République Algérienne Démocratique et Populaire
Ministère de l'enseignement supérieur et de la recherche scientifique

UNIVERSITE Dr. TAHAR MOULAY SAIDA

FACULTE : TECHNOLOGIE

DEPARTEMENT : INFORMATIQUE

POLYCOPIES PEDAGOGIQUES

Intitulé

Nouveaux Paradigmes en Intelligence Artificielle

Présenté par :
RAHMANI Mohamed El Hadi

Intitulé du cours : Nouveaux Paradigmes de l'Intelligence Artificielle

Crédit : 01

Coefficient : 01

Objectifs du cours : Ce cours intitulé le « Nouveaux Paradigmes de l'Intelligence Artificielle » permet de vous familiariser avec quelques techniques des métaheuristiques, ces derniers sont un sous domaine de l'IA, qui consiste à étudier des méthodes inspirées de la nature afin de les appliquer pour résoudre des problèmes qui n'ont pas de solutions exactes.

Le cours est divisé en cinq unités d'apprentissages, chaque unité d'apprentissage est traitée à travers des séquences pédagogiques permettant l'assimilation des concepts prévus, cette assimilation est consolidée par des activités d'apprentissages où ces notions sont mises en œuvre, c'est une des forces de ce cours. L'ensemble des unités d'apprentissage sont décrites ici.

Introduction générale à l'IA : Cette unité consiste à comprendre l'IA, son historique, son rôle et les différentes applications ainsi que les différentes méthodes.

Les algorithmes génétiques AG : Cette unité permet d'étudier les AG inspirés du développement génétique des êtres vivant, l'ensemble des exercices proposé permet d'appliquer les AG pour résoudre des problèmes réels.

Les réseaux de neurones RN : Cette unité permet d'étudier les RN inspirés du réseaux neuronaux des êtres humain, leur type et les différentes applications.

La logique floue : Cette unité permet d'étudier la logique floue et ces applications, l'ensemble des exercices proposé permet d'appliquer la logique floue pour résoudre des problèmes réels.

Les automates cellulaires : Cette unité permet d'étudier les automates cellulaires et ces applications.

Modalité d'évaluation : 67% examen écrit, 33% contrôle contenu.

Table de matière

Chapitre 1 : Introduction à l'intelligence artificielle

Définitions :

Avant de parler de l'intelligence artificielle (IA), il faut d'abord définir le mot intelligence, les chercheurs à travers les années ne peuvent pas donner une définition claire de l'intelligence, Lorenz la définit comme : « C'est collectif et cela émerge du comportement collectif ». Piaget a dit que « l'intelligence est la capacité d'adaptation d'un sujet à son milieu ». En générale, l'intelligence est liée à la capacité d'apprendre, comprendre, et observer selon le comportement de l'individu.

Le mot artificiel veut dire que la machine soit indépendante de l'être humain. Turing a défini l'IA : « C'est ce qui rend difficile la distinction entre une tâche réalisée par un être humain et par une machine ».

L'IA est généralement l'idée de simuler le raisonnement de l'être vivant afin de résoudre des problèmes d'une façon automatique.

Caractéristiques de l'IA :

L'IA a changé le concept de l'informatique grâce a ses caractéristiques qui peuvent être résumé en :

- La représentation symbolique : est la représentation abstraite des connaissances, elle permet de présenter les connaissances qualitatives par des phrases claire et compréhensible par l'être humain.
- Les heuristiques : est un type de résolution de problèmes qui n'ont pas de solution algorithmique facile. L'heuristique veut dire appliquer des algorithmes itératifs afin de trouver une solution optimale a un problème donné.
- La représentation des connaissances : permet de créer une base de connaissances d'un problème qui peut être utilisée dans la résolution de ce problème.
- La capacité de résoudre des problèmes avec des données incomplètes : est une des caractéristiques fortes de l'IA. Avec l'IA, on peut trouver des solutions a des problèmes qui ont des valeurs manquantes dans leur représentation.
- La capacité de traiter les données conflictuelles : Cela signifie qu'avec les techniques de l'IA, on peut affecter des probabilités de vérité à chaque donnée, ce qui aide de traiter le conflit entre ses données.

Les deux pôles de l'IA :

L'IA est divisée en 2 axes principaux :

- *L'ingénierie* qui se base purement sure les notions mathématiques pour résoudre les problèmes, elle prend en considération la notion de complexité pour trouver des solutions le plutôt possible avec le moindre de calcule possible.
- *Les sciences cognitives* qui se base sure l'étude de plusieurs domaines qui ont lié aux fonctions cognitives de l'être humain pour les simuler après la compréhension et la modélisation du problème afin de le résoudre.

Historique de l'IA :

L'idée de l'intelligence artificielle a commencé depuis les année 1950 avec Alan Turing qui a répondu à la question "Can a machine think?". Son idée été simple, deux agents dialoguent d'où la personne ne sais pas qu'elle parle avec une autre personne ou machine. Après ça, Dartmouth présente dans une conférence en 1956 le principe de l'IA, mais le nom Intelligence Artificielle a été proposé après par John McCarthy. Depuis ce temps-là, l'IA a commencé d'être plus vaste.

Les différentes approches de l'IA :

Il existe plusieurs familles d'approche de l'IA, dans cette section on discutera ses différentes familles :

- *Approche cognitive :* consiste à comprendre et reproduire le comportement humain et animal.
- *Approche basée sur une logique mathématique :* consiste à utiliser la logique avec le concept d'enregistrement pour conclure des solutions au questions. Ce genre d'approche nécessite des connaissances parfaites du problème.
- *Approche pragmatiste :* consiste à produire une machine qui réagit comme un être humain en prenant en compte les contraintes du problème.
- *Approche basée sur l'optimisation :* consiste à chercher la meilleure solution qui permet d'atteindre un objectif maximal, noter que cette solution ne soit pas forcément la meilleure solution qui existe dans l'espace de recherche.
- *Approche connexionniste :* consiste à simuler le comportement des réseaux de neurones naturel afin de générer un modèle qui peut apprendre depuis un ensemble de données.

Domaines d'application de l'IA :

L'IA a connu un succès qui la permet d'être utilisée dans plusieurs domaines tel que :

- L'informatique : Traitement des langages naturels, traitement d'image/vidéo, les moteurs de recherches…etc
- Mathématiques : Logique, outils mathématique (graphe et les arbres).
- La médicine
- Robotique
- ….etc

Représentation de la connaissance :

- *Réseaux sémantiques :* base sur la représentation des termes par des nœuds reliés par des arcs qui présente la relation entre ses nœuds, Ces relations peuvent être agrégation (la relation IS A ou EST UN), composition (la relation HAS A ou A UN) ou bien une relation d'instanciation (la relation IS KIND OF ou EST UNE SORTE DE). La figure 1.1. présente un réseau sémantique :

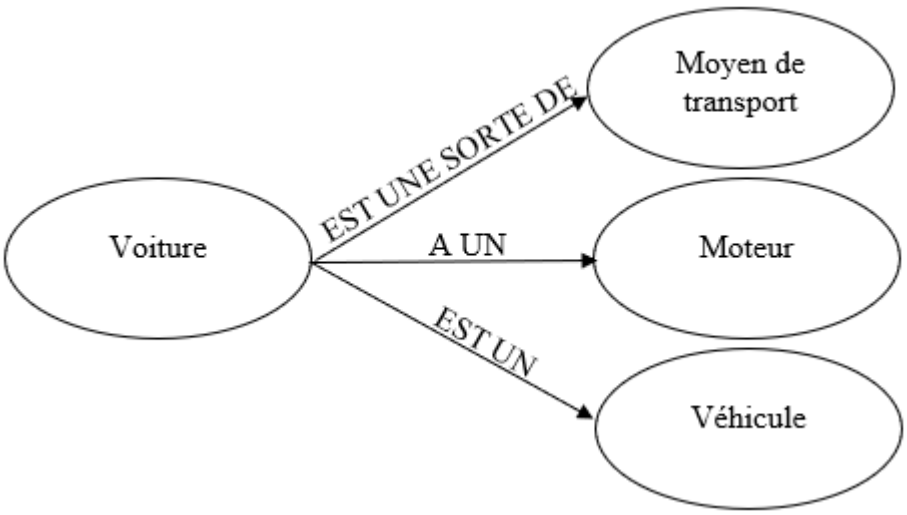


Figure 1.1. Réseau Sémantique

- *Règles de production :* permet de créer une base de règles pour représenter les connaissances du problème. Elle est constituée de trois composants :
  - Contexte : est la description du problème et son état actuel.
  - Une base de règles : sous forme SI <condition> ALORS <conclusion>
  - Un interprète : est le programme qui permet de tirer les conclusions à partir les règles en utilisant un moteur d'inférence.
- *Représentation logique :* Cette technique permet de représenter les connaissances sous forme des formules avec un syntaxe prédéfinis.
- *Représentation en ontologie :* est la représentation des connaissances par une structure constituée d'un ensemble d'individus (concepts, termes,

objets), classes (types d'objets), attributs (caractéristiques de chaque objet) et les relations (liens entre les objets). Cette représentation assure la clarté (définition complète de chaque objet), la cohérence entre les objets, et l'extensibilité (la possibilité d'ajouter des nouveaux objets).

- *Inférence :* est l'utilisation des opérations logiques pour interpréter les connaissances. Il existe trois types d'opérations :
  - Déduction : est l'utilisation d'une base de règles pour tirer des nouvelles connaissances à partir des données initiales.
  - Abduction : est l'utilisation d'une base de règles pour revenir des conclusions au données initiales.
  - Induction : est l'utilisation des données initiales et les conclusions pour trouver la base de règles.
- *Systèmes Experts :* sont un genre de systèmes dédiés pour répondre à des questions des utilisateurs dans un domaine fixe. Un système expert est constitué d'une interface graphique pour faciliter la communication avec les utilisateurs, une base de connaissances du domaine, un moteur d'inférence pour faciliter la recherche des réponses dans la base des règles, et une base de faits contenant les données spécifiques du domaine.

Apprentissage automatique :

L'apprentissage automatique est une discipline de l'IA qui permet de créer des modèles à base d'un ensemble de données pour prendre des décisions au future pour des nouvelles données. Il existe plusieurs types d'apprentissage :

- *Apprentissage Non Supervisé :* consiste à diviser l'ensemble de données en sous-ensembles similaires selon le problème à traiter.
- *Apprentissage Supervisé :* permet de déduire une sortie d'une donnée par un modèle créé à base d'un ensemble de données déjà étiquetées.
- *Apprentissage par Renforcement :* est un type de technique qui permet de prédire les actions en maximisant une récompense donnée.

Chapitre 2 : Les algorithmes Génétiques

Introduction :

Les algorithmes génétiques sont des techniques de la famille des algorithmes évolutionnaires, Ils sont inspirés du théorème de l'évolution de Charles Darwin, Dans son livre « The origin of species », Darwin a prétendu que l'apparition des espèces distinctes se fait par la sélection naturelle. Cette dernière est basée sur l'idée que les individus les plus adoptés à l'environnement survivent plus longtemps que les autres individus. L'adaptation à l'environnement se fait selon les lois de reproduction qui sont le croisement et la mutation, ce qui permet à s'évoluer et s'adapter au changement des conditions d'environnement.

Les algorithmes génétiques ont été inventés par J. Holland en 1975. Ils ont utilisé principalement pour la modélisation des systèmes qui cherche dans un espace de recherche en bits. L'algorithme génétique le plus connu et utilisé est seul de Goldberg 1989, cet algorithme suit plusieurs étapes : le codage, la sélection, le croisement et la mutation. Ces étapes sont effectuées afin de trouver une valeur optimale pour la fonction de fitness, tout ça va être discuté dans ce chapitre.

Le codage des individus :

Le codage des individus est la première étape, elle est considérée comme l'étape clef du l'algorithme vu a son effet sur les prochaines étapes. Il existe deux types de codage :

- *Le codage binaire :* consiste à représenter les individus par des chaines binaires, ce type est le plus utilisé. Soit *f(x)* la fonction à optimiser, la population des solutions possibles *x* est codée en binaire en *n* bits.

  L'avantage du codage binaire est la facilité des opérateurs de croisement et mutation.

  Exemple : Soit $f(x) = x^2$, on cherche le maximum de x pour que la valeur de f(x) soit maximale dans l'intervalle $x \in [0,31]$

  Donc on peut représenter chaque solution possible par 8 bits, cela signifie que 00000000 représente x=0 et 11111111 représente x=31.
- *Le codage réel :* le codage réel présente une deuxième solution pour les problèmes qui ont des difficultés de résolution en utilisant le codage binaire, cette représentation consiste à représenter chaque individu par une suite de nombres réels. Mais, ils demandent aussi une adaptation des opérateurs de croisement et mutation.

La population initiale :

Après le codage des informations, les algorithmes génétiques nécessitent une sélection d'une population initiale afin de commencer ses différents opérateurs. Le choix d'une population initiale n'est pas fixé par une méthode scientifique, généralement la sélection se fera aléatoirement, en cas d'existence des informations a priori sur le problème, la sélection peut être guidée pour trouver des meilleurs résultats. Un autre problème de sélection présente une difficulté, est la taille de la population, surtout que cette dernière peut affecter les résultats finaux. Figure 2.1 présente la hiérarchie d'une population :

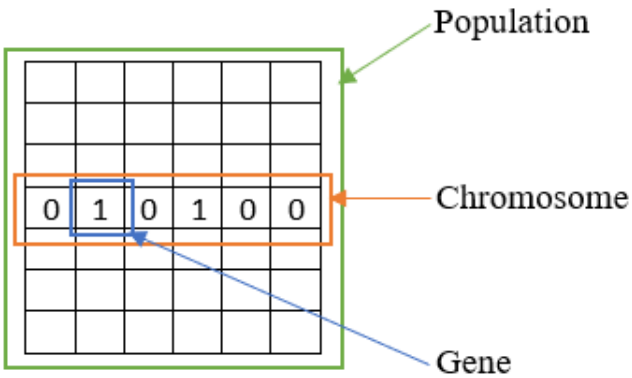


Figure 2.1 : la hiérarchie d'une population

La fonction de fitness :

La fonction de fitness (appelée aussi fonction d'adaptation, ou la fonction objective) est une formule appliquée sur chaque individu pour les comparer et sélection les mieux adapter pour survivre la prochaine itération. Elle peut être une valeur dans des problèmes simple où il y a peu de paramètres, ou bien elle peut être une somme pondérée de plusieurs fonction (ce quand appel optimisation multi-objective) dans des problèmes où il y a plusieurs paramètres, dans ce cas, l'utilisation des poids pondérée pour chaque fonction est nécessaire.

Sélection :

La sélection est la première étape des algorithmes génétiques, elle présente l'étape de choix des individus les plus adaptés à l'environnement pour survivre la prochaine itération, en biologie, la sélection présente la vitalité des chromosomes ce qui cause l'évolution des espèces.

La sélection artificielle est faite par évaluer tous les individus de la population en utilisant la fonction de fitness et choisir les meilleurs individus, soit de l'ensemble de la population, soit un sous-ensemble de la population construit par les meilleurs individus, soit par fixer un seuil et les individus qui vérifie la condition en comparant leur fonction de fitness avec se seuil vont être choisis. Il y a quatre types de sélection :

- *Sélection par roulette (Goldberg) :* Appelé aussi *Loterie Biaisé*. Cette méthode est basée sur probabilité de sélection pour chaque individu, la probabilité est calculée par la formule suivante :

$$P(x_i) = \frac{f(x_i)}{\sum_{j=1}^{N} f(x_j)}$$

Où x_i est l'individu, $f(x_i)$ est sa fonction de fitness, N est le nombre d'individus de la population.

