MIASC 5 : Modèles du Vivant

Adaptation et systèmes évolutifs

Damien.Olivier@univ-lehavre.fr

Université du Havre

December 14, 2004

MIASC 5 : Modèles du Vivant

Adaptation et systèmes évolutifs

Damien.Olivier@univ-lehavre.fr

Université du Havre

December 14, 2004

Autour des systèmes complexes

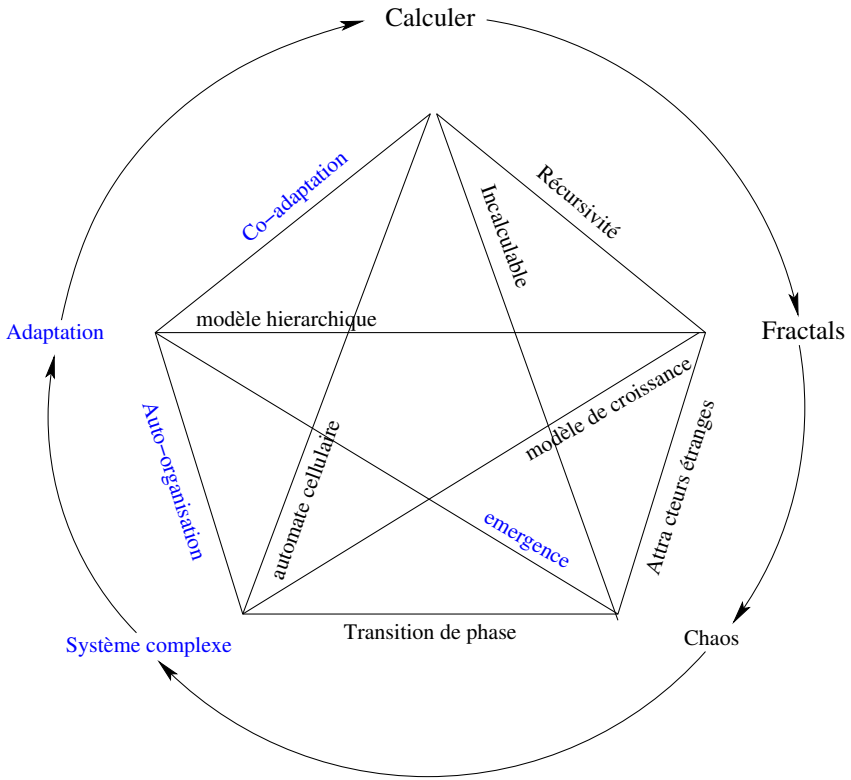


Figure: Cartes des concepts

Adaptation

Définition

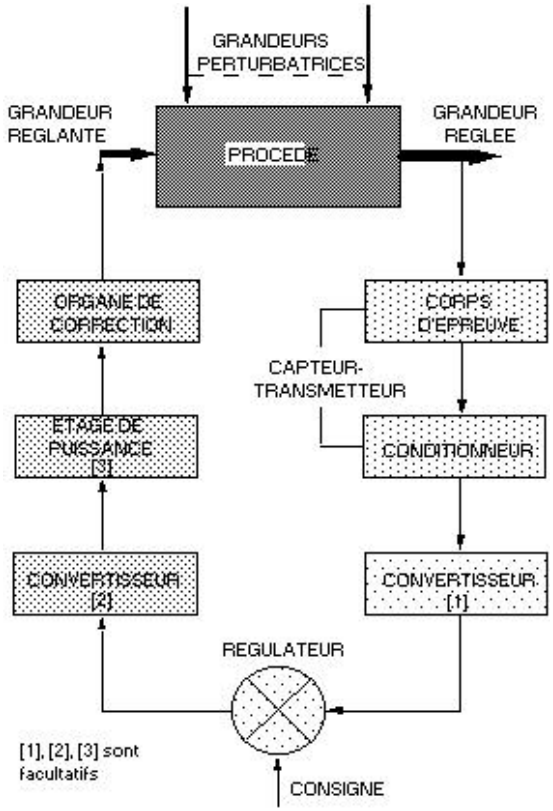
Un système adaptatif s'adapte aux changements de son environnement par des modifications internes. C'est un processus de feedback (rétroaction).

Adaptation

Définition

Un système adaptatif s'adapte aux changements de son environnement par des modifications internes. C'est un processus de feedback (rétroaction).

Exemple : Régulation



IA numérique

Par opposition à l'IA symbolique

- IA Symbolique, représentation symbolique des connaissances, souvent basée sur de la logique ;

IA numérique

Par opposition à l'IA symbolique

- IA Symbolique, représentation symbolique des connaissances, souvent basée sur de la logique ;
- IA numérique, le vivant reste le modèle mais c'est souvent réactif.

IA numérique

Par opposition à l'IA symbolique

- IA Symbolique, représentation symbolique des connaissances, souvent basée sur de la logique ;
- IA numérique, le vivant reste le modèle mais c'est souvent réactif.
  - Perception, fonctionnement réflexe . . .

IA numérique

Par opposition à l'IA symbolique

- IA Symbolique, représentation symbolique des connaissances, souvent basée sur de la logique ;
- IA numérique, le vivant reste le modèle mais c'est souvent réactif.
  - Perception, fonctionnement réflexe . . .
  - Adaptation sorte d'apprentissage sans formalisation et raisonnement.

La métaphore biologique

S'inspire du Néo-Darwinisme, synthèse de la théorie de la sélection naturelle (Darwin) et de la théorie de l'hérédité (Mendel)

Définition

$$adaptation = variation + hérédité + sélection$$

La métaphore biologique

S'inspire du Néo-Darwinisme, synthèse de la théorie de la sélection naturelle (Darwin) et de la théorie de l'hérédité (Mendel)

Définition

$$adaptation = variation + hérédité + sélection$$

- *Variation* : Différence entre individus ;

La métaphore biologique

S'inspire du Néo-Darwinisme, synthèse de la théorie de la sélection naturelle (Darwin) et de la théorie de l'hérédité (Mendel)

Définition

$$adaptation = variation + hérédité + sélection$$

- *Variation* : Différence entre individus ;
- *Hérédité* : Persistance temporelle ;

La métaphore biologique

S'inspire du Néo-Darwinisme, synthèse de la théorie de la sélection naturelle (Darwin) et de la théorie de l'hérédité (Mendel)

Définition

$$adaptation = variation + hérédité + sélection$$

- *Variation* : Différence entre individus ;
- *Hérédité* : Persistance temporelle ;
- *Sélection naturelle* : Avantage à ceux qui sont "le plus adapté" dans le mécanisme de reproduction.

La métaphore biologique

Une des caractéristiques du vivant : les individus n'ont pas été "programmés" pour répondre à un problème spécifique, mais ils changent en s'adaptant à leur environnement. Comment ?

La métaphore biologique

Une des caractéristiques du vivant : les individus n'ont pas été "programmés" pour répondre à un problème spécifique, mais ils changent en s'adaptant à leur environnement. Comment ?

- Évolution : résultat d'une altération progressive des êtres vivants au cours des générations ;

La métaphore biologique

Une des caractéristiques du vivant : les individus n'ont pas été "programmés" pour répondre à un problème spécifique, mais ils changent en s'adaptant à leur environnement. Comment ?

- Évolution : résultat d'une altération progressive des êtres vivants au cours des générations ;
- Reproduction basée sur le caractère génétique qui subit au cours des générations des recombinaisons et des mutations ;

La métaphore biologique

Une des caractéristiques du vivant : les individus n'ont pas été "programmés" pour répondre à un problème spécifique, mais ils changent en s'adaptant à leur environnement. Comment ?

- Évolution : résultat d'une altération progressive des êtres vivants au cours des générations ;
- Reproduction basée sur le caractère génétique qui subit au cours des générations des recombinaisons et des mutations ;
- Mécanisme de sélection naturelle.

La métaphore biologique

Une des caractéristiques du vivant : les individus n'ont pas été "programmés" pour répondre à un problème spécifique, mais ils changent en s'adaptant à leur environnement. Comment ?

- Évolution : résultat d'une altération progressive des êtres vivants au cours des générations ;
- Reproduction basée sur le caractère génétique qui subit au cours des générations des recombinaisons et des mutations ;
- Mécanisme de sélection naturelle.

Objectif

construire des systèmes informatiques sur la base de ce modèle.

Principe

La propriété d'adaptation des individus s'interprète comme une recherche d'optimum d'une fonction (la survie).

Principe

On génère initialement un grand nombre d'individus (ou solutions) et l'algorithme les fait évoluer en 3 phases.

- Reproduction
- Croisement
- Mutation

Sélection

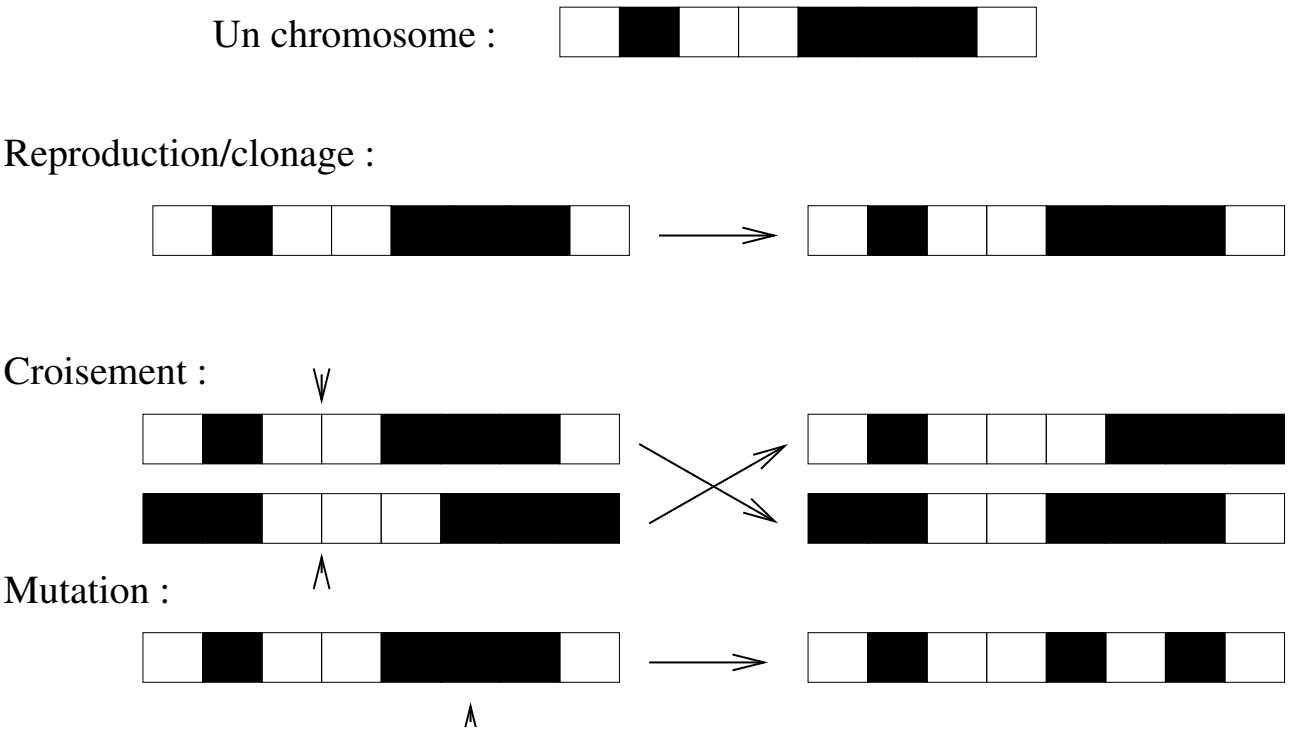
- Le processus de sélection est basé sur une fonction d'évaluation (ou fonction objectif) qui correspond à une *performance* de l'individu.
- On en déduit une probabilité pour chaque individu de se reproduire ou de générer des clones (cette probabilité s'appelle *fitness*).
- Finalement, ce processus contribue à produire une population ayant une meilleure adaptabilité (convergence vers l'optimum).

Formalisation

- Les individus sont représentés par des *chromosomes* constitués *d'allèles* (chromosome = chaîne d'informations sur un alphabet fini).

Formalisation

- Les individus sont représentés par des *chromosomes* constitués *d'allèles* (chromosome = chaîne d'informations sur un alphabet fini).
-



Algorithme

Initialiser la population
Calculer le degré d'adaptation de chaque individu
Tant que (**non** fini **ou non** convergence) **faire**
    Reproduction des individus
    Appliquer les opérateurs génétiques
    Sélectionner les survivants parmi les parents et les enfants
    Calculer le degré d'adaptation de chaque individu
Fait
Conclure

Nombreuses variantes

Exemple de base

- Problème : On recherche à optimiser (le maximum) la fonction $f(x) = x^2$ sur l'intervalle [0..31]

Exemple de base

- Problème : On recherche à optimiser (le maximum) la fonction $f(x) = x^2$ sur l'intervalle [0..31]
- Population totale possible : nombre compris entre 0 et 31 en codage binaire sur 5 bits (00000 à 11111).

