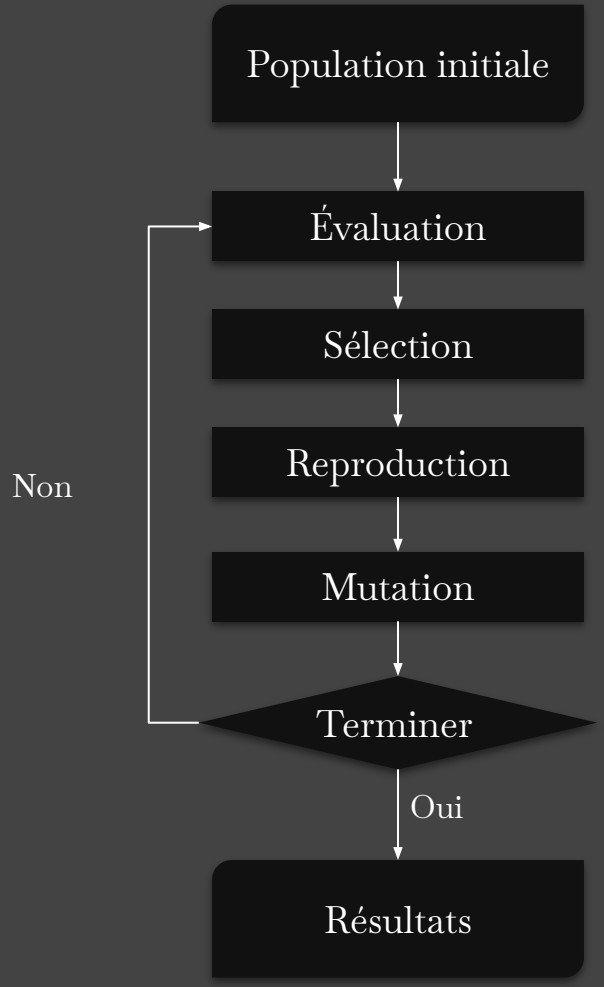
ALGORITHME GÉNÉTIQUE (TETRIS)


Population initiale
Évaluation
Sélection
Reproduction
Mutation
Terminer
Non
Oui
Résultats

ÉVALUATION

Tous les individus de la population initiale sont évalués (ils jouent à Tetris jusqu'au *Game Over* ou jusqu'à un nombre d'itérations définies (*e.g. 500*)

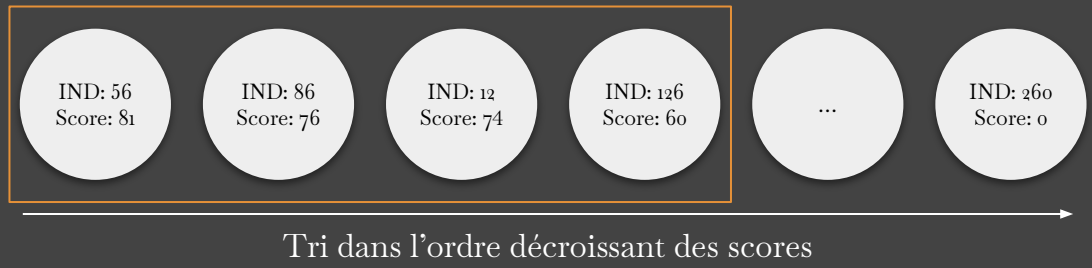
SÉLECTION

La sélection permet de trier les individus éligibles à la phase de reproduction

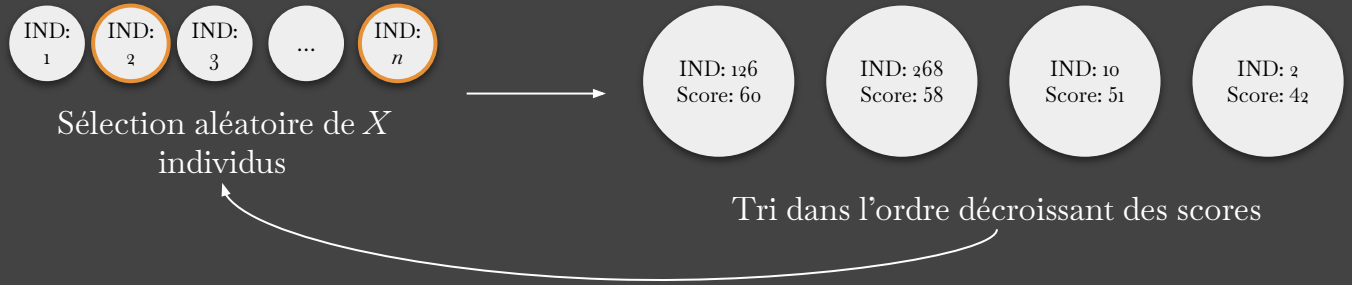
Il existe plusieurs stratégies de sélection. La plus simple est la sélection *Greedy* dans laquelle les *X* meilleurs individus sont sélectionnés. Toutefois, il est important de conserver l'aspect exploration de l'algorithme et la constante sélection des *X* meilleurs peut contribuer à l'*overfitting* des résultats.