Après calcule de tous les probabilités, on les place dans une roulette de loterie, et on lance un tournement de cette roulette, et on sélectionne l'individu après l'arrêt de la roulette, en fin en relance la roulette pour une autre itération. L'inconvénient majeur de cette technique est que les individus qui ont la plus grande probabilité, ont plus de chance d'être sélectionner.

Exemple : revenant à notre exemple de la recherche de x pour que $f(x) = x^2$ soit maximale dans l'intervalle [0,31].

L'individus	Fonction de fitness	Probabilité d'être sélectionné
12	144	0.169
25	625	0.735
9	81	0.095
Somme des fonctions de fitness	850	1

On tri ces probabilité dans une roulette de loterie comme suite :

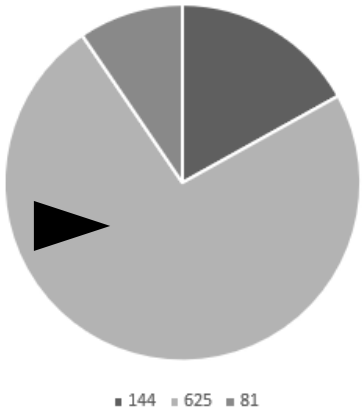


Figure 2.2 : roulette de loterie pour la sélection des individus

On tourne la roulette plusieurs fois d'où a chaque fois on sélectionne la valeur à côté du triangle noir.

- *Sélection par élitisme :* cette base sur un tri des individus selon leur valeurs de fonctions de fitness, puis on sélectionne les M meilleurs individus d'où $0 < M \leq N$. La sélection par élitisme a le même inconvénient comme la sélection par roulette, les individus qui ont mieux de valeurs, ont plus de chance d'être sélectionnés à chaque itération.
- *Sélection par tournoi :* cette méthode est très simple, on tire M individus aléatoirement, et on sélectionne parmi eux le meilleur individu qui a la meilleure valeur de fonction de fitness.
- *Sélection par rang :* cette technique se base sur une probabilité appelée rang, la probabilité est calculée comme suite : rang = $\frac{\sum f_i}{N}$ où N est le nombre d'individus. La sélection après est faite selon le rang d'où l'individu qui a le plus mauvais rang va être sélectionné.

Croisement :

Le croisement est la première phase de génération des nouveaux individus, elle est faite par la prise d'une partie d'un individu et l'autre partie d'un autre individu. L'opération du croisement diffère selon le codage d'individus choisi au début.

- *Croisement des individus binaire :*
  - *Croisement en 1 point :* On choisit un point (appelé break point) qui sépare deux bits aléatoirement, puis, pour le premier fils on prend la première partie (les bits du gauche de point choisi) du premier parent, et la deuxième partie (les bits de la droite de point choisi) du deuxième parent. Pour le deuxième fils, on prend la première partie du deuxième parent, et la deuxième partie du premier parent.
    Exemple : On continue avec notre exemple, prenant les deux parents 2 et 8, avec la représentation binaire on obtiendra :
    2 => 00010 et 8=> 01000
    Si on effectue un croisement dans le deuxième point, on obtiendra :
    00|010 => 00000 = 0
    01|000 => 01010 = 9
    Donc nos nouveaux individus sont 0 et 9 générer à partir leur parent 2 et 8.
  - *Croisement à deux points :* On choisit deux points aléatoirement et on échange les bits des parents entre ces deux points afin d'obtenir des nouveaux individus fils.

- *Croisement en n-points :* On divise les parents par n points choisis aléatoirement, puis on concatène les fragments résultant afin d'obtenir les fils.
- *Croisement uniforme :* On définit une chaine binaire (appelée masque) de même taille que les parents, puis on parcourt ce masque bit par bit, si le bit est égal 1, on échange les bits des parents, sinon, les bits des parents restent tels qu'ils sont. Figure 2.3 présente un exemple de croisement uniforme :

Parent 1	1	1	0	1	0	0
Parent 2	0	0	0	1	1	0
Masque	0	1	1	0	1	0

=>

Fils 1	1	0	0	1	1	0
Fils 2	0	1	0	1	0	0

Figure 2.3 : croisement uniforme

- *Croisement par trois parents :* On compare les bits des deux premiers parents, s'ils ont la même valeur, on la garde, s'ils ont différentes valeurs, on garde la valeur du même bit du troisième parent. Figure 2.4 présente un exemple de croisement par trois parents :

Parent 1	1	1	0	1	0	0
Parent 2	0	0	0	1	1	0
Parent 3	0	1	1	0	1	0

=>

Fils	0	1	0	1	1	0

Figure 2.4 : croisement par trois parents

- *Croisement réel :*
- *Croisement par la méthode Partially Mapped Crossover (PMX) :* On choisit 2 points de croisement d'où on échange les valeurs entre eux, puis on construit une base de règles pour changer les valeurs des gènes à l'extérieur des deux points choisis. Figure 2.5 montre un exemple du croisement PMX :

Parent 1	5	3	7	1	2	9	4	8
Parent 2	0	8	2	7	3	5	1	5

Fils 1				7	3	9		
Fils 2				1	2	5		

7 -> 1 ; 3 -> 2 ; 9 -> 5

Fils 1	**9**	**2**	**1**	7	3	9	4	8
Fils 2	0	8	**3**	1	2	5	**7**	**9**

Figure 2.5 : croisement PMX

- *Croisement Cycle crossover (CX) :* chaque gène du fils 1 prend la valeur du même gène du parent 1 si la valeur de son correspondant dans parent 2 existe dans parent 1. A la fin, on complète les gènes qui restent par le parent opposé. Figure 2.6 présente un exemple détaillé :

Étape 1 :

Parent 1	5	3	7	1	2	8	4	0
Parent 2	8	9	2	7	3	4	1	6

On copie 5 dans fils 1 et 8 dans fils 2 puis on cherche 8 dans parent 1

Étape 2 :

Fils 1	5							
Fils 2	8							

On copie 8 dans fils 1 et 4 dans fils 2 puis on cherche 4 dans parent 1

Étape 3 :

Fils 1	5					8		
Fils 2	8					4		

On copie 4 dans fils 1 et 1 dans fils 2 puis on continue avec tous les valeurs

Étape n :

Fils 1	5	3	7	1	2	8	4	6
Fils 2	8	9	2	7	3	4	1	0

En fin, on échange les valeurs des gènes qui restent

Figure 2.6 : croisement CX

Mutation :

La mutation est la deuxième étape de génération des nouveaux individus. Elle permet de changer une (plusieurs) valeur(s) d'un (plusieurs) gène(s). Les techniques de changement diffère selon le codage choisit au début.

- *Mutation en codage binaire :* est la technique la plus simple, elle consiste à choisir un (plusieurs) gène(s) aléatoirement, et changer leur valeur (si le gène est égal 1, il devient 0, et si le gène est égal 0, il devient 1).
- *Mutation en codage réel :*
  - *Mutation par inversion :* on choisit deux points aléatoires et en inverse les valeurs des gènes situés entre ces deux points. Figure 2.7 montre un exemple de mutation par inversion :

Parent	5	8	1	7	9	3
Fils	5	9	7	1	8	3

Figure 2.6 : mutation par inversion

- *Mutation par insertion :* on choisit au hasard deux gènes, puis on déplace la valeur du deuxième gène sélectionné avant le premier gène sélectionné. Figure 2.7 montre un exemple de mutation par insertion

Parent	5	8	1	7	9	3
Fils	5	9	8	1	7	3

Figure 2.7 mutation par insertion

- *Mutation par déplacement :* on choisit deux points aléatoires, et on déplace les gènes entre eux en une position choisit aussi au hasard. Figure 2.8 montre un exemple de mutation par déplacement :

Parent	5	8	1	7	9	3
Fils	8	1	7	9	5	3

Figure 2.8 mutation par déplacement

- *Mutation par permutation :* on choisit deux gènes aléatoires, et on permute leurs valeurs. Figure 2.9 montre un exemple de mutation par permutation :

Parent	5	8	1	7	9	3
Fils	5	9	1	7	8	3

Figure 2.9 mutation par permutation

Exercices :

- *Exercice 01 :*
  Soit à calculer le maximum de la fonction $f(x)=((1-x)^2)/((1+x))$ définie sur l'intervalle $x \in [0, 31]$. On veut trouver le maximum optimal en utilisant les algorithmes génétiques.

  Donnez la taille en nombre de bits d'un individu. Justifiez votre réponse.

  Commençant par les deux solutions 12 et 5. Appliquez les différents opérateurs de l'algorithme génétique pour quatre générations :

  De croisement (3ème point dans la première itération, 4ème point dans la deuxième itération, 2ème point dans la troisième itération, et 3ème point dans la quatrième itération)

  De mutation (1er bit dans la première itération, et 3ème bit dans la deuxième itération, 4ème bit dans la troisième itération, et 3ème bit dans la quatrième itération)

  De sélection (justifiez votre sélection pour chaque itération).

  Choisissez la solution optimale parmi toutes les solutions générées dans les différentes itérations. Justifiez votre choix.
- *Exercice 02 :*

Le but du problème des huit dames est de placer huit dames d'un jeu d'échecs sur un échiquier de 8 × 8 cases sans que les dames ne puissent se menacer mutuellement, conformément aux règles du jeu d'échecs (la couleur des pièces étant ignorée). Par conséquent, deux dames ne devraient jamais partager la même rangée, colonne, ou diagonale. Afin de trouver une solution pour placer 8 dames (D1, D2, D3, D4, D5, D6, D7 et D8) en utilisant les algorithmes génétiques, on représente chaque position d'une dame par 8 bits (les 4 premier bits représentent la ligne et les autres 4 bits représentent la colonne). Initialement, on commence par les positions suivantes :

D1 (3,1), D2 (8,2), D3 (7,3), D4 (7,5), D5 (1,5), D6 (6,6), D7 (2,7), D8 (3,7).

Donnez la représentation binaire de chaque position.

Donnez la fonction objective et les critères de sélection pour un algorithme génétique.

Appliquez les différents opérateurs de l'algorithme génétique pour deux itérations :

De Croisement (7ème point dans la première itération, et 4ème point dans la deuxième itération)

De mutation (2ème bit dans la première itération, et 6ème bit dans la deuxième itération)

De sélection (Justifiez votre sélection pour chaque itération).

Solutions des exercices :

- *Exercices 01 :*

1. La taille est 5 bits, parce que le maximum dans l'intervalle [0-31] peut être représenté en 5 bits (11111). 0.5
2. 01100 => 12 00101 => 05

croisement iter 1: mutation iter 1:

01101 => 13 0.25 11101 => 29 0.25

00100 => 04 0.25 10100 => 20 0.25

On sélectionne 29 et 20 car le critère (29 et 20 appartient à l'intervalle [0-31]) 0.5

croisement iter 2: mutation iter 2:

11100 => 28 0.25 11000 => 24 0.25

10101 => 21 0.25 10001 => 17 0.25

On sélectionne 24 et 17 car le critère (24 et 17 appartient à l'intervalle [0-31]) 0.5

croisement iter 3: mutation iter 3:

11001 => 25 0.25 11011 => 27 0.25

10000 => 16 0.25 10010 => 18 0.25

On sélectionne 27 et 18 car le critère (27 et 18 appartient à l'intervalle [0-31]) 0.5

croisement iter 4: mutation iter 4:

11010 => 26 0.25 11110 => 30 0.25

10011 => 19 0.25 10111 => 23 0.25

On sélectionne 30 et 23 car le critère (30 et 23 appartient à l'intervalle [0-31]) 0.5

3. La solution optimale est 30 car selon la fonction objective, f(30) = 27.12 est la valeur maximale parmi tous les valeurs testées sur f(x). 0.5

x	12	5	29	20	24	17	27	18	**30**	23
f(x)	9,3	2,66	26,13	17,19	21,16	14,22	24,14	15,21	**27,12**	20,16

- *Exercice 02 :*

1. D1 (3,1), D2 (8,2), D3 (7,3), D4 (7,5), D5 (1,5), D6 (6,6), D7 (2,7), D8 (3,7).
   D1 => 00110001 D2 => 10000010 D3 => 01110011 D4 => 01110101
   D5 => 00010101 D6 => 01100110 D7 => 00100111 D8 => 00110111

0.25*8=2

2. La fonction objective est que dans deux dames ne devraient jamais partager la même rangée, colonne, ou diagonale. Et les critères sont :
   0.5
   $0 > xi \geq 8$ (xi est la ligne du dame i)
   $0 > yi \geq 8$ (yi est la colonne du dame i)
3. D1 => 00110001 =>(3,1) D2 => 10000010 => (8,2)

Croisement iter 1 : mutation iter1 :

00110000 => (3,0) 0.25 01110000 => (7, 0) 0.25

10000011 => (8, 3) 0.25 11000011 => (12, 3) 0.25

On sélectionne les anciennes valeurs de D1 et D2 car y=0 dans le premier fils ne vérifie pas le critère $0 > yi \geq 8$ et la ligne 12 dans (12,3) ne vérifie pas le critère $0 > xi \geq 8$ 0.5

Croisement iter 1 : mutation iter1 :

00110010 => (3,2) 0.25 00110110 => (3, 6) 0.25

10000001 => (8, 1) 0.25 10000101 => (8, 5) 0.25

On sélectionne les nouvelles valeurs de D1 et D2 car elles vérifient tous les critères 0.5

Chapitre 3 : Les réseaux de Neurones

Définition :

Tout d'abord, ce que l'on désigne habituellement par réseau de neurones est en fait un réseau de neurones artificiels basé sur un modèle simplifié de neurone. Ce modèle permet certaines fonctions du cerveau, comme la mémorisation associative, l'apprentissage par l'exemple, le travail en parallèle, mais le neurone artificiel est loin de posséder toutes les capacités du neurone biologique. Les réseaux de neurones biologiques sont ainsi beaucoup plus compliqués que les modèles mathématiques et informatiques.

Il n'y a pas de définition universellement acceptée de réseau de neurones. On considère généralement qu'un réseau de neurones est constitué d'un grand ensemble d'unités. (Ou neurones), ayant chacune une petite mémoire locale. Ces unités sont reliées par des canaux de communication (les connexions, aussi appelées synapses d'après le terme biologique correspondant), qui transportent des données numériques. Les unités peuvent uniquement agir sur leurs données locales et sur les entrées qu'elles reçoivent par leurs connexions.

Certains réseaux de neurones sont des modèles de réseaux biologiques, mais d'autres ne le sont pas. Historiquement l'inspiration pour les réseaux de neurones provient cependant de la volonté de créer des systèmes artificiels sophistiqués, voire intelligents capables d'effectuer des opérations semblables à celles que le cerveau humain effectue de manière routinière, et d'essayer par-là d'améliorer la compréhension du cerveau.

