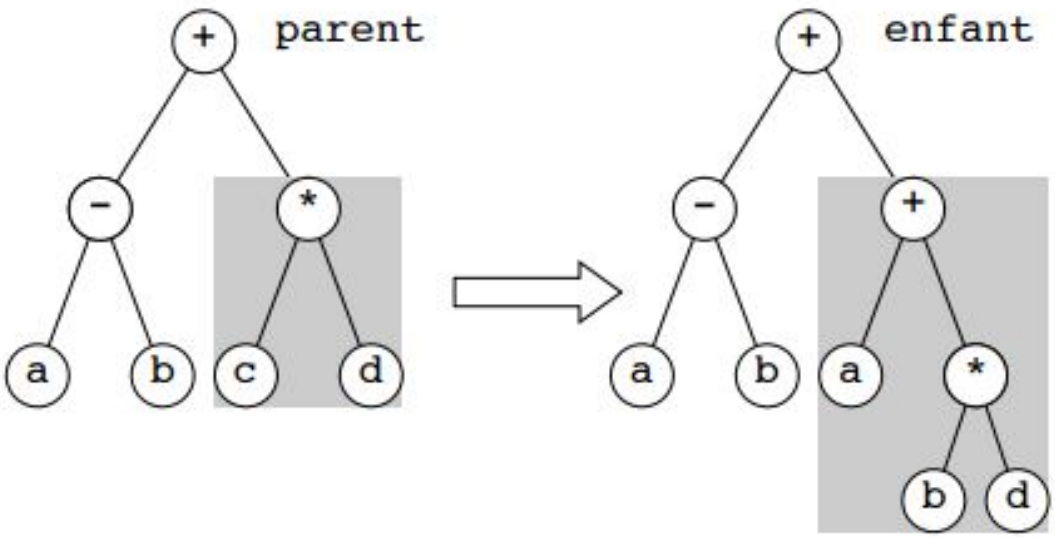

FIGURE 2 – Mutation KOZA

L'opérateur de mutation est réalisé en sélectionnant de manière aléatoire un sous-arbre du programme et en lui substituant un autre sous-arbre et ce dernier est typiquement généré au hasard à partir des ensembles F et T du problème.

1.6 Projection des concept générale sur ce TP

Ici on code un PG pqr une suite d'instructions aui seront exécutées sequentiellement.
Par exemple :
L'instruction A place la valeur de la variable A sur une pile.
Les opérations doivent adaptées pour tolérer le manque d'un argument sur la pile
L'espace ds programme est donc défini par l'ensemble des instructions possibles, qui comprend des variables, constantes et opérations.
Pour le **Crossover** on peut utiliser le crossover à un point vu pour les chaînes de bits.
Pour la **mutation**, elle est obtenue en chageant une instruction par une autre au hasard.
L'output est le sommet de la pule après exécution du programme.

2 Results

On prend une pupulation de `popSize =200` et `pc=0.1` et `pm=0.5` on obtient les résultats suivant :
This is a result for one of the execution with $p_c = 0.6$, $p_m = 0.5$, $popSize = 200$