Exemple de base

- Problème : On recherche à optimiser (le maximum) la fonction $f(x) = x^2$ sur l'intervalle [0..31]
- Population totale possible : nombre compris entre 0 et 31 en codage binaire sur 5 bits (00000 à 11111).
- Population initiale constituée de 4 individus choisis arbitrairement :

13	de	code	01101
24	de	code	11000
8	de	code	01000
19	de	code	10011

Exemple de base

- Problème : On recherche à optimiser (le maximum) la fonction $f(x) = x^2$ sur l'intervalle [0..31]
- Population totale possible : nombre compris entre 0 et 31 en codage binaire sur 5 bits (00000 à 11111).
- Population initiale constituée de 4 individus choisis arbitrairement :

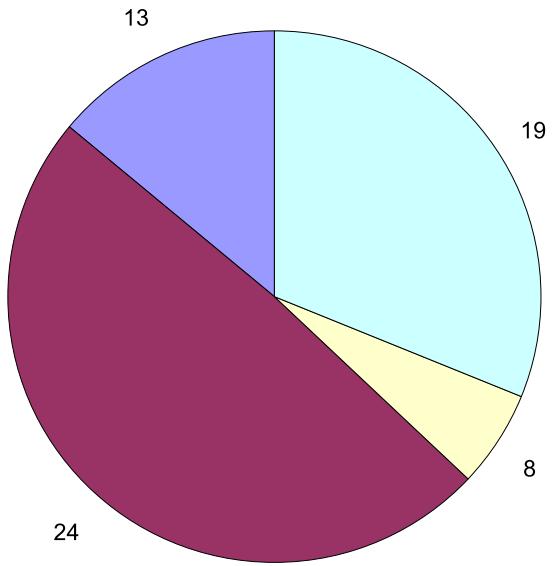
13	de	code	01101
24	de	code	11000
8	de	code	01000
19	de	code	10011

- Fonction de performance pour chaque individu : correspond à $f(x) = x^2$. On en déduit la fonction fitness qui est une répartition par pourcentage de ces performances.

Reproduction

Individu	perf.	fitness
13	169	0,14
24	576	0,49
8	64	0,06
19	361	0,31

Roulette biaisée



On tire au hasard 4 nouvelles chaînes parmi les existantes en tenant compte de la valeur de répartition. On obtient, par exemple : 13, 24, 24 et 19.

Croisement

On prend les individus 2 par 2. On coupe les chromosomes à une position aléatoire et on croise les parties coupées

$$\begin{cases} 13 : & 0110 \| 1 \\ 24 : & 1100 \| 0 \end{cases} \quad \begin{matrix} \searrow & \nearrow \\ \nearrow & \searrow \end{matrix} \quad \begin{matrix} 01100 & : 12 \\ 11001 & : 21 \end{matrix}$$

$$\begin{cases} 24 : & 11 \| 000 \\ 19 : & 10 \| 011 \end{cases} \quad \begin{matrix} \searrow & \nearrow \\ \nearrow & \searrow \end{matrix} \quad \begin{matrix} 11011 & : 27 \\ 10000 & : 16 \end{matrix}$$

Mutation

On opère par tirage aléatoire à partir d'une probabilité de mutation définie initialement. On prend ici 0,05 donc sur les 20 bits des 4 chromosomes, on suppose qu'un seul allèle a été sélectionné et mute.

$$12 : \quad 01\underline{1}00 \quad \rightarrow \quad 01000 \quad : 8$$

Nouvelle génération

8, 21, 27, 16.
Si on somme les performances, on obtient 1490 au lieu de 1170 à la génération précédente

Mise en œuvre

- Codage compatible avec les mécanismes génétiques ;

Mise en œuvre

- Codage compatible avec les mécanismes génétiques ;
- Comment réaliser la sélection sans tomber une démarche élitiste stérilisante ? Perte de la diversité biologique

Mise en œuvre

- Codage compatible avec les mécanismes génétiques ;
- Comment réaliser la sélection sans tomber une démarche élitiste stérilisante ? Perte de la diversité biologique
  - Les populations suivantes deviennent de plus en plus homogènes $\Rightarrow$ fragilité au changement

Mise en œuvre

- Codage compatible avec les mécanismes génétiques ;
- Comment réaliser la sélection sans tomber une démarche élitiste stérilisante ? Perte de la diversité biologique
  - Les populations suivantes deviennent de plus en plus homogènes $\Rightarrow$ fragilité au changement
  - Évolution d'une population = Évolution d'un individu !!!

Mise en œuvre

- Codage compatible avec les mécanismes génétiques ;
- Comment réaliser la sélection sans tomber une démarche élitiste stérilisante ? Perte de la diversité biologique
  - Les populations suivantes deviennent de plus en plus homogènes $\Rightarrow$ fragilité au changement
  - Évolution d'une population = Évolution d'un individu !!!
  - Découverte du plus proche optimum puis enlisement ;

Mise en œuvre

- Codage compatible avec les mécanismes génétiques ;
- Comment réaliser la sélection sans tomber une démarche élitiste stérilisante ? Perte de la diversité biologique
  - Les populations suivantes deviennent de plus en plus homogènes $\Rightarrow$ fragilité au changement
  - Évolution d'une population = Évolution d'un individu !!!
  - Découverte du plus proche optimum puis enlisement ;
  - Dans la pratique une population ne se rediversifie pas.

Mise en œuvre

- Codage compatible avec les mécanismes génétiques ;
- Comment réaliser la sélection sans tomber une démarche élitiste stérilisante ? Perte de la diversité biologique
  - Les populations suivantes deviennent de plus en plus homogènes $\Rightarrow$ fragilité au changement
  - Évolution d'une population = Évolution d'un individu !!!
  - Découverte du plus proche optimum puis enlisement ;
  - Dans la pratique une population ne se rediversifie pas.
- Quelques pistes

Mise en œuvre

- Codage compatible avec les mécanismes génétiques ;
- Comment réaliser la sélection sans tomber une démarche élitiste stérilisante ? Perte de la diversité biologique
  - Les populations suivantes deviennent de plus en plus homogènes $\Rightarrow$ fragilité au changement
  - Évolution d'une population = Évolution d'un individu !!!
  - Découverte du plus proche optimum puis enlisement ;
  - Dans la pratique une population ne se rediversifie pas.
- Quelques pistes
  - Roulette biaisée, les chromosomes les plus "aptes" dominent ;

Mise en œuvre

- Codage compatible avec les mécanismes génétiques ;
- Comment réaliser la sélection sans tomber une démarche élitiste stérilisante ? Perte de la diversité biologique
  - Les populations suivantes deviennent de plus en plus homogènes $\Rightarrow$ fragilité au changement
  - Évolution d'une population = Évolution d'un individu !!!
  - Découverte du plus proche optimum puis enlisement ;
  - Dans la pratique une population ne se rediversifie pas.
- Quelques pistes
  - Roulette biaisée, les chromosomes les plus "aptes" dominent ;
  - Sélection suivant le rang, probabilité proportionnelle au rang, les moins aptes ont une chance ;

Réglages ?????

- Longueur des chaînes ;
- Nombre de génération ;
- Taille de la population ;
- Probabilité de mutation/d'hybridation.

Définition(s)

Programmation automatique

- Génération automatique de programmes (J. Koza)

Définition(s)

Programmation automatique

- Génération automatique de programmes (J. Koza)
- Génération automatique de comportements représentés par des programmes exécutables (P. Angeline).

Définition(s)

Programmation automatique

- Génération automatique de programmes (J. Koza)
- Génération automatique de comportements représentés par des programmes exécutables (P. Angeline).

Approche évolutionniste

Une majorité de travaux s'inspirent de la philosophie des algos génétiques $\Rightarrow$ **Programmation génétique**.

Le modèle de Koza

- Programmes structurés en s-expressions ;

Le modèle de Koza

- Programmes structurés en s-expressions ;
- Définition d'un ensemble de fonctions primitives et de terminaux, seuls constituants autorisés des expressions.

Le modèle de Koza

- Programmes structurés en s-expressions ;
- Définition d'un ensemble de fonctions primitives et de terminaux, seuls constituants autorisés des expressions.
- Type de retour unique pour toutes les expressions.

Le modèle de Koza

- Programmes structurés en s-expressions ;
- Définition d'un ensemble de fonctions primitives et de terminaux, seuls constituants autorisés des expressions.
- Type de retour unique pour toutes les expressions.

Constat

Programmation génétique démarche très voisine des algorithmes génétiques
La donnée est le programme

s-expression ou arbre

Une s-expresion :

- Issue du LISP ;

s-expression ou arbre

Une s-expresion :

- Issue du LISP ;
- Représentée par une liste ;

s-expression ou arbre

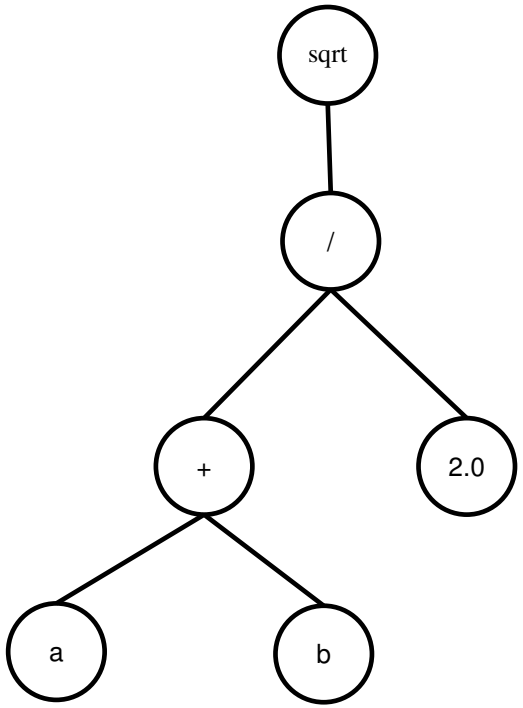
Une s-expresion :

- Issue du LISP ;
- Représentée par une liste ;
- Notation préfixée ;

s-expression ou arbre

Une s-expresion :

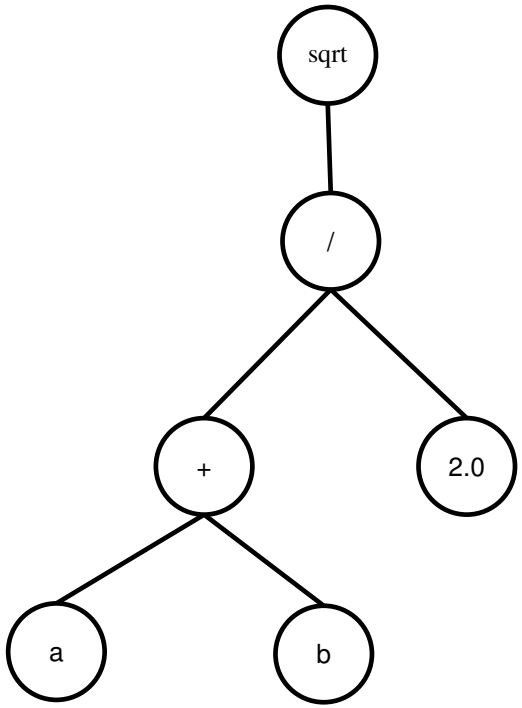
- Issue du LISP ;
- Représentée par une liste ;
- Notation préfixée ;
- Ex : (sqrt (/ (+ a b) 2.0))



s-expression ou arbre

Une s-expresion :

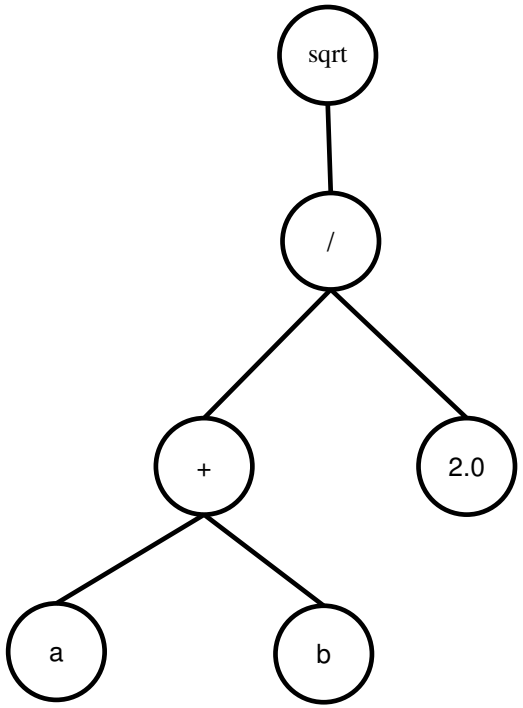
- Issue du LISP ;
- Représentée par une liste ;
- Notation préfixée ;
- Ex : (sqrt (/ (+ a b) 2.0))
- Implantation possible dans d'autre langage (C++, C, Java ...).



s-expression ou arbre

Une s-expresion :

- Issue du LISP ;
- Représentée par une liste ;
- Notation préfixée ;
- Ex : (sqrt (/ (+ a b) 2.0))
- Implantation possible dans d'autre langage (C++, C, Java ...).
- Programme = Arbre syntaxique



Les mots du langage

- Les terminaux :
  - Ce sont les feuilles de l'arbre syntaxique ;
  - Pseudo-variables contenant les entrées du programme et variables ordinaires ;
  - Constantes, fixées d'après la connaissance préliminaire du problème, ou générées aléatoirement (random ephemeral constants) ;
  - Fonctions sans arguments mais avec effets de bord ;

Les mots du langage

- Les fonctions et les opérateurs :
  - Nœuds internes de l'arbre ;
  - Exemple : fonctions booléennes, arithmétiques, transcendantales, à effet de bord (assignation de variable, déplacement d'un robot, ...), fonctions implantant des structures de contrôle : alternative, boucle, appel de routines, ?
  - Préférer un ensemble de fonctions petit et bien ajusté au domaine du problème, pour réduire l'espace de recherche.
  - Attention à ne pas le réduire trop, sous peine de perdre la possibilité de trouver des solutions intéressantes !

