Algorithmes pour l'IA – TD 6
Retour sur ce qu'on a vu

Ci-dessous sont une collection d'exercices sur les différents types d'algorithmes vus en cours, avec quelques indications du type de tâches à réaliser dans la mise en place de chaque type d'algorithme.

1 Résolution, parcours, recherche, jeux, CSP

Pour les algorithmes de parcours, il s'agit en premier lieu de modéliser le problème, c'est-à-dire, de :

- Décider des états initiaux et finaux
- Décider d'une fonction de transition entre états
- Décider, s'il y a lieu, d'une fonction heuristique
- Choisir l'algorithme

Dans le cas de jeux à deux joueurs (algorithme minimax, potentiellement avec élagage AlphaBeta), la fonction heuristique est remplacée par une fonction d'évaluation et le reste est établi sur la base des règles du jeu.

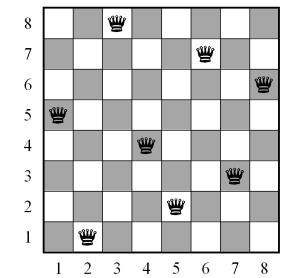
Pour les problèmes de résolution de contraintes (CSP, ou en programmation par contraintes), on va modéliser l'état initial et l'état final au travers :

- D'un ensemble de variables
- Du domaine de chaque variable
- D'un ensemble de contraintes portant sur ces variables

Les heuristiques ici portent sur le choix des variables à affecter en premier, et des valeurs à leur affecter.

Il conviendra dans tous les cas de savoir dérouler les algorithmes, afin de savoir les implémenter et les débugger.

1.1 Parcours de graphe et recherche heuristique

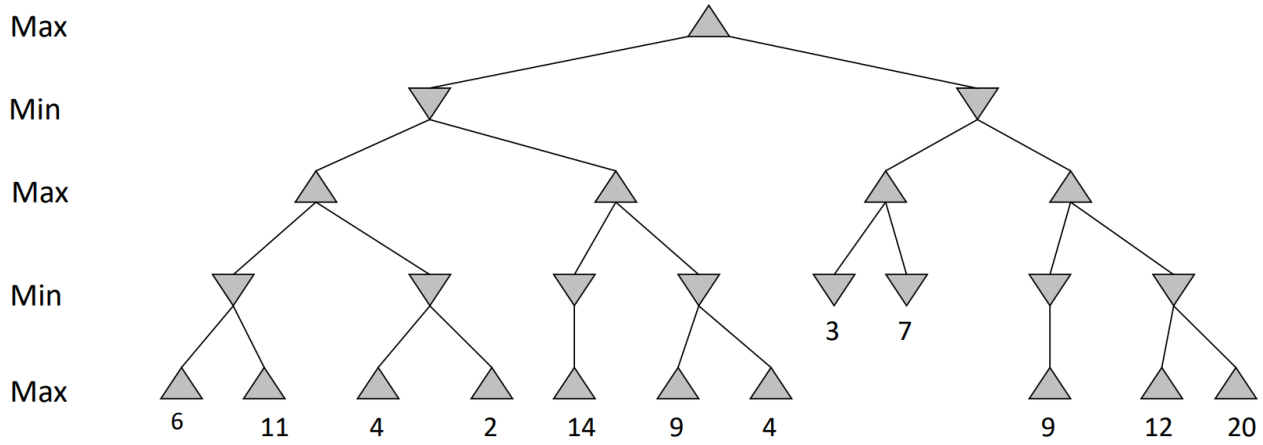


Vous connaissez déjà le problème des huit reines : trouver un emplacement pour 8 reines sur un échiquier de façon à ce qu'aucune reine ne puisse en prendre une autre. Modéliser le problème comme un problème de recherche, décidez du meilleur algorithme à appliquer (et justifier la réponse) et dérouler cet algorithme.

1.2 Jeux à deux joueurs

Quelle fonction d'évaluation utiliseriez vous pour le "jeux de dame". Est-ce un jeu difficile en termes de complexité et quel impact a cette complexité sur la profondeur de la recherche minimax ?

Dérouler l'algorithme minimax avec élagage AlphaBeta sur l'arbre suivant. Quel coup est choisi ? Combien de noeuds n'ont pas à être exploré grâce à l'élagage ?