File - unknown

```
1  C:¥Users¥mod¥AppData¥Local¥Programs¥Python¥Python35-32¥python.exe C
   :/Users/mod/OneDrive/2015-2016/Uni/AutumnSem2015/aMHOA/cTP/
   TP7_MHOA/TP7_-_Genetic_Programming/gppp.py
2  prog : ['XOR', 'OR', 'X1', 'X2', 'X2', 'OR']
3  output of execute(prog, cpu, data) :  0
4  ------------
5  [['NOT', 'OR', 'NOT', 'X3', 'X1', 'X3'], ['X1', 'AND', 'OR', 'X4', 'X3', 'X2'], ['X2', 'X4', '
   XOR', 'AND', 'X4', 'X2'], ['X3', 'OR', 'NOT', 'X2', 'AND', 'XOR'], ['X4', 'X4', 'XOR', '
   OR', 'X1', 'X4'], ['XOR', 'OR', 'NOT', 'OR', 'X4', 'AND'], ['OR', 'X2', 'AND', 'X4', '
   AND', 'X2'], ['OR', 'X3', 'NOT', 'OR', 'X4', 'X4'], ['X2', 'XOR', 'X2', 'X2', 'X4', 'NOT
   '], ['X4', 'X2', 'NOT', 'X4', 'X1', 'AND'], ['X3', 'X4', 'XOR', 'XOR', 'NOT', 'X3'], ['X4', '
   OR', 'X1', 'AND', 'OR', 'X2'], ['XOR', 'XOR', 'X4', 'AND', 'OR', 'OR'], ['X4', 'NOT', '
   NOT', 'X2', 'X4', 'X3'], ['X2', 'XOR', 'AND', 'X1', 'X1', 'AND'], ['X4', 'X3', 'X3', 'NOT
   ', 'X4', 'X3'], ['X2', 'AND', 'NOT', 'XOR', 'NOT', 'X4'], ['XOR', 'X1', 'X1', 'X2', 'X1', '
   X1'], ['XOR', 'NOT', 'X3', 'AND', 'AND', 'X2'], ['XOR', 'XOR', 'X2', 'AND', 'OR', 'OR
   '], ['X3', 'X4', 'AND', 'AND', 'XOR', 'X4'], ['AND', 'XOR', 'NOT', 'X4', 'X4', 'X2'], ['X1
   ', 'OR', 'OR', 'X4', 'XOR', 'X2'], ['X4', 'X2', 'OR', 'NOT', 'NOT', 'AND'], ['X2', 'OR', '
   X1', 'X3', 'X2', 'OR'], ['AND', 'NOT', 'X4', 'X4', 'X4', 'X2'], ['X3', 'AND', 'X1', 'AND', '
   X4', 'X1'], ['X2', 'AND', 'NOT', 'AND', 'X4', 'OR'], ['X1', 'NOT', 'OR', 'X4', 'X2', 'NOT
   '], ['X4', 'NOT', 'X1', 'NOT', 'X1', 'NOT'], ['X1', 'X2', 'X4', 'X3', 'OR', 'OR'], ['NOT', '
   OR', 'XOR', 'NOT', 'AND', 'XOR'], ['XOR', 'AND', 'X2', 'AND', 'AND', 'NOT'], ['X2', '
   AND', 'X3', 'AND', 'X1', 'XOR'], ['X1', 'X1', 'X4', 'NOT', 'AND', 'X4'], ['AND', 'X3', '
   XOR', 'X4', 'XOR', 'XOR'], ['NOT', 'XOR', 'NOT', 'X4', 'X2', 'XOR'], ['X3', 'OR', 'OR
   ', 'X3', 'AND', 'X4'], ['AND', 'NOT', 'NOT', 'X4', 'AND', 'X3'], ['XOR', 'X2', 'NOT', '
   AND', 'X3', 'OR'], ['OR', 'X4', 'X1', 'X4', 'X2', 'XOR'], ['X3', 'X1', 'X2', 'X1', 'X4', 'X4
   '], ['X4', 'AND', 'OR', 'X2', 'X1', 'X3'], ['AND', 'X3', 'OR', 'AND', 'NOT', 'X3'], ['XOR
   ', 'AND', 'OR', 'NOT', 'XOR', 'X2'], ['X1', 'OR', 'NOT', 'AND', 'OR', 'XOR'], ['X4', 'X4
   ', 'NOT', 'X1', 'X4', 'X4'], ['X1', 'AND', 'OR', 'OR', 'AND', 'OR'], ['X4', 'NOT', 'AND', '
   NOT', 'X2', 'X3'], ['X2', 'OR', 'NOT', 'AND', 'AND', 'X1'], ['XOR', 'X2', 'X4', 'OR', 'X4
   ', 'X3'], ['AND', 'NOT', 'X3', 'AND', 'X4', 'NOT'], ['X2', 'AND', 'NOT', 'AND', 'X4', 'X4
   '], ['NOT', 'X4', 'X2', 'X2', 'AND', 'X4'], ['X4', 'X4', 'NOT', 'NOT', 'NOT', 'OR'], ['AND
