Immagine che contiene Elementi grafici, Carattere, grafica, logo

Descrizione generata automaticamente

Politecnico di Torino

Corso di Laurea in Matematica per l’Ingegneria

**Discrete Fracture Network**

Progetto PCS 2024

|  |  |
| --- | --- |
| Relatori: | Candidati: |
| S. Berrone  M. Cicuttin  G. Teora  F. Vicini | M. Buffoblin  G. Calabrese  S. Silvestro |

# **Sommario**

1. INTRODUZIONE AL PROBLEMA 2

2. STRUTTURE DATI E DOCUMENTAZIONE UML 3

3. FUNZIONALITÀ ED IMPLEMENTAZIONE 6

4. TESTING 7

5. CONCLUSIONI 11

# **INTRODUZIONE AL PROBLEMA**

Il Discrete Fracture Network (DFN) è una metodologia utilizzata in geologia per modellare le fratture presenti nelle rocce.

Un DFN è costituito da un insieme di fratture rappresentate da poligoni planari che si intersecano nello spazio tridimensionale. Queste intersezioni sono chiamate tracce, segmenti di diverse lunghezze che possono essere classificate per passanti o non passanti. Una traccia passante ha entrambi gli estremi del segmento che giacciono sul bordo della frattura, mentre una traccia non-passante ha almeno un estremo all'interno della frattura stessa. Questo progetto ha l’obiettivo di identificare e classificare le tracce per ciascuna frattura.

La parte 1 del progetto consiste nel leggere un DFN da un file e calcolarne le tracce. Per ogni traccia si memorizzerà la coppia degli identificativi delle fratture che la definiscono e le coordinate tri-dimensionali dei punti che la identificano. Invece per ciascuna frattura, si differenzieranno le tracce in passanti e non-passanti, tramite un booleano [passante (false) oppure non-passante (true)]. Si ordineranno i due sottoinsiemi separatamente per lunghezza (decrescente). Il tutto verrà stampato in un file traces.txt.

La parte 2 del progetto riguarda la determinazione dei sotto-poligoni generati dal taglio delle fratture con le loro tracce. Ciascuna frattura, dato il suo sottoinsieme delle tracce, sarà inizialmente tagliata dalle tracce passanti. Per ciascuna traccia passante la frattura sarà divisa in due parti, di conseguenza l’elenco delle fratture sarà aggiornato con i nuovi sotto-poligoni generati. Una volta completato il taglio con le tracce passanti, si procederà con le tracce non-passanti, che verranno prolungate fino al bordo della frattura, generando così altri sottopoligoni.

Un’attenzione particolare andrà nell'introdurre delle condizioni che escludano la probabilità che due fratture si intersechino, evitando così il costo relativo alla risoluzione dei sistemi lineari necessari per il calcolo delle tracce.

Infine, per la validazione dei risultati, verranno utilizzati dei test, implementando GoogleTest per verificare le varie unità logiche del codice.

# **STRUTTURE DATI E DOC. UML**

**2.1 STRUTTURE DATI**

In questa sezione verranno descritte le strutture dati utilizzate nel progetto, che influenzano direttamente le prestazioni computazionali e la facilità di implementazione del programma.

Nel namespace FractureLib sono definite due struct: Fractures e Traces.

La struct Fractures è stata progettata per rappresentare un insieme di fratture all’interno del DFN. Le sue componenti sono:

* NumberFractures
  + Variabile di tipo unsigned int che tiene traccia del numero totale di fratture.
* Id
  + Variabile di tipo unsigned int che rappresenta l’identificatore univoco per ogni frattura.
* Vertices
  + Mappa che associa un id di frattura a una matrice di vertici. La matrice ha 3 righe (coordinate x, y, z) e un numero dinamico di colonne (il numero dei vertici è variabile).
* Piano
  + Mappa che associa un id di frattura ad un array di 4 elementi di tipo double, che rappresentano i coefficienti a, b, c, d dell'equazione del piano ax + by + cz + d = 0 su cui è situato il poligono.

La struct Traces è stata progettata per rappresentare le tracce generate dalle intersezioni delle fratture. Le sue componenti sono:

* NumberTraces
  + Variabile di tipo unsigned int che tiene traccia del numero totale di tracce.
* Id
  + Variabile di tipo unsigned int che rappresenta l’identificatore univoco per ogni traccia.
* Vertices
  + Mappa che associa un id di traccia a un array contenente due vettori tridimensionali (che rappresentano le coordinate dei due vertici della traccia).
* Tips
  + Mappa che associa un id di traccia a un array contenente due valori booleani (che indicano se la traccia è passante [false] o non-passante [true] per ciascuna delle due fratture che definiscono la traccia).