CM 8 cognition artificielle

L’algorithme génétique

1. **Algorithmes génétiques**

Les algorithmes génétiques appartiennent à la famille des algorithmes évolutionnistes. Leur but est d’obtenir une solution approchée à un problème d’optimisation, lorsqu’il n’existe pas de méthode exacte (ou que la solution est inconnue) pour le résoudre en un temps raisonnable.

Les algorithmes génétiques utilisent la notion de sélection naturelle de Charles Darwin (The Origin of species, 1859)

L’algorithme met en œuvre 3 m´mécanismes :

1. Sélection : on garde les individus enclins à obtenir les meilleurs résultats.
2. Enjambement, croisement ou recombinaison : 2 chromosomes s’échangent des parties de leurs chaines, pour donner de nouveaux chromosomes, avec une probabilité dans [0, 1].
3. Mutations : de façon aléatoire, un gène peut, au sein d’un chromosome être substitué à un autre, avec une probabilité faible (eg entre 0,001 et 0,01).

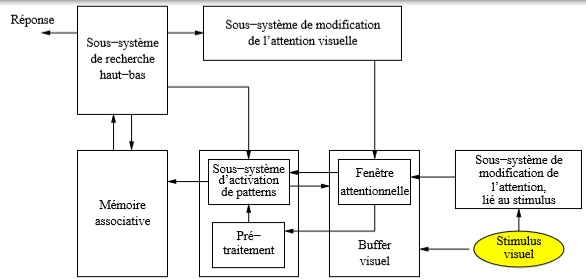
Remarque : la mutation permet d’éviter de converger trop rapidement vers une solution qui n’est pas la meilleure.

Exemple d’applications industrielles

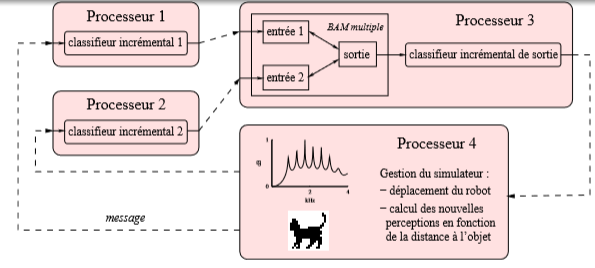
* Motorola : tests d’applications informatiques
* Optimisation de l’aérodynamique, optimisation structurelle, etc.
* Recherche d’itinéraires, par exemple Pathﬁnder
* Robotique, par exemple Aibo (pour apprendre à marcher)

1. **Algorithmes génétiques pour déterminer les hyperparamètres**

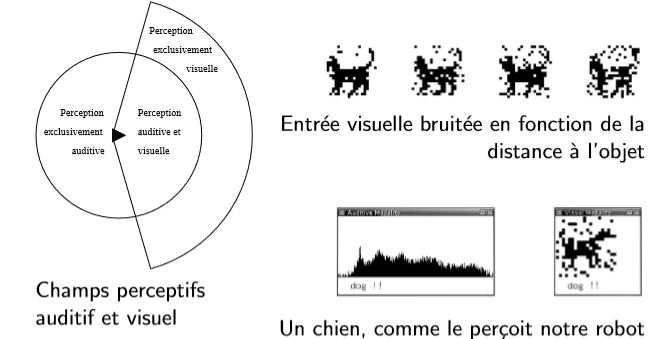
Modélisation informatique de la mémoire

Architecture fonctionnelle simpliﬁée d’un modèle de mémoire proposé par Kosslyn & Koenig pour la modalité visuelle (1995).

Implémentation et simulation d’un petit robot

Architecture informatique proposée par Emanuelle Reynaud et Didier Puzenat (2001) – hyper paramètres optimisés par un algorithme génétique

Simulation d’un robot comme base d’expérimentation du modèle multimodal (plongé dans un environnement virtuel)



1. **Problème du voyageur de commerce**

Trouver la distance minimale pour passer par toutes les villes sachant les distances entre chaque ville

⇒ Il s’agit d’un problème d’optimisation pour lequel on ne connait pas d’algorithme permettant de trouver une solution exacte rapidement dans tous les cas

Problème : « explosion combinatoire »

En eﬀet : pour n villes : on a 1 2(n−1) chemins à étudier !

Il y a environ 1080 atomes dans l’univers connu

⇒ impossible d’explorer tous les chemins possibles !

Le voyageur de commerce en python

On va utiliser un algo génétique pour trouver un bon chemin !

Un individu : un chemin passant par toutes les villes ﬁtness ≡ critère de sélection : longueur du chemin

4 = 24 individus possibles : en comptant les chemins dans le sens inverse

Exemple d’implémentation pour un trajet de 10 villes

1. Création (aléatoirement) d’une population de 50 individus sur 3 628 800 individus possibles
2. Évolution sur 10 générations :
   1. On fait 100 croisements
   2. On s´sélectionne les 10 meilleurs

Remarques :

* On n’a pas fait de mutations
* On part sur des distances symétriques
* On a pris un critère d’arrêt arbitraire

Précisions sur le croisement pour le voyageur de commerce

Remarques :

* Il faut garder quelques individus de la génération précédente, par exemple ceux qui n’ont pas été croisés. Sinon on risque de perdre des individus prometteurs.
* On ne peut pas simplement échanger des bouts de chromosomes car il faut que ça reste un chemin valide.

Exemple de solution pour le croisement :

1. On choisit 2 « parents » I1 et I2
2. On recopie le début (ou la ﬁn) de I1 dans un premier ﬁls
3. On complète ce ﬁls en parcourant I2 et en recopiant les villes non déjà présentes dans le ﬁls
4. On fait de même en partant d’I2 pour faire un second ﬁls