   ', 'AND', 'X3', 'AND', 'X2', 'X2'], ['AND', 'NOT', 'NOT', 'NOT', 'X4', 'X4'], ['AND', 'X4
   ', 'X1', 'OR', 'X1', 'AND'], ['X4', 'NOT', 'AND', 'NOT', 'AND', 'X1'], ['X2', 'X2', 'X2', '
   X3', 'X3', 'X2'], ['X3', 'X2', 'X1', 'X1', 'X1', 'X3'], ['X1', 'XOR', 'OR', 'X2', 'X4', 'X3'], ['
   X4', 'AND', 'NOT', 'X4', 'X4', 'X3'], ['X4', 'AND', 'NOT', 'X1', 'X1', 'OR'], ['NOT', 'OR
   ', 'NOT', 'XOR', 'XOR', 'XOR'], ['AND', 'X2', 'OR', 'X4', 'X4', 'NOT'], ['X1', 'XOR', '
   X3', 'X1', 'X2', 'X1'], ['X3', 'X4', 'NOT', 'X3', 'X1', 'AND'], ['X3', 'X4', 'X2', 'NOT', 'X3
   ', 'XOR'], ['X2', 'X3', 'X4', 'X3', 'AND', 'X3'], ['AND', 'AND', 'NOT', 'X1', 'OR', 'X4'], ['
   AND', 'XOR', 'XOR', 'X4', 'X1', 'OR'], ['OR', 'X2', 'X3', 'X1', 'X3', 'X3'], ['NOT', 'OR
   ', 'XOR', 'X3', 'XOR', 'OR'], ['X3', 'OR', 'X3', 'X3', 'X4', 'X2'], ['X2', 'NOT', 'XOR', '
   X1', 'X1', 'X2'], ['X2', 'X3', 'X2', 'X3', 'X3', 'X2'], ['NOT', 'X4', 'X2', 'XOR', 'X3', 'X1
   '], ['XOR', 'XOR', 'XOR', 'X4', 'AND', 'X2'], ['XOR', 'NOT', 'X1', 'X2', 'X4', 'X3'], ['X1
   ', 'XOR', 'AND', 'OR', 'X2', 'AND'], ['OR', 'X4', 'AND', 'NOT', 'NOT', 'X1'], ['NOT', '
   XOR', 'X2', 'XOR', 'X3', 'X2'], ['XOR', 'X2', 'X1', 'XOR', 'X2', 'X3'], ['X2', 'X4', 'AND
   ', 'X4', 'X1', 'X1'], ['X2', 'X4', 'X1', 'XOR', 'OR', 'X2'], ['AND', 'X2', 'AND', 'XOR', '
   NOT', 'XOR'], ['AND', 'X2', 'XOR', 'NOT', 'NOT', 'OR'], ['OR', 'NOT', 'OR', 'X2', 'X2
   ', 'AND'], ['X2', 'XOR', 'NOT', 'X2', 'NOT', 'X1'], ['NOT', 'X3', 'X2', 'XOR', 'X4', '
   XOR'], ['OR', 'XOR', 'OR', 'X4', 'X4', 'NOT'], ['X4', 'X1', 'X2', 'X2', 'X1', 'NOT'], ['
   XOR', 'AND', 'X4', 'NOT', 'X3', 'OR'], ['X1', 'X3', 'X1', 'X2', 'X4', 'X4'], ['NOT', 'OR', '
   X3', 'NOT', 'AND', 'OR'], ['OR', 'OR', 'NOT', 'AND', 'X2', 'X1'], ['X4', 'X3', 'X2', 'X1
   ', 'X4', 'OR'], ['NOT', 'NOT', 'X1', 'X3', 'NOT', 'X1'], ['X2', 'X3', 'X2', 'X1', 'X1', 'NOT
   ']]
6  (['X4', 'X4', 'XOR', 'OR', 'X1', 'X4'], 5)
7  (['X2', 'X3', 'X2', 'X3', 'OR', 'NOT'], 3)
8  (['X2', 'X3', 'X2', 'X3', 'OR', 'NOT'], 3)
9  (['X2', 'X3', 'X2', 'X3', 'OR', 'NOT'], 3)
10 (['X2', 'X3', 'X2', 'X3', 'OR', 'NOT'], 3)
```

```
11 (['X2', 'X3', 'X2', 'X3', 'OR', 'NOT'], 3)
12 (['X2', 'X3', 'X2', 'X3', 'OR', 'NOT'], 3)
13 (['X2', 'X3', 'X2', 'X3', 'OR', 'NOT'], 3)
14 (['X2', 'X2', 'NOT', 'NOT', 'XOR', 'OR'], 3)
15 (['X1', 'OR', 'AND', 'X2', 'X2', 'XOR'], 3)
16 (['X1', 'OR', 'AND', 'X2', 'X2', 'XOR'], 3)
17 (['X1', 'OR', 'AND', 'X2', 'X2', 'XOR'], 3)
18 (['X1', 'OR', 'AND', 'X2', 'X2', 'XOR'], 3)
19 (['XOR', 'X1', 'NOT', 'XOR', 'X1', 'AND'], 3)
20 (['X2', 'AND', 'X2', 'NOT', 'NOT', 'XOR'], 3)
