

# Pierre-Feuille-Ciseaux



Sujet proposé par Jean-Christophe Filliâtre

`mailto:Jean-Christophe.Filliatre@lri.fr`

Difficulté : dans la moyenne (★★)

**URL de suivi :** `http://www.enseignement.polytechnique.fr/profs/informatique/Jean-Christophe.Filliatre/08-09/INF431/rps/`

Le but de ce projet est de réaliser un programme jouant à « pierre-feuille-ciseaux » par des techniques de *programmation génétique*. Pierre-feuille-ciseaux est un jeu répandu dans les cours d'écoles où deux joueurs s'affrontent à l'aide des trois objets que sont la *pierre*, la *feuille* et les *ciseaux*. Simultanément, les deux joueurs indiquent leur choix en matérialisant de la main l'objet choisi. La pierre l'emporte alors sur les ciseaux (en les émoussant), qui l'emportent eux-mêmes sur la feuille (en la coupant), qui l'emporte elle-même sur la pierre (en l'enveloppant). La partie peut comporter un ou plusieurs tours. Ce jeu connaît du succès bien au delà des cours d'écoles, comme en témoigne la très sérieuse *World RPS Society*.

Il y a peu d'intérêt à écrire un programme qui statistiquement ne perd pas, car il suffit pour cela de jouer au hasard (ce qui reste difficile pour un joueur humain). En revanche, on peut écrire un programme qui tente d'anticiper les comportements humains et qui, pour cette raison, sera efficace face à un adversaire humain. C'est l'objectif de ce projet et on se propose d'utiliser la programmation génétique pour cela.

## 1 Programmation génétique

L'idée de la programmation génétique consiste à appliquer à des programmes informatique le principe de la sélection naturelle. Une population de programmes existe, au sein de laquelle de nouveaux programmes sont obtenus par *croisements* ou *mutations* et où inversement des programmes disparaissent car jugés inefficaces.

Pour appliquer cette idée au jeu pierre-feuille-ciseaux, il faut définir ce qu'est un programme jouant à ce jeu. On adopte ici une notion très simple : un programme est un

ensemble de règles, chaque règle prenant la forme

$$C_1A_1 C_2A_2 \dots C_nA_n \rightarrow C$$

où les  $C_iA_i$  désignent un motif, c'est-à-dire les  $n$  derniers coups joués par le programme ( $C_i$ ) et son adversaire ( $A_i$ ). On dit qu'une règle s'applique si les  $n$  derniers coups de la partie sont effectivement égaux à  $C_iA_i$ ; le programme joue alors le coup suivant  $C$ . Ainsi la règle

PC FP  $\rightarrow$  C

dit exactement « si j'ai battu mon adversaire avec une pierre à l'avant-dernier coup puis avec une feuille au dernier coup, alors je joue les ciseaux ». Pour rendre la notion de règle plus souple, on ajoute dans les motifs un symbole  $_$  désignant n'importe lequel des trois objets. Ainsi, le programme suivant joue-t-il exactement ce que son adversaire a joué au coup précédent :

$_P \rightarrow P$   
 $_F \rightarrow F$   
 $_C \rightarrow C$

Un programme peut contenir des règles dont les motifs sont de longueurs différentes. Ainsi le programme

$\rightarrow P$   
 $P_ \rightarrow F$   
 $F_ \rightarrow C$   
 $C_ \rightarrow P$

joue *pierre* en premier lieu, puis alternativement *feuille*, *ciseaux*, *pierre*, etc. On convient que si plusieurs règles s'appliquent, on choisit parmi celles-ci une règle au hasard dont le motif est le plus long possible. Si aucune règle ne s'applique, on convient de jouer *pierre*.

Le principe de la programmation génétique peut alors s'appliquer de la manière suivante. Le croisement de deux programmes est simplement obtenu en concaténant les deux ensembles de règles. On se fixe un nombre maximal de règles et, lorsqu'il est dépassé, on se contente de supprimer certaines règles aléatoirement. Une mutation est obtenue en modifiant aléatoirement certains éléments d'un programme (par exemple 10% d'entre eux). Enfin, la sélection proprement dite peut être obtenue en faisant s'affronter entre eux tous les programmes, sur un très grand nombre de parties, tout en maintenant le score de chaque programme. On convient qu'un coup gagnant rapporte un point, qu'un coup nul ne rapporte rien et qu'un coup perdant fait perdre un point. Les opérations de sélection et de génération de nouveaux individus peuvent être répétées plusieurs fois.

## 2 Introduction du facteur humain

Pour rendre le processus intéressant, il faut introduire le facteur humain, c'est-à-dire un certain nombre de programmes qui correspondent à des stratégies effectivement utilisées par des joueurs humains. Pour cela, il suffit de faire jouer une personne contre l'un des programmes de la population et d'enregistrer les motifs utilisés par le joueur. Ainsi, si la partie est de la forme

PC FF FC PC CF FC CC PC FF

où les coups du joueur sont notés en premier, on peut décider de recueillir les motifs de longueur 3 utilisés par le joueur, qui sont donc les suivants :

PC FF FC -> P  
FF FC PC -> C  
FC PC CF -> F  
PC CF FC -> C  
CF FC CC -> P  
FC CC PC -> F

À cela, on ajoute les trois motifs correspondant aux trois premiers coups, c'est-à-dire

-> P  
PC -> F  
PC FF -> F

Ces neuf règles constituent un nouveau programme, qui est ajouté à la population. Avec suffisamment de tels programmes recueillis, le processus de sélection devient intéressant. On peut alors faire rejouer des humains contre les programmes de deuxième génération, et ainsi de suite.

### 3 Travail demandé

Le travail demandé se compose de deux tâches distinctes :

1. d'une part, un programme effectuant le processus de sélection proprement dite, faisant passer la population de programmes d'une génération à la suivante ;
2. d'autre part, une interface permettant à un utilisateur de jouer une ou plusieurs parties contre un adversaire choisi dans la population courante, et d'enregistrer les motifs utilisés par le joueur pour définir de nouveaux programmes.

Ces deux phases devront pouvoir être alternées plusieurs fois. Pour initier le processus, on pourra écrire « à la main » une population initiale de quelques programmes. Voici quelques idées simples : jouer le coup précédent de l'adversaire ; jouer un coup plus fort que son propre coup précédent ; jouer au contraire un coup plus faible ; etc.

L'objectif à atteindre est de surprendre le correcteur avec un programme jouant à pierre-feuille-ciseaux qui donne réellement l'illusion d'anticiper les coups de l'adversaire.

Un certain nombre de paramètres peuvent influencer le résultat. En voici quelques uns, avec des valeurs suggérées :

| paramètre | signification                                        | valeur |
|-----------|------------------------------------------------------|--------|
| $T$       | nombre de coups d'une partie                         | 9      |
| $R$       | nombre maximal de règles dans un programme           | 20     |
| $M$       | probabilité de modification dans une mutation        | 10%    |
| $P$       | longueur des motifs recueillis                       | 3      |
| $O$       | nombre d'oppositions pour chaque paire de programmes | 1000   |

On pourra en dégager d'autres, si cela est pertinent, et mener des expériences pour déterminer les valeurs donnant les meilleurs résultats.

## Références

- [1] Page de suivi du projet. <http://www.enseignement.polytechnique.fr/profs/informatique/Jean-Christophe.Filliatre/08-09/INF431/rps/>