La plupart des réseaux de neurones ont une certaine capacité d'apprentissage. Cela signifie qu'ils apprennent à partir d'exemples, de même que les enfants apprennent à distinguer les chiens des chats à partir d'exemples de chiens et de chats. Le réseau peut ensuite dans une certaine mesure être capable de généraliser, c'est-à-dire de produire des résultats corrects sur des nouveaux cas qui ne lui avaient pas été présentés au cours de l'apprentissage.

Les réseaux de neurones artificiels

Les réseaux de neurones biologiques réalisent facilement un certain nombre d'applications telles que la reconnaissance de formes, le traitement du signal, l'apprentissage par l'exemple, la mémorisation, la généralisation. Ces applications sont pourtant, malgré tous les efforts déployés en algorithmique et en intelligence artificielle, à la limite des possibilités actuelles. C'est à partir de l'hypothèse que le comportement intelligent émerge de la structure et du comportement des éléments de base du

cerveau que les réseaux de neurones artificiels se sont développés. Les réseaux de neurones artificiels sont des modèles, à ce titre ils peuvent être décrit par leurs composants, leurs variables descriptives et les interactions des composants.

1. Composant (le neurone artificiel)

1.1. Structure

La Figure.19 montre la structure d'un neurone artificiel. Chaque neurone artificiel est un processeur élémentaire. Il reçoit un nombre variable d'entrées en provenance de neurones amont. A chacune de ces entrées est associé un poids w abréviation de Wight (poids en anglais) représentatif de la force de la connexion. Chaque processeur élémentaire est doté d'une sortie unique, qui se ramifie ensuite pour alimenter un nombre variable de neurones avals. A chaque connexion est associé un poids.

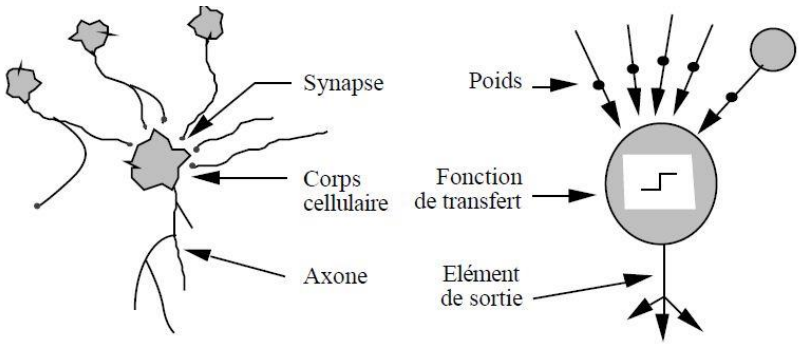


Figure.3.1. Mise en correspondance neurone biologique / neurone artificiel

La Figure.20 donne les notations que nous utilisons dans cet ouvrage.

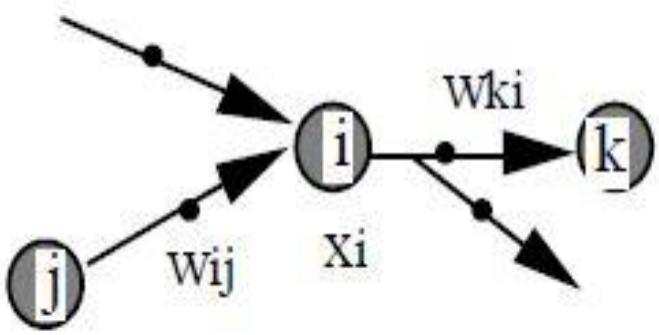


Figure.3.2. Structure d'un neurone artificiel. Pour le neurone d'indice i, les entrées sur celui-ci sont de poids w_{ij} alors que les connexions avals sont de poids w_{ki}.

1.2. Comportement

On distingue deux phases. La première est habituellement le calcul de la somme pondérée des entrées (a) selon l'expression suivante : $a = \sum (w_i . e_i)$

A partir de cette valeur, une fonction de transfert calcule la valeur de l'état du neurone. C'est cette valeur qui sera transmise aux neurones avals. Il existe de nombreuses formes possibles pour la fonction de transfert. Les plus courantes sont présentées sur la Figure. On remarquera qu'à la différence des neurones biologiques dont l'état est binaire, la plupart des fonctions de transfert sont continués, offrant une infinité de valeurs possibles comprissent dans l'intervalle [0, +1] (ou [-1, +1]).

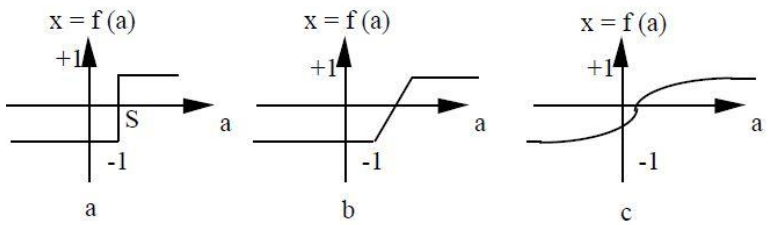


Figure.3.3. Différents types de fonctions de transfert pour le neurone artificiel

a: fonction à seuil (S, la valeur du seuil)

b: linéaire par morceaux

c: sigmoïde.

Nous constatons que les équations décrivant le comportement des neurones artificiels n'introduisent pas la notion de temps. En effet, et c'est le cas pour la plupart des modèles actuels de réseaux de neurones, nous avons affaire à des modèles à temps discret, synchrone, dont le comportement des composants ne varie pas dans le temps.

2. Variables descriptives

Ces variables décrivent l'état du système. Dans le cas des réseaux de neurones qui sont des systèmes non autonomes, un sous-ensemble des variables descriptives est constitué par les variables d'entrée, variables dont la valeur est déterminée extérieurement au modèle.

3. Structure d'interconnexion

Les connexions entre les neurones qui composent le réseau décrivent la topologie du modèle.

Elle peut être quelconque, mais le plus souvent il est possible de distinguer une certaine régularité.

Réseau multicouche (au singulier) : les neurones sont arrangés par couche. Il n'y a pas de connexion entre neurones d'une même couche et les connexions ne se font qu'avec les neurones des couches avalent (fig. 3.22). Habituellement, chaque neurone d'une couche est connecté à tous les neurones de la couche suivante et celle-ci seulement. Ceci nous permet d'introduire la notion de sens de parcours de l'information (de l'activation) au sein d'un réseau et donc définir les concepts de neurone d'entrée, neurone de sortie. Par extension, on appelle couche d'entrée l'ensemble des neurones d'entrée, couche de sortie l'ensemble des neurones de sortie. Les couches intermédiaires n'ayant aucun contact avec l'extérieur sont appelés couches cachées.

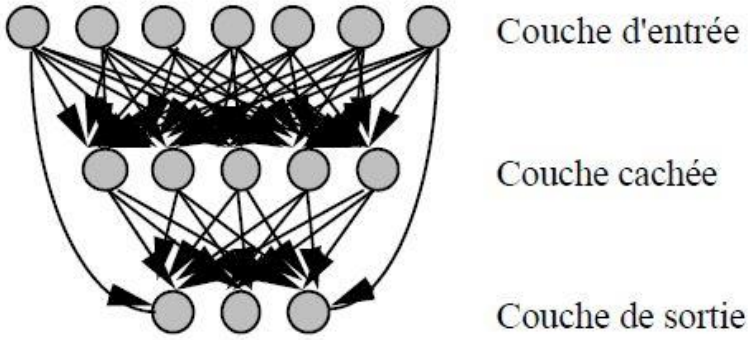


Figure.3.4. Définition des couches d'un réseau multicouche.

Réseau à connexions locales : Il s'agit d'une structure multicouche, mais qui à l'image de la rétine, conserve une certaine topologie. Chaque neurone entretien des relations avec un nombre réduit et localisé de neurones de la couche avale (fig. 3.23). Les connexions sont donc moins nombreuses que dans le cas d'un réseau multicouche classique.

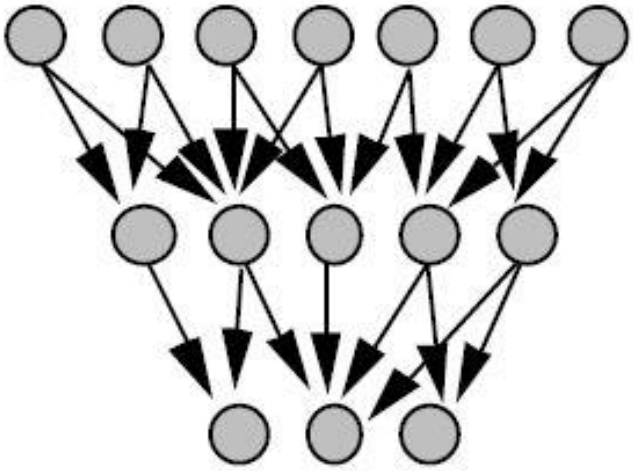


Figure.3.5. Réseau à connexions locales

Réseau à connexions récurrentes : les connexions récurrentes ramènent l'information en arrière par rapport au sens de propagation défini dans un réseau multicouche. Ces connexions sont le plus souvent locales (fig. 6).

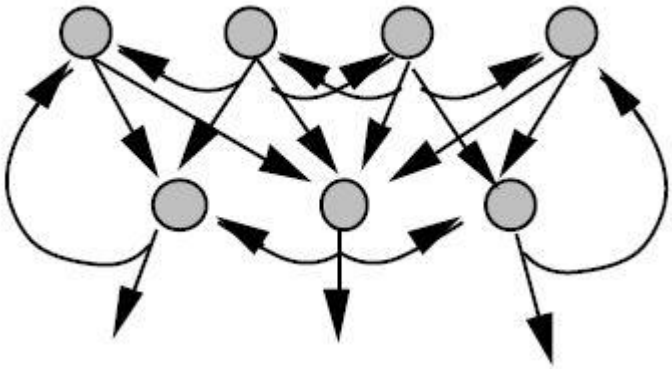


Figure.3.6. Réseau à connexions récurrentes

Réseau à connexion complète : c'est la structure d'interconnexion la plus générale (fig. 7). Chaque neurone est connecté à tous les neurones du réseau (et à lui-même).

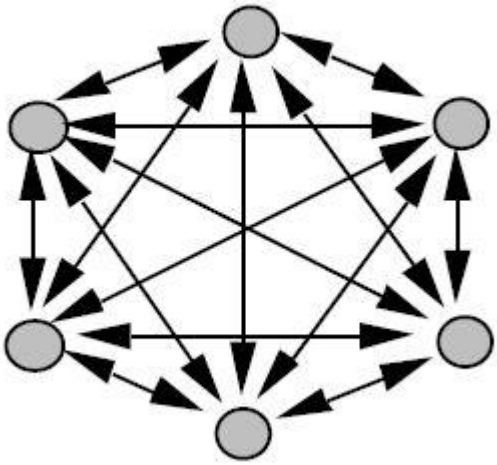


Figure.3.7. Réseau à connexions complète

Il existe de nombreuses autres topologies possibles, mais elles n'ont pas eu à ce jour la notoriété des quelques-unes que nous avons décrite ici.

Modélisation d'un neurone :

Le fonctionnement d'un neurone artificiel peut être modélisé par le schéma suivant :

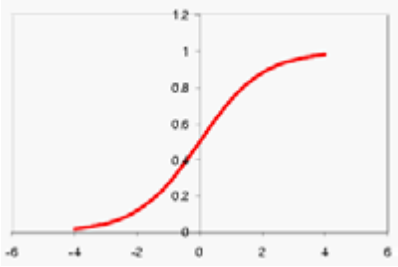
Sur ce schéma, le neurone a trois connexions en entrée le reliant à trois autres neurones. Il reçoit de l'information provenant de chacun de ces trois neurones. Les valeurs qu.il reçoit ainsi en entrée par chacune de ses connexions sont respectivement notées e1, e2 et e3 : ce sont les **entrées** du neurone.

Toutes les connexions n'ont pas une importance égale pour le neurone. Certaines sont plus importantes que d'autres. Le **poids** w affecté à chaque connexion mesure cette importance relative : le poids est proportionnel à l'importance de la connexion. En fait, tout se passe comme si le neurone ne recevait qu'une entrée E et que celle-ci prenait la valeur

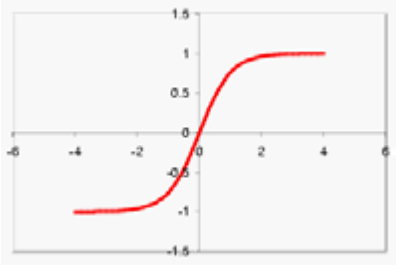
E=w1*e1+w2*e2+w3*e3.

Une fois l'entrée connue, le neurone effectue une opération qui dépend de E. Cela revient à dire qu.il applique une fonction f à la valeur E. Cette fonction f est appelée **fonction d'activation.** Le choix de cette fonction f se révèle être un élément constitutif important des réseaux de neurones. Ainsi, l'identité est rarement suffisante et des fonctions non linéaires et plus évoluées sont nécessaires. A titre illustratif voici quelques fonctions couramment utilisées comme fonctions d'activation :

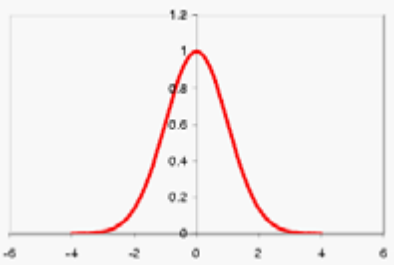
- le sigmoïde standard (encore appelé fonction logistique) :



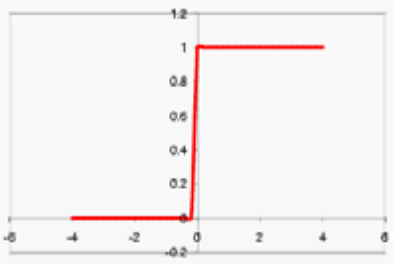
- la tangente hyperbolique :



- la fonction Gaussienne :



- une fonction à seuil :



La valeur f(E) calculée par le neurone constitue la **sortie** S. Cette valeur en sortie devient ensuite une valeur d'entrée pour tous les neurones avec lesquels notre neurone de référence est connecté.

Modélisation d'un réseau de neurones :

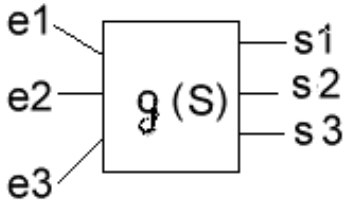


schéma général d'un réseau de neurone

Un réseau de neurone présente la même structure que chacun de ses neurones si on le regarde dans sa globalité. Il doit pouvoir calculer des valeurs de sorties (s1, s2, s3) en fonction de variables d'entrées (e1, e2, e3). Une première série de neurones applique aux entrées e1, e2, e3 leur propre fonction d'activation, ce qui fournit un certain nombre de résultats. Une seconde série de neurones prend ces résultats en entrée et calculent de nouveau, avec leur propre fonction d'activation, des résultats qu'ils

transmettent à la série de neurones suivante, etc..., jusqu'à atteindre la dernière série de neurones : les sorties de ces derniers neurones sont alors les sorties du réseau s1, s2, s3.

Contrairement à chacune des fonctions d'activation f, la fonction g qui transforme les valeurs d'entrée en valeurs de sortie à l'échelle du réseau ne peut pas être explicitée facilement. Elle est en effet beaucoup plus compliquée puisqu'elle est en quelque sorte constituée de la superposition de toutes les fonctions f de chaque neurone.