21 (['X2', 'AND', 'X2', 'NOT', 'NOT', 'XOR'], 3)
22 (['X2', 'AND', 'X2', 'NOT', 'NOT', 'XOR'], 3)
23 (['X2', 'AND', 'X2', 'NOT', 'NOT', 'XOR'], 3)
24 (['X2', 'AND', 'X2', 'NOT', 'NOT', 'XOR'], 3)
25 (['X2', 'AND', 'X2', 'NOT', 'NOT', 'XOR'], 3)
26 (['X2', 'AND', 'X2', 'NOT', 'NOT', 'XOR'], 3)
27 (['X3', 'X4', 'X4', 'X1', 'AND', 'XOR'], 3)
28 (['X1', 'XOR', 'OR', 'OR', 'X1', 'XOR'], 3)
29 (['X1', 'XOR', 'OR', 'OR', 'X1', 'XOR'], 3)
30 (['X1', 'XOR', 'OR', 'OR', 'X1', 'XOR'], 3)
31 (['X1', 'XOR', 'OR', 'OR', 'X1', 'XOR'], 3)
32 (['X1', 'XOR', 'OR', 'OR', 'X1', 'XOR'], 3)
33 (['X4', 'XOR', 'X4', 'X3', 'XOR', 'AND'], 3)
34 (['X4', 'XOR', 'X4', 'X3', 'XOR', 'AND'], 3)
35 (['X4', 'XOR', 'X4', 'X3', 'XOR', 'AND'], 3)
36 (['X4', 'XOR', 'X4', 'X3', 'XOR', 'AND'], 3)
37 (['X2', 'X2', 'XOR', 'X3', 'AND', 'XOR'], 3)
38 (['X2', 'X2', 'XOR', 'X3', 'AND', 'XOR'], 3)
39 (['X2', 'X2', 'XOR', 'X3', 'AND', 'XOR'], 3)
40 (['XOR', 'X1', 'X1', 'XOR', 'X4', 'AND'], 3)
41 (['XOR', 'X1', 'X1', 'XOR', 'X4', 'AND'], 3)
42 (['XOR', 'X1', 'X1', 'XOR', 'X4', 'AND'], 3)
43 (['XOR', 'X1', 'X1', 'XOR', 'X4', 'AND'], 3)
44 (['XOR', 'X1', 'X1', 'XOR', 'X4', 'AND'], 3)
45 (['XOR', 'X1', 'X1', 'XOR', 'X4', 'AND'], 3)
46 (['XOR', 'X1', 'X1', 'XOR', 'X4', 'AND'], 3)
47 (['XOR', 'X1', 'X1', 'XOR', 'X4', 'AND'], 3)
48 (['XOR', 'X1', 'X1', 'XOR', 'X4', 'AND'], 3)
49 (['XOR', 'X1', 'X1', 'XOR', 'X4', 'AND'], 3)
50 (['XOR', 'X4', 'XOR', 'X4', 'XOR', 'XOR'], 3)
51 (['X2', 'X1', 'X3', 'NOT', 'X4', 'AND'], 3)
52 (['X4', 'OR', 'X1', 'X1', 'XOR', 'AND'], 3)
53 (['X3', 'X2', 'OR', 'NOT', 'NOT', 'NOT'], 3)
54 (['X3', 'X2', 'OR', 'NOT', 'NOT', 'NOT'], 3)
55 (['X3', 'X2', 'OR', 'NOT', 'NOT', 'NOT'], 3)
56 (['X3', 'X2', 'OR', 'NOT', 'NOT', 'NOT'], 3)
57 (['X2', 'X4', 'X4', 'NOT', 'OR', 'NOT'], 3)
58 (['NOT', 'NOT', 'X4', 'NOT', 'X4', 'AND'], 3)
59 (['NOT', 'NOT', 'X4', 'NOT', 'X4', 'AND'], 3)
60 (['NOT', 'NOT', 'X4', 'NOT', 'X4', 'AND'], 3)
61 (['NOT', 'NOT', 'X4', 'NOT', 'X4', 'AND'], 3)
62 (['NOT', 'NOT', 'X4', 'NOT', 'X4', 'AND'], 3)
63 (['OR', 'X1', 'X3', 'X3', 'X3', 'XOR'], 3)
64 (['AND', 'X3', 'OR', 'X4', 'X4', 'XOR'], 3)
65 (['AND', 'X3', 'OR', 'X4', 'X4', 'XOR'], 3)
66 (['AND', 'X3', 'OR', 'X4', 'X4', 'XOR'], 3)
67 (['AND', 'X3', 'OR', 'X4', 'X4', 'XOR'], 3)
```

```
68  (['AND', 'X3', 'OR', 'X4', 'X4', 'XOR'], 3)
69  (['NOT', 'AND', 'X3', 'X1', 'X1', 'XOR'], 3)
70  (['NOT', 'AND', 'X3', 'X1', 'X1', 'XOR'], 3)
71  (['NOT', 'AND', 'X3', 'X1', 'X1', 'XOR'], 3)
72  (['X1', 'X2', 'X4', 'X2', 'NOT', 'AND'], 3)
73  (['X2', 'X4', 'X4', 'X1', 'NOT', 'AND'], 3)
74  (['X2', 'X4', 'OR', 'AND', 'X3', 'AND'], 5)
