2I013 Groupe 1

Projet Foot

Maxime Sangnier – Nicolas Baskiotis
maxime.sangnier@lip6.fr

2019

Laboratoire d'Informatique de Paris 6 (LIP6)
Sorbonne Université

Outline

Séance précédente

Bilan du TME

Vous avez :

- organisé votre code à l'aide des motifs de conception ;
- paramétré les stratégies (tir, défense, anticipation de la position de la balle. . .) ;
- optimisé les stratégies par recherche en grille.

Les motifs de conceptions les plus utiles :

- le décorateur, pour ajouter des fonctionnalités à un objet ;
- l'observateur, pour mettre en place des plans d'expériences.

Motifs structuraux : Decorator (très similaire à Proxy et Adaptor)

Comment ajouter des fonctionnalités de manière dynamique à un objet

Exemple : tirer au but

```
class Decorator:
      def __init__(self,state):
         self.state = state
      def __getattr__(self,attr):
         return getattr(self.state,attr)
class Shoot(Decorator):
      def __init__(self,state):
          Decorator.__init__(self,state)
      def shoot(self,p):
         return SoccerAction(Vector2D(...))
class Passe(Decorator):
      def __init__(self,state):
          Decorator.__init__(self,state)
      def passe(self,p):
         return SoccerAction(Vector2D(...))

mystate = Shoot(Passe(state))
```

Motifs de comportement : Observer

Simulation

- Une simulation possède une liste d'observateurs (`listeners`).
- A chaque événement marquant, tous les observateurs sont avertis par l'appel d'une fonction.
- Il est possible d'ajouter à la volée ou de supprimer des observateurs de la simulation.

Actions déclenchées lors d'une simulation

- `begin_match(team1,team2,state)` : au début de la simulation
- `end_match(team1,team2,state)` : à la fin
- `begin_round(team1,team2,state)` : au début de chaque engagement
- `end_round(team1,team2,state)` : à chaque but marqué
- `update_round(team1,team2,state)` : à chaque fin de tour

Tournoi 1v1

Équipe	Points	Matchs (gagnés, nuls, perdus)	Buts (marqués, encaissés)
Nasser's ␣ Team (Nasser)	76	(24,4,2)	(149,18)
Lyna et Mehdi (dulmo)	68	(21,5,4)	(131,27)
Nicolas / Téo (module_teo_nicolas)	67	(21,4,5)	(85,42)
pSG (P_S_G)	60	(18,6,6)	(138,70)
HUP's Team (ballon)	58	(18,4,8)	(135,36)
baker's Team (m_liste_fonction)	58	(17,7,6)	(94,36)
pinkertons (pinkertons)	50	(14,8,8)	(103,41)
Antonio Milla's Team (MillaModule)	46	(15,1,14)	(161,66)
KhadijaManel's Team (SocceriaMK)	46	(13,7,10)	(55,29)
Unknown (Strategy_f)	43	(14,1,15)	(93,68)
Strange's Team (moduleH)	36	(12,0,18)	(92,72)
equipe 3701195 (Pain_de_mie)	26	(8,2,20)	(119,119)
Philippe 's ␣ Team (BestModule)	22	(7,1,22)	(59,235)
Lucas's Team (MbappePresident)	19	(6,1,23)	(42,208)
Barbès Football Club (BarbesFC)	18	(6,0,24)	(33,212)
Fateam & Rania (rania_fatema)	1	(0,1,29)	(0,210)

Table 1 – Tournoi 1v1.

Équipe non-qualifiée

`ahilh projet-2i013` : dépôt **github** privé.

Tournoi 2v2

Équipe	Points	Matchs (gagnés, nuls, perdus)	Buts (marqués, encaissés)
Nasser's ␣ Team (Nasser)	78	(25,3,2)	(72,11)
Nicolas / Téo (module_teo_nicolas)	74	(24,2,4)	(122,10)
Lyna et Mehdi (dulmo)	72	(23,3,4)	(97,12)
Strange's Team (moduleH)	71	(22,5,3)	(94,13)
baker's Team (m_liste_fonction)	58	(18,4,8)	(80,29)
Antonio Milla's Team (MillaModule)	56	(18,2,10)	(105,62)
HUP's Team (ballon)	46	(15,1,14)	(66,113)
equipe 3701195 (Pain_de_mie)	45	(13,6,11)	(87,41)
pinkertons (pinkertons)	41	(12,5,13)	(64,54)
KhadijaManel's Team (SocceriaMK)	28	(6,10,14)	(25,63)
Philippe 's ␣ Team (BestModule)	28	(8,4,18)	(40,120)
Lucas's Team (MbappePresident)	26	(5,11,14)	(24,94)
pSG (P_S_G)	23	(6,5,19)	(34,86)
Fateam & Rania (rania_fatema)	17	(0,17,13)	(0,19)
Barbès Football Club (BarbesFC)	12	(3,3,24)	(14,94)
Unknown (Strategy_f)	3	(0,3,27)	(0,103)

Table 2 – Tournoi 2v2.

