Rapport du projet Calcule d'un trajet de transports publics

1. Introduction

La représentation d'un réseau de transport en Prolog trouve sa pertinence dans la résolution efficace du problème complexe de la recherche d'itinéraires. En utilisant Prolog, un langage de programmation logique, nous pouvons modéliser les lignes, arrêts et horaires des transports publics, facilitant ainsi la mise en œuvre d'algorithmes de recherche pour déterminer les trajets optimaux entre deux points du réseau. Cette approche permet une représentation claire et logique des relations entre les éléments du réseau, offrant une solution élégante pour répondre aux besoins des utilisateurs en matière de planification de trajets.

2. Représentation du Réseau de Transport

J ai utiliser une structure qui permet de représenter de manière détaillée chaque aspect d'une ligne de transport, des arrêts desservis aux horaires de départ . Pour cela j'ai utiliser le prédicat ligne/5 est utilisé pour représenter chaque ligne du réseau de transport. Il est défini avec cinq arguments :

- **Nom:** Décrit le nom de la ligne, pouvant être un numéro ou une lettre majuscule.
- **Type:** Indique le type de moyen de transport, tel que métro.
- **LArret:** Liste des arrêts desservis par la ligne avec leur temps nécessaire à parcourir la distance entre eux.
- **LTrajetAller:** Détaille les horaires de départ de la ligne du premier arrêt au dernier pour l'aller.
- **LTrajetRetour:** Spécifie les horaires de départ de la ligne du dernier arrêt au premier pour le retour.

Un exemple de ligne définie serait :

ligne(2, metro, [[nation, 0],[avron, 1],[alexandre_dumas, 2],…,[porte_dauphine, 3], [[5,0],2,[1,45]],[[5,15],2,[1,55]]).

Explication des éléments

- **2:** Le numéro ou l'identifiant de la ligne.
- **metro:** Le type de moyen de transport, ici le métro.
- **LArret:** La liste des arrêts desservis avec leur temps nécessaire, par exemple, [nation, 0] signifie que depuis l'arrêt "nation", aucun temps n'est nécessaire pour atteindre le prochain arrêt.
- **LTrajetAller:** Les horaires de départ de la ligne de "nation" à "porte_dauphine", avec un départ toutes les 2 minutes à partir de 05h00 jusqu'à 01h45.
- **LTrajetRetour:** Les horaires de départ du dernier arrêt ("porte_dauphine") vers le premier ("nation"), avec un départ toutes les 2 minutes à partir de 05h15 jusqu'à 01h55.

Cette structure permet de faciliter la recherche d'itinéraires pour les utilisateurs du réseau de transport.

3. Algorithmes pour la Recherche d'Itinéraires

J ai développé des prédicats pour manipuler les horaires, vérifier la faisabilité des connexions entre les arrêts et trouver des itinéraires entre deux points du réseau. Les prédicats addh/3, affiche/1, lig/3, ligtot/4, ligtard/4, itinTot/4et itinTard/4ont été élaborés pour réaliser ces fonctionnalités, offrant ainsi des outils pour la recherche d'itinéraires.

Addh/3 : Ce prédicat ajoute un certain nombre de minutes à un horaire donné. En utilisant des calculs de minutes et de modulos pour obtenir les heures et les minutes, il effectue l'addition de manière appropriée. C'est une fonction utilitaire pour manipuler les horaires.

- **Exemple d'utilisation :** `addh([13, 34], 30, [14, 4])` est vrai, car 13h34 + 30 minutes donne 14h04.

Affiche/1 : Ce prédicat affiche l'heure dans un format lisible par l'utilisateur en utilisant la fonction de formatage de Prolog. Il présente l'horaire sous la forme HH:MM, facilitant la lisibilité des heures.

- **Exemple d'utilisation :** `affiche([5,37])` affichera "05:37" à l'écran.

lig /3: Ce prédicat vérifie si une ligne de transport passe d'un arrêt à un autre. Il examine si les arrêts sont membres de la liste des arrêts desservis par une ligne spécifique, permettant de vérifier la faisabilité de connexion entre ces arrêts.

- **Exemple d'utilisation :** lig(nation, avron, 2). % Résultat : Vrai (car la ligne 2 passe de "nation" à "avron")

ligtot /4: Ce prédicat identifie la ligne qui part le plus tôt possible après un horaire spécifié entre deux arrêts donnés. Il recherche les horaires de départ correspondants à ces arrêts et sélectionne la ligne dont les horaires sont après ou égaux à l'horaire spécifié.

- **Exemple d'utilisation** : ligtot(jourdain, chatelet, Ligne, [5, 0]). % Résultat : Ligne = 11

Ligtard/4 : Cet autre prédicat identifie la ligne qui arrive le plus tard possible avant un horaire spécifié entre deux arrêts donnés. Il recherche les horaires d'arrivée correspondants à ces arrêts et sélectionne la ligne dont les horaires sont avant ou égaux à l'horaire spécifié.

- **Exemple d'utilisation :** ligtard(place_fetes, jaures, Ligne, [5, 30]).

% Résultat : Ligne = bis_7

itinTot/4 : Ce prédicat identifie un itinéraire partant le plus tôt possible après un horaire spécifié entre deux arrêts. Il recherche les horaires de départ et d'arrivée correspondants à ces arrêts et sélectionne l'itinéraire dont l'heure de départ est après ou égale à l'horaire spécifié.

