Rapport - Projet thorie des graphes

Younes Ben Yamna - Malek Zemni

20 avril 2016

1 Introduction

2 Graphe

2.1 Sommets du graphe

Considration des sommets

On considre comme sommets les points o se croisent plusieurs pistes. La carte fournie a t simplife, c'est dire qu'on considre en fait comme sommets les zones (et non les points) o se croisent plusieurs pistes.
Les noms des sommets sont donc choisis par rapport au nom de la zone o se trouve le sommet. Si la zone ne porte pas de nom, le nom du sommet sera choisi par rapport au nom de la piste ou de la remonte la plus proche.

Indices des sommets

Chaque sommet est repr par un indice allant de 0 44 : on a donc 45 sommets au total.
Voici la liste complte des sommets, ainsi que leurs indices respectifs :

- 0 PIC BLANC
- 1 GROTTE DE GLACE
- 2 SOMMET 3060
- 3 SARENNE BASSE
- 4 CLOCHER DE MACLE
- 5 LAC BLANC
- 6 LIEVRE BLANC
- 7 PLAT DES MARMOTTES
- 8 MINE DE L'HERPIE
- 9 SOMMET 2100
- 10 SOMMET DES VACHETTES
- 11 SIGNAL DE L'HOMME
- 12 L'ALPETTE
- 13 COL DU COUARD
- 14 CASCADE
- 15 CLOS GIRAUD

— 16 MONFRAIS
— 17 SIGNAL
— 18 RIFNEL EXPRESS
— 19 CHALVET
— 20 AURIS EXPRESS
— 21 FONTFROIDE
— 22 LOUVETS
— 23 POUTRAN
— 24 CHAMP CLOTURE
— 25 STADE
— 26 SCHUSS
— 27 ALPE D'HUEZ
— 28 GRANDE SURE
— 29 ECLOSE
— 30 SURES
— 31 COL
— 32 AURIS EN OISANS
— 33 LA VILLETTE
— 34 VAUJANY
— 35 L'EVERSIN D'OZ
— 36 OZ EN OISANS
— 37 PETIT PRINCE
— 38 VILLAGE
— 39 MARONNE
— 40 VILLARD RECULAS
— 41 HUEZ
— 42 DOME DES PETITES ROUSSES
— 43 ALPAURIS
— 44 L'ALPETTE

2.2 Arcs du graphe

Considration des arcs

2.3 Fichier du graphe

3 Structure

4 Algorithme

```
1 void initialise_tableau_antecedant(antecedant a[])
{ //initialise le tableau antecedant
3   int i;
   for (i = 0; i < V; i++)
5   {
```

```
    a[i].sommet=i;
    a[i].pere=-1;
  }
}

void initialise_tableau_parcour(parcour p[],int depart)
{ //initialise le tableau parcour
  int i;
  for (i = 0; i < V; i++)
  {
    p[i].sommet=i;
    p[i].parcouru= 0;
    p[i].poid=G[depart][i].poids;
  }
}

int recherche_pere(parcour p[])
{ //recherche le sommet de poid minimum (le poid=1000 a modifier)
  int indice_pere=0;
  while((p[indice_pere].poid<1)&&(indice_pere<(V-1))) indice_pere
    ++;
  int i;
  for (i = (indice_pere+1); i < V; i++)
  {
    if( ( p[i].poid < p[indice_pere].poid ) && ( p[i].poid > 0 ) &&
     (p[i].parcouru!=1))
      indice_pere = i;
  }
  if(( p[indice_pere].poid < 1 ) || (p[indice_pere].parcouru == 1 )
    )
    return -1;
  return indice_pere;
}

int recherche_fils(int pere,int f[])
{//recherche tout les fils de pere
  int nombre_fils=0;
  int i;
  for (i = 0; i < V; i++)
  {
    if ( G[pere][i].poids < 1000 ){
      f[nombre_fils] = i;
      nombre_fils++;
    }
  }
  return nombre_fils;
}

void dijkstra(int depart,int arrivee,antecedant a[],parcour p[])
{
  int pere=depart;
  int i,j,fils;
  int f[10];
  if (depart==arrivee){
     printf("Vous etes deja a destination '%s'\n\n\n",nomSommet(
    depart));
    return;}
```

```
59    while(pere!=-1)
      {
61        printf("le pere est %d\n",pere);
        int nbf=recherche_fils(pere,f);
63      for (i = 0; i < nbf; i++)
        {
65        fils=f[i];
          printf("le fils est %d\n",fils);
67        if((p[fils].parcouru==0)&&((p[fils].poid==-1)||(p[pere].poid+
        G[pere][fils].poids<p[fils].poid)))
          {
69          p[fils].poid=p[pere].poid+G[pere][fils].poids;
            a[fils].pere=pere;
71        }
        }
73      p[pere].parcouru=1;
        pere=recherche_pere(p);
75    }
      //###############affichage####################
77    int chemin[50];
      i=0;
79    while(-1!=arrivee)
      {
81      chemin[i]=arrivee;
        i++;
83      arrivee=a[arrivee].pere;
      }
85    chemin[i]=depart;
      i++;
87    for (j = 0; j < i-1 ; j++)
      {
89      printf("%d ) vous devez partir du sommet '%s' et prendre '%s'
        jusqu'au sommet '%s'\n\n",j,nomSommet(chemin[i-j-1]),G[chemin[i
        -j-1]][chemin[i-j-2]].nom,nomSommet(chemin[i-j-2]));
      }
91    printf("%d ) vous etes arriv  a votre destination '%s' en %d
        secondes\n\n\n",j,nomSommet(chemin[0]),p[chemin[0]].poid);
  }
```

src/dijkstra.c

4.1 Lecture du graphe

4.2 Affichage du graphe

5 Conclusion