Équipe non-qualifiée

`ahilh projet-2i013` : dépôt **github** privé.

Optimisation d'une stratégie

Plan d'expérience

Définir :

1. une action à optimiser ;
2. un modèle ;
3. les paramètres du modèle à optimiser ;
4. un critère ;
5. les conditions environnementales.

Plan d'expérience

Définir :

1. une action à optimiser ;
2. un modèle ;
3. les paramètres du modèle à optimiser ;
4. un critère ;
5. les conditions environnementales.

Stratégie :

1. discrétisation et recherche en grille ;
2. s'il y a plus de trois paramètres ?
3. pas de recherche exhaustive possible $\Longrightarrow$ recherche aléatoire par cohorte.

Algorithmes génétiques

Problème :

$$\text{maximiser } f(x)$$
$$\text{pour } x \in \mathbb{R}^d,$$

Problème :

$$\text{maximiser } f(x)$$
$$\text{pour } x \in \mathbb{R}^d,$$

d trop grand pour une recherche exhaustive.

Les algorithmes génétiques

Problème :

$$\text{maximiser } f(x)$$
$$\text{pour } x \in \mathbb{R}^d,$$

d trop grand pour une recherche exhaustive.

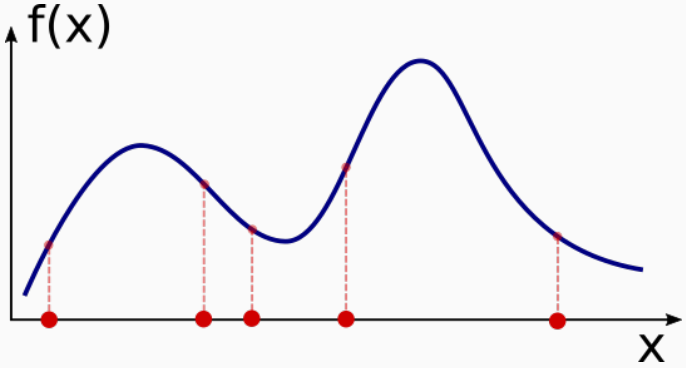
Algorithmes génétiques :

- le but est d'obtenir une solution approchée à un problème d'optimisation ;
- inspirés de la théorie de l'évolution ;
- utilisent la notion de sélection naturelle, appliquée à une population de solutions potentielles.

Les algorithmes génétiques

Idée générale :

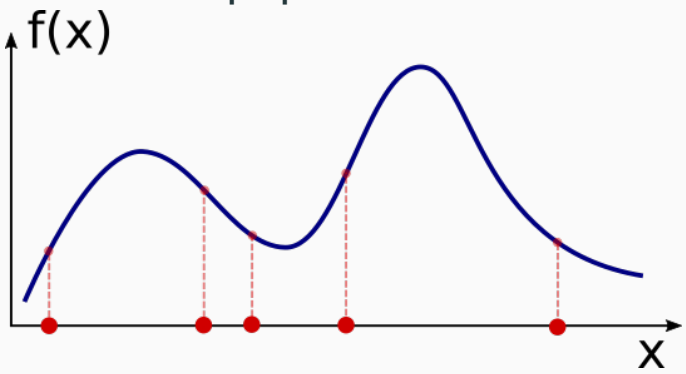
- ensemble de candidats $\{x_1, \ldots, x_n\}$ (aussi appelés individus ou chromosomes) ;
- chaque candidat est défini par un ensemble de propriétés. Ces propriétés sont appelés *gènes* (ils forment le *génotype*) ;
- les différentes valeurs que peuvent prendre les gènes sont les allèles ;
- les propriétés des candidats sont altérées dans le temps de sorte à faire évoluer la population vers le maximum de f ;