SÉLECTION

Greedy



Tournament



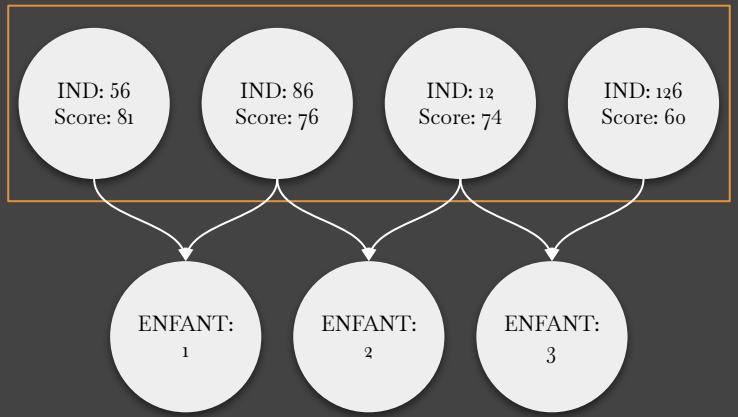
REPRODUCTION

La reproduction permet de créer de nouveaux individus pour la prochaine génération qui viendront prendre la place des individus les moins performants au sein de la population

Il existe plusieurs manières de générer de nouveaux individus et elles dépendent essentiellement de la fonction à optimiser