Un réseau de neurones peut donc être représenté par les poids w des différents neurones. Ces poids peuvent varier au cours du temps, en fonction des entrées présentées E. Le grand problème est alors de savoir comment modifier ces poids (c'est-à-dire en d'autres termes de trouver une loi équivalente à dw/dt=g(E,w)).

Les réseaux de neurones peuvent donc réaliser un filtre séparateur (les réseaux filtrent ou classent les entrées, la sortie étant la classe correspondant à l'entrée) non linéaire (la fonction g n est pas linéaire, c'est-à-dire que les sorties ne s'obtiennent pas linéairement à partir des entrées), et adaptatif (les poids peuvent varier au cours du temps, ce qui modifie le réseau).

Sans avoir besoin d'apprendre au réseau des règles logiques ni de stocker des données, en leur présentant successivement des exemples, ces réseaux sont ainsi capables de trouver une régularité, et de séparer les entrées en différentes classes. Les premiers réseaux de neurones formels ont été conçus par W. McCulloch et W. Pitts en 1943.

Connectivité :

La connectivité des réseaux est la manière dont les neurones sont connectés entre eux. Elle peut être totale (tous les neurones sont connectés entre eux) ou organisée par couche (les neurones d'une couche ne sont connectés qu'à la couche précédente en entrée et à la couche suivante en sortie). Il existe des réseaux **monocouches** ou **multicouches**.

Apprentissage supervisé / non supervisé

Une caractéristique des réseaux de neurones est leur capacité à apprendre (par exemple à reconnaître une lettre, un son...). Mais cette connaissance n'est pas acquise dès le départ. La plupart des réseaux de neurones apprennent par l'exemple en suivant un algorithme d'apprentissage. Il y a deux algorithmes principaux : l'apprentissage supervisé et l'apprentissage non supervisé.

Lors d'un **apprentissage supervisé**, les résultats corrects (c'est-à-dire les valeurs que l'on désire que le réseau obtienne en sortie) sont fournis au réseau, si bien que celui-ci peut ajuster ses poids de connexions pour les obtenir. Après l'apprentissage, le réseau est testé en lui donnant seulement

les valeurs d'entrée mais pas les sorties désirées, et en regardant si le résultat obtenu est proche du résultat désiré.

Lors d'un **apprentissage non supervisé**, on ne fournit pas au réseau les sorties que l'on désire obtenir. On le laisse évoluer librement jusqu'à ce qu'il se stabilise.

Calcul des poids synaptiques :

La **rétropropagation** est une méthode de calcul des poids (aussi appelés poids synaptiques du nom des synapes, terme désignant la connexion biologique entre deux neurones) pour un réseau à apprentissage supervisé qui consiste à minimiser l'erreur quadratique de sortie (somme des carrés de l'erreur de chaque composante entre la sortie réelle et la sortie désirée).

D'autres méthodes de modification des poids sont plus "locales", chaque neurone modifie ses poids en fonction de l'activation ou non des neurones proches

Quelques réseaux célèbres :

Il y a de très nombreuses sortes de réseaux de neurones actuellement. Personne ne sait exactement combien. De nouveaux réseaux (ou du moins des variations de réseaux plus anciens) sont inventés chaque semaine. On en présente ici de très classiques.

1. Perceptron :

C'est un des premiers réseaux de neurones, conçu en 1958 par Rosenblatt. Il est linéaire et monocouche. Il est inspiré du système visuel. La première couche (d'entrée) représente la rétine. Les neurones de la couche suivante (unique, d'où le qualificatif de monocouche) sont les cellules d'association, et la couche finale les cellules de décision. Les sorties des neurones ne peuvent prendre que deux états (-1 et 1 ou 0 et 1).

Seuls les poids des liaisons entre la couche d'association et la couche finale peuvent être modifiés. La règle de modification des poids utilisée est la règle de Widrow-Hoff : si la sortie du réseau (donc celle d'une cellule de décision) est égale à la sortie désirée, le poids de la connexion entre ce neurone et le neurone d'association qui lui est connecté n'est pas modifié.

Dans le cas contraire le poids est modifié. Proportionnellement. À la différence entre la sortie obtenue et la sortie désirée :

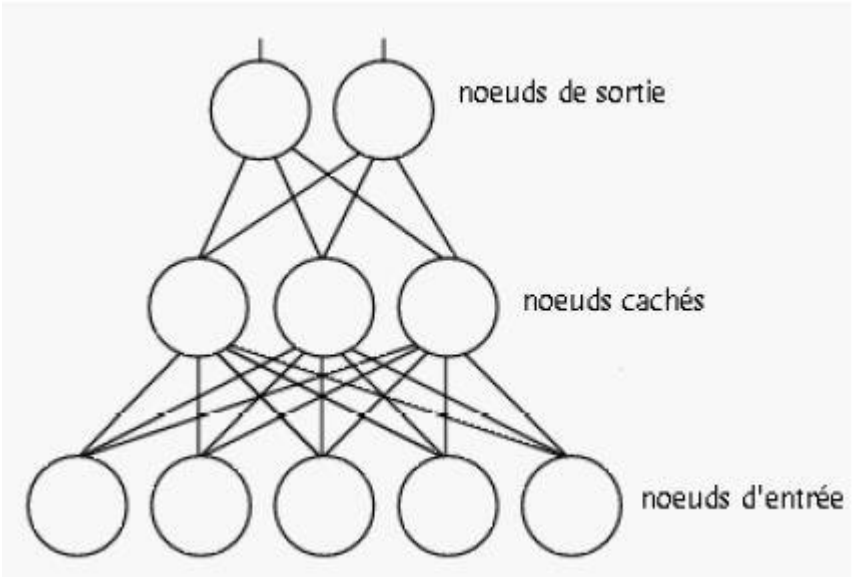
w <= w + k (d - s)

Où s est la sortie obtenue, d la sortie désirée et k une constante positive.

En 1969, Papert et Minsky ont démontré les limites du perceptron classique, incapable, par exemple de simuler la fonction ou exclusif (xor).

2. Les perceptrons multicouches (PMC) :

Ils sont une amélioration du perceptron comprenant une ou plusieurs couches intermédiaires dites couches cachées, dans le sens où elles n'ont qu'une utilité intrinsèque pour le réseau de neurones et pas de contact direct avec l'extérieur. Chaque neurone n'est relié qu'aux neurones des couches directement précédente et suivante, mais à tous les neurones de ces couches.



Les PMC utilisent, pour modifier leurs poids, un algorithme de rétropropagation du gradient qui est une généralisation de la règle de Widrow-Hoff. Il s'agit toujours de minimiser l'erreur quadratique. On propage la modification des poids de la couche de sortie jusqu'à la couche d'entrée. Les PMC agissent comme un séparateur non linéaire et peuvent être utilisés pour la classification, le traitement de l'image ou l'aide à la décision.

3. Les réseaux de Hopfield :

Il s'agit d'un réseau constitué de neurones à deux états (-1 et 1, ou 0 et 1), dont la loi d'apprentissage est la règle de Hebb (1949), qui veut qu'une synapse améliore son activité si et seulement si l'activité de ses deux neurones est corrélée (c'est-à-dire que le poids d'une connexion entre deux neurones augmente quand les deux neurones sont activés au même temps).

4. Les réseaux de Kohonen :

Contrairement aux réseaux de Hopfield où les neurones sont modélisés de la façon la plus simple possible, on recherche ici un modèle de neurone plus proche de la réalité. Ces réseaux sont inspirés des observations biologiques du fonctionnement des systèmes nerveux de perception des mammifères.

Une loi de Hebb modifiée (tenant compte de l'oubli) est utilisée pour l'apprentissage. La connexion est renforcée dans le cas où les neurones reliés ont une activité simultanée, et diminuée dans le cas contraire (alors qu.il ne se passait précédemment rien dans ce cas). Ceci se résume par la formule :

d w / dt = k S e - B(S) w

Où w est le poids associé à une certaine connexion, e la valeur que le neurone reçoit en entrée par cette connexion, S la valeur qu.il renvoie en sortie (toujours positive), B(S) la fonction d'oubli et k une constante positive.

Une loi d'interactions latérale. (Observée biologiquement) est aussi modélisée. Les neurones très proches (physiquement) interagissent positivement (le poids des connexions est augmenté autour d.une certaine zone quand une synapse est activée), négativement pour les neurones un peu plus loin, et pas du tout pour les neurones éloignés. Ceci crée un "amas" de neurones activés et contribue à spécialiser certains neurones : pour une entée donnée, une sortie particulière sera activée alors que les autres resteront inertes. On utilise aussi parfois des lois de concurrence entre les neurones (création et destruction de neurones selon certains critères).

Ceci permet de résoudre certains problèmes, dits NP complets, tel le problème du voyageur de commerce (comment relier n villes par le chemin le plus court).

Les réseaux de Kohonen ont des applications dans la classification, le traitement de l'image, l'aide à la décision et l'optimisation.

Chapitre 4 : La logique floue

Introduction :

La logique floue est un ensemble de techniques qui permettent de vérifier un ensemble de condition en introduisant la notion de degré et un ensemble d'état des conditions différent de vrai et faux. Autrement dit, la logique floue est une modélisation qui vise à prédire une variable subjective (par exemple : la moyenne de cet étudiant est bien) et n'est pas objective (la moyenne de cet étudiant est 14.31).

Afin de comprendre les principes de la logique floue, on va concevoir au fil de ce cours un système d'inférence qui a pour but de noter les applications du transport privé en Algérie tels que Yassir, Temtem, Heetch…etc.

Les sous-ensembles flous :

En 1965, Lotfi Zadeh a présenté la notion de la logique booléenne qui se base sur les ensembles classiques. La logique floue est donc une extension de la logique de Zadeh, en ajoutant les théories mathématiques des ensembles flous. Dans ce chapitre, on utilisera les termes *« les sous-ensembles flous »* et *« les ensembles flous »*, d'où les ensembles classiques sont appelés **ensembles nets**.

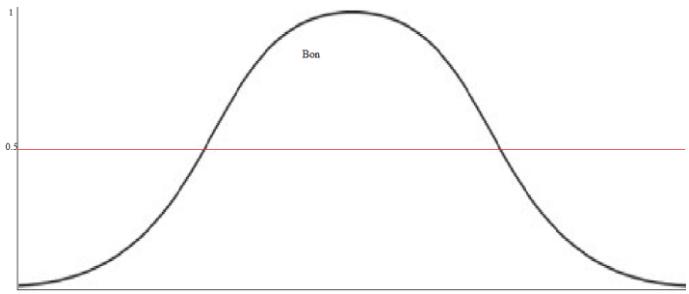


Figure 4.1 : Fonction d'appartenance des notes des services de transport caractérisant les sous-ensembles bons

Définition 1 : Soit X un ensemble, un sous-ensemble Y de X est caractérisé par la fonction d'appartenance : f(x) : X → [0,1]

Dans notre exemple, il faut redéfinir des fonctions d'appartenance de chaque sous-ensemble flou pour chacune des quatre variables :

- Input 1 : état de la voiture : Nouvelle, Ancienne.
- Input 2 : qualité de la conduite : Confortable, Inconfortable.
- Input 3 : temps de réponse à la demande : Rapide, Retard.
- Output : note du service : Bon, Moyen, Mauvais.

La forme de chaque sous-ensemble est choisie selon les expériences des autres clients en faisant des études statistiques.

Prenant par exemple l'état de la voiture, la figure 2.1 représente la fonction caractéristique d'un ensemble classique, et la figure 2.2 représente la fonction d'appartenance d'un ensemble flou :

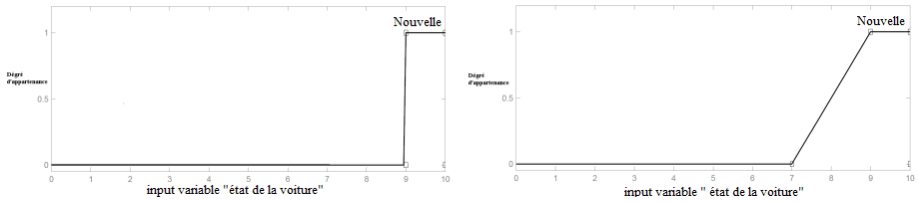


Figure 4.2 : Comparaison entre fonction caractéristique d'un ensemble classique et fonction d'appartenance d'un ensemble flou

Afin de détailler les caractéristiques d'un ensemble flou, on a besoin de définir les caractéristiques d'un ensemble classique

Soit un ensemble X, Y un sous-ensemble flou de X et A_Y la fonction d'appartenance qui le caractérise.

Définition 2 : La **Hauteur** de Y, notée h(Y), est la borne supérieure de la fonction d'arrivée de sa fonction d'appartenance : $h(Y) = \sup\{A_Y(x)|x \in X\}$

Définition 3 : Y est dit **normalisé** si et seulement si h(Y)=1 en pratiques.

Définition 4 : Le **support** de Y est tout élément x de X d'où supp(Y)={x∈X|A_Y(x)>0}

Définition 5 : Le **noyau** de Y est tout élément x de X d'où supp(Y)={x∈X|A_Y(x)=1}

Définition 6 : un **α-coupe** de Y est tout élément x de X d'où α-coupe(x)={x∈X|A_Y(x)≥α}

Si on applique ces caractéristiques sur la fonction d'appartenance pour note = bon, on obtient la figure 3

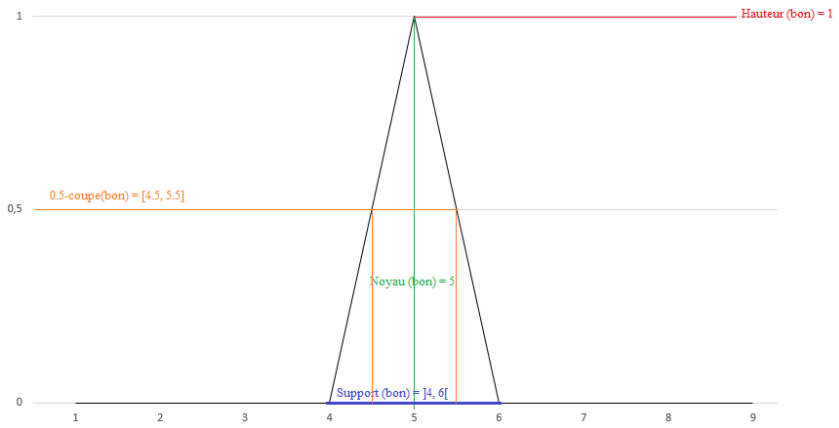


Figure 4.3 : Caractéristiques d'un ensemble flou

Note : Si Y est un ensemble classique, on obtient :

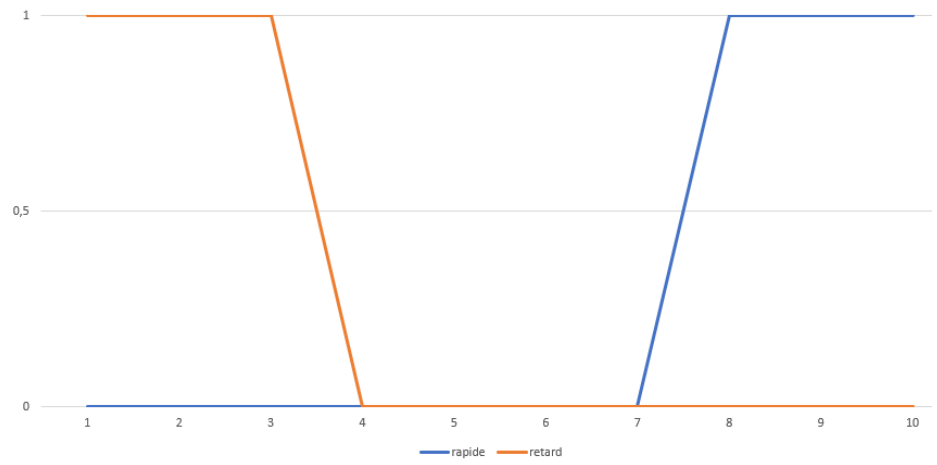
- Supp(Y) = Noyau(Y)
- h(Y) = $\begin{cases} 1 \; si \; Y \neq \emptyset \\ 0 \; si \; Y = \emptyset \end{cases}$

Les variables linguistiques :

Une variable linguistique est le triplet qui relie une variable, sa plage de valeurs, et l'ensemble des sous-ensembles flous.