1.3 Problèmes de satisfaction de contraintes

Il y a quatre familles A, B, C, D vivant dans 4 maisons, 1, 2, 3, et 4.

- La famille C est dans une maison dont le numéro est plus élevé que celle de D
- La famille D est dans une maison voisine de A, avec un plus petit numéro
- Il y a au moins une maison entre celles des familles D et B
- La famille C n'est pas au numéro 3
- La famille B n'est pas au numéro 1

Modélisez le problème qui consiste à trouver dans quelle maison habite quelle famille comme un problème de satisfaction de contraintes et résolvez à l'aide des algorithmes SimpleRetourArrière et anticipation. Vous pouvez aussi essayer de le résoudre avec ORtools.

2 Partitionnement

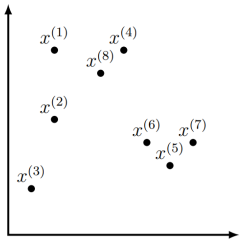
Pour une problème de partitionnement, comme pour les autres problèmes d'apprentissage, on va s'intéresser à comment préparer les données (normaliser, encoder), quel algorithme choisir et quels paramètres utiliser. Pour K-Means, le paramètre important est k. Pour DBScan, c'est epsilon, la mesure de distance et la taille minimal d'une partition. Pour le partitionnement hiérarchique, il s'agit de la mesure de distance et de l'agrégation utilisée. Ces algorithmes peuvent facilement être déroulées sur des exemples simples.

Partitionnez le jeu de données (à deux dimensions) ci-dessous en utilisant :

- k-means avec k=3 (voir tableau pour les distances euclidiennes) et en partant de trois points existants choisis au hasard.
- le partitionnement hiérarchique ascendant avec la distance euclidienne, agrégée en utilisant la distance minimum
- DBScan avec la distance euclidienne, une distance maximale epsilon de 3, et un nombre minimal d'objet par partition de 2

Comparez les résultats obtenus.

$x^{(1)} = (2, 8)$, $x^{(2)} = (2, 5)$, $x^{(3)} = (1, 2)$, $x^{(4)} = (5, 8)$,
$x^{(5)} = (7, 3)$, $x^{(6)} = (6, 4)$, $x^{(7)} = (8, 4)$, $x^{(8)} = (4, 7)$.



	$x^{(1)}$	$x^{(2)}$	$x^{(3)}$	$x^{(4)}$	$x^{(5)}$	$x^{(6)}$	$x^{(7)}$	$x^{(8)}$
$x^{(1)}$	0	3.0000	6.0828	3.0000	7.0711	5.6569	7.2111	2.2361
$x^{(2)}$	3.0000	0	3.1623	4.2426	5.3852	4.1231	6.0828	2.8284
$x^{(3)}$	6.0828	3.1623	0	7.2111	6.0828	5.3852	7.2801	5.8310
$x^{(4)}$	3.0000	4.2426	7.2111	0	5.3852	4.1231	5.0000	1.4142
$x^{(5)}$	7.0711	5.3852	6.0828	5.3852	0	1.4142	1.4142	5.0000
$x^{(6)}$	5.6569	4.1231	5.3852	4.1231	1.4142	0	2.0000	3.6056
$x^{(7)}$	7.2111	6.0828	7.2801	5.0000	1.4142	2.0000	0	5.0000
$x^{(8)}$	2.2361	2.8284	5.8310	1.4142	5.0000	3.6056	5.0000	0

3 Apprentissage supervisé

Les algorithmes d'apprentissage supervisé sont plus variés que ce que l'on a vu jusqu'à maintenant. Il s'agit tout d'abord de reconnaître si on a affaire à une tâche de classification ou de régression. Il faudra ensuite préparer les données (discrétisassions, si nécessaire, encodage, si nécessaire, normalisation, division entre jeu d'entraînement et jeu de test, etc.), choisir l'algorithme et établir les paramètres de l'algorithme pour obtenir de bon résultats. Pour les algorithmes simples, on cherchera à les dérouler pour pouvoir les débugger, et dans tous les cas, l'évaluation et l'interprétation des résultats sont des éléments importants.