75  (['X1', 'AND', 'X2', 'X3', 'X3', 'XOR'], 3)
76  (['XOR', 'X1', 'OR', 'X1', 'XOR', 'OR'], 3)
77  (['OR', 'X2', 'X4', 'NOT', 'X4', 'AND'], 3)
78  (['X4', 'X4', 'X1', 'AND', 'XOR', 'OR'], 3)
79  (['X4', 'X4', 'X1', 'AND', 'XOR', 'OR'], 3)
80  (['X4', 'X4', 'X1', 'AND', 'XOR', 'OR'], 3)
81  (['X4', 'X4', 'X1', 'AND', 'XOR', 'OR'], 3)
82  (['X4', 'X4', 'X1', 'AND', 'XOR', 'OR'], 3)
83  (['X4', 'X4', 'X1', 'AND', 'XOR', 'OR'], 3)
84  (['X4', 'X4', 'X1', 'AND', 'XOR', 'OR'], 3)
85  (['X4', 'X4', 'AND', 'AND', 'XOR', 'OR'], 5)
86  (['X1', 'X4', 'XOR', 'X4', 'X1', 'AND'], 5)
87  (['NOT', 'X1', 'NOT', 'X2', 'X2', 'XOR'], 3)
88  (['XOR', 'OR', 'X3', 'X3', 'XOR', 'OR'], 3)
89  (['X4', 'NOT', 'X4', 'XOR', 'NOT', 'AND'], 3)
90  (['X4', 'NOT', 'X4', 'XOR', 'NOT', 'AND'], 3)
91  (['X4', 'NOT', 'X4', 'XOR', 'NOT', 'AND'], 3)
92  (['AND', 'NOT', 'X2', 'X4', 'X4', 'XOR'], 3)
93  (['AND', 'NOT', 'X2', 'X4', 'X4', 'XOR'], 3)
94  (['X2', 'X2', 'XOR', 'AND', 'OR', 'XOR'], 3)
95  (['X4', 'OR', 'X1', 'X4', 'X4', 'XOR'], 3)
96  (['X4', 'OR', 'X1', 'X4', 'X4', 'XOR'], 3)
97  (['X4', 'OR', 'X1', 'X4', 'X4', 'XOR'], 3)
98  (['X4', 'OR', 'X1', 'X4', 'X4', 'XOR'], 3)
99  (['XOR', 'X1', 'X3', 'X3', 'XOR', 'AND'], 3)
100 (['X3', 'X2', 'AND', 'X4', 'AND', 'AND'], 3)
101 (['X3', 'X2', 'AND', 'X4', 'AND', 'AND'], 3)
102 (['X4', 'X4', 'OR', 'XOR', 'X1', 'AND'], 3)
103 (['X4', 'X4', 'OR', 'XOR', 'X1', 'AND'], 3)
104 (['X4', 'X4', 'OR', 'XOR', 'X1', 'AND'], 3)
105 (['X4', 'X4', 'OR', 'XOR', 'X1', 'AND'], 3)
106 (['X4', 'X4', 'OR', 'XOR', 'X1', 'AND'], 3)
107 (['AND', 'X1', 'XOR', 'X1', 'XOR', 'XOR'], 3)
108 (['XOR', 'X1', 'NOT', 'AND', 'X1', 'X4'], 5)
109 (['XOR', 'X1', 'NOT', 'AND', 'X1', 'X4'], 5)
110 (['X4', 'X2', 'OR', 'X4', 'NOT', 'NOT'], 5)
111 (['OR', 'X4', 'XOR', 'X4', 'X4', 'XOR'], 3)
112 (['OR', 'NOT', 'X3', 'X3', 'OR', 'XOR'], 3)
113 (['OR', 'NOT', 'X3', 'X3', 'OR', 'XOR'], 3)
114 (['X4', 'XOR', 'OR', 'AND', 'X4', 'XOR'], 3)
115 (['X1', 'AND', 'OR', 'X1', 'XOR', 'AND'], 3)
116 (['X1', 'X1', 'XOR', 'OR', 'X1', 'AND'], 3)
117 (['X1', 'X1', 'XOR', 'OR', 'X1', 'AND'], 3)
118 (['X1', 'X1', 'XOR', 'OR', 'X1', 'AND'], 3)
119 (['NOT', 'NOT', 'OR', 'X2', 'X2', 'XOR'], 3)
120 (['X4', 'AND', 'AND', 'X4', 'X4', 'XOR'], 3)
121 (['X4', 'AND', 'AND', 'X4', 'X4', 'XOR'], 3)
122 (['X3', 'X3', 'XOR', 'OR', 'AND', 'AND'], 3)
123 (['X3', 'XOR', 'AND', 'NOT', 'X4', 'AND'], 3)
124 (['X3', 'XOR', 'AND', 'NOT', 'X4', 'AND'], 3)
```

```
125 (['X3', 'XOR', 'AND', 'NOT', 'X4', 'AND'], 3)
126 (['X3', 'XOR', 'AND', 'NOT', 'X4', 'AND'], 3)
127 (['X1', 'NOT', 'XOR', 'X1', 'XOR', 'NOT'], 3)
128 (['X1', 'NOT', 'XOR', 'X1', 'XOR', 'NOT'], 3)