Algorithme

4 étapes principales :

Algorithme

4 étapes principales :

- Générer une population aléatoire initiale à l'aide des fonctions, opérateurs et terminaux qui composent le problème.

Algorithme

4 étapes principales :

- Générer une population aléatoire initiale à l'aide des fonctions, opérateurs et terminaux qui composent le problème.
- Exécuter chaque programme de la population et affecter à chacun d'entre-eux une fitness en fonction de la façon où il résout le problème.

Algorithme

4 étapes principales :

- Générer une population aléatoire initiale à l'aide des fonctions, opérateurs et terminaux qui composent le problème.
- Exécuter chaque programme de la population et affecter à chacun d'entre-eux une fitness en fonction de la façon où il résout le problème.
- Créer une nouvelle population de programmes.

Algorithme

4 étapes principales :

- Générer une population aléatoire initiale à l'aide des fonctions, opérateurs et terminaux qui composent le problème.
- Exécuter chaque programme de la population et affecter à chacun d'entre-eux une fitness en fonction de la façon où il résout le problème.
- Créer une nouvelle population de programmes.
  - Copier les meilleurs programmes ;

Algorithme

4 étapes principales :

- Générer une population aléatoire initiale à l'aide des fonctions, opérateurs et terminaux qui composent le problème.
- Exécuter chaque programme de la population et affecter à chacun d'entre-eux une fitness en fonction de la façon où il résout le problème.
- Créer une nouvelle population de programmes.
  - Copier les meilleurs programmes ;
  - Créer des nouveaux programmes par mutation ;

Algorithme

4 étapes principales :

- Générer une population aléatoire initiale à l'aide des fonctions, opérateurs et terminaux qui composent le problème.
- Exécuter chaque programme de la population et affecter à chacun d'entre-eux une fitness en fonction de la façon où il résout le problème.
- Créer une nouvelle population de programmes.
  - Copier les meilleurs programmes ;
  - Créer des nouveaux programmes par mutation ;
  - Créer des nouveaux programmes par reproduction sexuée (crossing-over) ;

Algorithme

4 étapes principales :

- Générer une population aléatoire initiale à l'aide des fonctions, opérateurs et terminaux qui composent le problème.
- Exécuter chaque programme de la population et affecter à chacun d'entre-eux une fitness en fonction de la façon où il résout le problème.
- Créer une nouvelle population de programmes.
  - Copier les meilleurs programmes ;
  - Créer des nouveaux programmes par mutation ;
  - Créer des nouveaux programmes par reproduction sexuée (crossing-over) ;
- Déterminer quel est le meilleur programme apparu dans chaque génération,

Algorithme

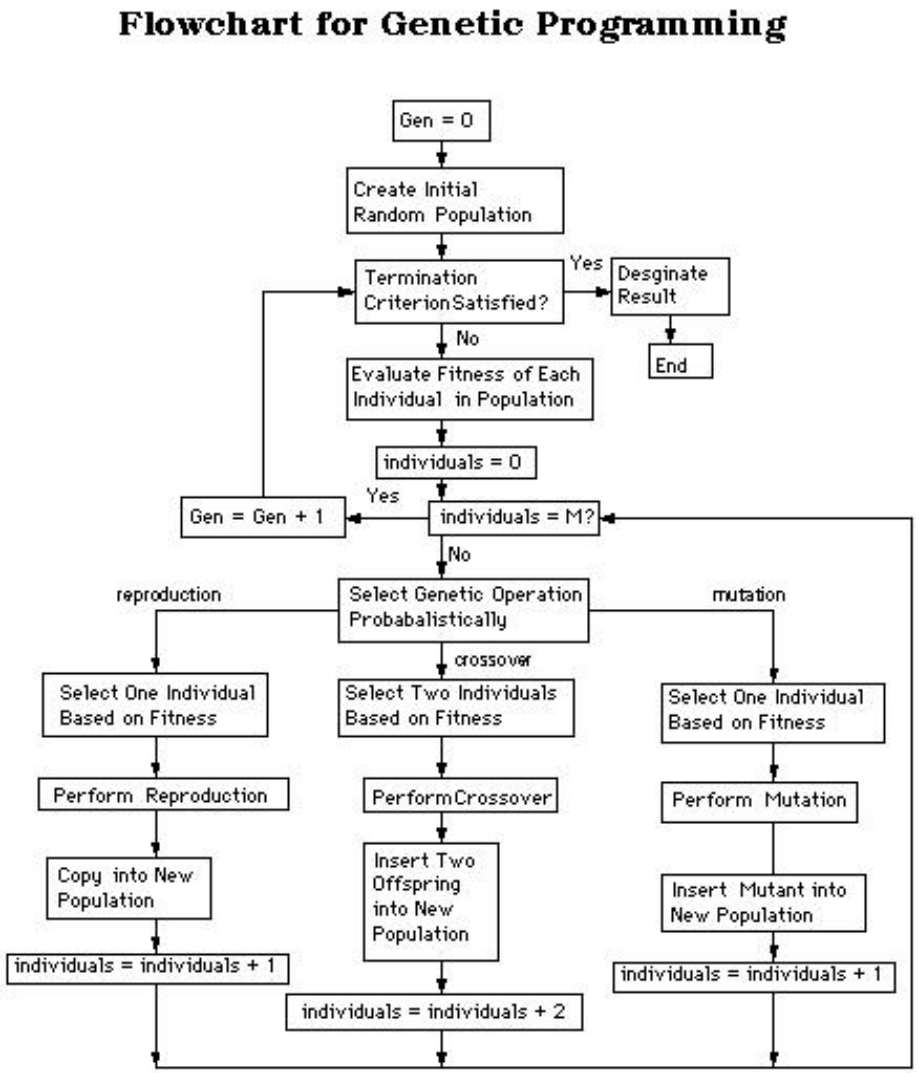
4 étapes principales :

- Générer une population aléatoire initiale à l'aide des fonctions, opérateurs et terminaux qui composent le problème.
- Exécuter chaque programme de la population et affecter à chacun d'entre-eux une fitness en fonction de la façon où il résout le problème.
- Créer une nouvelle population de programmes.
  - Copier les meilleurs programmes ;
  - Créer des nouveaux programmes par mutation ;
  - Créer des nouveaux programmes par reproduction sexuée (crossing-over) ;
- Déterminer quel est le meilleur programme apparu dans chaque génération,

Résultat

Le meilleur d'entre-eux est le résultat.

Algorithme


Flowchart for Genetic Programming
Gen = 0
Create Initial Random Population
Termination Criterion Satisfied?
Yes
Desginate Result
End
No
Evaluate Fitness of Each Individual in Population
individuals = 0
Yes
Gen = Gen + 1
individuals = M?
No
reproduction
Select Genetic Operation Probabalistically
mutation
crossover
Select One Individual Based on Fitness
Select Two Individuals Based on Fitness
Select One Individual Based on Fitness
Perform Reproduction
Perform Crossover
Perform Mutation
Copy into New Population
Insert Two Offspring into New Population
Insert Mutant into New Population
individuals = individuals + 1
individuals = individuals + 2
individuals = individuals + 1

Tirage aléatoire

- Problème un arbre construit aléatoirement a peu (aucune) chance d'être syntaxiquement correct ;

Tirage aléatoire

- Problème un arbre construit aléatoirement a peu (aucune) chance d'être syntaxiquement correct ;
- On peut partir de la racine :

Tirage aléatoire

- Problème un arbre construit aléatoirement a peu (aucune) chance d'être syntaxiquement correct ;
- On peut partir de la racine :
  - Le mot est terminal, on s'arrête ;

Tirage aléatoire

- Problème un arbre construit aléatoirement a peu (aucune) chance d'être syntaxiquement correct ;
- On peut partir de la racine :
  - Le mot est terminal, on s'arrête ;
  - C'est une fonction ou un opérateur, on tire ses descendants jusqu'à obtenir un arbre complet ;

Tirage aléatoire

- Problème un arbre construit aléatoirement a peu (aucune) chance d'être syntaxiquement correct ;
- On peut partir de la racine :
  - Le mot est terminal, on s'arrête ;
  - C'est une fonction ou un opérateur, on tire ses descendants jusqu'à obtenir un arbre complet ;
- On impose une profondeur minimale pour éviter les arbres réduits à une variable ;

Tirage aléatoire

- Problème un arbre construit aléatoirement a peu (aucune) chance d'être syntaxiquement correct ;
- On peut partir de la racine :
  - Le mot est terminal, on s'arrête ;
  - C'est une fonction ou un opérateur, on tire ses descendants jusqu'à obtenir un arbre complet ;
- On impose une profondeur minimale pour éviter les arbres réduits à une variable ;
- On impose une taille maximale ;

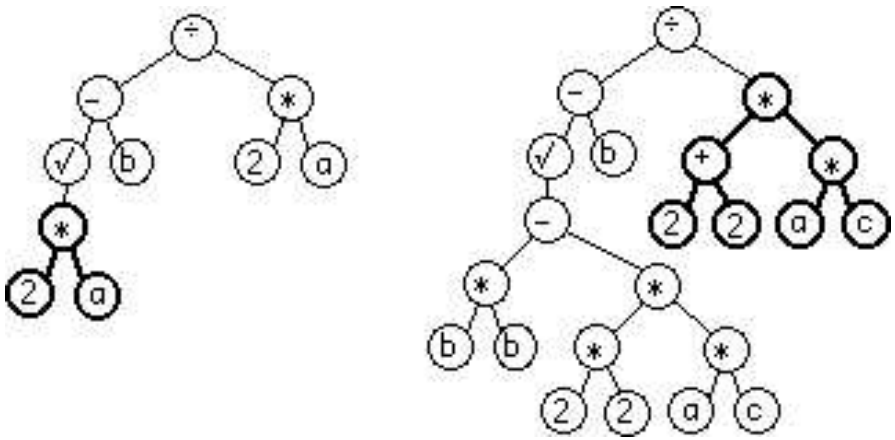
Tirage aléatoire

- Problème un arbre construit aléatoirement a peu (aucune) chance d'être syntaxiquement correct ;
- On peut partir de la racine :
  - Le mot est terminal, on s'arrête ;
  - C'est une fonction ou un opérateur, on tire ses descendants jusqu'à obtenir un arbre complet ;
- On impose une profondeur minimale pour éviter les arbres réduits à une variable ;
- On impose une taille maximale ;
- ...

Crossing-Over

- Choisir les nœuds aléatoirement ;

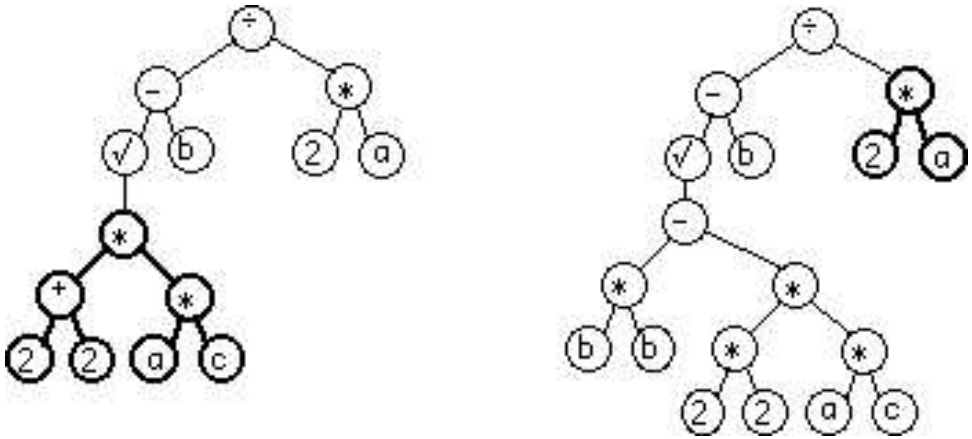
Deux parents différents



Crossing-Over

- Choisir les nœuds aléatoirement ;
- Échanger les sous-arbres.

