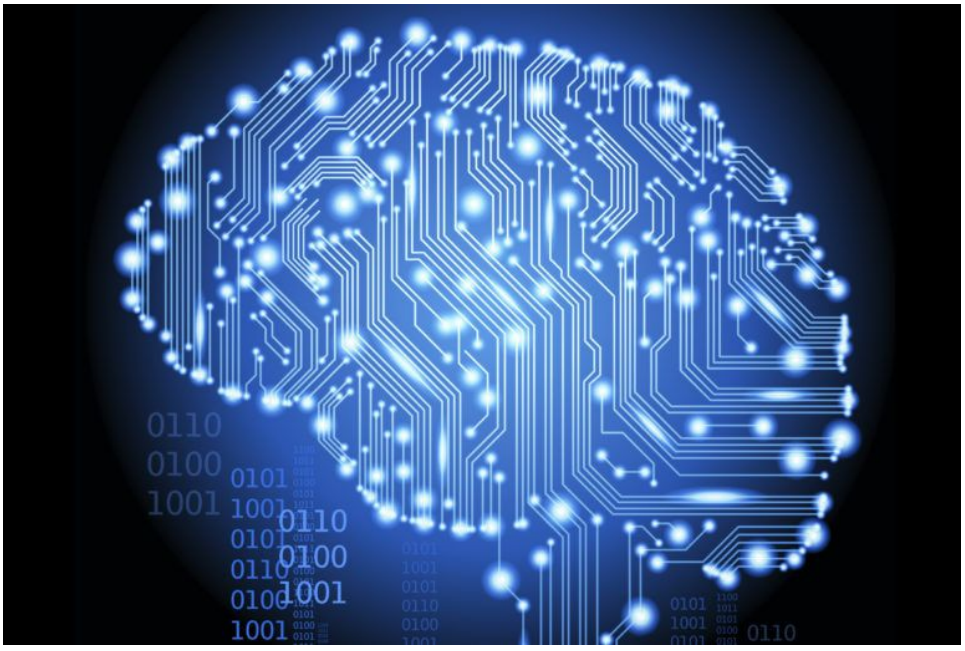


Rapport du projet
Intelligence Artificielle

Biccheri Lucas - Renaud Florian

1er Décembre 2018

SOMMAIRE

Dans le cadre de l'UE Intelligence Artificielle, nous avons dû faire un projet comportant deux parties majeures. Dans un premier temps, nous aborderons la partie questionnaire et Winform puis dans un second temps nous verrons la partie sur l'algorithme de Dijkstra. Dans chacune des deux parties, nous expliquerons nos choix d'interfaces ainsi que de conception.

I. Questionnaire

La première partie de ce projet consiste à réaliser un questionnaire sur le module d'Intelligence Artificielle. Pour ce faire, plusieurs choix ont été fait :

1) Ergonomie

Concernant l'ergonomie de notre application, une des exigences techniques de ce projet était d'utiliser un WindowsForm en langage C#. L'interface présente une première partie qui fera office d'accueil permettant d'ouvrir le questionnaire. Nous avons donc deux Winforms de la forme suivante :

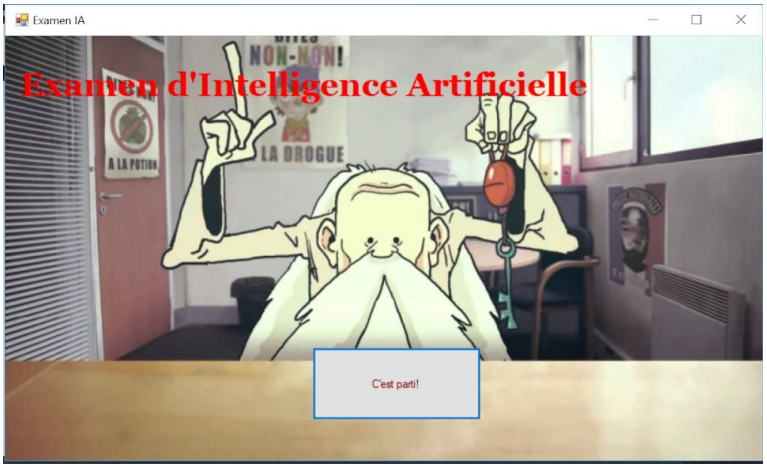


Figure 1 : Interface de lancement du questionnaire

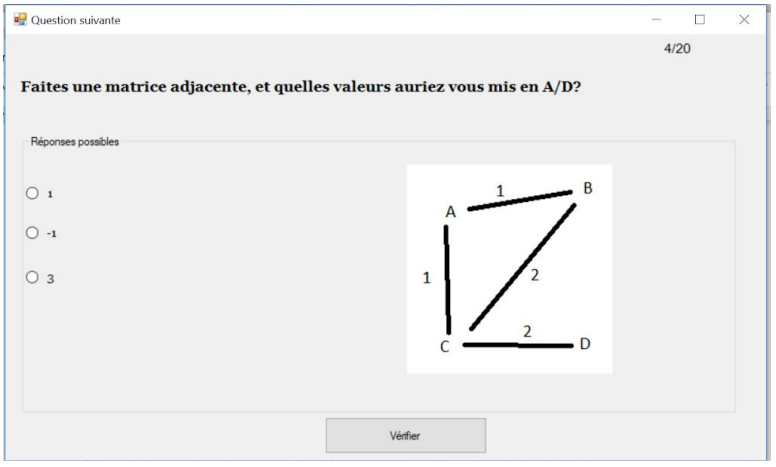


Figure 2 : Interface du questionnaire

Tout d'abord, nous avons choisi un modèle assez simple dans son ergonomie avec les informations minimales nécessaires tels que le titre de la question, trois réponses

possibles et la bonne réponse qui s'affiche lorsqu'on clique sur le bouton vérifier. Pour passer à la question il suffit d'appuyer sur le bouton "Question suivante" qui apparaît et ainsi le compteur s'incrémente, ce qui nous permet de voir l'évolution du questionnaire.