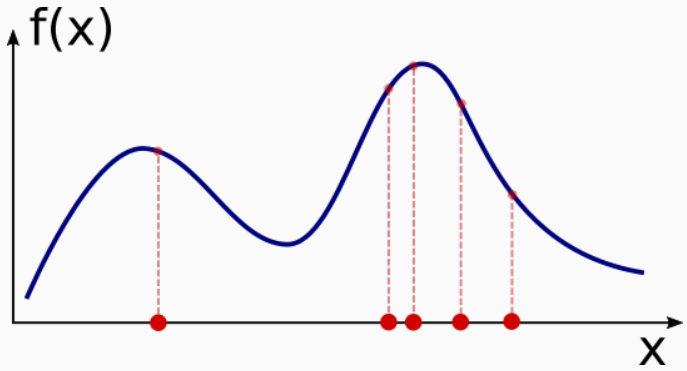


Les algorithmes génétiques

Idée générale :

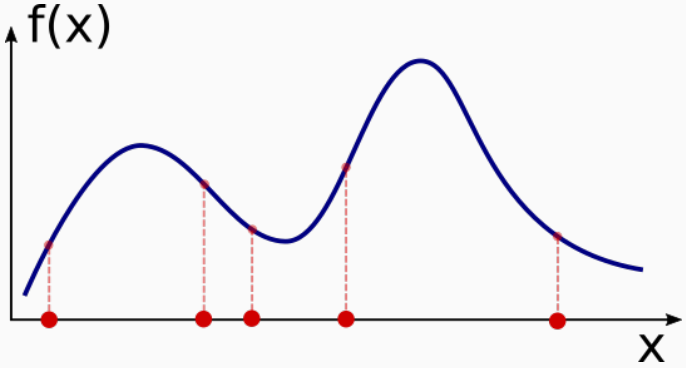
- ensemble de candidats $\{x_1, \ldots, x_n\}$ (aussi appelés individus ou chromosomes) ;
- chaque candidat est défini par un ensemble de propriétés. Ces propriétés sont appelés *gènes* (ils forment le *génotype*) ;
- les différentes valeurs que peuvent prendre les gènes sont les allèles ;
- les propriétés des candidats sont altérées dans le temps de sorte à faire évoluer la population vers le maximum de f ;

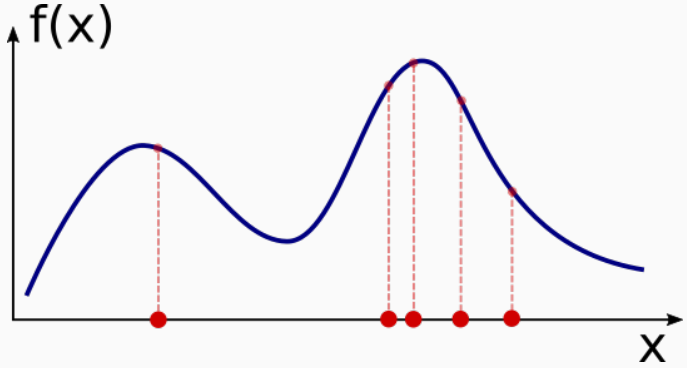




Idée générale :

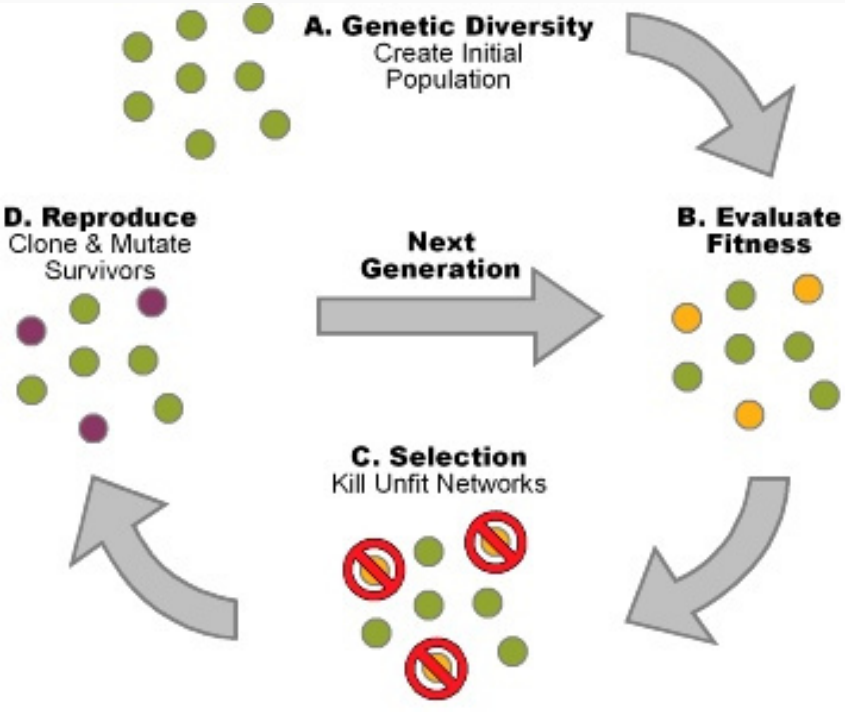
- ensemble de candidats $\{x_1, \ldots, x_n\}$ (aussi appelés individus ou chromosomes) ;
- chaque candidat est défini par un ensemble de propriétés. Ces propriétés sont appelés *gènes* (ils forment le *génotype*) ;
- les différentes valeurs que peuvent prendre les gènes sont les allèles ;
- les propriétés des candidats sont altérées dans le temps de sorte à faire évoluer la population vers le maximum de f ;