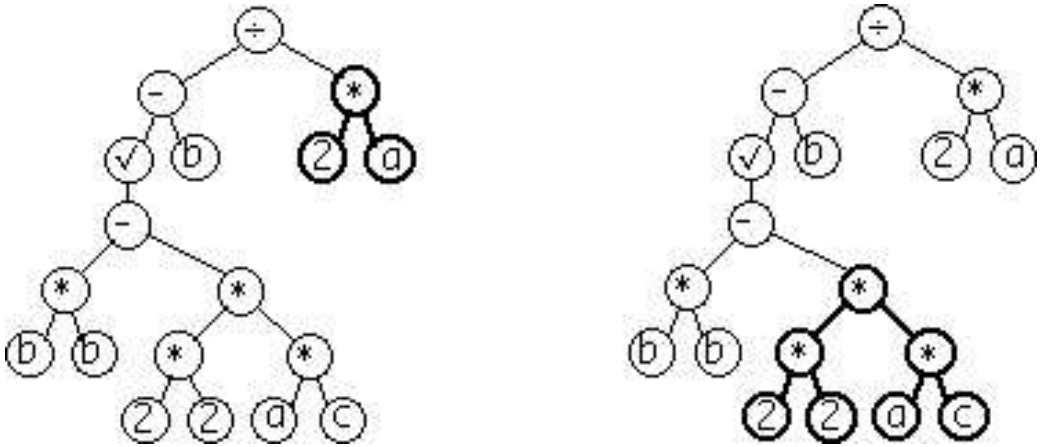
Deux parents différents



Crossing-Over

- Choisir les nœuds aléatoirement ;

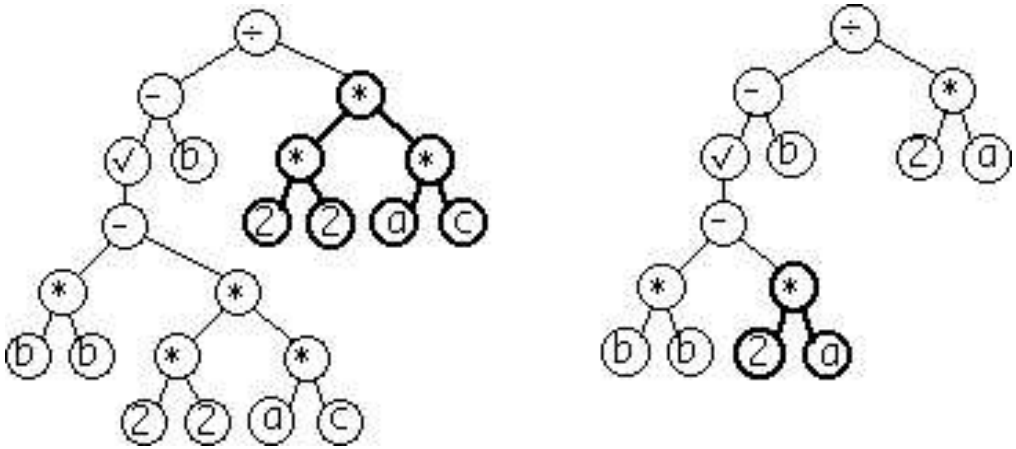
Deux parents identiques



Crossing-Over

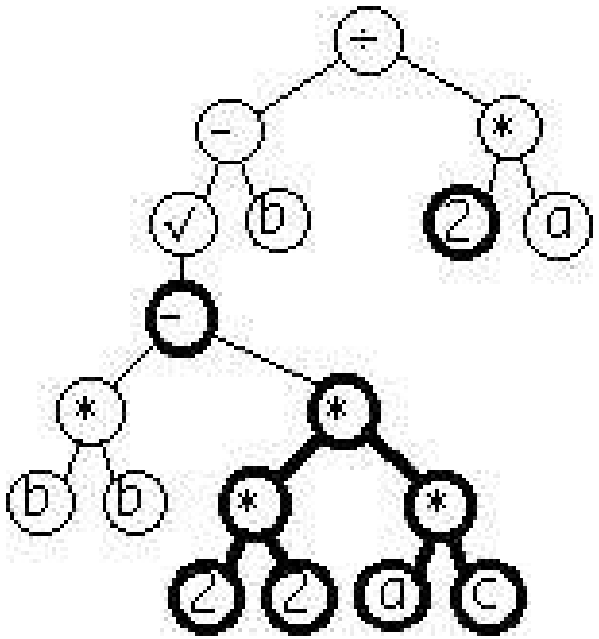
- Choisir les nœuds aléatoirement ;
- Échanger les sous-arbres.

Deux parents identiques



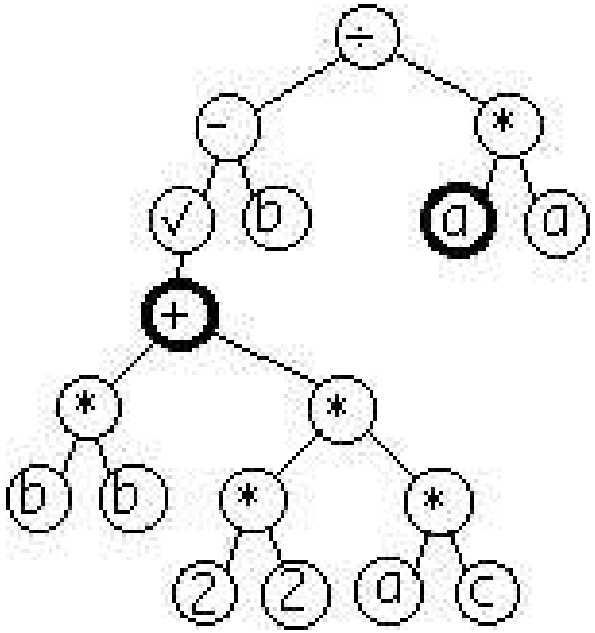
Mutation

- Deux types de mutations sont possibles :



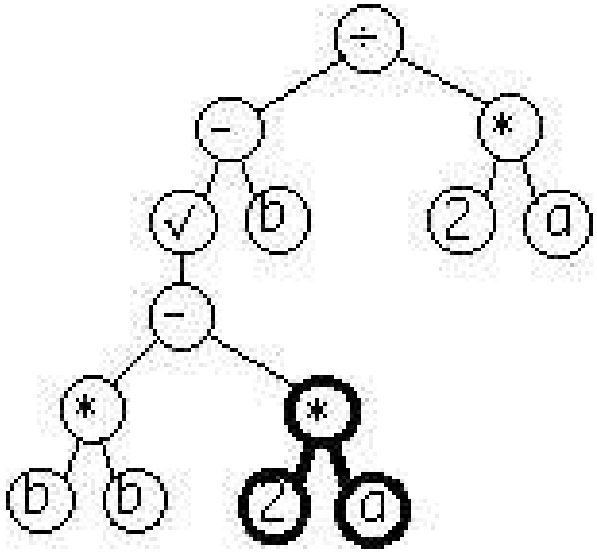
Mutation

- Deux types de mutations sont possibles :
- On remplace uniquement des nœuds de l'arbre.



Mutation

- Deux types de mutations sont possibles :
- On remplace uniquement des nœuds de l'arbre.
- Un sous-arbre est remplacé par un sous-arbre généré aléatoirement.

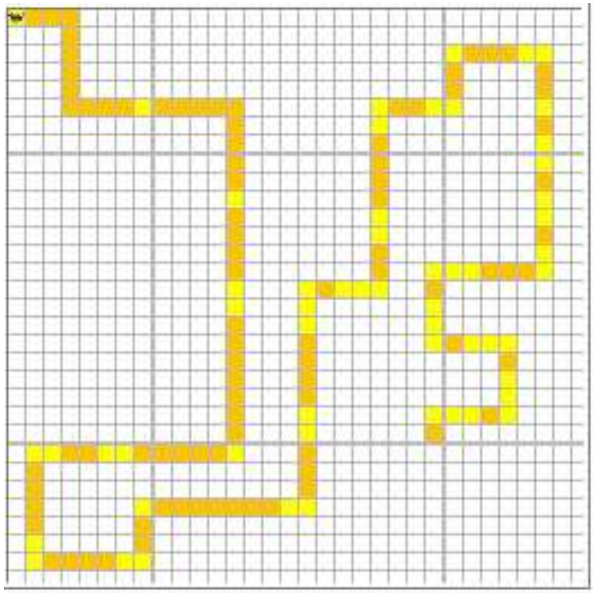


Exemple : Santa Fe Trail

- Fourmis à la recherche de nourriture ;
- Un chemin de nourriture qui comporte des trous
- 89 tas de nourriture.

Objectifs

Déterminer une stratégie pour consommer tous les tas de nourriture en un temps raisonnable.



Les fourmis

- Les terminaux : `{enAvant, droite, gauche }` ;
  - Provoque le déplacements des fourmis ;

Exemple de code

```
(siNourritureEnAvant (enAvant) (progn2 (gauche) (enAvant)))
```

Les fourmis

- Les terminaux : `{enAvant, droite, gauche }` ;
  - Provoque le déplacements des fourmis ;
- Les fonctions : `{siNourritureEnAvant(),progn2(), progn3() }` ;
  - `progn2()` et `progn3()` prennent respectivement 2 et 3 arguments qui sont exécutés en séquence.

Exemple de code

```
(siNourritureEnAvant (enAvant) (progn2 (gauche) (enAvant)))
```

Les fourmis

- Les terminaux : `{enAvant, droite, gauche }` ;
  - Provoque le déplacements des fourmis ;
- Les fonctions : `{siNourritureEnAvant(),progn2(), progn3() }` ;
  - `progn2()` et `progn3()` prennent respectivement 2 et 3 arguments qui sont exécutés en séquence.
- Fitness : Nombre de tas trouvés ;

Exemple de code

```
(siNourritureEnAvant (enAvant) (progn2 (gauche) (enAvant)))
```

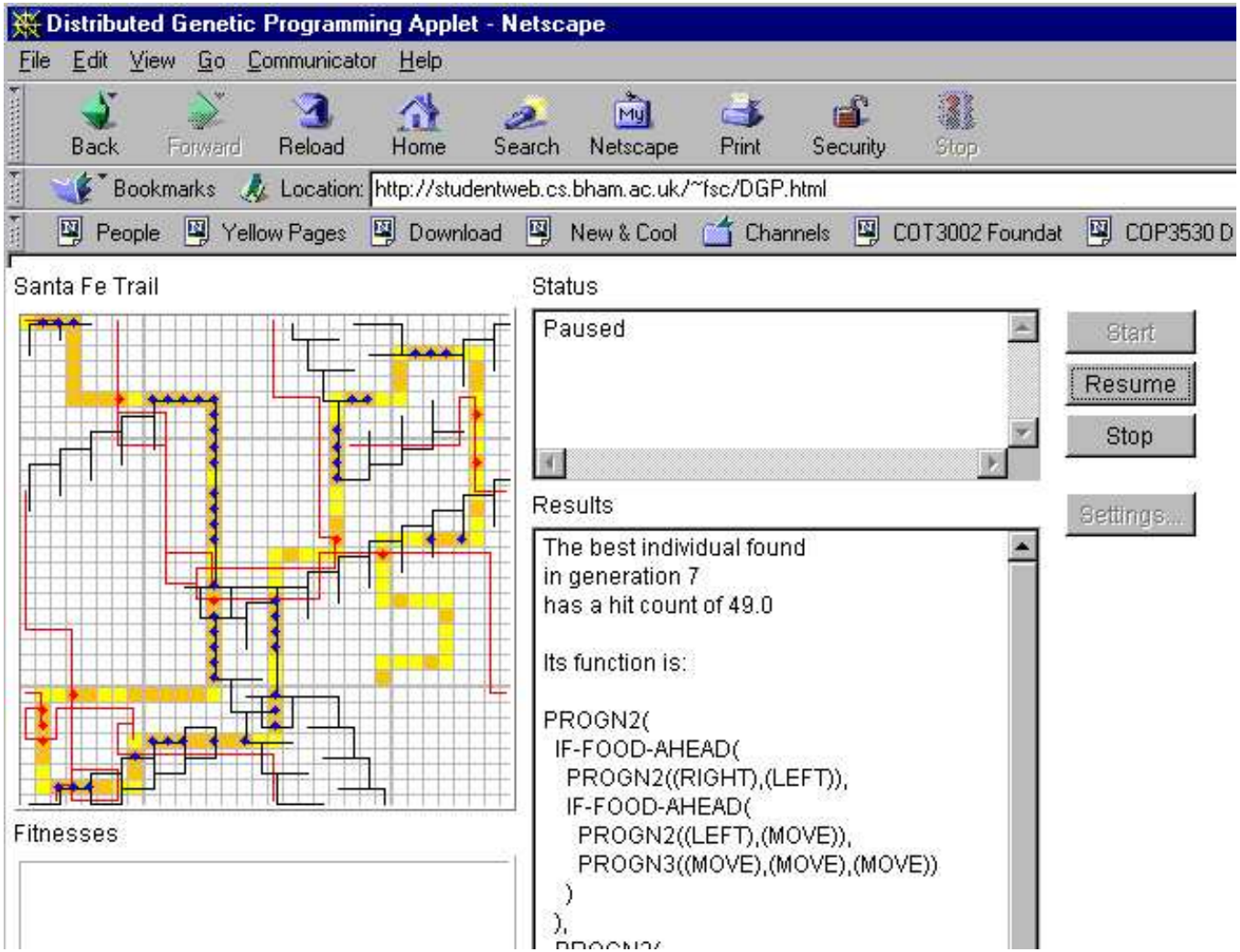
Les fourmis

- Les terminaux : `{enAvant, droite, gauche }` ;
  - Provoque le déplacements des fourmis ;
- Les fonctions : `{siNourritureEnAvant(),progn2(), progn3() }` ;
  - `progn2()` et `progn3()` prennent respectivement 2 et 3 arguments qui sont exécutés en séquence.
- Fitness : Nombre de tas trouvés ;
- Critère d'arrêt : time-out = 400 opérations