3.1 Régression linéaire simple

Considérant le tableau suivant, réalisez les calculs nécessaires pour la régression linéaire simple en gardant les deux dernières lignes du tableau comme exemples de test. Quelle erreur obtenez vous sur ces exemples ?

Pays	*PIB X*	*Taux de scolarisation Y*
Pays en développement	*4775*	*63*
Pays les plus pauvres	*1350*	*45*
Pays arabes	*5680*	*62*
Asie de l'Est et Pacifique	*5872*	*69*
Amérique latine et Caraïbes	*7964*	*81*
Asie du Sud	*3072*	*56*
Afrique Sub-saharienne	*1942*	*50*
Europe centrale,orientale et CEI	*8802*	*83*

3.2 Arbres de décision

Le tableau ci-dessous contient des informations sur des étudiants pour lesquels on cherche à prédire la variable "Admis?".

- S'agit-il d'une tâche de classification ou de régression ?
- Que devez vous faire pour appliquer l'algorithme ID3 en termes de préparation des données ici ?
- Quels mesures pouvez vous utiliser pour calculer le gain d'information ?
- Appliquez ID3 en conservant les exemples 13, 14, 15 et 16 comme jeu de test.
- Quels erreurs obtenez vous sur ces exemples ?

	Doublant	Série	Mention	Admis?
1	Non	Maths	ABien	Admis
2	Non	Techniques	ABien	Admis
3	Oui	Sciences	ABien	Non Admis
4	Oui	Sciences	Bien	Admis
5	Non	Maths	Bien	Admis
6	Non	Techniques	Bien	Admis
7	Oui	Sciences	Passable	Non Admis
8	Oui	Maths	Passable	Non Admis
9	Oui	Techniques	Passable	Non Admis
10	Oui	Maths	TBien	Admis
11	Oui	Techniques	TBien	Admis
12	Non	Sciences	TBien	Admis
13	Oui	Maths	Bien	Admis
14	Non	Sciences	ABien	Non Admis
15	Non	Maths	TBien	Admis
16	Non	Maths	Passable	Non Admis

3.3 Réseaux de neurones

Considérant le jeu de données jouet ci-contre, sachant que les trois premières colonnes représentent les variables d'entrée et la suivant la variable de sortie:

1	1	1	R
1	1	2	R
1	2	1	R
2	1	1	R
2	2	1	B
2	1	2	B
1	2	2	B
2	2	2	B

1. S'agit il d'une tâche de classification ou de régression ?
2. Peut-on réaliser cette tâche avec une précision de 100% avec un simple perceptron ?
3. Combien de neurones en entrée et en sortie pour classifier les points 3D en R ou B ?
4. Combien de couches cachées ? Combien de neurones sur chaque couche cachée ?
5. Déroulez les premières étapes de l'apprentissage avec 2 neurones en une couche cachée et un taux d'apprentissage de 0.1.

4 Algorithmes génétiques

Pour un algorithmes génétique, la modélisation du problème consiste à trouver une représentation qui soit applicable et facile à appliquer, une fonction d'évaluation, un mode de sélection (on n'a vu que la roue de la fortune biaisée), une fonction de mutation et une fonction de croisement. On devra aussi choisir les valeurs des paramètres, incluant la taille des générations, le nombre de générations, le taux de mutation, le taux de croisement et, si pertinent, le taux d'élitisme.

Considérant la table de vérité à droite, on veut utiliser un algorithme génétique pour trouver des règles "si...alors" avec des conditions sur les variables *pair* et *impair* et des conclusions sur la variable *carré*. Les conditions peuvent utiliser les opérateurs logiques *et* et *non*.

	pair	impair	carré
1		X	X
2	X		
3		X	
4	X		X
5		X	
6	X		
7		X	
8	X		
9		X	X
10	X		

- Quelle représentation pourriez vous utiliser et comment la générer aléatoirement ?

- Quelle fonction de mutation ?
- Quelle fonction de croisement ?
- Quelle fonction d'évaluation ?
- Générez les trois premières générations de cet algorithme avec 5 individus par générations, un taux de mutation de 0.3 et un taux de croisement de 0.2.

Et si on ajoute *ou* comme opérateur logique ?