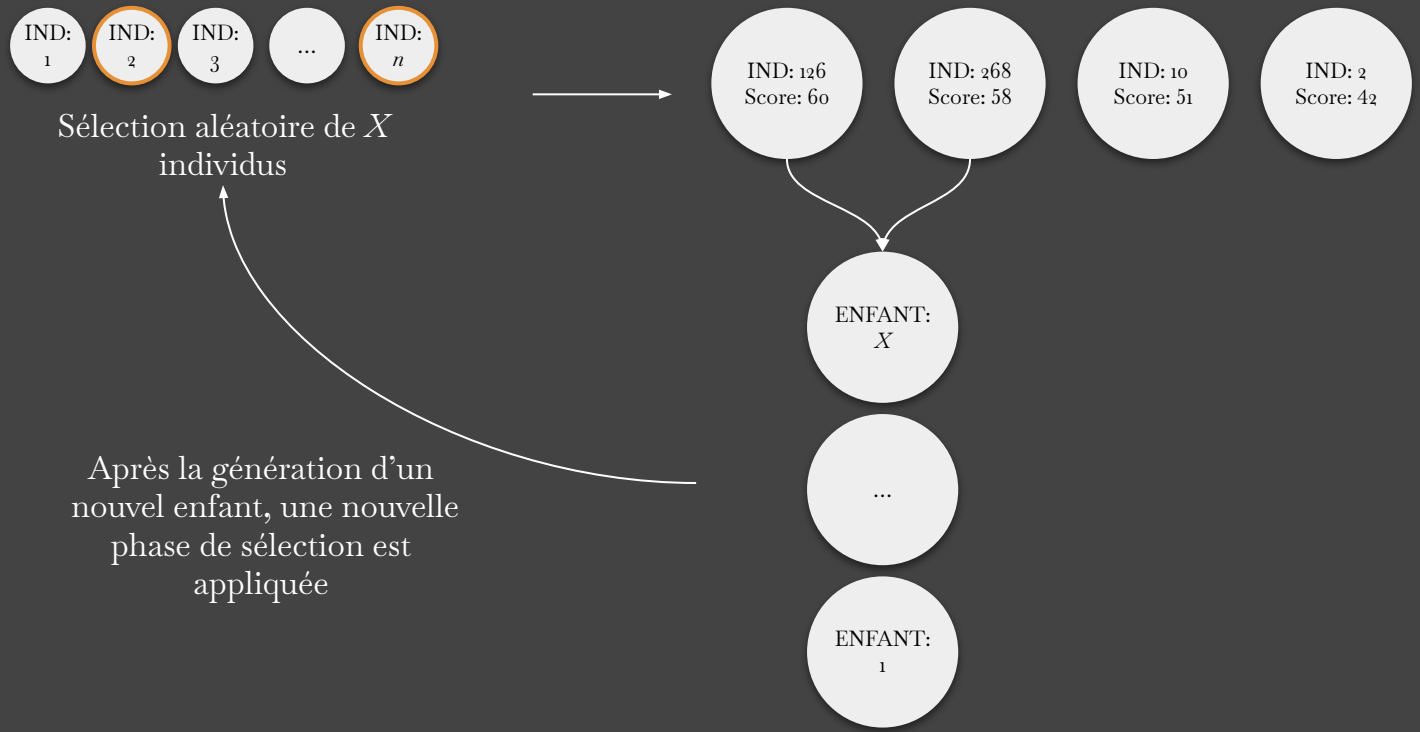
REPRODUCTION

Greedy



REPRODUCTION

Tournament



REPRODUCTION

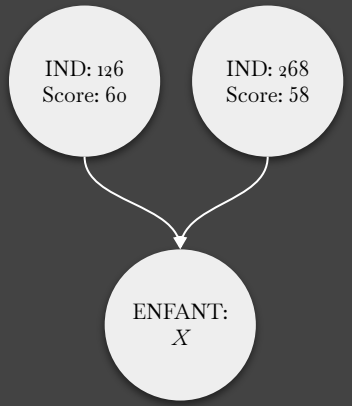
La problématique d'optimisation des emplacements des tetrominos peut être modélisée à travers l'optimisation de 4 principaux paramètres

- Minimiser la hauteur maximum (hauteur de la plus haute colonne), H
- Maximiser le nombre de lignes complètes, N
- Minimiser le nombre de trous, T
- Minimiser les variations de hauteur entre les colonnes, V

$$score = wH + xN + yT + zV$$

REPRODUCTION

Moyenne pondérée (par les scores) des constantes



$$score_{126} = w_{126}H + x_{126}N + y_{126}T + z_{126}V \qquad score_{268} = w_{268}H + x_{268}N + y_{268}T + z_{268}V$$

$$w_{ENFANT} = (w_{126}Score_{126} + w_{268}Score_{268}) / (Score_{126} + Score_{268})$$

$$x_{ENFANT} = (x_{126}Score_{126} + x_{268}Score_{268}) / (Score_{126} + Score_{268})$$

$$y_{ENFANT} = (y_{126}Score_{126} + y_{268}Score_{268}) / (Score_{126} + Score_{268})$$

$$z_{ENFANT} = (z_{126}Score_{126} + z_{268}Score_{268}) / (Score_{126} + Score_{268})$$

MUTATION

Les mutations permettent de minimiser les chances de rester bloquer sur des optimums locaux et ainsi permettre à l'algorithme de couvrir une plus grande partie de l'espace des possibilités

Dans l'exemple de l'équation précédente, chaque constante à une chance d'être modifiée suite à la reproduction

$$score = wH + xN + yT + zV$$

$$x_{NEW} = x + ALEA(-0.1, 0.1)$$

$$score = wH + x_{NEW}N + yT + zV$$

EXEMPLES DE VARIABLES

POPULATION_MAX = 200 (nombre d'individus au sein de la population)
MAX_ITER = 500 (nombre d'itérations maximum s'il n'y a pas de *Game Over*)
MAX_GENERATION = 50 (nombre de générations avant de présenter les résultats finaux).
NB_ENFANTS = 0.30 (30% des individus de la population vont être remplacés par de nouveaux enfants)
TAUX_MUTATION = 0.05 (probabilité qu'une constante mute)

Pour la stratégie *Tournament*:

SELECTION_POPULATION = 0.10 (10% des individus de la population sont sélectionnés aléatoirement lors de chaque phase de sélection)