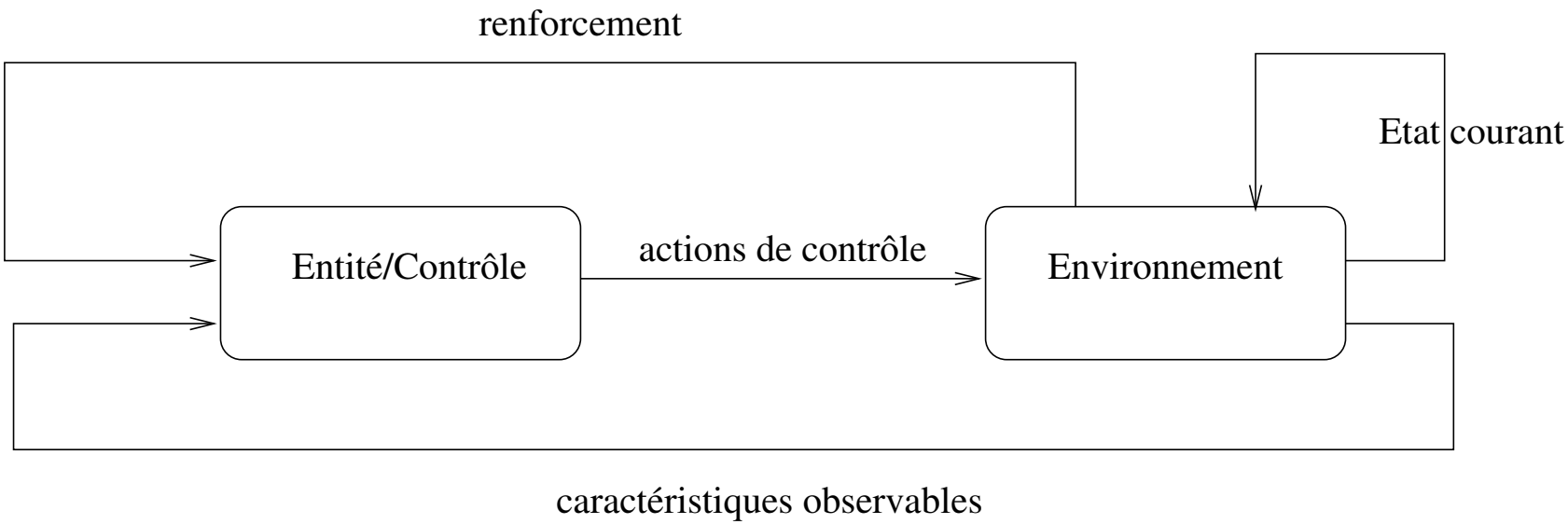
Exemple de code

```
(siNourritureEnAvant (enAvant) (progn2 (gauche) (enAvant)))
```

Trace d'exécution



Feedback et contrôle



Architecture

Système capables d'apprendre des règles pour optimiser leurs performances dans un environnement perçu sous forme de messages. Il est constitué de 4 parties principales :

- Une base de classifieurs qui représente les connaissances du système ;

Architecture

Système capables d'apprendre des règles pour optimiser leurs performances dans un environnement perçu sous forme de messages. Il est constitué de 4 parties principales :

- Une base de classifieurs qui représente les connaissances du système ;
- Une interface d'entrée composée de capteurs ;

Architecture

Système capables d'apprendre des règles pour optimiser leurs performances dans un environnement perçu sous forme de messages. Il est constitué de 4 parties principales :

- Une base de classifieurs qui représente les connaissances du système ;
- Une interface d'entrée composée de capteurs ;
- Une interface d'entrée composée d'actionneurs ;

Architecture

Système capables d'apprendre des règles pour optimiser leurs performances dans un environnement perçu sous forme de messages. Il est constitué de 4 parties principales :

- Une base de classifieurs qui représente les connaissances du système ;
- Une interface d'entrée composée de capteurs ;
- Une interface d'entrée composée d'actionneurs ;
- Une liste de messages ;

Architecture

Système capables d'apprendre des règles pour optimiser leurs performances dans un environnement perçu sous forme de messages. Il est constitué de 4 parties principales :

- Une base de classifieurs qui représente les connaissances du système ;
- Une interface d'entrée composée de capteurs ;
- Une interface d'entrée composée d'actionneurs ;
- Une liste de messages ;
- Deux algorithmes :

Architecture

Système capables d'apprendre des règles pour optimiser leurs performances dans un environnement perçu sous forme de messages. Il est constitué de 4 parties principales :

- Une base de classifieurs qui représente les connaissances du système ;
- Une interface d'entrée composée de capteurs ;
- Une interface d'entrée composée d'actionneurs ;
- Une liste de messages ;
- Deux algorithmes :
  - Un algorithme de répartition de crédits qui favorisent les classifieurs efficaces et pénalisent les inefficaces ;

Architecture

Système capables d'apprendre des règles pour optimiser leurs performances dans un environnement perçu sous forme de messages. Il est constitué de 4 parties principales :

- Une base de classifieurs qui représente les connaissances du système ;
- Une interface d'entrée composée de capteurs ;
- Une interface d'entrée composée d'actionneurs ;
- Une liste de messages ;
- Deux algorithmes :
  - Un algorithme génétique qui fait évoluer la base de classifieurs.

Structure d'un classifieur

- $<$ *classifieur* $>:=<$ *condition* $>:<$ *message* $>$
  - $<$ *condition* $>:= \{0, 1, \#\}^+$ # joker
  - $<$ *message* $>:= \{0, 1\}^+$

Structure d'un classifieur

- $< \textit{classifieur} >:=< \textit{condition} >:< \textit{message} >$
  - $< \textit{condition} >:= \{0, 1, \#\}^+$ # joker
  - $< \textit{message} >:= \{0, 1\}^+$
- Une condition est satisfaite lorsqu'il y a égalité entre la condition et le message;

Structure d'un classifieur

- $<$ *classifieur* $>:=<$ *condition* $>:<$ *message* $>$
  - $<$ *condition* $>:= \{0, 1, \#\}^+$ # joker
  - $<$ *message* $>:= \{0, 1\}^+$
- Une condition est satisfaite lorsqu'il y a égalité entre la condition et le message;
- Une condition satisfaite est candidate à l'envoi de message ;

Structure d'un classifieur

- $<$ *classifieur* $>:=<$ *condition* $>:<$ *message* $>$
  - $<$ *condition* $>:= \{0, 1, \#\}^+$ # joker
  - $<$ *message* $>:= \{0, 1\}^+$
- Une condition est satisfaite lorsqu'il y a égalité entre la condition et le message;
- Une condition satisfaite est candidate à l'envoi de message ;
- L'envoi du message dépend du mécanisme d'enchère qui dépend du poids du classifieur.

Mécanisme d'enchère

Comment choisir le classifieur qui envoie le message?

1. 0 1 # # : 0 0 0 0
2. 1 # # 0 : 1 0 0 1
3. 0 0 # # : 0 1 1 0 $\Rightarrow$
4. # # 1 1 : 1 1 0 1
5. # 1 1 # : 1 1 0 0

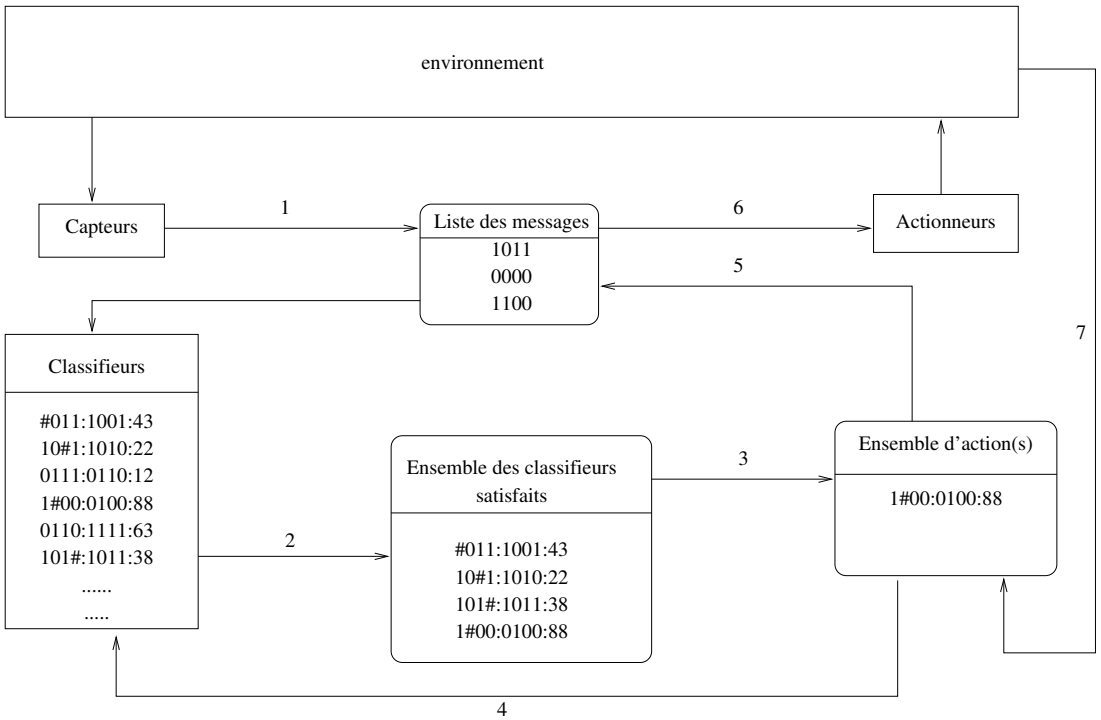
Algorithme de répartition de crédits.

