My Project

Generated by Doxygen 1.8.11

Contents

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 3

Class Documentation

3.1 Donnee Struct Reference

Structure définissant les données du problème.

`#include <Donnee.h>`

Public Attributes

- int m_nombre_lieux

  Nombre de lieux (incluant le village)
- int m_capacite

  Capacité du reservoir du drone.
- int * m_quantite_eau

  Demande de chaque lieu (la case 0 est inutilisée car le dépôt n'a aucune demande à voir satisfaire)
- int ** m_distancier

  distancier (les lignes et colonnes 0 correspondent au dépôt)

3.1.1 Detailed Description

Structure définissant les données du problème.

The documentation for this struct was generated from the following file:

- Donnee.h

3.2 Enumeration Class Reference

Public Member Functions

- Enumeration (Donnee data)

  Constructeur.
- Donnee get_m_d ()

  permet d'acceder aux données du problème
- void set_m_ListeTournees (Tournee tour)

  permet d'ajouter une tournee à l'ensemble de tournées à visiter
- void addToEnum (int array[ ], int size)

  permet de créer une tournée et de l'ajouter à l'énumeration
- std::vector< Tournee > get_m_ListeTournees ()

  permet d'accéder à la liste des tournées
- bool capaciteSuffisante (int array[ ], int size)

  permet de determiner si le reservoir du drone à une contenance suffisante pour visiter tout les points de la tournée
- void powerset (int n)

  permet de determiner l'ensemble des différentes tournées que le drône peut réaliser sans vider son réservoir
- int size ()

  permet d'accéder au nombre nombre de tournée qu'il est possible de faire

3.2.1 Constructor & Destructor Documentation

3.2.1.1 Enumeration::Enumeration (Donnee *data*)

Constructeur.

Parameters

data	données du problème

3.2.2 Member Function Documentation

3.2.2.1 void Enumeration::addToEnum (int *array[ ],* int *size*)

permet de créer une tournée et de l'ajouter à l'énumeration

Parameters

array[ ]	tableau contenant des indices de points d'eau
size	intervalle sur lequel on doit considérer le tableau

3.2.2.2 bool Enumeration::capaciteSuffisante (int *array[ ],* int *size*)

permet de determiner si le reservoir du drone à une contenance suffisante pour visiter tout les points de la tournée

Parameters

array[ ]	tableau contenant des indices de points d'eau
size	intervalle sur lequel on doit considérer le tableau

Returns

bool retourne oui si le drone à une capacite suffisante pour visiter tout les points d'eau dont l'indice est donné dans le tableau

3.2.2.3 Donnee Enumeration::get_m_d ( )

permet d'acceder aux données du problème

Returns

m_d données du problème

3.2.2.4 vector< Tournee > Enumeration::get_m_ListeTournees ( )

permet d'accéder à la liste des tournées

Returns

m_ListeTournees vecteur des tournées

3.2.2.5 void Enumeration::powerset (int *n*)

permet de determiner l'ensemble des différentes tournées que le drône peut réaliser sans vider son réservoir

Parameters

n	taille de l'ensemble à traiter

3.2.2.6 void Enumeration::set_m_ListeTournees (Tournee *tour*)

permet d'ajouter une tournee à l'ensemble de tournées à visiter

Parameters

Tournee	tournée à ajouter à l'énumeration

3.2.2.7 int Enumeration::size ( )

permet d'accéder au nombre nombre de tournée qu'il est possible de faire

Returns

m_ListeTournees.size()

The documentation for this class was generated from the following files:

- Enumeration.h
- Enumeration.cpp

3.3 Tournee Class Reference

Public Member Functions

- Tournee ()
  Constructeur de Tournee.
- int size ()
  Retourne le nombre de points d'eau visités.
- std::vector< int > getPointsdeau ()
  Getter sur m_pointsdeau.
- void setPermutationMin (Donnee d)
  permet de mettre à jour la valeur de la permutation offrant la plus petite distance à parcourir
- std::vector< int > getPermutationMin ()
  Getter sur m_permutationMin.
- int getDistance ()
  Getter sur m_distance.
- void setDistance (Donnee d)
  permet de mettre à jour la plus petite distance à parcourir afin de visiter tout les points d'eau de la tournée
- void add (int i, Donnee d)
  permet d'ajouter l'indice d'un nouveau point d'eau à visiter pendant la tournée
- std::vector< int > calculePlusPetiteDistancePerm (std::vector< int > tour, Donnee d)
  calcule la permutation offrant la plus petite distance à parcourir en brute force
- int calculeDistanceTour (std::vector< int > tour, Donnee d)
  calcule la distance parcourue pendant le tour fourni

3.3.1 Member Function Documentation

3.3.1.1 void Tournee::add (int *i*, Donnee *d*)

permet d'ajouter l'indice d'un nouveau point d'eau à visiter pendant la tournée

Parameters

i	indice du point d'eau à visiter pendant la tournée
d	données du problème

3.3.1.2 int Tournee::calculeDistanceTour (std::vector< int > *tour*, Donnee *d*)

calcule la distance parcourue pendant le tour fourni

Parameters

tour	indices des points d'eau visités
d	données du problème

Returns

res la distance entre deux points

3.3.1.3 vector< int > Tournee::calculePlusPetiteDistancePerm (std::vector< int > *tour*, Donnee *d*)

calcule la permutation offrant la plus petite distance à parcourir en brute force

Parameters

tour	indices des points d'eau visités
d	données du problème

Returns

meilleur_tour permutation minimale

3.3.1.4 int Tournee::getDistance ( )

Getter sur m_distance.

