**Rapport sur l'utilisation de l'IA dans la conception du tracé du graphe**

L'objectif de ce rapport est de documenter comment l'intelligence artificielle, ici en l’occurrence ChatGPT (GPT-o4) a été utilisée pour assister le développement des fonctions de tracé de graphe. Grâce aux suggestions et aux améliorations proposées par l'IA, on a pu partir de ce qu’on savait déjà faire grâce au semestre précédent (System.Drawing) pour obtenir un affichage optimisé et du code donc plus lisible.

1. **Fonction DessinerGraphe**

Initialement, l’idée était de créer une fonction permettant de dessiner un graphe avec ses liens et ses nœuds en évitant la superposition des éléments.

Voici le prompt que nous lui avons donné :

*Comment est-il possible dessiner un graphe en C#, de préférence en utilisant System.Drawing, en évitant les superpositions des sommets ?*

L'IA a suggéré plusieurs points d'amélioration, notamment :

* L'utilisation de Graphics.SmoothingMode, c’est à dire AntiAlias pour améliorer la qualité du rendu.
* Une approche pour tracer d'abord les liens, puis dessiner les sommets pour éviter qu'ils ne soient masqués. Il y a une sorte de superpositions de couches.
* L'utilisation d'un dictionnaire pour stocker les positions des nœuds afin de simplifier l'accès et l'affichage.

Grâce à ces conseils, j'ai pu structurer ma méthode et améliorer la lisibilité du graphe.

**2. Fonction GenererPositionsSansSuperposition**

Mon premier essai consistait à placer les nœuds aléatoirement, mais cela entraînait de nombreuses superpositions. Je n’ai malheureusement pas de screen écran de cette version car le programme C# écrit par-dessus le même chemin à chaque génération de l’image.

Alors il fallait demander à ChatGPT de désormais les superpositions (ce qu’il n’avait donc pas gérer après le premier prompt) :

*Je voudrais que les positions des nœuds soient bien aléatoires tout en garantissant une distance minimale entre eux ? Ce qui évitera la superposition des nœuds.*

L'IA conseille à la suite de cela de :

* Utiliser une boucle de tentatives avec un nombre maximal d’essais pour garantir qu’un nœud soit placé correctement sans chevauchement.
* Créer une liste des positions déjà utilisées et vérifier la distance minimale avant de valider une nouvelle position.
* Implémenter une fonction de calcul de distance Distance(PointF p1, PointF p2) pour vérifier si deux points sont trop proches.

En appliquant ces suggestions, on a pu assurer un espacement minimal entre les nœuds, améliorant ainsi la lisibilité du graphe.

**3. Exploration d'une approche avec coordonnées polaires**

Avant de finaliser la solution actuelle, j'avais de mon côté exploré une autre méthode basée sur des coordonnées polaires pour organiser les nœuds autour d'un cercle, plus précisément autour du nœud avec le plus grand degré. J'avais formulé la demande suivante à l'IA :

*Peux-tu m'aider à positionner des nœuds de graphe en cercle en utilisant des coordonnées polaires ? Je voudrais que soit positionné au centre du cercle le sommet avec le plus haut degré.*

Même si cette méthode était efficace, elle était illisible car il y a beaucoup trop de liens. Pareil que l’exemple précédent, les images générées par cet version de l’algo ont été « perdues ».

**Conclusion**

ChatGPT nous a permis de concevoir et d'optimiser ces fonctions. En fournissant des prompts assez précis, la structuration de notre classe GraphDrawer a été simple et plus qu’essentielle puisque l’on va s’en resservir pour le projet final. En plus de cela, le visuel permet aux utilisateurs de comprendre ce que fait le programme.

Mathieu ANTUNES-HOLUE & Antonin BIRON – TD B