Nous avons choisi d'afficher l'état d'avancement des questions en haut à droite du type "X/20" plutôt qu'une "progress bar" car il nous semblait plus pertinent d'avoir une information précise. Lorsque le questionnaire est fini, une fenêtre s'affiche pour nous donner le score que l'utilisateur a fait. Nous voyons bien que le compteur est à 20/20 et dès que l'on clique sur OK, le Form se ferme et l'on revient sur l'interface de départ.

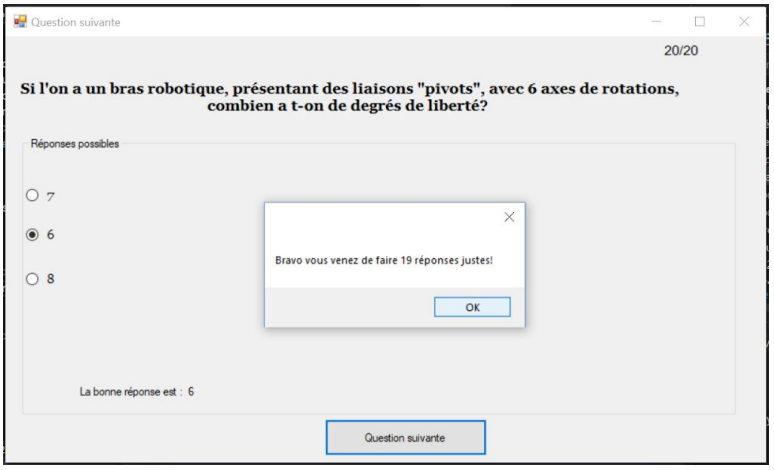


Figure 3 : MessageBox fin de questionnaire

S'il souhaite s'exercer à nouveau, il suffit à l'utilisateur de cliquer sur le bouton du Form principale.

2) Choix des questions

Pour ce qui est du choix des questions, nous avons pris 20 questions traitant des cours d'Intelligence Artificielle qu'un étudiant de l'ENSC suit au cours du semestre 7. Il y a divers niveaux de difficultés dans ces questions et nous avons choisi de balayer toutes les diapositives qu'un étudiant peut observer durant un cours d'Intelligence Artificielle, c'est à dire du cours de robotique en passant par les graphes et algorithmes. L'étudiant va ainsi pouvoir se tester sur tout l'étendu du cours et espérer réussir son examen de fin de semestre. Les questions sont choisies au hasard à partir d'un fichier XML, nommé "Data.Xml", grâce à une fonction Random qui choisit un nombre au hasard dans la liste des questions établies. Si l'utilisateur le souhaite, il peut rajouter une question directement depuis le program ou alors en l'écrivant directement dans le fichier Xml afin de ne pas recompiler le code déjà écrit. Il suffira ainsi de lancer le Form afin de tester les nouvelles questions.

Le fichier Xml de la forme suivante :

<Question>
    <titre> <titre/>
    <A><A/>
    <B><B/>
    <C><C/>
    <reponseBonne><reponseBonne/>
    <image><image/>

<Question/>

3) Réalisation

Pour le développement du code, nous avons essayé de réduire le plus possible la duplication de code. Pour ce faire, nous avons créer des fonctions afin d'alléger le code et le rendre plus lisible. Nous avons ainsi créé :

- la fonction AfficherQuestions qui permet d'afficher le titre dans le label associé, les réponses à la questions dans des CheckBox qui nous permettent d'avoir une seule et unique réponse pouvant être coché.
- La fonction ResultatJuste permet de déterminer si la réponse choisi par l'utilisateur est correct, si oui un compteur des réponses justes sera incrémenté, sinon le compteur ne bouge pas.

Le principe était de programmer une boucle pour 20 questions afin d'afficher le titre de la question et les réponses associés tout en incrémentant le compteur des questions. Une question ne peut pas être posée deux fois. Lorsque le compteur arrive à 20, l'utilisateur répond à la dernière question puis les résultats s'affichent sous la forme d'un MessageBox. La fonction d'affichage des questions va chercher les noeuds du fichier xml et va lire ce qu'il y a dans les balises.

II. Algorithme de Dijkstra

La deuxième partie du projet consiste à évaluer la compréhension de l'algorithme de Dijkstra par l'utilisateur. Cette évaluation se découpe en deux partie : la première qui consiste à remplir les ouverts et fermés, et la seconde où il faut remplir le squelette de l'arbre final par les noeuds qui ont permis à l'utilisateur de trouver le chemin le plus court.