- **Exemple d'utilisation :** itinTot(mairie_lilas, chatelet,[5, 30] , Parcours).
  % Résultat : Parcours = [mairie_lilas, 0, chatelet, 1]

itinTard /4: Ce prédicat identifie un itinéraire arrivant le plus tard possible avant un horaire spécifié entre deux arrêts. Il recherche les horaires de départ et d'arrivée correspondants à ces arrêts et sélectionne l'itinéraire dont l'heure d'arrivée est avant ou égale à l'horaire spécifié.

- **Exemple d'utilisation :** itinTard(nation, pere_lachaise, [8, 20], Parcours). % Résultat:

Parcours = [nation, 0, pere_lachaise, 2]

4. Amélioration des Prédicats pour les Options d'Itinéraires

Une seconde partie du projet a consisté à enrichir les prédicats existants en ajoutant des options pour les recherches d'itinéraires. Des prédicats tels que ligne_passe/5, ligne_arrivant_plus_tard/6, ligne_partant_plus_tard/6, itineraire_arrivant_plus_tard/7et itineraire_partant_plus_tard/7ont été conçus pour tenir compte des préférences des utilisateurs, comme le choix du réseau, la longueur du trajet ou le nombre de correspondances.

ligne_passe/5 : le prédicat lig avec amélioration

- **Exemple d'utilisation :** ligne_passe(nation, pere_lachaise, Ligne, metro, Longueur).
  % Résultat : Ligne = 2,Longueur = [[5, 0], 2, [1, 45]]

ligne_arrivant_plus_tard/6 : le prédicat ligtot avec amélioration
Exemple d'utilisation : ligne_arrivant_plus_tard(nation, pere_lachaise, Ligne, [8, 20], metro, Longueur). % Résultat :

Ligne = 2,Longueur = [[5, 0], 2, [1, 45]]

ligne_partant_plus_tard/6 : le prédicat ligtard avec amélioration

- **Exemple d'utilisation :** ligne_partant_plus_tard(nation, pere_lachaise, Ligne, [0,0], metro, Longueur). % Résultat : Ligne = 2, Longueur = [[5, 0], 2, [1, 45]]

itineraire_arrivant_plus_tard/7 : le prédicat intintard avec amélioration

- **Exemple d'utilisation :** itineraire_arrivant_plus_tard(nation, pere_lachaise, [5,0], Parcours, metro, Longueur, Correspondances).
  % Résultat : Longueur = [[5, 0], 2, [1, 45]],Correspondances = [[5, 15], 2, [1, 55]]

itineraire_partant_plus_tard/7 : le prédicat intintot avec amélioration

- **Exemple d'utilisation :** itineraire_partant_plus_tard(nation, alexandre_dumas, [0, 0], Parcours, metro, Longueur, Correspondances). % Résultat :

Longueur = [[5, 0], 2, [1, 45]],Correspondances = [[5, 15], 2, [1, 55]]

5. Interface utilisateur

L'objectif de cette section du projet était de créer une interface utilisateur permettant aux utilisateurs de choisir le mode de transport (métro ou bus), de sélectionner leurs stations de

départ et d'arrivée, ainsi que de préciser l'heure d'arrivée. J ai également mis en place des fonctions pour afficher les stations disponibles et les parcours possibles entre les stations choisies.

5.1 Description de l'Interface Utilisateur

L'interface utilisateur commence par demander à l'utilisateur de choisir le mode de transport (métro ou bus). En fonction de ce choix, le programme affiche les stations disponibles pour le mode de transport sélectionné. L'utilisateur est ensuite invité à entrer dans la station de départ, la station d'arrivée et l'heure d'arrivée souhaitée.
Si le mode de transport sélectionné est le métro, le programme affiche les stations de métro disponibles et propose les parcours possibles entre les stations de départ et d'arrivée. Pour le bus, le processus est similaire, mais avec les gares de bus.

5.2 Description des Prédicats

interface_utilisateur/0 :
Ce prédicat est le point d'entrée de l'interface utilisateur. Il demande à l'utilisateur de choisir le mode de transport et d'appeler ensuite le prédicat choisir_transport/1pour afficher les stations correspondantes

choisir_transport/1 :
Ce prédicat prend le mode de transport choisi par l'utilisateur en entrée et appelle la fonction appropriée (afficher_stations_metro afficher_stations_bus) en fonction de ce choix. Si le mode de transport n'est pas reconnu, un message d'erreur est affiché.

afficher_stations_metro/0etafficher_stations_bus/0 :
Ces prédicats affichent respectivement les stations disponibles pour le métro et le bus. Ils utilisent le prédicat afficher_liste_stations/1pour imprimer la liste des stations.

afficher_liste_stations/1 :
Ce prédicat affiche une liste de stations. Il est utilisé par les prédicats afficher_stations_metroet afficher_stations_bus.

afficher_parcours/1
Ce prédicat affiche les parcours possibles entre les stations de départ et d'arrivée. Il utilise le prédicat itinTot/4pour obtenir les itinéraires et les affiches à l'utilisateur.

L'implémentation de l'interface utilisateur offre une expérience utilisateur intuitive pour l'utilisation du programme. Les prédicats associés fonctionnent de manière cohérente pour afficher les informations nécessaires. Cela contribue à rendre le programme convivial et accessible pour les utilisateurs.

6. Conclusion

Ce projet a été une opportunité de plonger dans la programmation logique et d'appliquer les concepts théoriques à la résolution pratique de problèmes réels. Malgré les défis rencontrés, nous avons réussi à créer un système fonctionnel pour aider les utilisateurs à planifier leurs trajets en transports en commun. Ce projet a permis d'explorer les possibilités et les défis de la programmation en Prolog dans un contexte concret et applicatif.