Définition 7 : Soit V une variable, X_V la plage des valeurs possibles de V, et T_V l'ensemble des sous-ensembles flous de la variable V. Prenant

l'exemple de la variable V= temps de réponse à la demande, la **variable linguistique** correspond au triplet (V, X_V, T_V) peut être représentée dans la figure 4 :



V = temps de réponse à la demande

$X_V = R^+$

T_V = {rapide, retard}

Figure 4.4 : Variable linguistique « temps de réponse à la demande »

Les opérateurs flous

Dans la logique, on trouve trois opérateurs principaux : la négation (NON), l'intersection (ET) et l'union (OU). Ces opérateurs diffèrent de la théorie des ensembles classiques présenté par Zadeh et la théorie des ensembles flous. Le tableau suivant démontre la différence :

Dénomination	Intersection ET : $A_{Y \cap Z}(x)$	Réunion OU : $A_{YUZ}(x)$	Complément NON : $A_{\bar{Y}}(x)$
Opérateurs de Zadeh MIN/MAX	Min ($A_Y(x)$, $A_Z(x)$)	Max ($A_Y(x)$, $A_Z(x)$)	1- $A_Y(x)$
Probabiliste	$A_Y(x) \times A_Z(x)$	$A_Y(x) + A_Z(x)$	1- $A_Y(x)$

PROD/PROBOR		- $A_Y(x) \times A_Z(x)$	

Avec les définitions usuelles des opérateurs flous, nous retrouvons toujours les propriétés de commutativité, distributivité et associativité des opérateurs classiques. Cependant, relevons deux exceptions notables :

- En logique floue, on peut trouver un élément x appartient à l'ensemble X et en même temp appartient à non X. C-à-d : $Y \cap \overline{Y} \neq Ø$.
- En logique floue, le principe du tiers exclu est contredit : $Y U \overline{Y} \neq X$.

Le raisonnement en logique floue

En logique classique, le raisonnement suit des règles de la forme :

$$\begin{cases} si\ p\ alors\ q \\ p\ vrai\ alors\ q\ vrai \end{cases}$$

En logique floue, le raisonnement suit des règles floue de la forme : si $x \in X$, et $y \in Y$ alors $z \in Z$. Ces règles peuvent être exprimer en langage naturel par les variables linguistiques discutées dans la première partie. Prenant notre exemple des services de transport :

Si le temp de réponse à la demande est rapide alors la note du service est Bon.

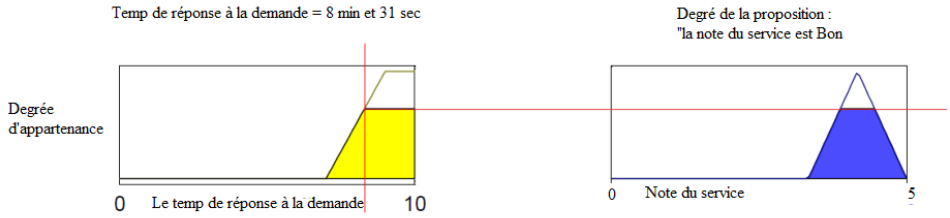
Dans ce cas, la variable *note de service* appartient à l'ensemble flou *Bon*, Cette appartenance dépend du degré de l'appartenance de la variable *temp de réponse à la demande* dans l'ensemble flou *rapide*. Par conséquent, on déduit que plus les règles proposées sont vérifiées, plus l'action préconisée pour les sorties doit être respectée. Pour connaître le degré de vérité de la proposition floue *la note du service* sera élevé, nous devons définir l'implication floue. Il n'existe pas une méthode unique de l'application floue, parmi tous les implications floue proposées, le concepteur doit choisir ses implications pour créer son système flou, ou bien les définis à nouveau. Il existe deux définitions de l'implication floue qui sont les plus couramment utilisées :

- Mamdani : Min ($A_Y(x)$, $A_Z(x)$)
- Larsen : $A_Y(x) \times A_Z(x)$

En appliquant la définition de Mamdani sur notre règle :

Si le temp de réponse à la demande est rapide alors la note du service est Bon.

Si la réponse à la demande est faite dans 8 min et 31 sec, on obtiendra :



On conclut alors que l'application d'une règle floue dépend de trois facteurs principaux :

1. L'implication floue choisie
2. La fonction d'appartenance de la conclusion de règle
3. Le degré de la validité des propositions situées

En intégrant les opérateurs NON, ET et OU, on peut maintenant former une conjonction de propositions floues, ce qui nous donne un ensemble de règles qui forment un système flou, cet ensemble est appelé ***La matrice des décisions***.

Une matrice des décisions est l'ensemble des règles d'un système flou. Continuant dans notre exemple, on peut construire une matrice des décisions comme suite :

Si *état de voiture* est *Nouvelle* ET *temp de réponse à la demande* est *Rapide*	Alors *la note du service* est *Bon*
Si *état de voiture* est *Nouvelle* ET *temp de réponse à la demande* est *Retard*	Alors *la note du service* est *Moyen*
Si *état de voiture* est *Ancienne* OU *temp de réponse à la demande* est *Retard*	Alors *la note du service* est *Mauvais*

On conclut alors que l'application d'une règle floue dépend de trois facteurs principaux :

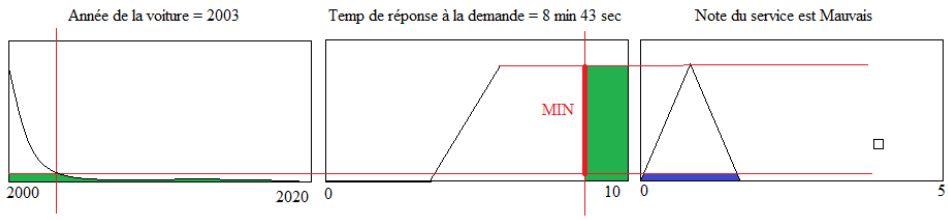
4. L'implication floue choisie
5. La fonction d'appartenance de la conclusion de règle
6. Le degré de la validité des propositions situées

En intégrant les opérateurs NON, ET et OU, on peut maintenant former une conjonction de propositions floues, ce qui nous donne un ensemble de règles qui forment un système flou, cet ensemble est appelé ***La matrice des décisions***.

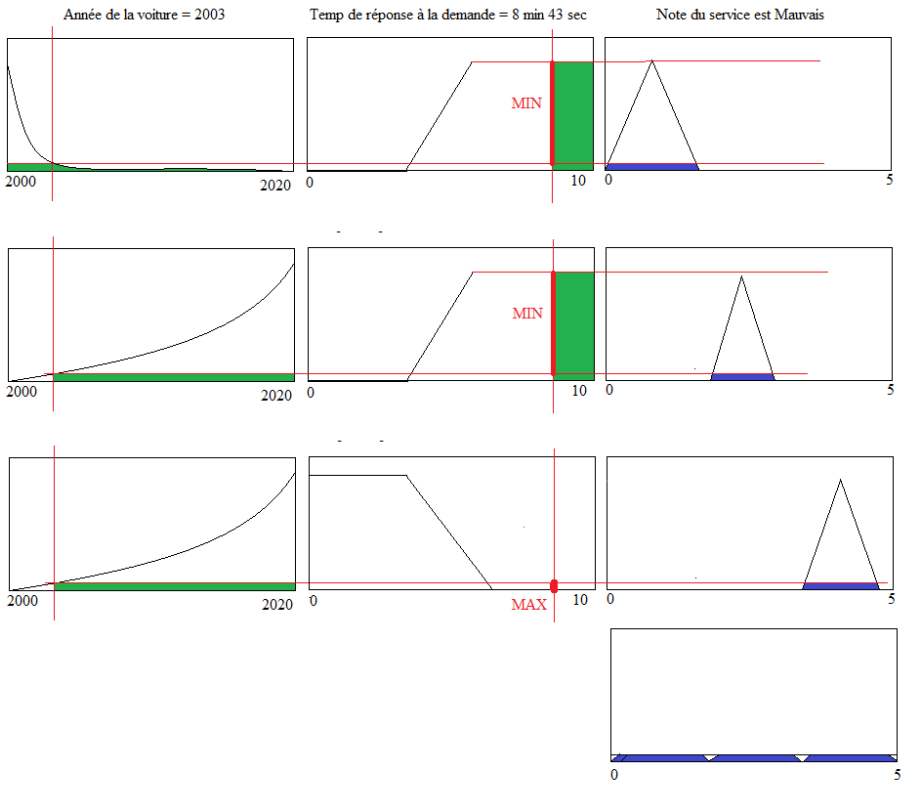
Une matrice des décisions est l'ensemble des règles d'un système flou. Continuant dans notre exemple, on peut construire une matrice des décisions comme suite :

Si *état de voiture* est *Ancienne* ET *temp de réponse à la demande* est *Retard*	alors *la note du service* est *Mauvais*
Si *état de voiture* est *Nouvelle* ET *temp de réponse à la demande* est *Retard*	alors *la note du service* est *Moyen*
Si *état de voiture* est *Nouvelle* OU *temp de réponse à la demande* est *Rapide*	alors *la note du service* est *Bon*

Prenant la règle « Si *état de voiture* est *Ancienne* ET *temp de réponse à la demande* est *Retard* alors *la note du service* est *Mauvais* » lorsque l'année de la voiture est 2002 et le temp de réponse à la demande est 8 minutes 43 secondes, si on applique l'implication de Mamdani avec la traduction de ET par min, on obtiendra :



De cette façon, on continue avec tous les règles, on obtiendra :



Après l'application de tous les règles de notre système flou, on peut prendre une décision sachant que l'année de la voiture est 2003 et le chauffeur a pris 8 minutes et 43 secondes pour répondre à la demande. Cette étape permet de passer de l'ensemble flou à une décision, s'appelle ***la défuzzification***.

La Défuzzification

Ils existent plusieurs méthodes de défuzzification, dans ce chapitre, on définira les 2 méthodes : moyenne des maximas, et centre de gravité.

- Moyenne des maximas consiste à calculer la moyenne des abscisses des maximas de l'ensemble flou.
- La méthode du centre de gravité consiste à calculer le centre de gravité de la surface couverte par la fonction d'appartenance de la conclusion.

Chapitre 5 : Les automates cellulaires

Introduction :

La littérature sur les automates cellulaires est immense et les ressources Internet qui y sont consacrées sont légions. L'objectif ici est beaucoup plus limité. Ce site se voulant vouer aux profanes, je me contenterai d'apporter des éléments de réponse aux deux questions essentielles que se pose toute personne découvrant les auto- mates cellulaires, généralement après une phase d'intense perplexité :

- Mais qu'est-ce que cela peut bien être ?
- À quoi est-ce que cela peut bien servir ?

La réponse à ces questions est malheureusement loin d'être simple. Les automates cellulaires sont des constructions abstraites aux propriétés très complexes et dont l'abord n'est pas immédiat.

Historique

On fait généralement remonter l'histoire des automates cellulaires aux années quarante et à Stanislas Ulam. Ce mathématicien s'est intéressé à l'évolution de constructions graphiques engendrées à partir de règles simples. La base en était un espace à deux dimensions divisées en « cellules », soit une sorte de feuille quadrillée. Chacune des cellules pouvait avoir deux états : allumé ou éteint. Partant d'une configuration donnée, la génération suivante était déterminée en fonction de règles de voisinage. Par exemple, si une cellule donnée était en contact avec deux cellules allumées elle s'allumait sinon elle s'éteignait. Ulam, qui utilisait l'un des premiers ordinateurs, a rapidement constaté que ce mécanisme permettait de générer des figures complexes et esthétiques et que dans certains cas, ces figures pouvaient se répliquer. Des règles extrêmement simples permettaient de cons- truire des structures très complexes. À partir de là, se posait la question suivante : ces mécanismes récursifs (c'est-à-dire en l'occurrence dépendant de leur propre état antérieur) peuvent-ils expliquer la complexité du réel ? Cette complexité n'est-elle qu'apparente, les lois fondamentales étant elles-mêmes simples?

En parallèle, John von Neumann (fort des travaux de A. Turing) s'intéressait à la théorie des automates autoréplicateurs et travaillait à la conception d'une machine autoréplicatrice le « kinématon. » Une telle machine devait être capable, à partir de matériaux trouvés dans l'environnement, de produire n'importe quelle machine décrite dans son programme, y compris une copie d'elle-même. Von Neumann montrait

ici comment résoudre le problème de l'autoréférence de la description. Pour s'autorépliquer, la machine devrait en effet contenir une description d'elle-même, mais pour être complète, cette description doit également être décrite, etc. La solution réside dans la capacité donnée à la machine d'interpréter sa description à la fois comme un programme, une séquence d'instruction, et comme un composant. La description sera d'abord interprétée pour construire la nouvelle machine, elle sera ensuite simplement copiée afin de donner à la nouvelle machine une description d'elle-même. Ce mécanisme correspond de fait à l'interprétation actuelle du fonctionnement de la molécule d'ADN découverte après les travaux de von Neumann. A.C. Clarke a rendu les machines de von Neumann célèbre avec la série « *2001 Odyssée de l'espace.* » Pour transformer Jupiter en étoile, un premier monolithe se reproduit, les descendants font de même, la population croît ainsi de manière exponentielle pour atteindre rapidement la taille nécessaire à la réalisation d'une aussi gigantesque tâche.

C'est S. Ulam qui a suggéré à von Neumann d'utiliser ce qu'il appelait les « espaces cellulaires » (cellular spaces) pour construire sa machine autoréplicatrice. Il pouvait ainsi s'affranchir des conditions physiques réelles pour travailler dans un univers extrêmement simplifié pourtant apte à engendrer une haute complexité. Le passage à cet univers formel l'a amené à constater :

« En axiomatisant les automates [autoréplicateurs] de cette manière, on a jeté la moitié du problème par la fenêtre et c'est peut-être la moitié la plus importante. On s'est résigné à ne pas expliquer comme ces éléments sont constitués de choses réelles, particulièrement comment ces éléments sont constitués de particules élémentaires ou même de molécules (…) on considérera simplement que des parti- cules élémentaires dotées de certaines propriétés existent. La question à laquelle on espère répondre, ou au moins examiner, est : Quels principes sont mis en œuvre dans l'organisation de ces molécules dans les êtres vivants fonctionnels (…) Je discuterai de tout cela seulement de ce point de vue limité.»