- la population à chaque itération est appelée génération.

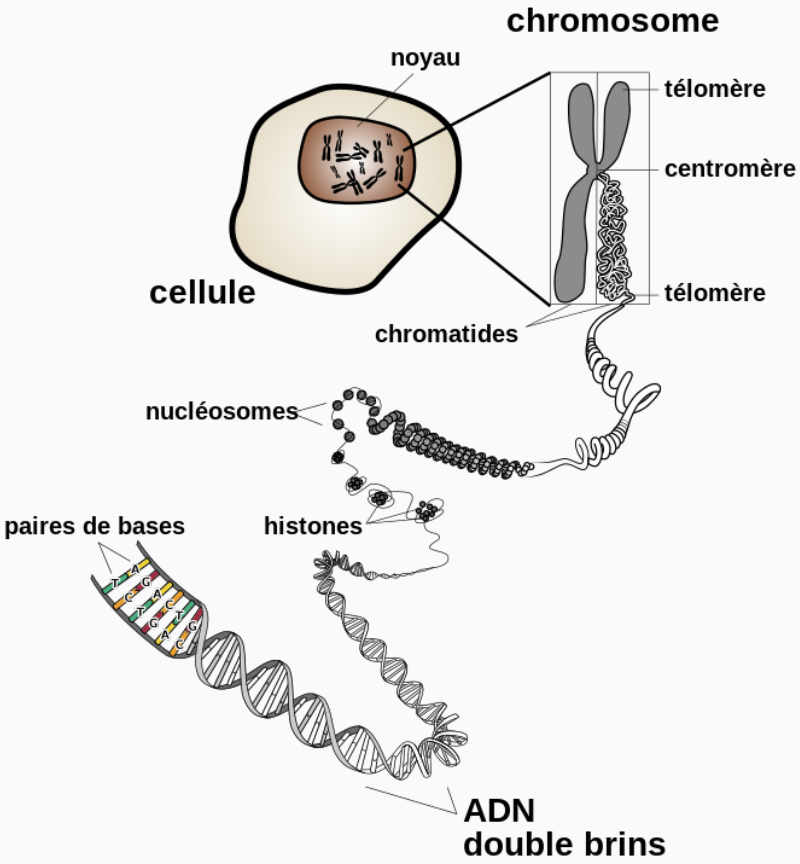
Procédure générale



A. Initialisation : génération aléatoire de candidats $\{x_1, \ldots, x_n\}$.
B. Évaluation : calcul de $\{f(x_1), \ldots, f(x_n)\}$.
C. Sélection : conservation des x_i les *plus prometteurs* (i.e. avec des valeurs $f(x_i)$ les plus grandes).
D. Reproduction : génération d'une nouvelle population (la future génération) à partir de celle sélectionnée.

Quelques notions de biologie génétique

- Les organismes vivants sont constitués de cellules, dont les noyaux comportent des chromosomes qui sont des chaînes d'ADN.



Quelques notions de biologie génétique

- Sur chacun de ces chromosomes, une suite de nucléotides constitue une chaîne qui code les fonctionnalités de l'organisme (la couleur des yeux par exemple).
- C'est le gène. Il correspond à une phrase fonctionnelle le long de la chaîne.