129 (['X1', 'NOT', 'XOR', 'X1', 'XOR', 'NOT'], 3)
130 (['X1', 'NOT', 'XOR', 'X1', 'XOR', 'NOT'], 3)
131 (['X1', 'NOT', 'XOR', 'X1', 'XOR', 'NOT'], 3)
132 (['X1', 'NOT', 'XOR', 'X1', 'XOR', 'NOT'], 3)
133 (['X1', 'NOT', 'XOR', 'X1', 'XOR', 'NOT'], 3)
134 (['X1', 'NOT', 'XOR', 'X1', 'XOR', 'NOT'], 3)
135 (['X1', 'NOT', 'XOR', 'X1', 'XOR', 'NOT'], 3)
136 (['XOR', 'X4', 'XOR', 'X3', 'X3', 'XOR'], 3)
137 (['X3', 'NOT', 'X3', 'X4', 'AND', 'AND'], 3)
138 (['X1', 'NOT', 'X3', 'X4', 'AND', 'AND'], 3)
139 (['X1', 'NOT', 'X3', 'X4', 'AND', 'AND'], 3)
140 (['X1', 'NOT', 'X3', 'X4', 'AND', 'AND'], 3)
141 (['X4', 'XOR', 'XOR', 'X4', 'XOR', 'OR'], 3)
142 (['X3', 'X4', 'X4', 'X4', 'X2', 'AND'], 5)
143 (['X2', 'X4', 'X4', 'NOT', 'OR', 'NOT'], 3)
144 (['NOT', 'X1', 'X1', 'X1', 'AND', 'XOR'], 3)
145 (['NOT', 'X1', 'X1', 'X1', 'AND', 'XOR'], 3)
146 (['NOT', 'X1', 'X1', 'X1', 'AND', 'XOR'], 3)
147 (['NOT', 'X1', 'X1', 'X1', 'AND', 'XOR'], 3)
148 (['X4', 'XOR', 'X4', 'X3', 'X3', 'XOR'], 3)
149 (['X2', 'X4', 'X4', 'NOT', 'NOT', 'XOR'], 3)
150 (['X2', 'X4', 'X4', 'NOT', 'NOT', 'XOR'], 3)
151 (['X3', 'NOT', 'X1', 'X3', 'X3', 'XOR'], 3)
152 (['X3', 'NOT', 'X1', 'X3', 'X3', 'XOR'], 3)
153 (['X4', 'X4', 'NOT', 'NOT', 'XOR', 'XOR'], 3)
154 (['X4', 'X4', 'NOT', 'NOT', 'XOR', 'XOR'], 3)
155 (['X4', 'X4', 'NOT', 'NOT', 'XOR', 'XOR'], 3)
156 (['X4', 'X4', 'NOT', 'NOT', 'XOR', 'XOR'], 3)
157 (['X4', 'X4', 'NOT', 'NOT', 'XOR', 'XOR'], 3)
158 (['X4', 'X4', 'NOT', 'NOT', 'XOR', 'XOR'], 3)
159 (['X4', 'NOT', 'X3', 'X2', 'OR', 'NOT'], 3)
160 (['X4', 'NOT', 'X3', 'X2', 'OR', 'NOT'], 3)
161 (['X4', 'NOT', 'X3', 'X2', 'OR', 'NOT'], 3)
162 (['X4', 'NOT', 'X3', 'X2', 'OR', 'NOT'], 3)
163 (['AND', 'X3', 'X3', 'X3', 'XOR', 'AND'], 3)
164 (['AND', 'X3', 'X3', 'X3', 'XOR', 'AND'], 3)
165 (['AND', 'X3', 'X3', 'X3', 'XOR', 'AND'], 3)
166 (['AND', 'X3', 'X3', 'X3', 'XOR', 'AND'], 3)
167 (['AND', 'X3', 'X3', 'X3', 'XOR', 'AND'], 3)
168 (['AND', 'X3', 'X3', 'X3', 'XOR', 'AND'], 3)
169 (['AND', 'X3', 'X3', 'X3', 'XOR', 'AND'], 3)
170 (['AND', 'X3', 'X3', 'X3', 'XOR', 'AND'], 3)
171 (['X1', 'X1', 'XOR', 'X1', 'AND', 'AND'], 3)
172 (['X1', 'X1', 'AND', 'X4', 'X2', 'AND'], 5)
173 (['X1', 'X1', 'AND', 'X4', 'X2', 'AND'], 5)
174 (['AND', 'NOT', 'AND', 'NOT', 'X4', 'X4'], 5)
175 (['X1', 'AND', 'X1', 'X4', 'OR', 'XOR'], 3)
176 (['X1', 'AND', 'X1', 'X4', 'OR', 'XOR'], 3)
177 (['X1', 'AND', 'X1', 'X4', 'OR', 'XOR'], 3)
178 (['X1', 'AND', 'X1', 'X4', 'OR', 'XOR'], 3)
179 (['X1', 'OR', 'AND', 'OR', 'X1', 'XOR'], 3)
180 (['X1', 'OR', 'AND', 'OR', 'X1', 'XOR'], 3)
181 (['X4', 'X2', 'AND', 'X4', 'XOR', 'XOR'], 3)
```