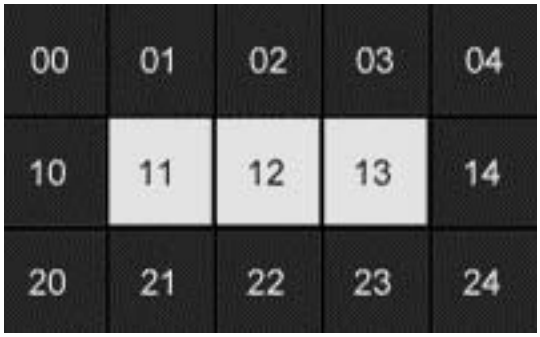
Sur cette base, il conçut un automate cellulaire de quelques 200.000 cellules à 29 états contenant un copieur universel, une description de lui-même et une machine de Turing pour la supervision.

Les automates cellulaires sont sortis des laboratoires en 1970 avec le désormais fameux Jeu de la vie (Life Game) de John Horton Conway.

Le Jeu de la vie

À l'origine, le Jeu de la vie fut présenté comme un jeu mathématique. Sa description va nous permettre de matérialiser et mieux comprendre ce que sont les auto- mates cellulaires.

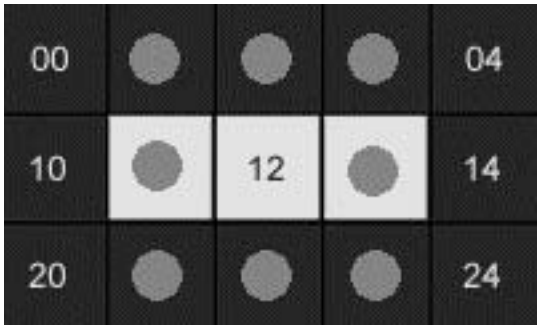
À l'instar des espaces cellulaires d'Ulam, le Jeu de la vie se présente sous la forme d'une grille constituée de cellules, par exemple :



Exemple de configuration de départ

L'univers est limité ici à un rectangle de 5 par 3. Pour faciliter l'explication, nous avons numéroté les cellules de 0 à 4 en horizontal et de 0 à 2 en vertical. Les cellules claires sont actives.

Dans le Jeu de la vie, est considérée comme voisine toute cellule contiguë, y compris les diagonales.



Détermination du voisinage

La figure ci-dessus montre le voisinage de la cellule 12. En l'occurrence, sur les huit voisins, deux sont actifs.

Les règles du Jeu de la vie sont simples :

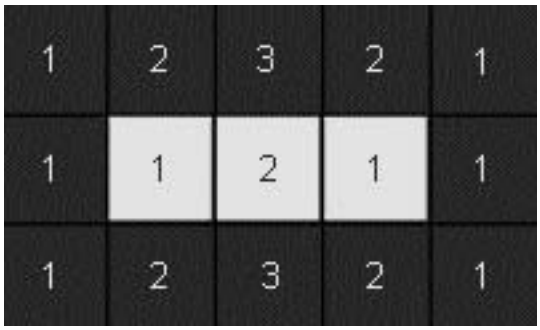
- Une cellule inactive entourée de 3 cellules actives devient active (« naît ») ;
- Une cellule active entourée de 2 ou 3 cellules actives reste active ;

- Dans tous les autres cas, la cellule « meurt » ou reste inactive.

On peut interpréter ces règles en considérant qu'une naissance nécessite un certain rassemblement de population (3 en l'occurrence), que les cellules ne peuvent survivre à un trop grand isolement (moins de 2 voisines) et qu'une trop forte con- centration (plus de 3 voisines) les étouffe.

Les automates cellulaires fonctionnent de manière *discrète*. C'est-à-dire que le temps s'écoule par à-coups. Ceci signifie dans notre cas qu'à la génération *t*, chaque cellule examine son environnement et détermine son état futur. Quand l'ensemble des cellules a été traité, et seulement à ce moment- là, toutes les cellules passent à l'état calculé. On simule ainsi un traitement parallèle.

Illustrons ce mécanisme à partir de la configuration précédente :



Valeurs de voisinage

Dans le schéma précédent, le nombre de voisins actifs est noté pour chaque cellule :

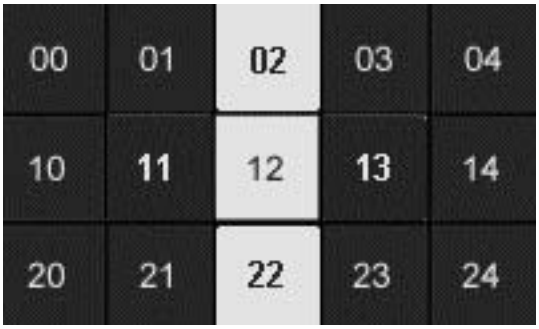
- Les cellules inactives 00, 04, 10, 14, 20 et 24 ont une voisine active et restent donc en l'état.

- Les cellules inactives 01, 03, 21 et 23 ont deux voisines, elles ne changent donc pas.

- Les deux cellules inactives restantes (02 et 22) ont trois voisines actives, la règle 1 s'applique : elles naissent.

- Les cellules actives 11 et 13 n'ont qu'une voisine active : elles meurent.

- Enfin la cellule active 12 ayant deux voisines actives elle reste en vie. À la génération suivante, seules les cellules 02, 12 et 22 seront donc actives.



Seconde génération

On met ici en évidence les trois propriétés fondamentales des automates cellulaires « standards » :

1. *Le parallélisme* : Un système est dit parallèle si ses constituants évoluent simultanément et de manière indépendante.

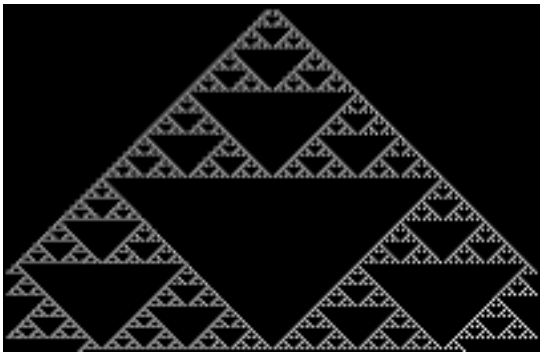
2. *La proximité* (locality) : Le nouvel état d'une cellule ne dépend que de son état actuel et de l'état du voisinage immédiat.

3. *L'homogénéité* : Les lois sont universelles, c'est-à-dire communes à l'ensemble de l'espace de l'automate.

Les autres types d'Automates Cellulaires

Le Jeu de la vie n'est qu'un type d'automate cellulaire parmi une infinité. Il est en effet possible de jouer sur l'ensemble des règles qui régissent l'univers de l'auto- mate cellulaire.

Le paramètre le plus évident est le nombre de dimensions. Rien n'oblige en effet à considérer des environnements à deux dimensions. L'analyse théorique des automates cellulaires s'est essentiellement effectuée à partir d'automates à une dimension. En réduisant le nombre de dimensions, on limite l'explosion combinatoire, donc le nombre d'automates possibles. Si l'on considère le cas simple d'un voisinage de trois cellules, soit la cellule concernée et ses voisines de droite et de gauche, dans un automate à une dimension et deux états, il n'existe que $2^3 = 256$ règles possibles. La représentation des automates à une dimension (soit une ligne), utilise la seconde dimension pour représenter le temps. À chaque génération, une nouvelle ligne est ajoutée au-dessous de la précédente, on peut visualiser ainsi la dynamique de ce type d'automate.



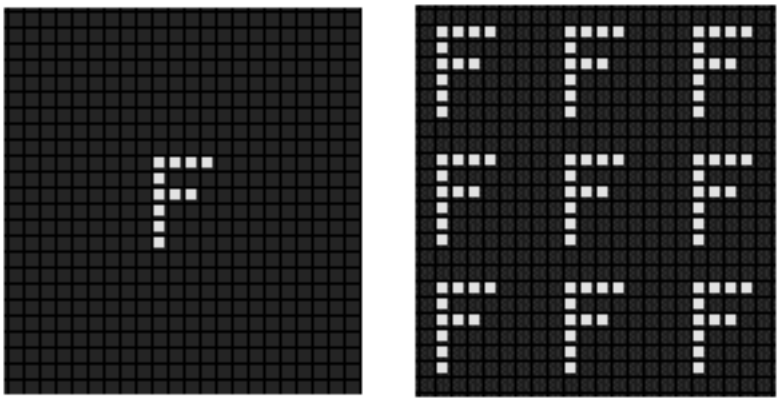
Exemple d'automate à une dimension (Triangle de Pascal)

Il est naturellement possible de créer des automates à trois dimensions voire plus.

Il est également possible de jouer sur la détermination du voisinage. Si l'on con- sidère les automates à deux dimensions, les voisinages les plus courants sont [1] :

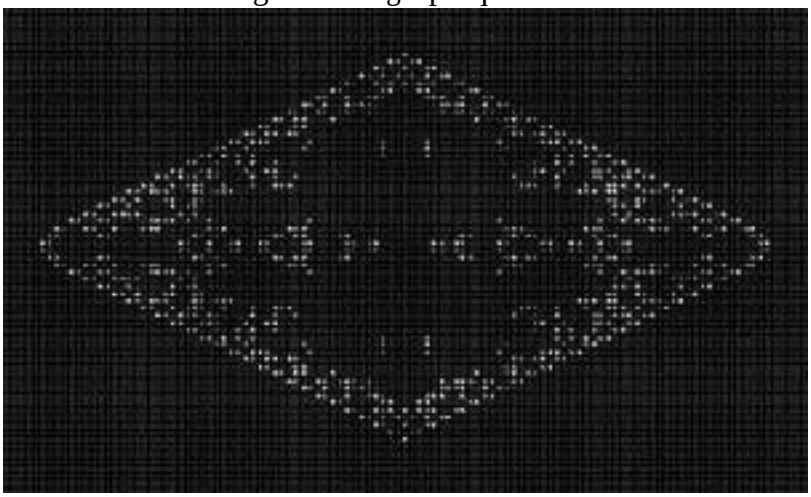
- *Von Neumann* : on considère les seuls voisins Nord/Sud/Est/Ouest.
- *Moore* : on ajoute les diagonales. C'est le cas du Jeu de la vie.
- *Moore étendu* : on étend la distance de voisinage au-delà de 1.
- *Margolus* : on considère des ensembles de 2x2 éventuellement alternés. C'est ce type de voisinage qui est utilisé dans la simulation du comporte- ment des gaz.

Par exemple, l'automate de Fredkin qui utilise un voisinage de Moore est basé sur la parité du voisinage. C'est un automate de type *sommatif*, c'est-à-dire que l'état des cellules dépend du nombre de voisins actifs, indépendamment de leur position. En l'occurrence, il n'y a reproduction que si la valeur de voisinage est impaire. Cet automate a la propriété remarquable de reproduire toute configuration de base en neuf exemplaires. La règle de Fredkin est généralisable à plus de deux dimensions.



Fredkin génération 0 et Fredkin génération 8

Il est également possible de jouer sur le nombre d'états. Rien n'oblige en effet à se cantonner aux deux états vie/mort. *Brian's Brain* par exemple, présenté par Brian Silverman en 1984 utilise trois états (vie/fantôme/mort) pour engendrer une grande diversité de planeurs complexes au sein de configurations graphiques étonnantes.



Brian's Brain

Des règles plus complexes sont imaginables. On peut par exemple construire des automates stochastiques dont les règles de transition intègrent une fonction de probabilité.

D'une manière générale, on peut construire tout type d'automate en jouant sur les règles structurelles et fonctionnelles. Les premières définissent la structure spatiale du réseau d'automates, soit son nombre de dimensions, le mode d'arrange- ment des cellules (carré, hexagonal... dans un automate à deux dimensions) et le mode de détermination du

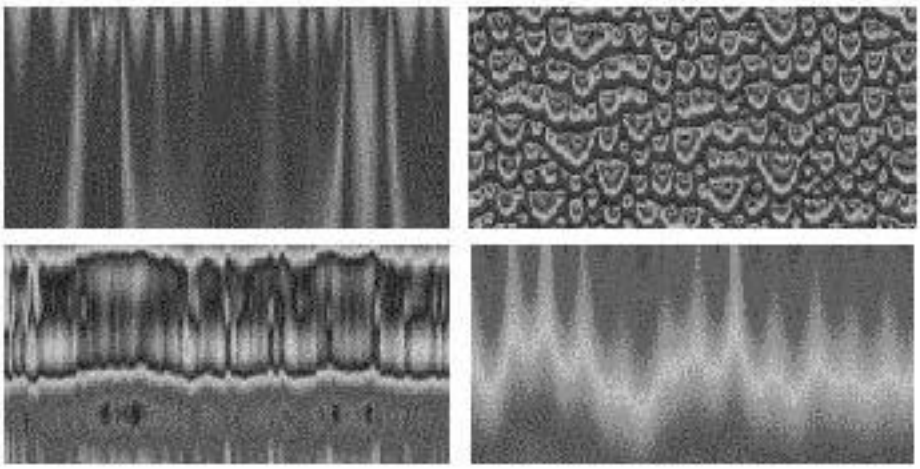
voisinage. Les secondes déterminent le nombre d'états et les règles de transition. Le choix de ces deux types de règles permet de construire un univers adapté à l'objectif recherché.

Les applications pratiques

Les applications pratiques des automates cellulaires sont nombreuses et diverses. Fondamentalement ils constituent des univers dont on fixe les lois. Notre Univers est soumis aux lois de la Physique. Ces lois ne sont que partiellement connues et apparaissent hautement complexes. Dans un automate cellulaire, les lois sont simples et complètement connues. On peut ainsi tester et analyser le comportement global d'un univers simplifié. Voici quelques exemples d'application :

- Simulation du comportement d'un gaz. Un gaz est composé d'un ensemble de molécules dont le comportement est fonction de celui des molécules voisines.
- Étude des matériaux magnétiques selon le modèle d'Ising : ce modèle (1925) représente le matériau à partir d'un réseau dont chaque nœud est dans un état magnétique donné. Cet état — en l'occurrence l'une des deux orientations du moment magnétique — dépend de l'état des nœuds voisins.
- Simulation des processus de percolation.
- Dans un domaine différent, les automates cellulaires peuvent être utilisés comme alternative aux équations différentielles [1].
- Conception d'ordinateurs massivement parallèles.
- Simulation et étude du développement urbain [2].
- Simulation des processus de cristallisation.
- Simulation de la propagation des feux de forêt.

Dans un domaine plus quotidien, les automates cellulaires peuvent être utilisés comme générateur graphique [3]. Les quelques figures ci-dessous, construites avec Capow [4] montrent certains effets graphiques.



E-Émergences, autoréplication et complexité

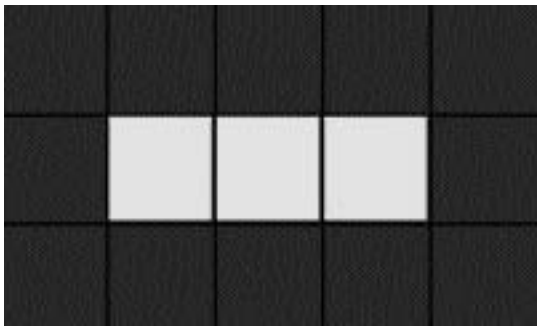
- *Émergences :* À la base du concept d'émergence on trouve l'idée commune selon laquelle : « le tout est plus que la somme des parties. » La notion d'émergence est ainsi liée à la non-linéarité en ce qu'un processus émergent ne respecte pas le principe de super- position.