1) Organisation

Cette partie a d'abord demandé une phase de reprise du code partagé par M. Salotti pour savoir comment fonctionnait le programme. Suite à quoi nous avons récupéré la majorité des classes fournies (GenericNode, Node, SearchTree) et le formulaire. Nous avons apporté des modifications sur le formulaire pour permettre l'évaluation de l'utilisateur; mais aussi au sein des classes, notamment la classe SearchTree pour pouvoir récupérer les résultats étape par étape des ouverts et fermés.

2) Interface de l'application

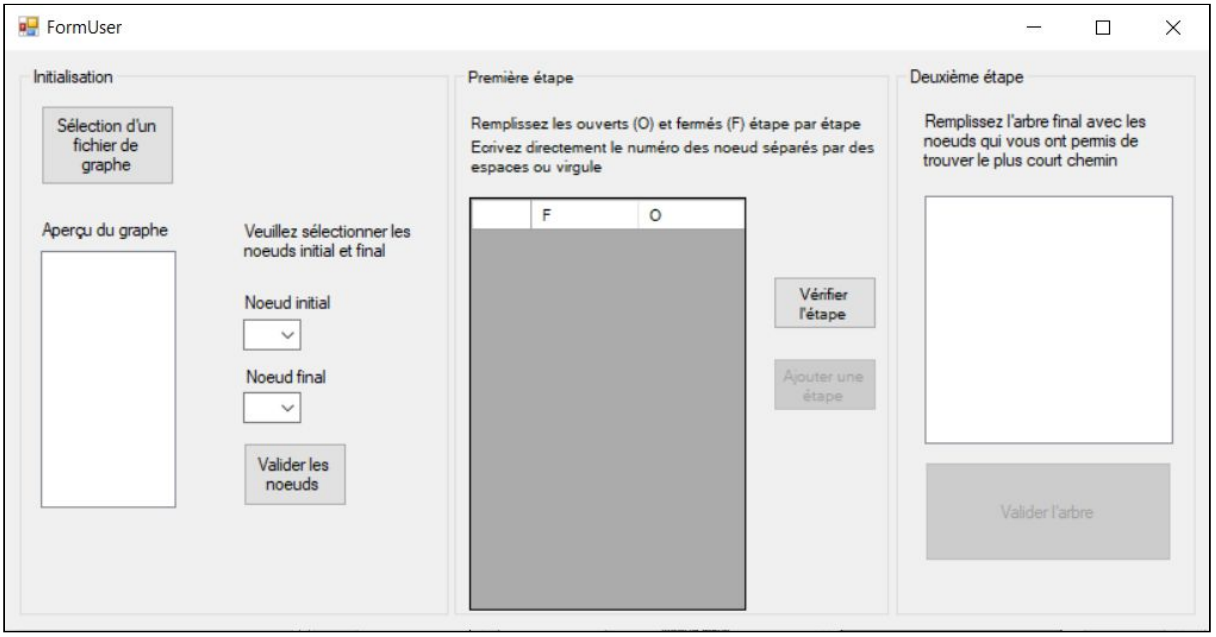


Figure 4 : interface utilisateur pour la partie 2

L'interface se sépare en trois parties :

- la première pour initialiser l'application : importation du fichier texte des données d'un graph (selon la construction conseillée)
- la deuxième pour "faire tourner Dijkstra à la main"
- la dernière pour évaluer l'arbre final des distances