File - unknown

```
182 (['X4', 'X2', 'AND', 'X4', 'XOR', 'XOR'], 3)
183 (['X4', 'X2', 'AND', 'X4', 'XOR', 'XOR'], 3)
184 (['AND', 'X1', 'X4', 'X3', 'X3', 'XOR'], 3)
185 (['X4', 'XOR', 'X4', 'X1', 'AND', 'AND'], 5)
186 (['AND', 'X2', 'X4', 'X2', 'NOT', 'AND'], 3)
187 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
188 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
189 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
190 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
191 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
192 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
193 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
194 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
195 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
196 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
197 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
198 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
199 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
200 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
201 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
202 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
203 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
204 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
205 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
206 (['X4', 'X3', 'X2', 'OR', 'NOT', 'AND'], 1)
207
208 Process finished with exit code 0
209
```

Dans cet expérience, on prend une population de 100, qu'on fait évoluer 100 fois avec selection, crossover et mutation, ainsi je fais fixer p_c et varier p_m d'une façon croissante.

TABLE 1 – Results with changing parameters of p_c & p_m

	popSize = 100				
Res\params	p_c = 0.6 p_m= 0.01	p_c = 0.6 p_m=0.1	p_c = 0.6 p_m=0.2	p_c = 0.6 p_m= 0.4	p_c = 0.6 p_m=0.5
Best F	3.0	3.0	1.0	3.0	3.0
Mean	3.0	3.4	1.48	2.98	3.0
STD	0.0	0.28141058827257937	0.8584693919818557	0.2	0.0
	['X1', 'X1', 'X2', 'X3', 'X3', 'XOR']	['X1', 'X2', 'X4', 'XOR', 'X4', 'AND']	['X2', 'X3', 'OR', 'NOT', 'X4', 'AND']	['X1', 'X2', 'X4', 'X1', 'X1', 'XOR']	['XOR', 'X3', 'X4', 'OR', 'X3', 'XOR']