« On peut appeler émergence les qualités ou propriétés d'un système qui pré- sentent un caractère de nouveauté par rapport aux qualités ou propriétés des composants considérés isolément ou agencés différemment dans un autre type de système. » L'association complexe d'éléments induit l'apparition de phénomènes, de mécanismes nouveaux. «À chaque niveau (de l'évolution prébiotique, bio- tique et sociale) émergent des propriétés nouvelles qui ne peuvent être expliquées par la somme des propriétés de chacune des parties qui constituent le tout. Il y a un saut qualitatif (...) La propriété d'émergence est liée à la complexité. L'accroissement de la diversité des éléments, l'accroissement du nombre de liaisons entre ces éléments et le jeu des interactions non linéaires conduisent à des comportements difficilement prédictibles. » Les émergences dites « globales » caractérisent donc les propriétés d'un système qui sont nouvelles par rapport aux propriétés de ses composants isolés ou organisés de manière différente. La vie en fait indéniablement partie.

Les processus émergents sont généralement fondés sur la multiplication des interactions parallèles entre éléments indépendants. Les automates cellulaires fonctionnent précisément selon ce principe. Ils constituent ainsi un outil précieux d'analyse de l'émergence.

On a examiné plus haut le comportement d'une ligne de trois cellules verticales : à la première génération on obtient trois cellules horizontales et à

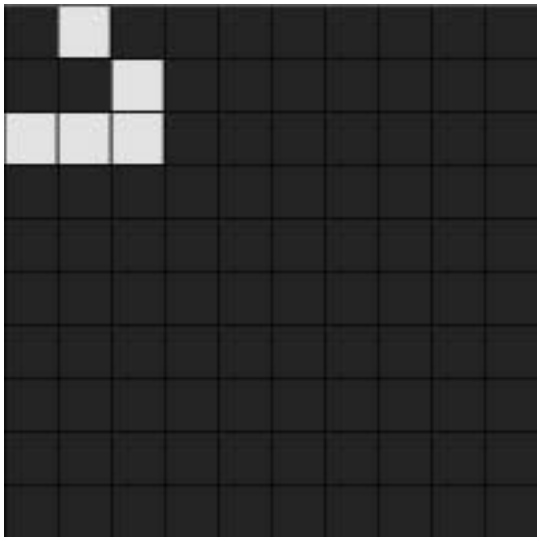
la seconde de nouveau trois cellules verticales, etc. Une ligne de trois cellules vivantes engendre donc un cycle.



Animation d'un clignotant

Cette figure appartient à la catégorie des « clignotants » (blinker) ou oscillateurs. Un oscillateur n'est pas constitué d'un groupe de cellules données, c'est une configuration dynamique au sein de l'espace de l'automate cellulaire. Le clignotant semble être autonome, il est une structure spécifique, particulière au sein de son milieu.

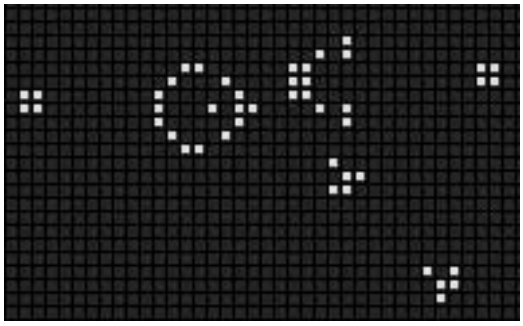
D'une manière générale, les règles du Jeu de la vie ont été fixées de façon à engendrer une grande diversité de structures imprévisibles. Les spécialistes ont recensé toute une faune de configurations aux comportements plus étonnants les uns que les autres. Des bibliothèques entières sont disponibles. L'une des plus fameuses est le « planeur » (glider) qui apparaît souvent après un remplissage aléatoire. Une configuration donnée de cinq cellules se réplique toutes les quatre générations à une cellule de distance.



Animation d'un planeur

Plus encore que les oscillateurs, les planeurs évoquent le phénomène d'émergence. On a l'illusion d'un être rampant, parcourant l'espace en ligne droite. Un planeur n'est pas un ensemble de cellules. À chaque génération, les cellules qui le composent sont remplacées. De la même manière que les atomes qui vous constituent ne sont pas ceux dont vous disposiez à votre naissance, les composants du planeur sont perpétuellement renouvelés. L'application des règles du Jeu de la vie fait ainsi apparaître une structure dynamique, cohérente et autonome, ayant des propriétés spécifiques, c'est le caractère même de l'émergence. Ces propriétés — en l'occurrence le déplacement — peuvent être utilisées à des fins spécifiques. Le planeur permet de représenter un signal, on en trouve un exemple dans LogiCell.

On peut également citer une autre figure remarquable : le canon à planeur (glider gun). Il s'agit d'un ensemble de cellules engendrant des planeurs. Il a permis de prouver que la population du Jeu de la vie peut croître indéfiniment. Le classique canon de période 30 est utilisé comme générateur dans LogiCell.



Le canon à planeur

Autoréplication : Avec le kinématon rencontré plus haut, von Neumann a essayé de rendre compte des conditions de l'autoréplication. Il se posait plus précisément la question de l'organisation logique d'un automate suffisante pour assurer l'autoréplication.

Face à l'impossibilité physique de réaliser cette machine, on a vu que von Neumann s'est tourné vers les automates cellulaires pour construire son automate autoréplicateur. Cet automate était extrêmement complexe car il intégrait un constructeur universel. En 1968, Edgar Codd a proposé une version simplifiée de l'automate de von Neumann n'utilisant que huit états, mais là encore, Codd intégrait un constructeur universel. Les choses ont changé dans les années 1980 avec Christopher Langton.

Langton a considéré que l'étude des systèmes vivants au sein d'un calculateur nécessitait de considérer les seuls éléments nécessaires et non les éléments suffisants. Il a ainsi abandonné l'idée d'universalité du réplicateur.

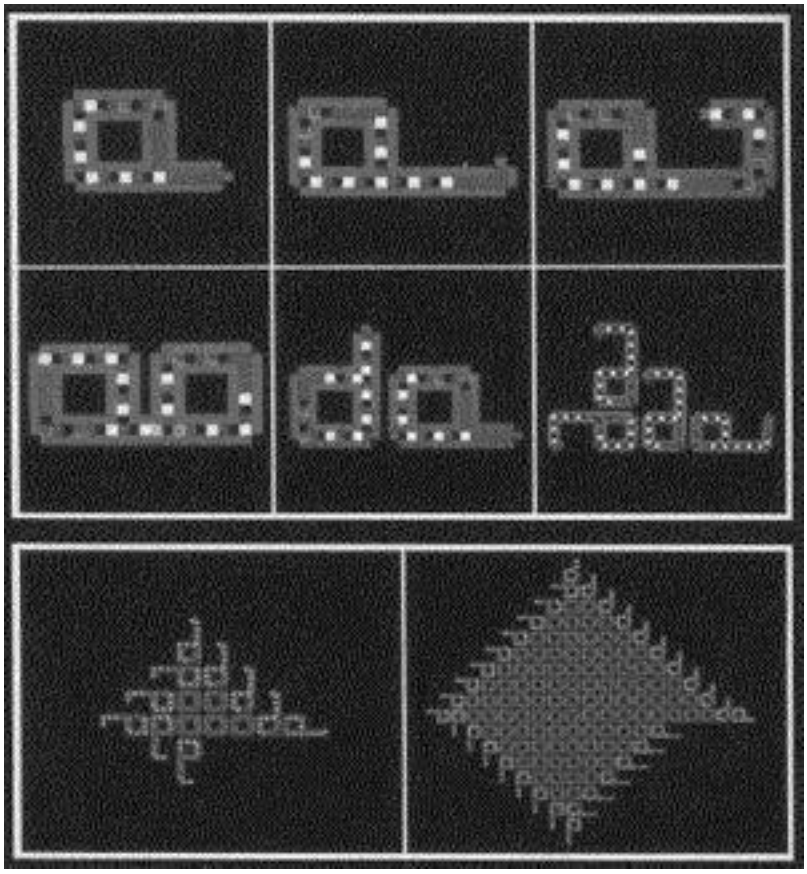
L'idée de base de Langton est qu'il est possible de concevoir un automate cellulaire supportant une structure dont les composants constituent l'information nécessaire à sa propre réplication. Cette structure est donc à la fois elle-même et représentation d'elle-même.

L’automate de Langton utilise huit états et vingt-neuf règles. La structure qui se réplique est une boucle constituée d'une « membrane » au sein de laquelle circule l'information nécessaire à la réplication.

La boucle de Langton

Les cellules à l'état 2 forment la membrane, les cellules internes contiennent l'information de réplication. D'une certaine manière, elles sont l'ADN de la boucle. Les séquences 7-0 et 4-0 se propagent vers la queue. Quand elles atteignent l'extrémité, les premières prolongent la queue, les secondes construisent un angle droit vers la gauche.

L'ajout d'une règle de « stérilisation » qui bloque l'évolution au bout d'un certain nombre de générations permet la cristallisation des boucles les plus anciennes et amène à la construction d'une sorte de corail.



D’après S. Levy, Artificial Life., Penguin, 1992.

- *Les boucles de Langton :* Les boucles de Langton, comme l'automate de von Neumann montrent que : « (…) l'une des propriétés fondamentales des organismes vivants, l'autoreproduction, est explicable en termes d'interactions d'éléments simples et qu'elle peut être étudiée dans ses principes logiques indépendamment de sa réalisation physique. »

En aucune manière, les boucles de Langton ne peuvent être considérées comme « vivantes », elles ne sont qu'une construction autoréplicatrice limitée.

- *Chaos et complexité :* Le nombre d'univers possibles est virtuellement infini. Dans ce contexte, Wolfram s’est interrogé sur l’existence de règles générales de comportement des automates cellulaires.

S.Wolfram s'est intéressé aux automates à une dimension, deux états avec un voisinage de deux. Il considère que ne sont « légaux » que les automates qui d'une part éliminent toute cellule dont le voisinage est vide,

et d'autre part sont symétriques. Il n'existe alors que 32 automates légaux dont l'auteur a réalisé une étude systématique.

Cette étude a montré que, selon l'auteur, de nombreux automates cellulaires (peut-être tous) s'intègrent dans quatre classes principales :

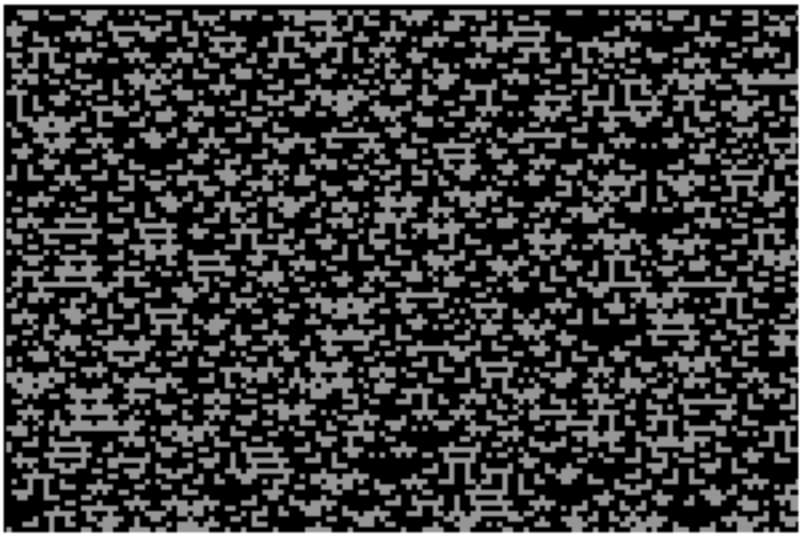
- *Classe I- L'évolution conduit à des configurations homogènes.*

Un automate de classe I (règle 36)

- *Classe II- L'évolution conduit à des structures simples ou périodiques.*

Un automate de classe II (règle 40)

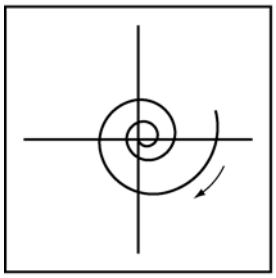
- *Classe III- L'évolution conduit à des configurations chaotiques.*



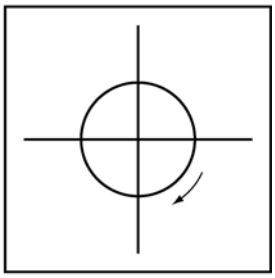
Un automate de classe III (règle 18)

Ces trois classes peuvent être liées à des comportements physiques connus que l'auteur présente en termes d'*attracteurs* dans l'espace des phases. Un tel espace est utilisé pour représenter la dynamique d'un système. Chacun des points d'une représentation graphique figure les états du système en fonction du temps. Par exemple, le mouvement d'un pendule sans frottement sera représenté par un cercle décrivant son caractère cyclique. Si l'on considère un pendule réel (c'est-à-dire soumis aux frottements) la trajectoire sera une spirale rejoignant progressivement le centre de la figure, soit le point représentant la position d'arrêt. Ce point, comme le cercle, est un attracteur, il correspond à la trajectoire du système stabilisé. On connaît depuis longtemps l'attracteur fixe où la stabilisation correspond à l'arrêt, ainsi que l'attracteur cyclique. Dans ce dernier cas, le système répète en permanence le même mouvement (votre cœur tant que vous êtes en vie par exemple), on obtient alors une figure plus ou moins circulaire, mais dans tous les cas bouclant sur elle-même. Plus récemment, en 1963, une nouvelle forme d'attracteur a été découverte : l'*attracteur de LORENZ* généralisé sous la forme des *attracteurs étranges*. Ce type de représentation dans l'espace des phases correspond aux systèmes chaotiques. La complexité des figures engendrées y évoque clairement la sensibilité aux conditions initiales.

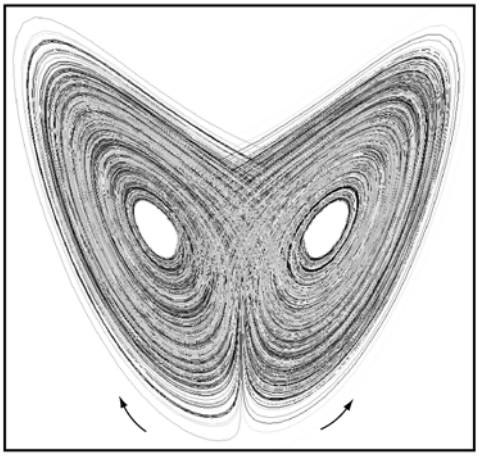
Attracteur fixe



Attracteur cyclique



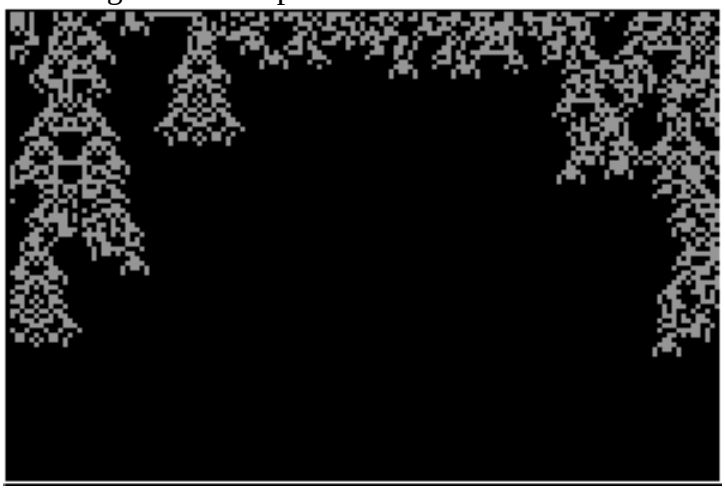
Un attracteur étrange : l'attracteur de LORENZ



Réalisé avec *Fractint* 20.0

Attracteurs

Dans le cas de la classe IV, le parallèle avec des mécanismes connus est beau- coup moins évident. Les automates de cette classe évoluent vers des configurations globales complexes.