3) Initialisation

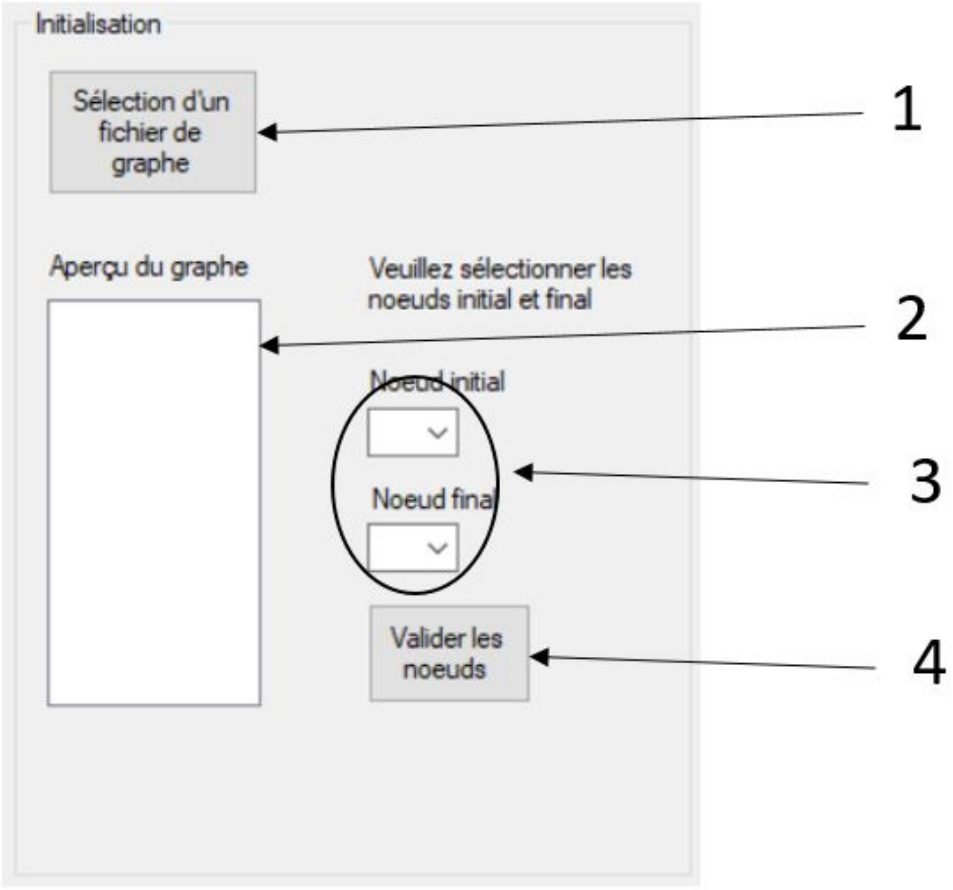


Figure 5 : partie initialisation

(1) - Bouton qui permet d'importer un fichier texte de la forme définie par le programme fourni (même si aucun fichier n'est importé, un graphe par défaut est initialisé).
(2) - ListBox qui permet d'afficher le graph importé : chaque ligne est construite de cette manière : "noeud1 ---> noeud2 : valeur".
(3) - ComboBox qui permettent de sélectionner le noeud initial et final pour trouver le chemin le plus court entre ces deux points.
(4) - Bouton qui valide les noeuds et permet de commencer la première partie de l'évaluation (si les noeuds sont valides, une ligne est ajouté au DataGridView de la première partie).

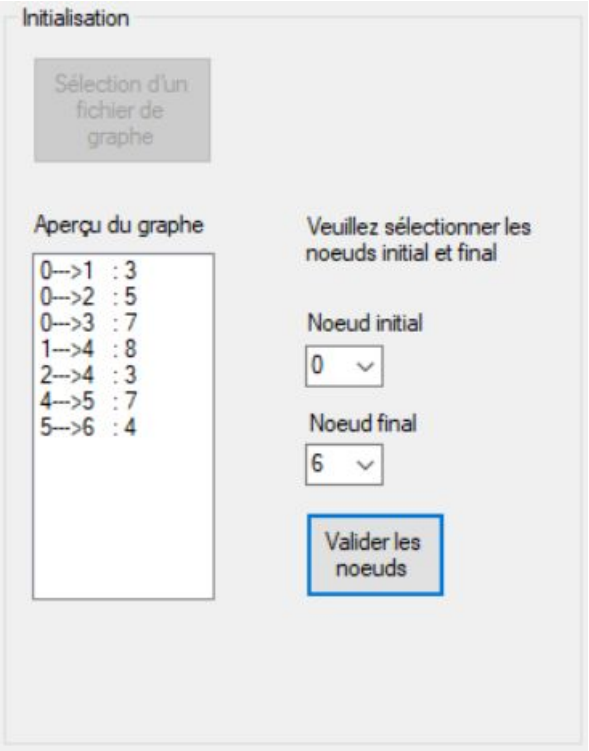


Figure 6 : initialisation faite (avec vérification des noeuds)

4) Première partie

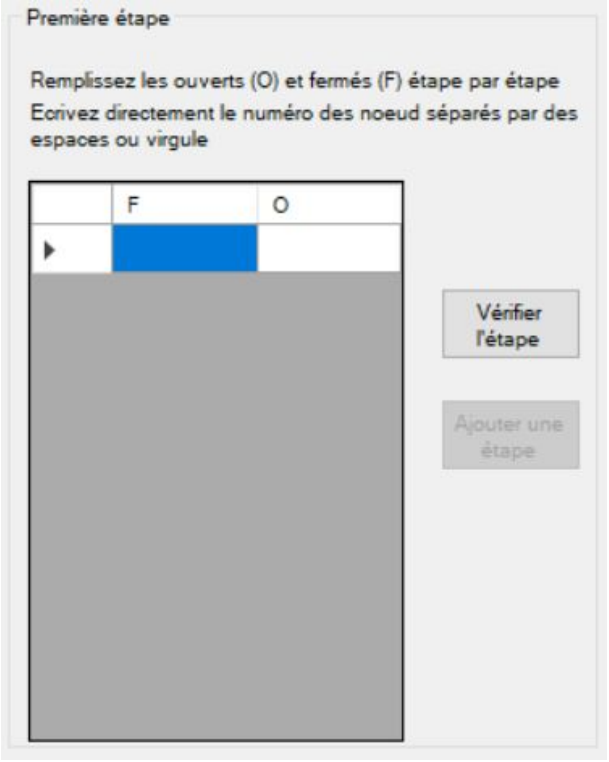


Figure 7 : première partie initialisée

Après avoir valider les noeuds, on peut remplir les ouverts et fermés étape par étape. Voici comment cette partie se déroule :

(1)