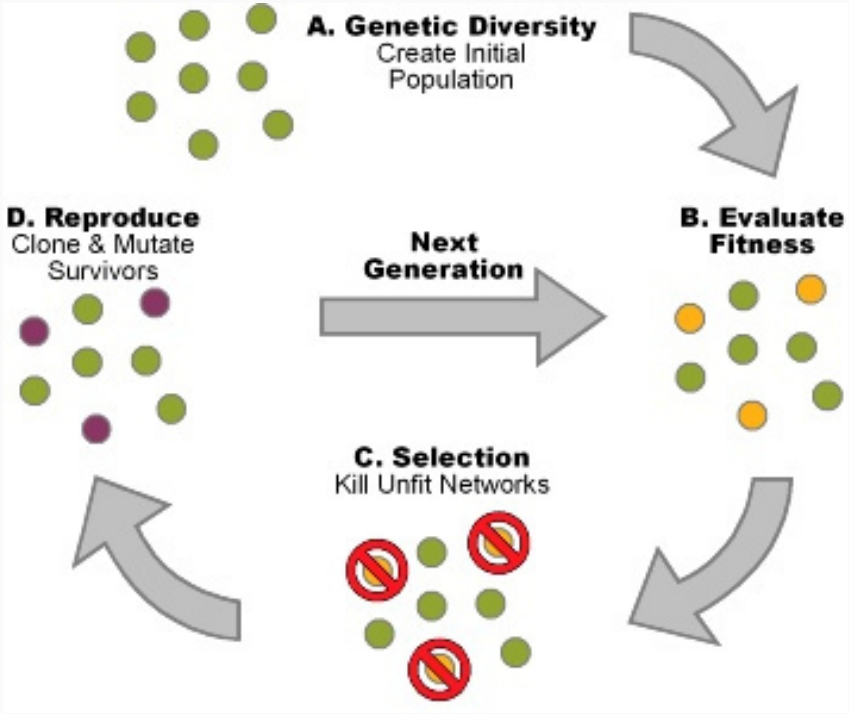
Quelques notions de biologie génétique

- Sur chacun de ces chromosomes, une suite de nucléotides constitue une chaîne qui code les fonctionnalités de l'organisme (la couleur des yeux par exemple).
- C'est le gène. Il correspond à une phrase fonctionnelle le long de la chaîne.

  - L'ensemble des gènes d'un individu est son génotype et l'ensemble du patrimoine génétique d'une espèce est le génome.
  - Les différentes versions d'un même gène sont appelées allèles.
  - Théorie de l'évolution : au fil du temps, les gènes conservés au sein d'une population donnée sont ceux qui sont le plus adaptés aux besoins de l'espèce vis-à-vis de son environnement.

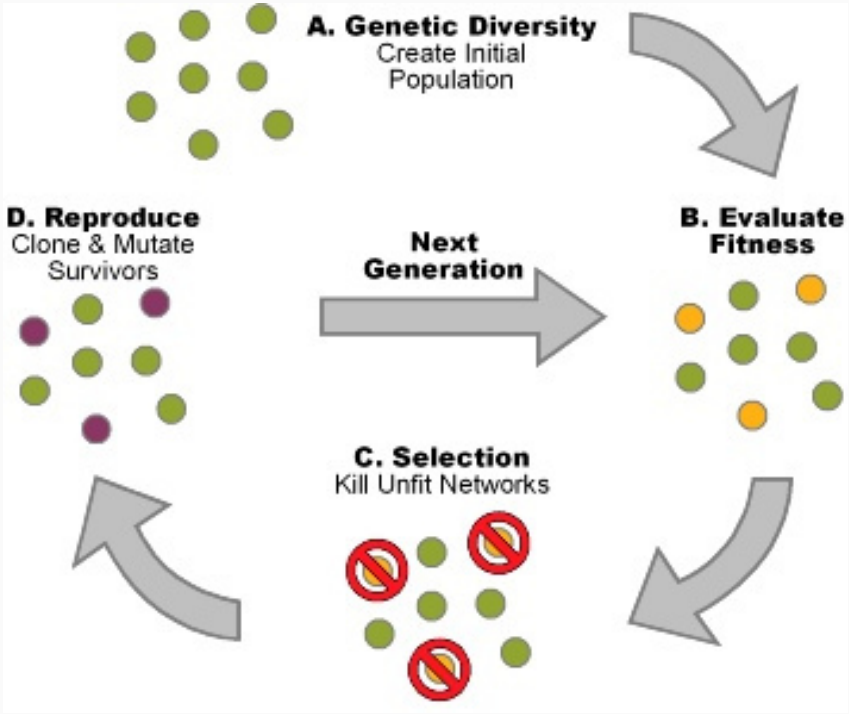
Le brassage génétique

- Le brassage génétique est l'ensemble de processus importants au sein d'un groupe d'organismes d'une même espèce, intervenant lors de la phase de reproduction (lorsque les chromosomes de deux organismes fusionnent en créant un nouvel organisme).