Un automate de classe IV (règle 20)

Pour Wolfram, « Les automates cellulaires peuvent être vus comme des ordina- teurs dans lesquels les données sont représentées par les configurations initiales, et traitées à travers l'évolution temporelle. Le calcul universel [dans le sens de la machine universelle de Turing] implique que des configurations initiales adéquates peuvent gérer des procédures algorithmiques arbitraire. » Wolfram constate que les automates cellulaires de classe IV engendrent des structures qui rappellent fortement le Jeu de la vie. Or on sait, parce que cela a été réalisé, que ce dernier permet de construire une machine universelle de Turing. À partir de là, il pose l'hypothèse selon laquelle cette classe caractérise les automates ayant des capacités de Calcul Universel. Pour que cette capacité émerge, les cellules doivent pouvoir communiquer entre elles et transmettre l'information. Dans les automates de classe I et II, l'interdépendance des cellules est trop forte pour qu'un traitement utile puisse avoir lieu. Les automates de classe III quant à eux se caractérisent par une interdépendance trop faible. Les classes I à III sont les plus fréquentes. Elles représentent 30 des 32 automates de Wolfram. Ce n'est donc que dans une minorité de cas — 2 sur 32 en l'occurrence — que l'on trouve des automates cellulaires de classe IV. Ces automates cellulaires qui sont à la limite entre les classes I et II d'une part et III d'autre part, sont seuls aptes à un traitement éventuel de l'information et sont donc les seuls « intéressants » .

C. Langton s'est intéressé lui aussi à l'existence de règles générales de classification des automates cellulaires. Le problème réside dans le nombre d'automates cellulaires possibles. Si l'on considère les seuls automates à une dimension, 8 états et un voisinage de 5, il existe 8^5, soit 8^{32768} (près de 10^{30000}) univers possibles. Langton a décidé de caractériser les automates cellulaires en fonction d'un paramètre général, le *paramètre* λ. λ est en fait la probabilité au sein de toutes les configurations de voisinages possibles, qu'une configuration donnée entraîne la « vie » de la cellule, soit :

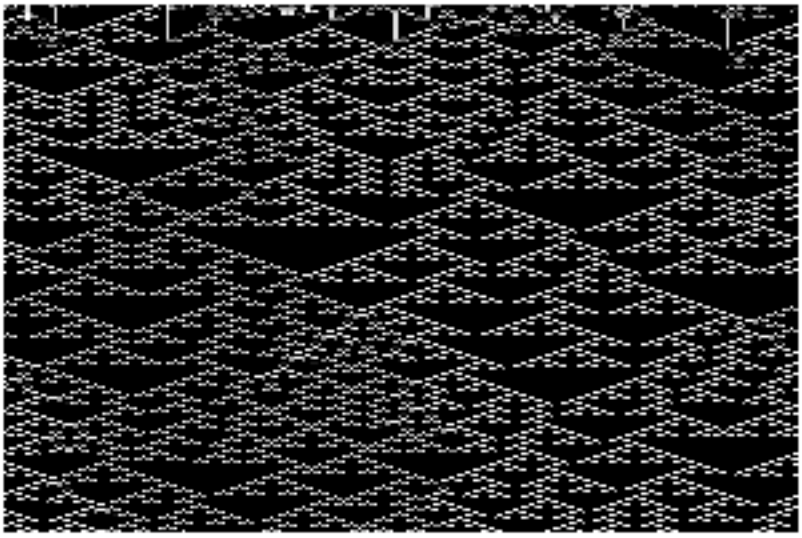
1- (nombre de transitions « mortelles » /nombre total de transitions).

En construisant des règles de transitions à partir de ce paramètre, Langton a pu proposer une classification des automates cellulaires. Pour une valeur faible, les cellules disparaissent rapidement. Si on élève la valeur grossièrement au-dessus de 0,2 on constate l'apparition de structures cycliques ou persistantes. Au-dessus de 0,3 des comportements complexes et imprévisibles apparaissent. Enfin, au-dessus de 0,5 la multiplication de structures induit un comportement chaotique. D'une

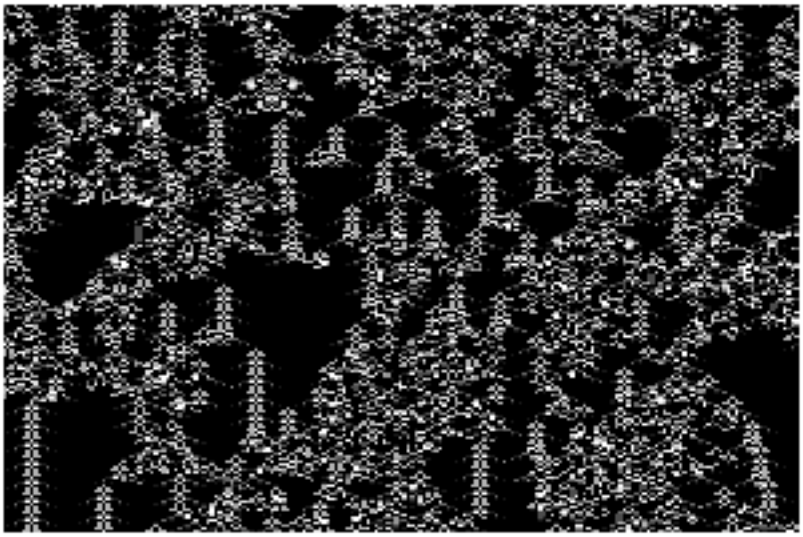
certaine manière, le paramètre λ indique la température de l'univers de l'automate cellulaire

Les quatre figures suivantes montrent des exemples d'automates une dimension à huit états avec un voisinage de cinq.

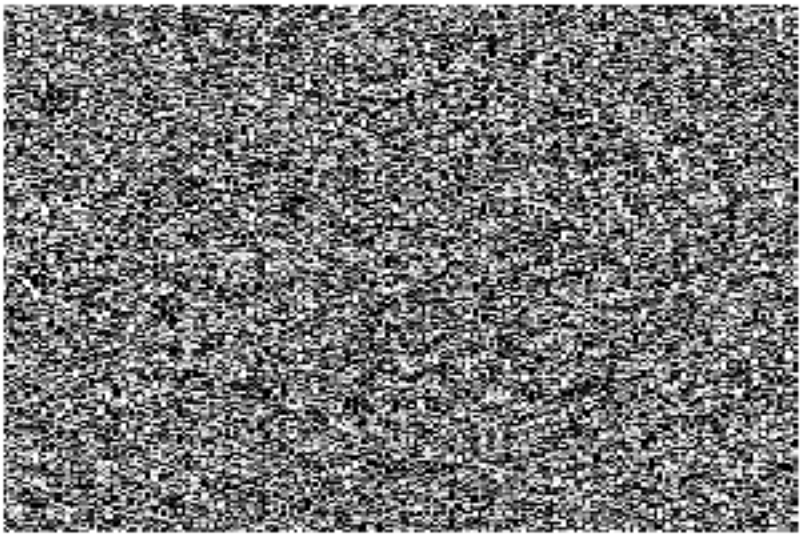
λ = 0.1



λ = 0.25



$\lambda = 0.45$



$\lambda = 0.8$

On retrouve donc bien chez Langton, mais dans un cadre plus général, la classification de Wolfram. D'après Langton, les automates de classe IV, ceux dont le paramètre se situe entre environ 0.3 et 0.5, sont ceux dont la capacité de transmission de l'information est la plus importante : « (…) les automates cellulaires capables de réaliser des calculs non triviaux — y compris la capacité de Calcul Universel — ont plus de chance de se trouver au voisinage de la transition de phase entre l'ordre et le chaos (…). » On aurait ainsi la progression suivante dans l'espace des phases :

Homogène -> cyclique -> complexe -> chaotique.

J.-C. Heudin utilise un paramètre (β) qui, dans des automates cellulaires à deux dimensions, correspond au nombre de cellules nécessaires pour qu'une cellule reste inchangée. Si l'on prend le cas de Conway, ce paramètre vaut 2. Il redéfinit les automates à partir de quatre règles soit :

R1 : le voisinage est inférieur à β : la cellule meurt.

R2 : une cellule entourée de β cellules vivantes conserve son état. R3 : une cellule ayant β +1 voisines vivantes devient vivante.

R4 : une cellule entourée de plus de β +1 voisines meurt.

Pour β = 1, les probabilités d'exécution des règles sont : R4>R3>R2>R1. Pour β = 2, les probabilités s'inversent. On obtient : R1>R2>R3>R4. Au-delà de 2, l'ordre reste identique mais les règles R2 à R4 ne s'exécutent que marginalement.

Heudin montre ainsi un changement fondamental dans l'univers des automates cellulaires pour □ = 2. C'est au cours de cette modification profonde des propriétés de l'automate — de cette *transition de phase* — que se manifeste la complexité :

« Pour apparaître [le complexe a] besoin d'ordre et d'une pincée de chaos. Cette situation n'est possible qu'à l'interface des deux régimes, à la frontière qui mène au chaos. »

La généralisation des enseignements des automates cellulaires, la diversité des structures universelles ou l'existence de la vie, amènent ainsi à penser que, parmi l'infinité d'univers possibles, les lois de notre Univers sont précisément à la frontière entre l'ordre et le chaos. C'est l'essence de l'approche de Langton dans son expression : « La vie au bord du chaos. »

Références

- Filippo Peverelli et al, « L'intelligence Artificielle », Rapport EIVD, Lausanne, 11 juin 2002
- F.Y. Villemin « Intelligence Artificielle B, Introduction et définition », 2001.
- deptinfo.cnam.fr/Enseignement/CycleSpecialisation, consulté le 20/11/2003.
- J.-M Karkan & G.Tjoem « Systèmes Experts : Un nouvel outil pour l'aide à la
- décision », Edition Masson, 1993.
- R. Voyer « Moteurs de Systèmes Experts », Edition Eyrolles, 1987.
- H. Farreny et M. Ghallab « Éléments d'intelligence artificielle », Edition Hermes, 1992
- François Denis, « Cours d'Intelligence artificielle, Systèmes Experts », www.grappa.univ-lille3.fr/polys/se consulté le 24/10/2003.
- F.Y- Villemin « Systèmes Intellignets » C 2001/2002, deptinfo.cnam.fr/Enseignement/CycleSpecialisation, consulté le 20/11/2003.
- James L Crowley, Cours, « Conception des systèmes intelligents, programmation des systèmes experts » 2ème Année ENSIMAG, 5 février 2003, consulté le 10/03/20.
- Sabas, A., Deliste S. Badri M. « Vers une unification des méthodologies de développement et la standardisation des plates formes SMA ». Maîtrise en Mathématique etInformatique Appliquée. Université du Québec à Trois-Rivière, 2000.
- Ktari, B. « Programmation orientée objets », Pour l'obtention du grade de Maîtriseès Sciences en Informatique, Université de Laval (Québec - Canada), 1998.
- Sabas, A., Delisle, S., Badri, M. « Etude comparative des méthodologies de développement des Systèmes Multi-Agents », Département de Mathématiques et d'Informatique, Université du Québec, Trois-Rivières, Canada.2001
- H. Farreny, « Les systèmes experts : principes et exemples », Techniques Avancées de L'informatique, Edition CEPADUES, 1985.
- Jacques Ferber, http://www.lirmm.fr/~ferber, Professeur d'informatique à l'Université de Montpellier II, Chercheur au Lirmm, Dernières modifications : 3 Octobre 2003.
- Jean Noel Chatain, Alain Dussauchoy, « Systèmes experts : Méthodes et Outils », Edition Eyrolles, 1987.
- F.Hayes-Roth « Building Expert Systems ». Edition Addison Wesley, 1983
- (Goldberg, 1989) D.E. Goldberg, " Genetic Algorithms in Search, Optimization and Machine Learning", Addison-Wesley, 1989.
- Leekwijck, W. V. and Kerre, E. E. (1999). Defuzzification : criteria and classification. Fuzzy Sets and Systems, 108(2) :159-178.Madau D., D. F. (1996). Influence value defuzzification method. Fuzzy
- Systems, Proceedings of the Fifth IEEE International Conference, 3 :1819-1824.
- Zadeh, L. (1965). Fuzzy sets. Information and Control, 8(3) :338-353.
- Adami Ch., *Introduction to Artificial Life*, Springer-Verlag, New-York, 1998.
- Bertalanffy (von) L., *Théorie générale des systèmes.* Dunod, Paris, 1973.
- Gutowitz H.A., Langton C.G., Methods for designing Cellular Automata with Interesting" Behavior, 1994.
- Heudin JC, La Vie Artificielle, Hermès, Paris, 1994.

- Heudin JC, L'évolution au bord du chaos, Hermès, Paris, 1998.
- Langlois A., Phipps M., Automates cellulaires. Application à la simulation urbaine. Hermès, Paris, 1997.
- Langton C.G., Studying Artificial Life with cellular automata, Physica D 22, 1986.
- Langton C.G., Life at the edge of chaos, Artificial Life II, Addison-Wesley, 1991.
- Langton C., Artificial Life in The philosophy of Artificial Life, Boden M. A. dir., Oxford readings in Philosophy, Oxford University Press, 1996.
- Langton ed., Artificial Life an overview, MIT press, 1997.
- Levy S., Artificial Life. The quest for a new creation, Penguin, 1992.
- Michell M., Hraber P.T., Crutchfield J., Revisiting the edge of chaos : Evolving Cellular Automate to perform Computations, Santa Fe Institute, Working Paper 93-03-014.
- Morin E., La Méthode. I-La Nature de la Nature., Points, Seuil, Paris 1977.
- Rennard J.-Ph., Vie artificielle. Où la biologie rencontre l'informatique, Vuibert, 2002.
- Rosnay (de) J., Le macroscope. Vers une vision globale. Points, Seuil, Paris, 1975. Rucker R., Walker J., Introduction to CelLab. Ce texte est disponible à : http://www.fourmilab.ch/cellab/
- Shatten A., Cellular Automata, Institute of General Chemistry Vienna University of Technology, Austria, 1997.
- Toffoli T., Cellular automata as an alternative to Differential equations, in Modelling Physics, Physica 10D, 1984.
- Von Neumann J. et Burks A. ed., Theory of Self-Reproduction Automata, University of Illinois Press, 1966.
- Wolfram S., Universality and complexity in cellular automata, Physica D, 10:1-35,1984.
- Wolfram S., A New Kind of Science, Wolfram Media, 2002