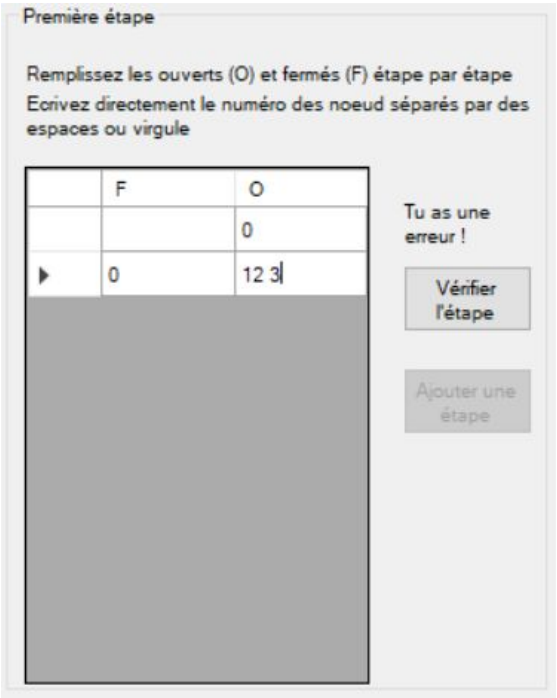


Figure 8a : erreur de remplissage

L'utilisateur remplit les cases comme spécifié, puis vérifie leur validité. S'il y a une erreur, il ne peut pas ajouter de ligne.

(2)

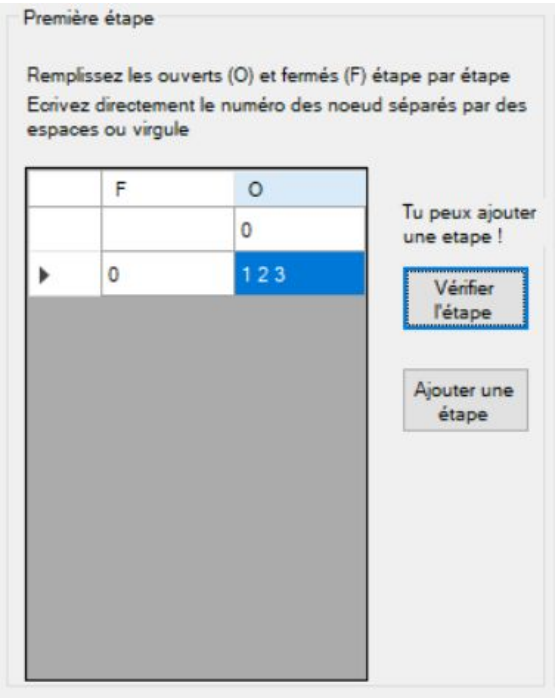


Figure 8b : bon remplissage

Si les cases sont bien remplies, on peut ajouter un étape (une ligne au DataGridView).

(3)

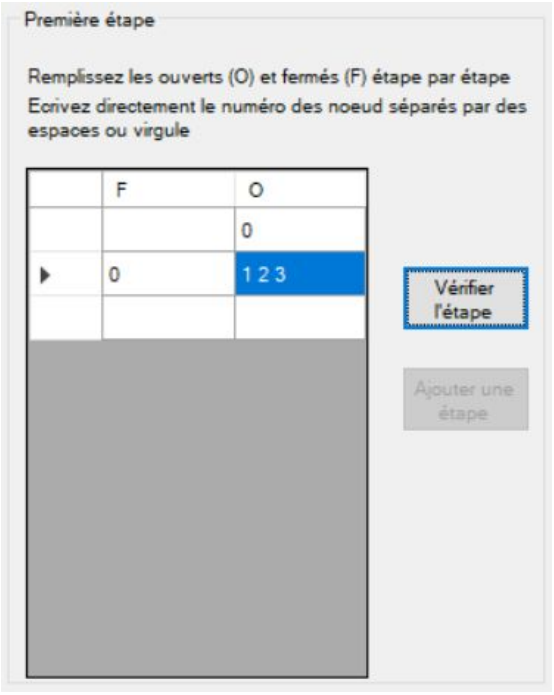


Figure 8c : ligne ajoutée

Une fois la ligne ajoutée, il peut continuer.

(4)

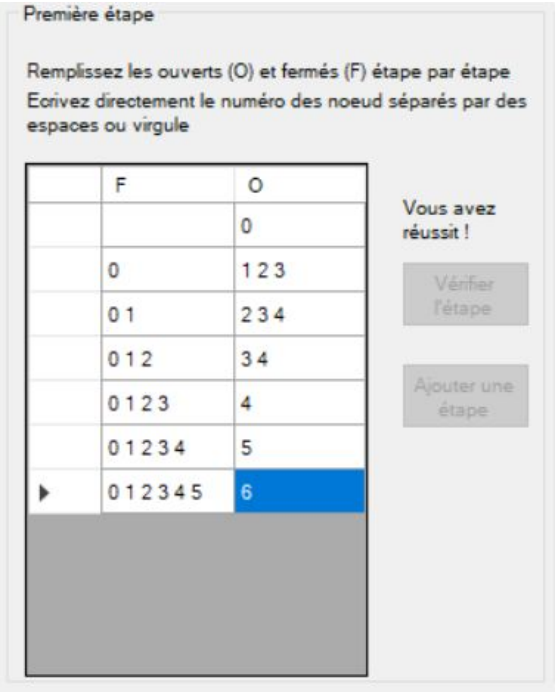


Figure 8d : tableau rempli

Une fois le tableau entièrement rempli, cliquer sur "Ajouter une étape" valide le tableau et permet de passer à la deuxième étape.