Le brassage génétique

- Le brassage génétique est l'ensemble de processus importants au sein d'un groupe d'organismes d'une même espèce, intervenant lors de la phase de reproduction (lorsque les chromosomes de deux organismes fusionnent en créant un nouvel organisme).



- Ces processus sont imités par les algorithmes génétiques afin de faire évoluer les populations de solutions de manière progressive.

Le brassage génétique

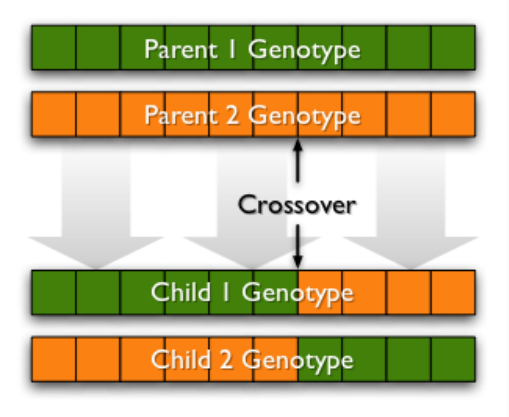
Avant le reproduction :

- la sélection déterminer quels individus sont plus enclins à obtenir les meilleurs résultats. Ce processus est analogue à un processus de sélection naturelle (les individus les plus adaptés gagnent la compétition de la reproduction tandis que les moins adaptés meurent avant la reproduction, ce qui améliore globalement l'adaptation).

Le brassage génétique

Pendant le reproduction :

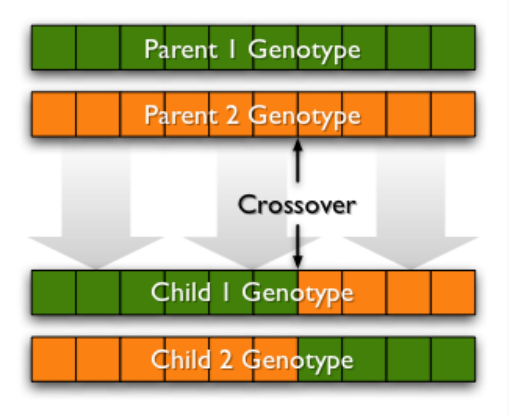
1. l'enjambement (aussi appelé croisement ou recombinaison) consiste pour deux chromosomes à s'échanger des parties de leurs chaînes, pour donner de nouveaux chromosomes. Ces enjambements peuvent être simples ou multiples.



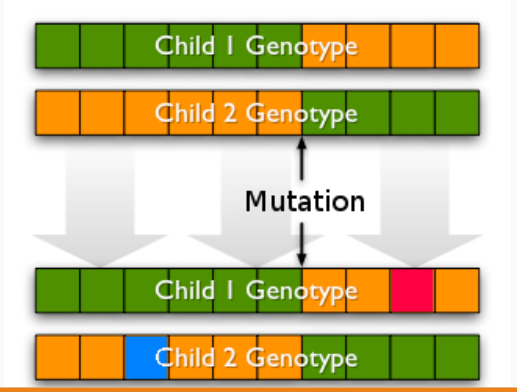
Le brassage génétique

Pendant le reproduction :

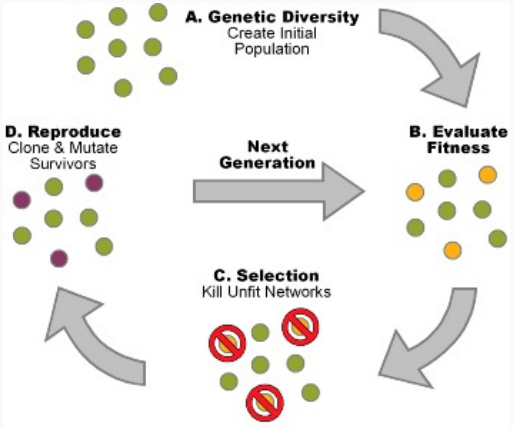
1. l'enjambement (aussi appelé croisement ou recombinaison) consiste pour deux chromosomes à s'échanger des parties de leurs chaînes, pour donner de nouveaux chromosomes. Ces enjambements peuvent être simples ou multiples.