Algorithme de répartition

Bucket brigade

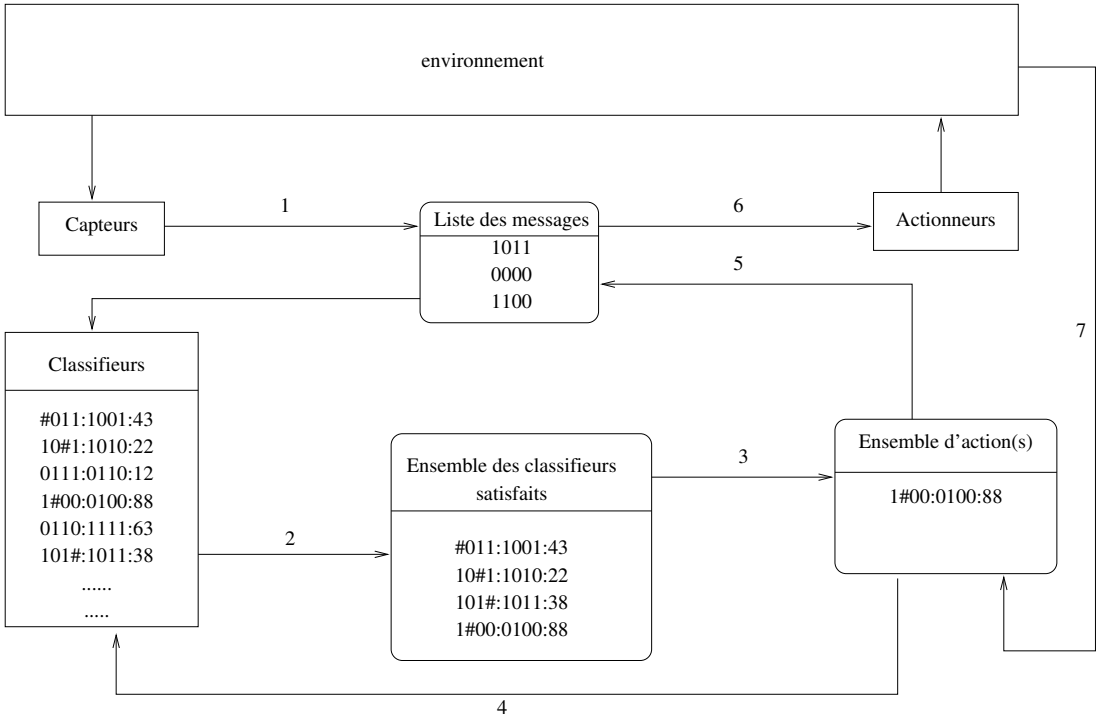
- Deux notions :
  - La vente aux enchères ;
  - Une chambre de compensation ;
- Chaque classifieur a un poids ;
- L'enchère d'un classifieur donné est proportionnelle à son poids ;
- Un classifieur est récompensé en fonction de l'environnement et de ses messages envoyés ;
- Un classifieur gagnant effectue un paiement à la chambre de compensation ;

Fonctionnement



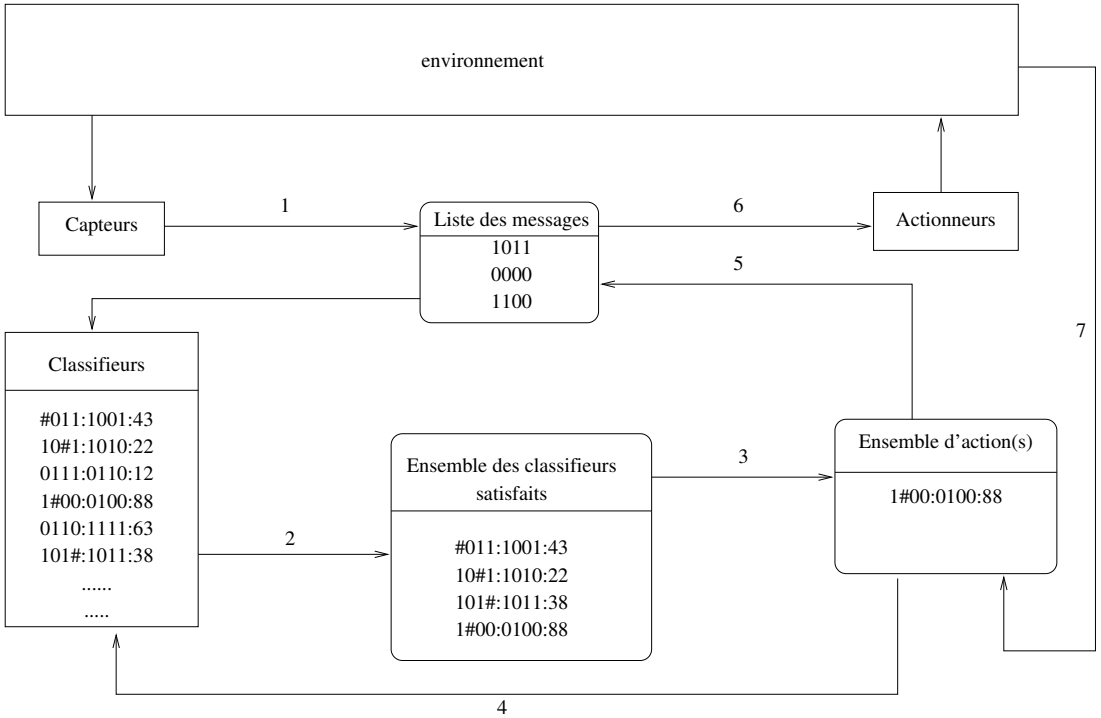
1 . Les capteurs placent des messages dans la liste des messages qui correspondent aux événements observés dans l'environnement

Fonctionnement



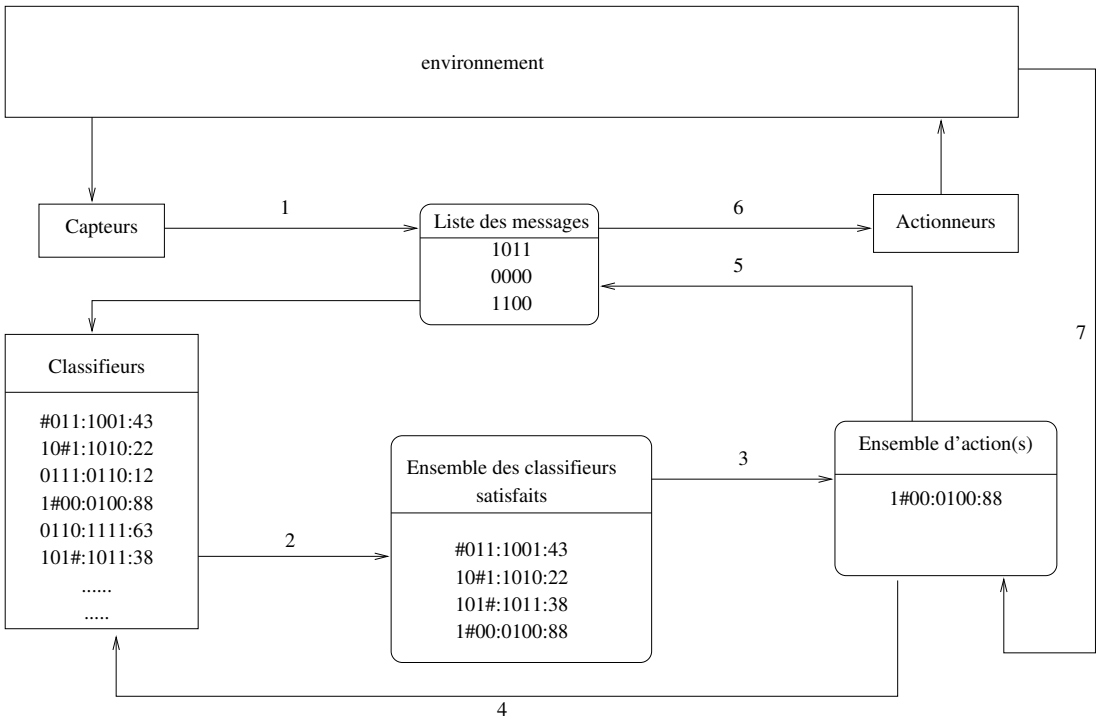
2. Un ensemble de classifieurs satisfaisant les messages est formé

Fonctionnement



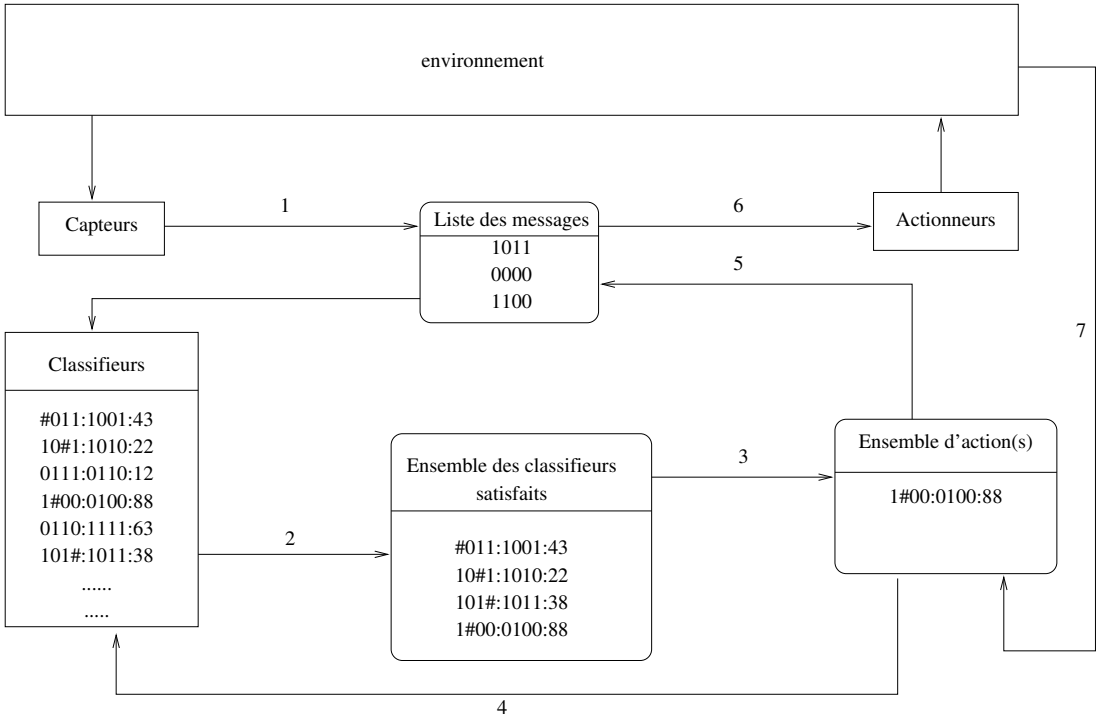
3. Les classifieurs luttent entre-eux (par enchère) pour déterminer ceux qui auront le droit de poster leur message. Le résultat est dépendant du poids mais peut intégrer d'autres conditions (le classifieur le moins général (par ex)). Un ensemble d'actions est formé (plusieurs classifieurs peuvent gagner).

Fonctionnement



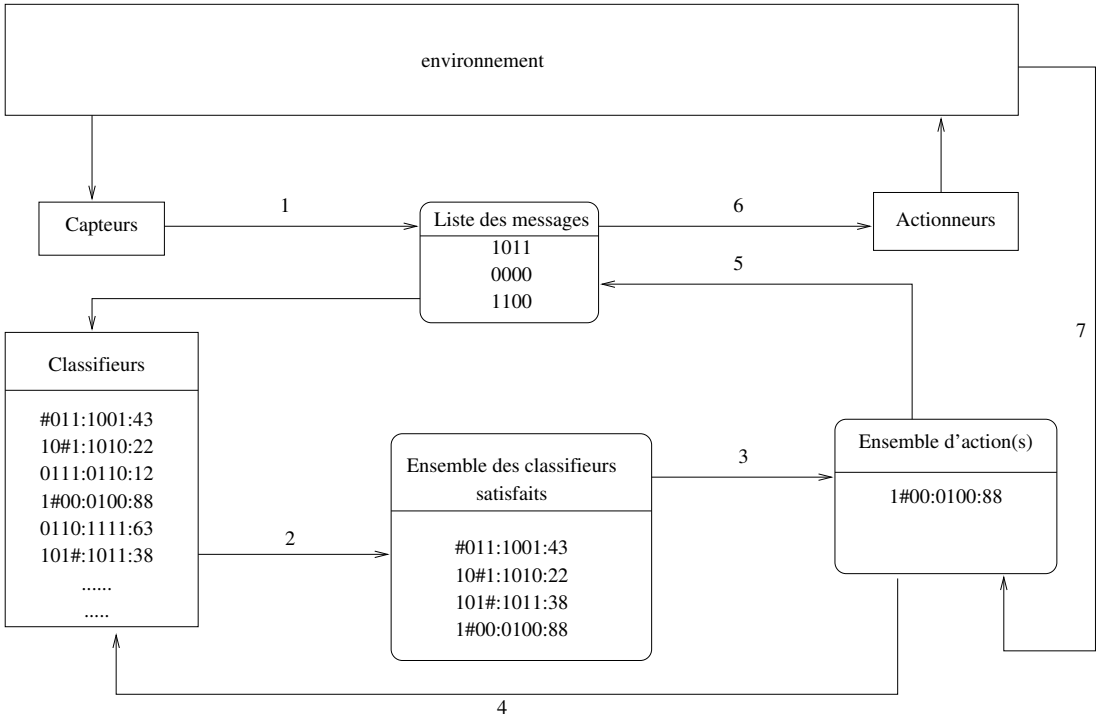
4. La somme des enchères gagnantes est partagée par les classifieurs ayant conduits aux messages émis (le poids des classifieurs correspondant augmente)