5) Seconde partie

Figure 9 : deuxième partie initialisée

Finir de remplir le tableau des ouverts et fermés permet de voir le squelette de l'arbre final. Mais celui-ci n'est pas rempli, il faut remplacer les "?" par les numéros des noeuds correspondant. Pour ce faire, il suffit de double-cliquer sur un noeud du TreeView pour l'éditer (figure 10). Une fois que l'utilisateur pense avoir bien rempli l'arbre, il peut le vérifier (figure 11a et 11b) (NB : le bouton de vérification est indisponible tant qu'aucune édition n'a été faite.

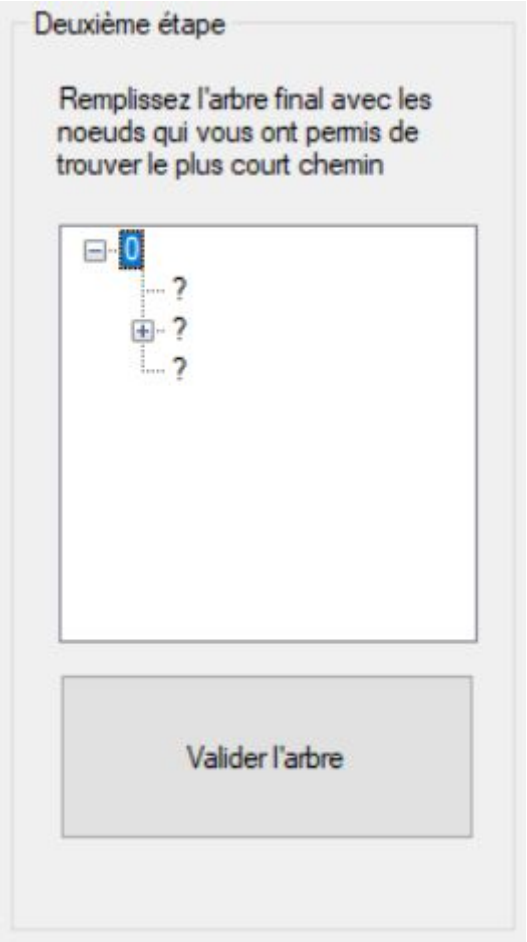


Figure 10 : édition d'un noeud

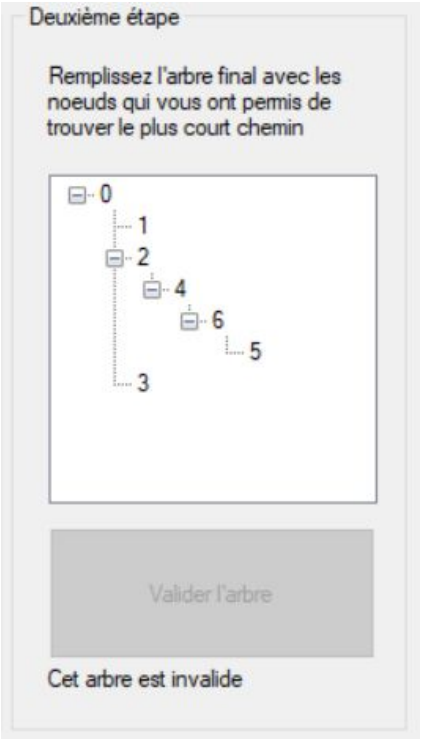


Figure 11a : erreur dans l'arbre

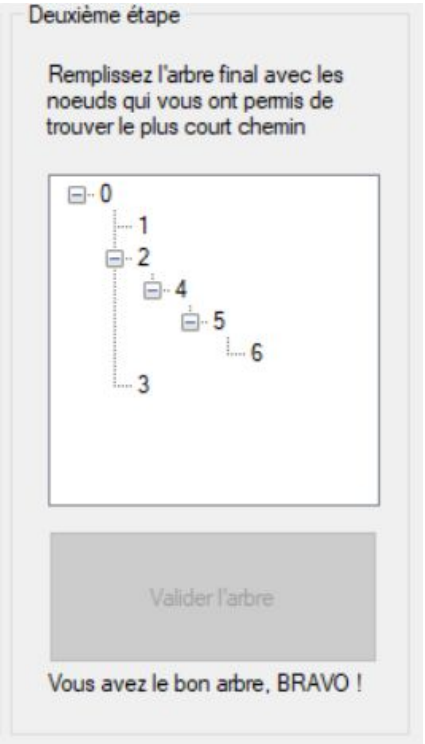


Figure 11b : arbre valide

III. Gestion de projet

Pour ce qui est de la gestion de projet, nous avons principalement utilisé GitHub pour le partage du code. Cela nous permettait de travailler sur deux parties différentes à distance et de partager notre avancement. Nous avons travaillé en parallèle, Florian a travaillé sur la partie 2 et Lucas sur la partie 1. A chaque avancée, nous avons fait tester le code par l'un par l'autre afin que le binôme comprenne les deux parties et confirme les choix faits en début de projet. Nous organisions des réunions toutes les semaines afin d'avoir un travail continue dans le temps.

IV. Conclusion

Nous avons réalisé dans un premier temps un questionnaire parcourant tout le cours d'Intelligence Artificielle qui est simple d'utilisation, et dans un second temps une interface pour pouvoir appliquer l'algorithme de Dijkstra et le comprendre. Cette application pourrait ainsi être ré-utilisée à des fins pédagogiques pour l'apprentissage et/ou l'évaluation des connaissances dans le domaine de l'IA.

