Ingé4 - Travaux Pratiques de C++
TP2 : Parcours du réseau férré parisien (binôme)

Johan Thapper
Polytech Paris-Sud, 2012-2013*

17 décembre 2012

1 Objectifs du TP

L'objectif applicatif de ce TP est de modéliser le réseau férré parisien (simplifié) et de le parcourir. L'objectif pédagogique de ce TP est de modéliser un problème énoncé succinctement en langue naturelle sans aucun support.

2 Rendu du TP

2.1 Archive

Archive au format tar.gz nommée `TP2_grpX_NOM1_NOM2.tar.gz` (remplacez *X* par votre numéro de groupe, *NOM1* et *NOM2* par vos noms, sans accents) à envoyer à `thapper@lri.fr`. La date limite, sans délai supplémentaire, est le 20 janvier. L'archive devra comporter les sources et un `Makefile` afin de pouvoir être compilée avec GCC sous Linux (vous êtes libres de faire une documentation).

3 Métro : durée d'un parcours

Un train se déplace entre deux stations du métro parisien, le temps de trajet dépend de la population des voyageurs. Votre programme doit pouvoir calculer le temps d'un parcours pour un voyageur particulier

*Crédits du TP : M. Falco, P. Le Bodic

entre deux stations saisies par l'utilisateur. Vous êtes libres de tous vos choix : pensez à la robustesse et l'optimisation. Si une donnée numérique n'est pas donnée dans l'énoncé, c'est à vous de l'estimer lors de vos tests d'implémentation.

Le temps entre deux stations sera calculé de la manière suivante : temps entre deux stations de métro d'une même ligne : deux minutes + (poids total des voyageurs / 10 000) ; temps entre deux stations de métro d'une ligne différente : huit minutes + 0,1 * âge du voyageur. Cette donnée intègre que plus il y a de monde dans la voiture, plus on perd de temps à chaque station en montée/descente de voyageurs.

On pose qu'un métro contient cinq voitures d'une capacité de cent voyageurs chacune. La population des voyageurs doit être aléatoire : vous êtes libres des bornes pour l'âge et le poids mais elles doivent être réalistes. La population des voyageurs dans les voitures de métro est triplement aléatoire : vous devez prévoir trois configurations que l'utilisateur pourra choisir à l'avance (**heure de pointe**, **heure normale**, **heure creuse**) ; configurations estimées de façon réaliste à partir de la capacité d'un métro. Sans aller jusqu'à chercher les statistiques de la RATP concernant les stations les plus bondées, il est tout de même attendu que les stations correspondances soient les plus fréquentées dans la répartition des voyageurs sur l'ensemble des stations.

Votre progamme doit également pouvoir intégrer une anomalie de parcours comme par exemple une ligne entière bloquée pendant le déroulement du parcours (incident technique, grève) ou une seule station inaccessible (colis suspect). Lorsque cette anomalie se produit, le programme doit être capable de calculer un autre parcours, si possible le plus avantageux. N'hésitez pas à modéliser d'autres anomalies.

3.1 Résumé

Le parcours d'un itinéraire se fera à l'aide d'un graphe non orienté qui comportera donc un ensemble de stations et un ensemble de connexions. Vous êtes toujours libres de l'implémentation, la clef est de gérer la liste des stations accessibles à partir d'une station et de savoir s'il s'agit d'une correspondance.

Vous allez estimer des données numériques non précisées dans l'énoncé, modéliser des anomalies de parcours (au moins celle de la ligne devenue inaccessible durant le parcours) et vous allez utiliser **l'algorithme de Dijkstra**[1].

Il est attendu que l'utilisateur puisse fournir quatre paramètres au lancement du programme : la station de départ, la station d'arrivée, la fréquentation (heure de pointe, heure normale, heure creuse) et une anomalie se déclenchant durant le parcours (au moins deux choix pour ce dernier paramètre : anomalie ou pas d'anomalie).

4 Partie facultative (sans bonus)

Vous pouvez considérer également le RER en plus du métro en ajoutant au fichier texte les stations de RER qui sont à l'intérieur de Paris (et estimez les capacités respectives de chaque ligne RER.)

5 Fichiers

Fichiers à récupérer dans le répertoire `http://www.lri.fr/~thapper/courses/cpp-2012/TP2` :
- `TP2.pdf`
- `metro.txt`

Le graphe du métro Parisien a été simplifié :
- ligne 13 : suppression de la bifurcation, le terminus est *LaFourche*
- ligne 10 : suppression de la distinction de sens de parcours, le terminus est *Javel-AndreCitroen*
- ligne 7 : suppression de la bifurcation, le terminus est *MaisonBlanche*

1. `http://fr.wikipedia.org/wiki/Algorithme_de_Dijkstra`