REPORT PROGETTO PCS 2024

Tommaso Ferraris 297720, Andrea Blotto 295621

Indice argomenti Trattati:

1- Introduzione Discrete Fracture Network (DFN)
2- Struttura utilizzata per il DFN
3- Strutture matematiche utilizzate per calcolo delle Tracce
4- Struttura utilizzata per la Mesh Poligonale
5- Strutture matematiche utilizzate per costruzione della Mesh

1- INTRODUZIONE

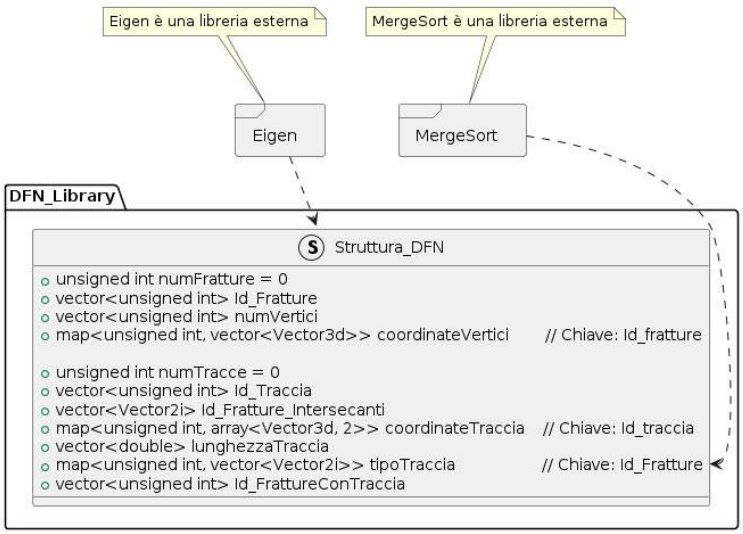
Un Discrete Fracture Network (DFN) è un sistema costituito da N fratture F_n , $n \in \{1, \dots, N\}$ rappresentate da poligoni planari che si intersecano tra di loro nello spazio tridimensionale.

Le M intersezioni tra le fratture T_m , $m \in \{1, \dots, M\}$ chiamate di seguito Tracce, possono essere identificate da un segmento, supponendo di non considerare le Tracce di misura nulla. Per ciascuna Frattura, una Traccia può essere **passante** o **non-passante**.

Una Traccia passante per una Frattura è un segmento con entrambi gli estremi che giacciono sul bordo della frattura stessa. Al contrario, una traccia non-passante per una frattura è un segmento che ha almeno un suo estremo all'interno della frattura stessa.

Il modello di soluzione proposto in seguito ha il compito di individuare la presenza di tracce individuate dall'intersezione tra le fratture, calcolarne la loro tipologia (passanti/ non-passanti) ed inserirle all'interno della struttura DFN. Successivamente individuate le Tracce di ciascuna Frattura sarà necessario costruire una mesh relativa alla specifica Frattura che permetta di individuare graficamente i risultati.

2- STRUTTURA DEL DFN



Cenni sulle dimensioni delle unità logiche della struttura:

- Id_Fratture è un vettore di dimensione (1 × NumFratture) dove l'id della n-esima Frattura corrisponde all'n-esimo elemento del vettore, la sua dimensione è variabile poiché il numero di fratture può variare.
- numVertici è un vettore di dimensione (1 × NumFratture) anch'esso dipendente dal numero di fratture che memorizziamo.
- coordinateVertici è una mappa con chiave l'id della frattura e dimensione (3 x numVertici) x numFratture, dove 3 indica che lo spazio in cui memorizziamo le coordinate è tridimensionale (x,y,z).
- Id_Traccia è un vettore di dimensione (1 × numTracce) che segue la stessa logica di Id_Fratture.
- Id_Fratture_intersecanti ha dimensione (2 x numTracce) dove 2 indica che devono essere memorizzati gli Id delle due Fratture intersecanti che generano la Traccia.
- coordinateTraccia è una mappa con chiave Id_Traccia e dimensione (3 x 2) x numTracce memorizzato come un array poiché il suo contenuto non varia in dimensione.
- tipoTraccia è una mappa con chiave id_fratture e dimensione 2 x (2 x numTracce) dove vengono memorizzati id_Traccia e la sua tipologia, di tutte le Tracce presenti sulla Frattura di un determinato Id.
- IdFratturaconTraccia è un vettore di dimensione (1 x dimensione TipoTraccia) che memorizza l'id delle sole fratture che hanno almeno una Traccia.

3- STRUTTURE MATEMATICHE PER IL CALCOLO DELLE TRACCE

a- Sottofunzioni per calcolo di Centroide:

Il **centroide** di un poligono, noto anche come baricentro, è il punto che rappresenta il centro geometrico del poligono. Per calcolare il centroide di un poligono, bisogna utilizzare le coordinate dei suoi vertici.