Ce projet nous a permis de nous améliorer dans la conception d'un WindowsForm et ainsi approfondir nos connaissances dans ce domaine. Nous avons aussi dû reprendre le code de quelqu'un d'autre, ce qui implique de comprendre le code écrit par une autre personne, puis de l'adapter à notre contexte.

En conclusion, nous avons alors un projet qui répond aux exigences demandées et qui est ergonomique pour l'utilisateur.

Annexe

Voici la liste des questions du questionnaire de la partie 1 :

Question 1 : Si l'on a un bras robotique, présentant des liaisons "pivots", avec 6 axes de rotations, combien a t-on de degrés de liberté?

Réponses :
_ 7
_ 6
_ 8

Question 2 : Quelle est la fonction d'un servo contrôleur?

Réponses :
_ Reçoit des pulsations d'une Arduino(exemple), et suivant les pulsations renvoie en sortie un signal à un servo moteur
_ Fonctionne comme un processeur d'un ordinateur
_ Décide des déplacements d'un bras robotique

Question 3 : Quel est le type de capteur utilisé pour la détection d'obstacle dans une voiture autonome?

Réponses :
_ Lidar
_ Capteur à ultrason
_ Sonar

Question 4 : Quelle est la différence entre une IA faible et une IA forte?

Réponses :
_L'efficacité des programmes est accrue
_ Les programmes traitent une situation plus rapidement
_ Les programmes ont autant de conscience que les humains

Question 5 : L'algorithme MinMax est :

Réponses :
_ Utilisé en IA pour jouer à deux contre deux
_ Utilisé en IA pour trouver le chemin le plus court
_ Utilisé pour la détection de contours en vision par ordinateur

Question 6 : Que faut il faire pour passer d'un volume de texte à des cadres syntaxiques?

Réponses :

_ Une analyse syntaxique
_ Une généralisation
_ Une agrégation statistique

Question 7 : Une ontologie est…

Réponses :
_ un modèle permettant de définir une suite d'événements réalisable par un processus
_ un modèle de données représentatif d'un ensemble de concepts dans un domaine, ainsi que des relations entre ces concepts
_ un ensemble de neurones formels interconnectés permettant la résolution de problèmes complexes tels que la reconnaissance des formes

Question 8 : Comment fonctionne la compréhension d'un texte par l'IA?

Réponses:
_ Par mots clés associés
_ Par compréhension du sens des phrases
_ Par analyse du contexte des phrases

Question 9 : Comment placer 8 reines sur un échiquier 8x8 sans qu'aucune ne soit prise avec une autre?

Réponses :
_ Pas sur les mêmes lignes mais sur les mêmes diagonales
_ Sur les mêmes lignes et sur les mêmes diagonales
_ Pas sur les mêmes lignes ni les mêmes diagonales

Question 10 : Quel est l'objectif de l'algorithme de Dijkstra?

Réponses :
_ Résoudre un système impliquant un réseau neuronal
_ Trouver le chemin le plus court entre un E initial et un E final
_ Déterminer une réponse logique lors d'un traitement de texte

Question 11 : Les limites de Dijkstra sont..

Réponses :
_ Lorsqu'il y a trop de noeuds ouverts, l'algorithme est inexploitable
_ Lorsqu'il y a un très grand nombre de successeurs, l'algorithme est inexploitable
_ Lorsqu'il y a trop de noeuds fermés, l'algorithme est inexploitable

Question 12 : Par quoi faut il remplacer X pour compter le nombre de relations non symétriques du graphe?

Réponses:
_ if ((M[i,j] == "a") || (M[i,j]== "b")) cpt++

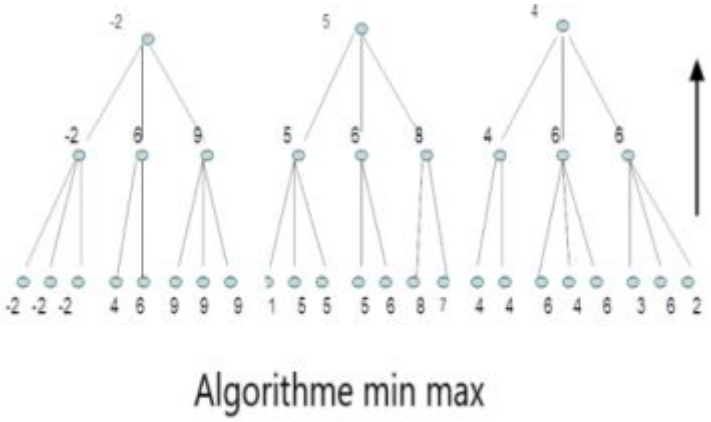
_ if ((M[i,j] != "0") et (M[j,i] != M[i,j])) cpt++
_ Ni a, ni b, il faut d'abord modifier les lignes précédentes.

Soit M une matrice d'adjacence de taille NxN remplie avec des « a », des « b » ou éventuellement avec des « 0 » en cas d'absence de relation. On considère le code ci-dessous :

```
int cpt = 0 ;
for (int i=0 ; i<n ; i++)
  for (int j=0 ; j<n ; j++)
      X
```

Question 13 : D'après cet arbre, quel est le meilleur chemin à adopter?