Fonctionnement



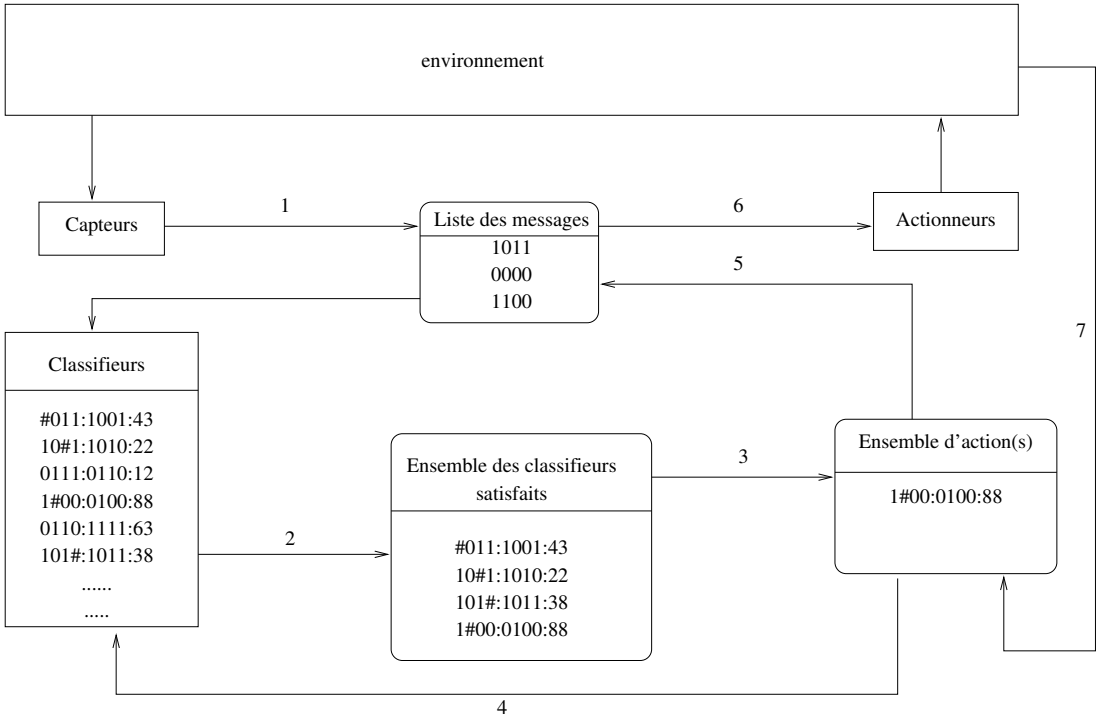
5. Une nouvelle liste de message est formée à partir des messages contenus dans l'ensemble des actions

Fonctionnement



6. Si des messages correspondent à des actions réelles les actionneurs les effectuent

Fonctionnement



7. L'environnement récompense les classifieurs contenus dans l'ensemble des actions

Algorithme génétique

- Régulièrement un AG produit de nouveaux classifieurs.
- chromosome = classifieur
- Le poids d'une règle donne son degré d'adaptation (pour l'AG).
- Seule une partie des règles est remplacée et la sélection utilise le plus souvent une roulette biaisée.
- Les nouvelles règles introduites participent alors aux enchères suivantes, etc.

Stratégies évolutives

D'après Ingo Rechenberge (Allemagne). Assez voisins des algorithmes génétiques (J. Holland / USA), les différences essentielles sont :

- Les individus ne sont pas codés par des chromosomes mais par des valeurs réelles ;
- Moins d'aléatoire : la reproduction est déterministe ;
- L'opérateur de mutation agit principalement et l'opérateur de croisement est moins important : il agit simplement pour assurer une diversité suffisante.

Vol d'oiseau

Au départ J. Kennedy et R. Eberhart cherchaient à simuler :

- la capacité des oiseaux à voler de façon synchrone
- et leur aptitude à changer brusquement de direction tout en restant en une formation optimale.

Modélisation

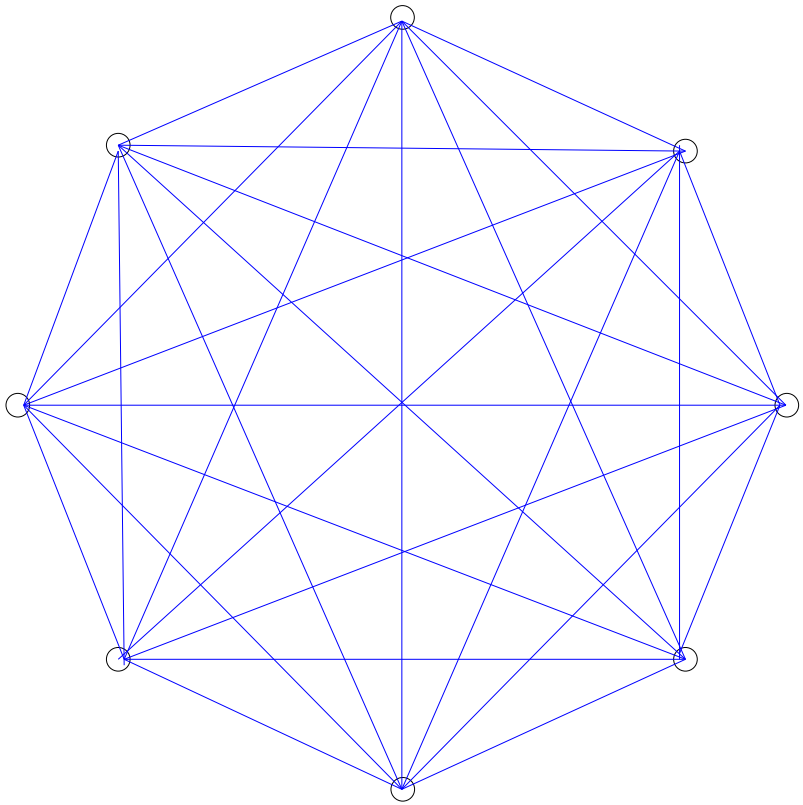
Les particules sont les individus et elles se déplacent dans l'hyperespace de recherche. Le processus de recherche est basé sur deux règles :

- Chaque particule est dotée d'une mémoire qui lui permet de mémoriser le meilleur point par lequel elle est déjà passée et elle a tendance à retourner vers ce point.
- Chaque particule est informée du meilleur point connu au sein de son voisinage et elle va tendre à aller vers ce point.

Voisinage

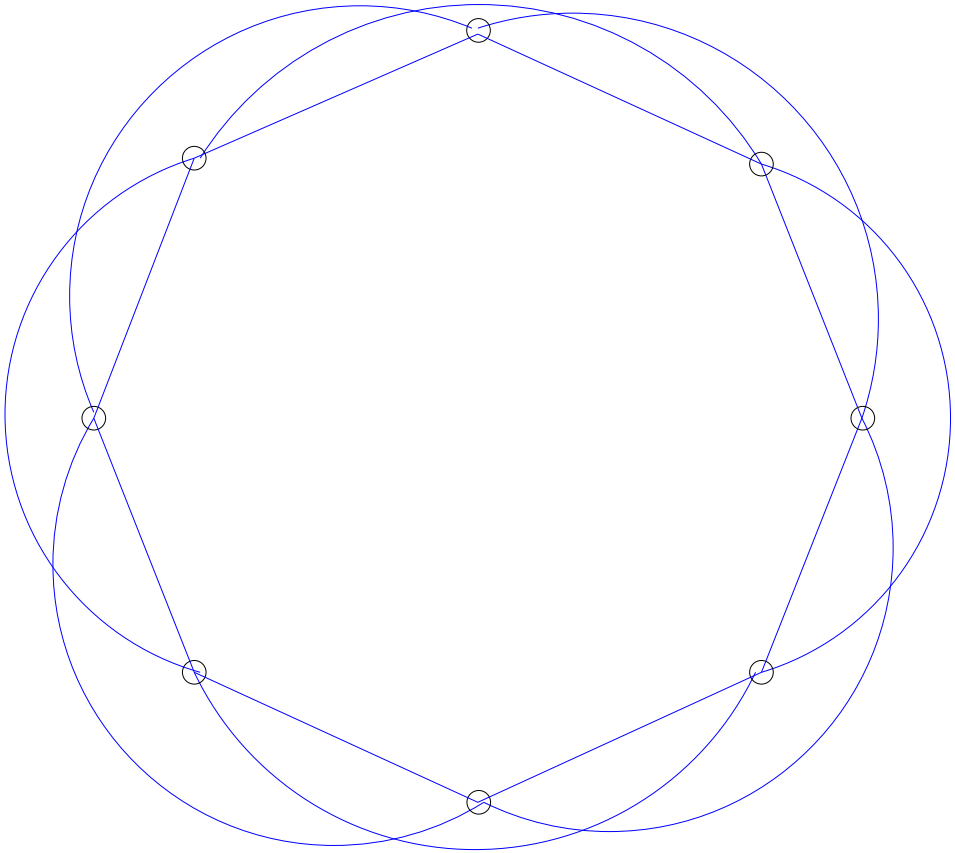
- Le voisinage constitue la structure du réseau social ;
- Les particules à l'intérieur d'un voisinage communiquent entre-elles ;
- Différents voisinages ont été étudiés et sont considérés en fonction des identificateurs des particules et non des informations topologiques comme les distances euclidiennes dans l'espace de recherche.

Différents voisinages



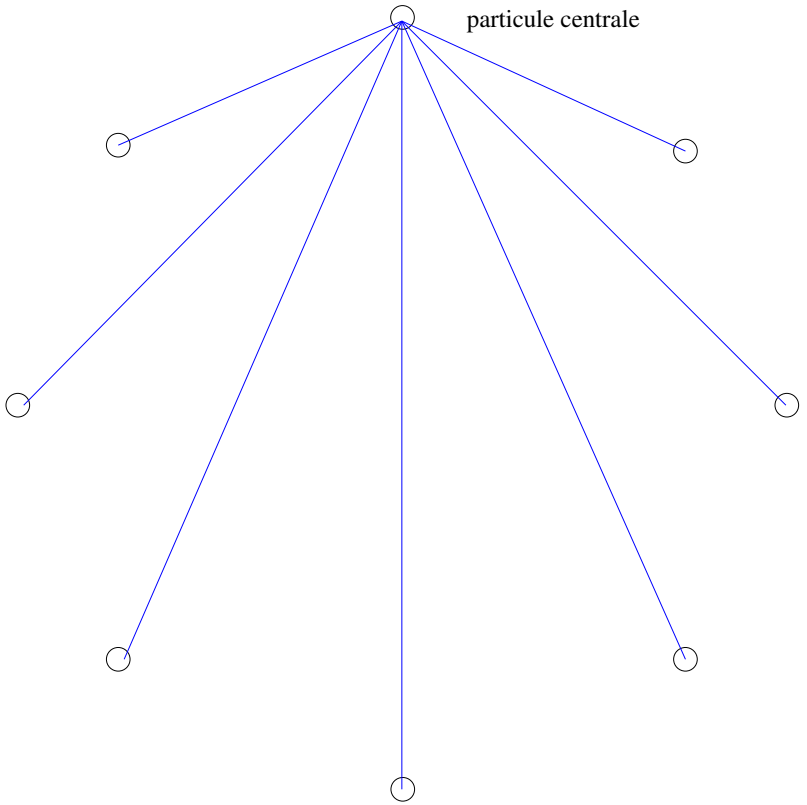
Topologie en étoile : le réseau social est complet, chaque particule est attirée vers la meilleure particule notée `gbest` et communique avec les autres.

Différents voisinages



Topologie en anneau : chaque particule communique avec n ($n = 3$) voisines immédiates. Chaque particule tend à se déplacer vers la meilleure dans son voisinage local notée `lbest`.

Différents voisinages



Topologie en rayon : une particule "centrale" est connectée à toute les autres. Seule cette particule centrale ajuste sa position vers la meilleure, si cela provoque une amélioration l'information est propagée aux autres.

Recherche d'une solution

- Chaque particule représente une solution potentielle dans l'espace de recherche ;
- La nouvelle position d'une particule est déterminée en fonction de sa propre valeur et celle de ses voisines.
- Soit $\overrightarrow{x_i}(t)$ la position de la particule P_i au temps t, sa position est modifiée en ajoutant une vitesse $\overrightarrow{v_i}(t)$ à sa positon courante :

$$\overrightarrow{x_i}(t) = \overrightarrow{x_i}(t-1) + \overrightarrow{v_i}(t) \quad (1)$$

- C'est le vecteur vitesse qui dirige le processus de recherche et reflète la "sociabilité" des particules.