2.1 discussion exp 1

Dans cet expérience ça se voit que la mutation nous faire éviter à s'enliser dans des optima locaux. Mais si elle est trop fréquente, le PG est orienté vers une recherche aléatoire de la bonne solution. Dans le tableau ci dessous on fait la même expérience d'avant mais en variant p_c

TABLE 2 – Results with varying $p_c = 0.6$, increasing it

	popSize = 100	popSize = 100	popSize = 100	popSize = 100	popSize = 100
Res\params	p_m = 0.6 p_c= 0.01	p_m = 0.6 p_c=0.1	p_m = 0.6 p_c=0.2	p_m= 0.6 p_c= 0.4	p_m= 0.6 p_c=0.5
Best F	3.0	3.0	3.0	3.0	1
Mean	4.5	3.08	3.12	3.02	1.62
STD	1.3349989596333858	0.3938927711338647	0.47736651315188405	0.2	0.9296463974234636
	['NOT', 'XOR', 'X4', 'AND', 'X4', 'XOR']	['X2', 'AND', 'X2', 'XOR', 'OR', 'XOR']	['X3', 'XOR', 'X3', 'NOT', 'X4', 'AND']	['AND', 'X4', 'XOR', 'X4', 'XOR', 'XOR']	['X4', 'X3', 'X2', 'NOT', 'XOR', 'AND']

2.2 discussion exp 2

Après regarder les résultat comme dans la page -5-6-7 pour chaque exécution on remarque bien que s'il n'y a pas de crossover où la valeur très proche de zero, les nouvelles générations sont l'exacte copie des racines alors que s'il y a du crossover, les générations sont composés d'une partie de chacun de leur racines et alors les nouveaux générations gardent la meilleur partie des anciens.et si la probabilité est au max tous les descendant sont générés par crossover
Ceci dans le but d'obtenir de meilleurs générations. néanmoins, il est quand même important qu'une partie de la population survivre à la nouvelle génération.

Dans cet expérience, on prend une population de 12, qu'on fait évoluer 50 fois avec selection, crossover et mutation.

TABLE 3 – Results with same params $P_m = 0.01$, $p_c = 0.6$ on 5 execution

	popSize = 12				
Exec	1	2	3	4	5
Best F	1.0	3.0	3.0	3.0	1
Mean	1.06	3.02	3.56	3.04	1.69
STD	0.3428932159955306	0.2	0.282842712474619	0.282842712474619	0.3938927711338647
	['X4', 'X3', 'X2', 'OR', 'NOT', 'AND']	['OR', 'NOT', 'X3', 'OR', 'X3', 'XOR']	['X2', 'XOR', 'AND', 'OR', 'XOR', 'X4']	['OR', 'X4', 'XOR', 'X4', 'OR', 'XOR']	['NOT', 'X3', 'X2', 'NOT', 'X4', 'AND']

2.3 discussion exp 3

J'ai fait cet expérience pour voir l'effet du size de la population sur les résultats et comme présent dans le tableau ci dessus, on remarque que le STD varie d'autant que dans les premiers deux expérience avec un optimum de 1 où la $p_c = 0.01$ et $p_c = 0.5$ alors que pour $p_c > 0.1$ et <0.5 le meilleur optimum est 3 et l'écart type est respectivement autour de 34% et 20 %

3 Conclusion et discussion générale

Durant toutes les expériences on ne trouve pas l'optimum à chaque run, et donc l'optimum a fitness 1.
Il y a des fonction qui sont curieusement plus difficile à trouve car l'une des éléments du programme est un attracteur.

Les algorithmes génétiques fournissent des solutions proches de la solution optimale à l'aide des mécanismes de sélection, de crossover et de mutation comme le montre les 3 tableau ci dessus. et c'est pour ça qu'ils sont applicables à nombreux problèmes, dont le problème de vol (crach vol ẽrreur).
Une note sur la méthode de séléction par tournoi, c'est que avec un nombre élevé de participant, un individu faible sera presque toujous sûr de perdre.

Finalement avec cette méthode on peut trouves des programmes qui répondes au mueux à une tache définie. Pour ce faire, elle permet à la machine d'apprendre, en utilisant cet algorithme évolutionniste afin d'optimiser la population de programmes.