Réponses :
_ Gauche
_ Milieu
_ Droite



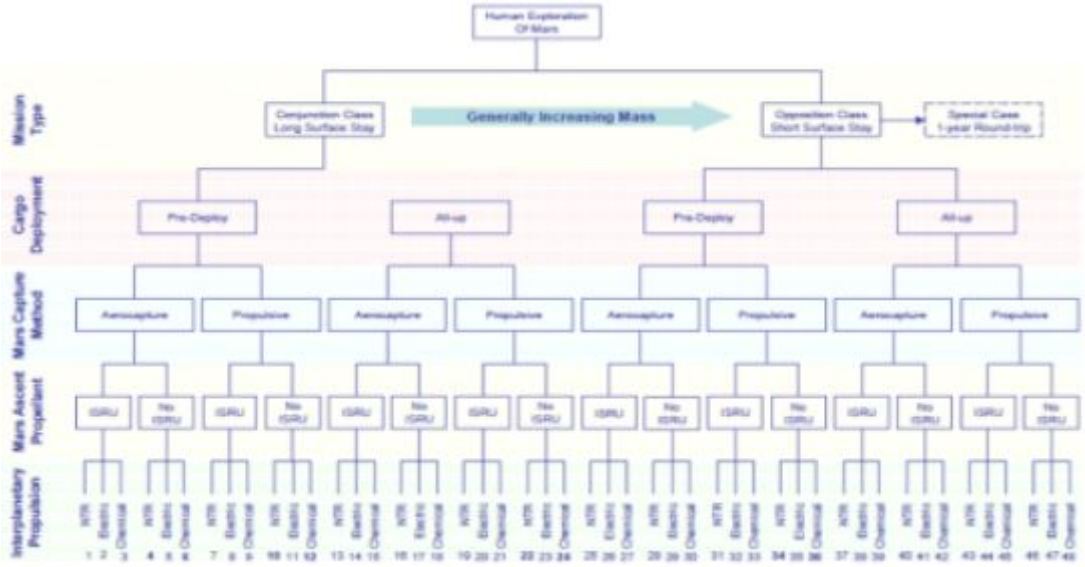
Algorithme min max

Question 14 : Quel est le mot à trouver?

Réponses:
_ Arbre
_ Titre
_ Cadre

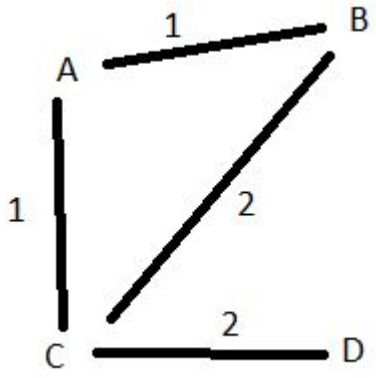
Question 15 : Que représente cette image?

Réponses :
_ Arbre de décision
_ Ontologie
_ Réseau Bayésien



Question 16 : Construisez la matrice adjacente associée, et quelle valeurs auriez vous mis sur la relation A/D?

Réponses :
_ 1
_ -1
_ 3



Question 17 : Quelle différence trouve t-on entre un algorithme Minmax et A*?

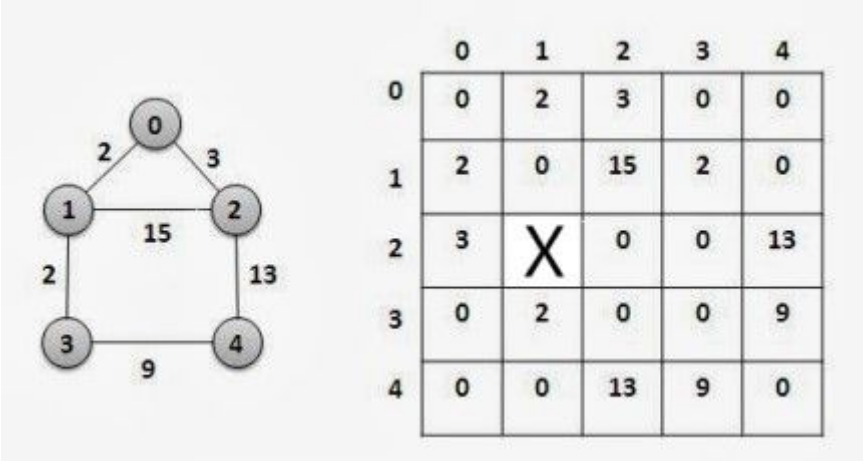
Réponses :
_ On n'utilise pas le même type de conception
_ On n'explorera pas les branches inutiles
_ Le contexte d'utilisation n'est pas le même

Question 18 : Dans quel cas A* garantit d'avoir le chemin le plus court?

Réponses :
_ Heuristique est un minorant du coût du chemin restant réel
_ Association avec l'algorithme de Dijkstra
_ On ne prend d'heuristique pour trouver le chemin le plus court

Question 19 : Construisez la matrice adjacente associée, et quelle valeurs auriez vous mis sur la relation 1/2?

Réponses :
_ 15
_ 0
_ 3



	0	1	2	3	4
0	0	2	3	0	0
1	2	0	15	2	0
2	3	X	0	0	13
3	0	2	0	0	9
4	0	0	13	9	0

Question 20 : Quel algorithme va être utilisé dans un puissance 4 pour déterminer le nombre de coup minimum pour gagner?

Réponses :
_ Algorithme de Dijkstra
_ Algorithme Minmax
_ Algorithme A*