   Deux parents engendrent deux enfants, conservant le *patrimoine génétique* du couple (en pratique, ils échangent des gènes).
2. les mutations altèrent aléatoirement un gène des nouveaux chromosomes.



Procédure générale



A. Initialisation : génération aléatoire d'une population candidats $\{x_1, \ldots, x_n\}$.

B. Évaluation : calcul de $\{f(x_1), \ldots, f(x_n)\}$.

C. Sélection : conservation des 25 à 50% des x_i avec $f(x_i)$ les plus grands.

D. Reproduction : création de la nouvelle génération.

   1. Croisement réalisé avec une probabilité de 70% (emplacement de l'enjambement aléatoire).
   2. Mutation des chromosomes enfants avec une probabilité de 0.1 à 1% (gène choisi aléatoirement).

Les représentations génétiques

Différents codages des chromosomes :

- binaire, par exemple :

$$x_i = 110100101011. \tag{1}$$

  Chaque gène peut prendre comme valeurs 0 ou 1.

- alphabétique, par exemple :

$$x_i = 214030187942. \tag{2}$$

  Chaque gène peut prendre comme valeurs 0, 1, . . ., 9.

- sous forme d'arbres.
- vectoriel, par exemple :

$$x_i = \boxed{9.3 \mid 4.5 \mid 3.2 \mid -0.1 \mid 102. \mid -58.4 \mid -8.7}. \tag{3}$$

Implémentation

Exemple de code

```
from sklearn.model_selection import ParameterGrid

def begin_match(self, team1, team2, state):
    self.last_step = 0  # Step of the last round
    self.criterion = 0  # Criterion to maximize (here, number of goals)
    self.cpt_trials = 0  # Counter for trials
    # Set of chromosomes
    # self.param_grid = iter(ParameterGrid(self.params))
    self.cur_param = next(self.param_grid, None)  # Current parameter
    if self.cur_param is None:
        raise ValueError('no parameter given.')
    self.res = dict()  # Dictionary of results
```

Exemple de code

```
def end_round(self, team1, team2, state):
    # A round ends when there is a goal of if max step is achieved
    if state.goal > 0:
        self.criterion += 1  # Increment criterion

    self.cpt_trials += 1  # Increment number of trials

    print(self.cur_param, end="␣␣␣␣")
    print("Crit:␣{}␣␣␣Cpt:␣{}".format(self.criterion, self.cpt_trials))

    if self.cpt_trials >= self.trials:
        # Save the result
        self.res[tuple(self.cur_param.items())] = self.criterion * 1. / sel

        # Reset parameters
        self.criterion = 0
        self.cpt_trials = 0

        # Next parameter value
        self.cur_param = next(self.param_grid, None)
        if self.cur_param is None:
            # Update the set of chromosomes
            # self.simu.end_match()
```

Passe à deux joueurs

Passe à deux joueurs

De quoi a-t-on besoin ?

Passe à deux joueurs

De quoi a-t-on besoin ?

- Chaque joueur **voit** qui a la balle.
- Chaque joueur **sait** qui a eu la balle en dernier.
- Si mon co-équipier a la balle : position d'attente. S'il l'envoie : je vais la récupérer.

Passe à deux joueurs

De quoi a-t-on besoin ?

- Chaque joueur **voit** qui a la balle.
- Chaque joueur **sait** qui a eu la balle en dernier.
- Si mon co-équipier a la balle : position d'attente. S'il l'envoie : je vais la récupérer.

Implémentation

- Enregistrer le dernier porteur du ballon.
- Grâce à un attribut de la classe `Stratety`.
- Chaque joueur **connaît** le dernier porteur du ballon.

Objectifs du TME

Objectifs du TME

- paramétrer les stratégies (tir, défense, anticipation de la position de la balle...) ;
- optimiser les stratégies grâce à un algorithme génétique ;
- implémenter des stratégies pour le 2v2 (à terme le 4v4).

A chaque TME

Mettre à jour le dépôt contenant le simulateur :

```
cd [DOSSIER DU SIMULATEUR]
git pull
```