Algorithme de base

Initialiser aléatoirement la population
Répéter
  Pour i **de** 1 **à** N **faire**
    Si $(\mathcal{F}(\overrightarrow{x_i}) > pbest_i)$ **Alors**
      $pbest_i \leftarrow \mathcal{F}(\overrightarrow{x_i})$
      $\overrightarrow{x}_{pbest_i} \leftarrow \overrightarrow{x_i}$
    Fin Si
    $\overrightarrow{v_i} \leftarrow \overrightarrow{v_i} + \rho(\overrightarrow{x}_{pbest_i} - \overrightarrow{x_i})$
    $\overrightarrow{x_i} \leftarrow \overrightarrow{x_i} + \overrightarrow{v_i}$
  Fin Pour
jusqu'à ce que (le processus converge)

N nombre de particules
$\overrightarrow{x_i}$ position de la particule P_i
$\overrightarrow{v_i}$ vitesse de la particule P_i
$pbest_i$ meilleure fitness obtenue pour la particule P_i
$\overrightarrow{x}_{pbest_i}$ position de la particule P_i pour la meilleure fitness
ρ valeur aléatoire positive

Algorithme avec un voisinage en étoile

Initialiser aléatoirement la population
Répéter
  Pour i **de** 1 **à** N **faire**
    Si $(\mathcal{F}(\overrightarrow{x_i}) > pbest_i)$ **Alors**
      $pbest_i \leftarrow \mathcal{F}(\overrightarrow{x_i})$
      $\overrightarrow{x}_{pbest_i} \leftarrow \overrightarrow{x_i}$
    Fin Si
    Si $(\mathcal{F}(\overrightarrow{x_i}(t)) > gbest)$ **Alors**
      $gbest \leftarrow \mathcal{F}(\overrightarrow{x_i})$
      $\overrightarrow{x}_{gbest} \leftarrow \overrightarrow{x_i}$
    Fin Si
  Fin Pour
  Pour i **de** 1 **à** N **faire**
    $\overrightarrow{v_i} \leftarrow \overrightarrow{v_i} + \rho_1(\overrightarrow{x}_{pbest_i} - \overrightarrow{x_i}) + \rho_2(\overrightarrow{x}_{gbest} - \overrightarrow{x_i})$
    $\overrightarrow{x_i} \leftarrow \overrightarrow{x_i} + \overrightarrow{v_i}$
  Fin Pour
jusqu'à ce que (le processus converge)

N nombre de particules
$\overrightarrow{x_i}$ position de la particule P_i
$\overrightarrow{v_i}$ vitesse de la particule P_i
$pbest_i$ meilleure fitness obtenue pour la particule P_i
$\overrightarrow{x}_{pbest_i}$ position de la particule P_i pour la meilleure fitness
$\overrightarrow{x}_{gbest}$ position de la particule ayant la meilleure fitness de toutes
ρ_1, ρ_2 valeurs aléatoires positives

Vitesse de variation

Plus une particule est éloignée de la meilleure solution globale et de sa meilleure solution, plus sera importante la variation de sa vitesse afin de faire bouger la particule vers les meilleures solutions. Les variables aléatoires ρ_1 et ρ_2 peuvent être définie de la façon suivante :

$$\begin{cases} \rho_1 = r_1 c_1 \\ \rho_2 = r_2 c_2 \end{cases}$$

r_1 et r_2 suivent une loi uniforme sur [0..1] et c_1 et c_2 sont constantes et représentent une accélération positive, avec $c_1 + c_2 \leq 4$.
L'algorithme s'exécute tant qu'un critère de convergence n'a pas été atteint. Cela peut être :

- Un nombre fixe d'itérations ;
- En fonction de la fitness ;
- Lorsque la variation de vitesse est proche de 0.

Paramètres de l'algorithme

Six paramètres rentrent en ligne de compte :

- La dimension du problème ;
- Le nombre de particules ;
- Les valeurs des coefficients ρ ;
- La taille du voisinage ;
- La vitesse maximale ;
- L'inertie.

Vitesse maximale

Pour éviter que les particules se déplacent trop rapidement d'une région à une autre dans l'espace de recherche, on fixe une vitesse maximale V_{max}. Ainsi si $v_{ij}(t)$ est la vitesse de la particule P_i au temps t dans la dimension j,

$$v_{ij}(t) = V_{max} \text{ si } v_{ij}(t) > V_{max} \text{ et } v_{ij}(t) = -V_{max} \text{ si } v_{ij}(t) < -V_{max}$$

V_{max} est généralement dépendant de l'échelle du problème. V_{max} n'est pas obligatoire si on utilise un coefficient de constriction (resserrement) κ

$$\overrightarrow{v_i}(t) = \kappa(\overrightarrow{v_i}(t-1) + \rho_1(\overrightarrow{x}_{pbest_i} - \overrightarrow{x_i}(t)) + \rho_2(\overrightarrow{x}_{gbest} - \overrightarrow{x_i}(t)))$$
$$\text{avec } \kappa = 1 - \frac{1}{\rho} + \frac{\sqrt{|\rho^2 - 4\rho|}}{2} \text{ et } \rho = \rho_1 + \rho_2 > 4$$

Facteur d'inertie

Pour contrôler l'influence de la vitesse obtenue au pas précédent on peut introduire un facteur d'inertie Φ qui décroît en fonction du temps.

$$\overrightarrow{v_i}(t) = \Phi \overrightarrow{v_i}(t-1) + \rho_1(\overrightarrow{x}_{pbest_i} - \overrightarrow{x_i}(t)) + \rho_2(\overrightarrow{x}_{gbest} - \overrightarrow{x_i}(t))$$

Un grand facteur d'inertie provoque une grande exploration de l'espace de recherche alors qu'un petit facteur d'inertie concentre la recherche sur un petit espace. La convergence n'est pas assurée quelques soient les valeurs de Φ et également de c_1 et c_2 [**?**], les études expérimentales montrent

$$\frac{1}{2}(c_1 + c_2) - 1 < \Phi \leq 1$$

Voisinage basé sur une distance euclidienne

Kennedy et Eberhart utilisent un voisinage basé sur les indices des particules, Suganthan utilise un voisinage spatial entre les particules. Une particule P_b est voisine d'une particule P_a si :

$$\frac{||\overrightarrow{x}_a - \overrightarrow{x}_b||}{d_{max}} < \xi \tag{2}$$

avec d_{max} la plus grande distance entre deux particules et

$$\xi = \frac{3t + 0.6t_{max}}{t_{max}} \tag{3}$$

avec t l'itération courante et t_max le nombre maximal d'itérations. On peut remarquer que la taille du voisinage croît avec le temps.

Le modèle Echo

- Ensemble d'agents situés dans un environnement nutritif ;
- Comportement des agents déduit de leur génomes ;
- Interactions entre agents de nature variée : reproduction, compétition, coopération ;
- Performance/fitness des agents : leur capacité de survie.

Caractéristiques

- Temps discret et monde représenté par une grille torique ;
- Déplacement possible d'un agent dans une case voisine à chaque pas de temps ;
- Interactions possibles entre agents que s'ils sont sur des cases voisines ;
- Les comportements des agents sont contrôlés par des gènes spécifiques appelés *conditions* ;
- Les agents ont des signes extérieurs apparents ou "tags" représentatifs de leur état social. Ces tags sont représentés par des gènes spécifiques ;

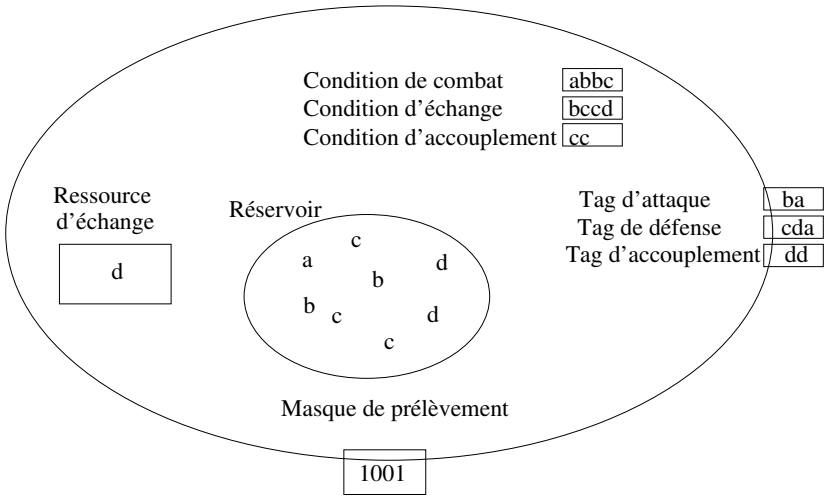
Caractéristiques

- Reproductions sexuelles ou asexuelles permettant l'évolution du génotype ;
- Chaque site produit une quantité de ressources spécifiques à chaque pas de temps. Il y a 4 variétés de ressources : a, b, c et d sur lesquelles est basée la survie de chaque agent ; Ce sont les "molécules" de base du monde Echo. Elles sont les constituants du génôme des agents et forment pour cela des chaînes (simple chaînage, pas de réaction chimique). Il n'y a pas d'interaction entre les gènes d'un génome.

Agents

Constitues de 2 éléments :

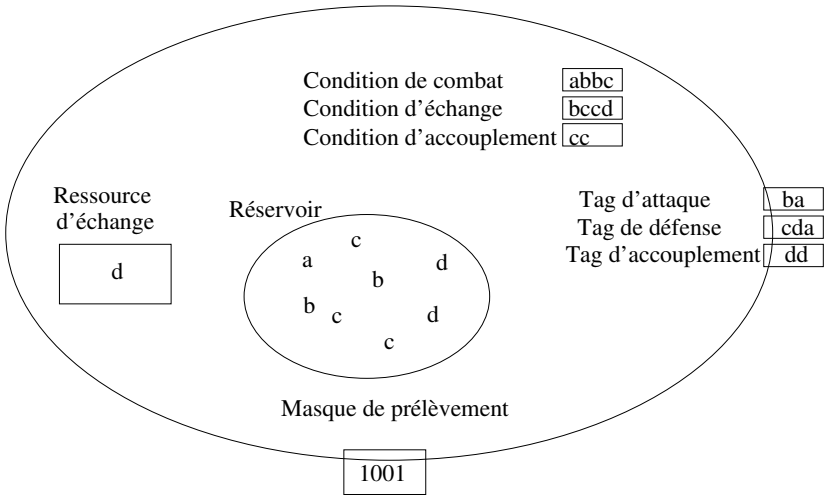
- Chromosome codant le comportement et l'apparence (tag) des agents : ce sont des chaînes de ressources (appelés aussi *nucléotide*) ;



Agents

Constitues de 2 éléments :

- Chromosome codant le comportement et l'apparence (tag) des agents : ce sont des chaînes de ressources (appelés aussi *nucléotide*) ;
- Un réservoir de ressources stockées dans chaque agent.



Chromosomes d'un agent

Constitués de 8 gènes au total

- 3 gènes de tag : tag d'attaque, tag de défense et tag d'accouplement ;
- 3 gènes de conditions (comportementales et sociales) : condition de combat, condition de négociation et condition d'accouplement ;
- 2 gènes de tags pour les échanges de ressources :
  - Ressources d'échanges : type de ressource que peut donner l'agent à un autre au cours d'une négociation ;
  - Masque de prélèvement : masque binaire d'un bit par type de ressources. Une ressource peut être acquise dans l'environnement que si le bit correspondant vaut 1.

D'autres possibilités d'échanges de ressources : suite à un combat ou par transmission filiale.

Auto-reproduction

Lorsque la quantité de ressources du réservoir de l'agent est suffisamment importante,

- il y a génération d'un clone auquel on transmet une quantité des ressources ;
- le chromosome du clone est obtenu par copie + mutation.

Mutation

Dans Echo, les mutations agissent sur les gènes.
Sur les tags et les conditions, 3 types de mutation sont permis :

- suppression du dernier nucléotide du gène ;
- insertion d'un nucléotide à la fin du gène ;
- modification d'un nucléotide dans le gène.

Sur les ressources d'échanges, seule la modification est permise. Sur le masque de prélèvement, la mutation correspond à la modification d'un bit.

Interactions entre les agents

Elles sont contrôlées par les tags et les conditions. Deux agents proches peuvent interagir :

- par indifférence ;
- par échange ;
- par reproduction et génération de nouveaux agents avec croisements génétiques.

Les interactions sont contrôlées par un mécanisme de préfixe : elles ne peuvent se produire que si la condition est le préfixe du tag associé.

Agent A	Agent B
Condition de combat : ab	Tag d'attaque : abb
Tag d'attaque : bcd Condition d'échange : ab	Condition d'échange : b Tag d'attaque : abb
Tag d'accoupl. : dcd Condition d'accoupl. : a	Condition d'accoupl. : dc Tag d'accoupl. : adb

- L'agent A peut attaquer l'agent B ;
- Les deux agents peuvent procéder à des échanges ;
- Les deux agents peuvent s'accoupler.

Combat

- Les combats se produisent si les conditions sont réalisées et avec une certaine probabilité ;
- Dans le cas d'un combat, on compare 1 à 1 les nucléotides du tag d'attaque d'un agent avec le tag de défense de l'autre afin d'obtenir un score. La probabilité d'être le vainqueur correspond au rapport de son score sur la somme des 2 scores.
- Le vainqueur remporte toutes les ressources du vaincu, y compris celles qui constituent son génôme. Le vaincu est retiré du monde.

Accouplement

Reproduction par croisement de deux sous-chaînes :

$$\begin{cases} agent\ A : & ab||bcbd||cd \\ agent\ B : & b||bd||ca \end{cases} \quad \begin{matrix} \searrow & \nearrow \\ \nearrow & \searrow \end{matrix} \quad \begin{matrix} ab||bd||cd \\ b||bcbd||ca \end{matrix}$$

Les longueurs des gènes des ressources d'échange et du masque de prélèvement doivent être constantes.

Cycle de base

1. interaction entre des agents sélectionnés ;
2. alimentation des agents par distribution des ressources de l'environnement ;
3. mort accidentelle des agents ;
4. renouvellement des ressources de l'environnement.