Returns

m_distance

3.3.1.5 vector< int > Tournee::getPermutationMin ( )

Getter sur m_permutationMin.

Returns

m_permutationMin

3.3.1.6 vector< int > Tournee::getPointsdeau ( )

Getter sur m_pointsdeau.

Returns

m_pointsdeau

3.3.1.7 void Tournee::setDistance (Donnee *d*)

permet de mettre à jour la plus petite distance à parcourir afin de visiter tout les points d'eau de la tournée

Parameters

d	données du problème

3.3.1.8 void Tournee::setPermutationMin (Donnee *d*)

permet de mettre à jour la valeur de la permutation offrant la plus petite distance à parcourir

Parameters

d	donnees du problème

3.3.1.9 int Tournee::size ( )

Retourne le nombre de points d'eau visités.

Returns

m_pointsdeau.size()

The documentation for this class was generated from the following files:

- Tournee.h
- Tournee.cpp

Chapter 4

File Documentation

4.1 Donnee.h File Reference

Définie les données du problème à résoudre, c'est à dire le nombre de points d'eau, la capacité du réservoir du drone accompagné du distancier.

```
#include <vector>
#include <string>
```
Include dependency graph for Donnee.h:

4.2 Enumeration.cpp File Reference

Définie une énumeration des différentes tournées que le drône peut réaliser sans vider son réservoir.

```
#include <vector>
#include "Enumeration.h"
```
Include dependency graph for Enumeration.cpp:

4.2.1 Detailed Description

Définie une énumeration des différentes tournées que le drône peut réaliser sans vider son réservoir.

4.3 Enumeration.h File Reference

Définie une énumeration des différentes tournées que le drône peut réaliser sans vider son réservoir.

```
#include <vector>
#include "Donnee.h"
#include "Tournee.h"
```
Include dependency graph for Enumeration.h: This graph shows which files directly or indirectly include this file:

Classes

- class Enumeration

4.3.1 Detailed Description

Définie une énumeration des différentes tournées que le drône peut réaliser sans vider son réservoir.

4.4 main.cpp File Reference

Permet de résoudre le problème du projet.

```
#include <iostream>
#include "stdio.h"
#include <stdlib.h>
#include <string.h>
#include <string>
#include <fstream>
#include <glpk.h>
#include <math.h>
#include "Donnee.h"
#include "Tournee.h"
#include "Enumeration.h"
#include "partitionEnsemble.h"
#include <ctime>
```

Include dependency graph for main.cpp:

Functions

- void lecture_data (char *file, Donnee *d)

  Lecture des données.
- int **main** (int argc, char *argv[ ])

4.4.1 Detailed Description

Permet de résoudre le problème du projet.

4.4.2 Function Documentation

4.4.2.1 void lecture_data (char * *file*, Donnee * *d*)

Lecture des données.

Parameters

*file	Le chemin du fichier de données
*d	Les données à remplir

4.5 partitionEnsemble.cpp File Reference

Résolution d'une instance de problème de partitionnement d'ensemble.

`#include "partitionEnsemble.h"`
Include dependency graph for partitionEnsemble.cpp:

Functions

- void partitionEnsemble (Enumeration enume, int nbLieux)

  //fonction permettant de résoudre le problème posé en utilisant GLPK et de déterminer les tournées que le drone doit choisir afin d'avoir un itinéraire optimal

4.5.1 Detailed Description

Résolution d'une instance de problème de partitionnement d'ensemble.

4.5.2 Function Documentation

4.5.2.1 void partitionEnsemble (Enumeration *enume*, int *nbLieux*)

//fonction permettant de résoudre le problème posé en utilisant GLPK et de déterminer les tournées que le drone doit choisir afin d'avoir un itinéraire optimal

Parameters

enum	Enumeration de l'ensemble des tournées possible
nbLieux	Le nombre de lieux du problème

4.6 Tournee.cpp File Reference

Définie une tournée de points d'eau, c'est à dire la suite que le drône va pouvoir visiter en une seule fois sans avoir à aller vider son réservoir.

`#include "Tournee.h"`
Include dependency graph for Tournee.cpp:

4.6.1 Detailed Description

Définie une tournée de points d'eau, c'est à dire la suite que le drône va pouvoir visiter en une seule fois sans avoir à aller vider son réservoir.

4.7 Tournee.h File Reference

Définie une tournée de points d'eau, c'est à dire la suite que le drône va pouvoir visiter en une seule fois sans avoir à aller vider son réservoir.

```
#include <iostream>
#include <vector>
#include <algorithm>
#include "Donnee.h"
```

Include dependency graph for Tournee.h: This graph shows which files directly or indirectly include this file:

Classes

- class Tournee

4.7.1 Detailed Description

Définie une tournée de points d'eau, c'est à dire la suite que le drône va pouvoir visiter en une seule fois sans avoir à aller vider son réservoir.

Index