Nel nostro caso non è strettamente necessario calcolare un centroide perfettamente centrato, poiché ci basta avere una stima di esso, dunque possiamo utilizzarne una forma semplificata che risulta essere esatta solo per poligoni regolari.

$$(C_x = \frac{1}{n}\sum_{i=1}^{n} x_i, \qquad C_y = \frac{1}{n}\sum_{i=1}^{n} y_i, \quad C_z = \frac{1}{n}\sum_{i=1}^{n} z_i)$$

Il centroide è utilizzato per controllare se tra due fratture ci possono essere intersezioni, infatti la distanza tra due centroidi di due fratture non deve superare la somma della massima distanza tra il centroide e un vertice della frattura, a meno di una tolleranza.

b- Calcolo Tracce:

Per calcolare una traccia bisogna trovare il segmento di intersezione tra le due fratture, per fare ciò sono necessari dei passaggi matematici intermedi.

- Il primo passaggio per il calcolo di una traccia consiste nell'individuare la retta di intersezione tra i due piani che contengono le Fratture, e dalla geometria computazionale si eseguono i seguenti passaggi matematici:
  Presi tre punti di una Frattura $P_0,\ P_1,\ P_2$ si ricava l'equazione del piano contenente la Frattura:

$$x = P_0 + \alpha_0(P_2 - P_0) + \alpha_1(P_1 - P_0) \qquad n = \frac{u \times v}{||u||||v||}$$

  Da cui sapendo che $n \perp u$ e $n \perp v$ ottengo la seguente equazione:

$$n^T x = n^T P_0$$

  Ora per trovare la retta di intersezione tra due piani è necessario calcolare la direzione tangente, che identifica la direzione della retta di intersezione e un punto P sulla retta, calcolati nel seguente modo:

$$t = n_1 \times n_2 \qquad \begin{pmatrix} -n_1 - \\ -n_2 - \\ -t - \end{pmatrix} \begin{pmatrix} P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ 0 \end{pmatrix}$$

  Aggiungendo inoltre la condizione per la quale se $Det\ (A) = 0$ non c'è intersezione.

- Avendo trovato la retta di intersezione dei due piani bisogna restringere il campo per ottenere Traccia generata dall'intersezione tra due fratture. Per fare ciò è necessario calcolare gli eventuali punti di intersezione tra la retta e i lati delle Fratture, ottenendo così dei possibili punti di intersezione.

$$\begin{cases} P = P_0 + \alpha(P_1 - P_0) \\ P = P_2 - \beta t \end{cases} \qquad \text{Da cui ottengo: } \begin{pmatrix} x_1^1 - x_1^0 & t_1 \\ x_2^1 - x_2^0 & t_2 \\ x_3^1 - x_3^0 & t_3 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} x_1^2 - x_1^0 \\ x_2^2 - x_2^0 \\ x_3^2 - x_3^0 \end{pmatrix}$$

  Risolvendo tale sistema lineare il punto di possibile intersezione trovato deve passare attraverso due controlli:
  1. È necessario limitare il lato del poligono, limitando il valore di $\alpha \in [0,1]$.
  2. Per il secondo controllo si definisce l'equazione del punto di intersezione
     $$I = P_0 + \alpha(P_1 - P_0), \qquad \alpha \in [0,1]$$
     E si verifica che il punto trovato risulti interno all'altra frattura intersecante, attraverso un metodo di triangolazione della Frattura.

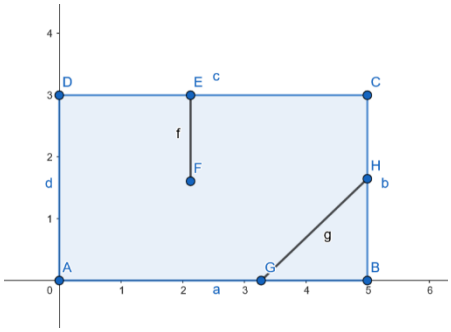
Superati entrambi i controlli possiamo considerare il punto I un effettivo punto della Traccia generata dall'intersezione di due Fratture.

Procedendo con i precedenti calcoli matematici per entrambe le fratture intersecanti si ottengono le Tracce relative a ciascuna Frattura.

c- Calcolo della tipologia delle Tracce:

Una Traccia di una Frattura può essere di due diverse tipologie:

1. **Passante**: Per la quale gli estremi della traccia sono entrambi appartenenti ai lati della Frattura.
2. **Non-passante**: Che identifica le Tracce di tutte le altre tipologie, ovvero con uno o nessuno degli estremi appartenenti ad un lato della Frattura.



Nella figura è possibile visualizzare un esempio di traccia passante (Traccia g) e non passante (Traccia f).

Dal punto di vista matematico calcolare la tipologia di una traccia è un conto relativamente veloce, è necessario solamente verificare la collinearità degli estremi della traccia con le coppie di vertici della Frattura e verificare se entrambi gli estremi appartengono a qualche lato della Frattura, controllando quindi che i valori delle coordinate siano sempre compresi tra quelli del segmento.

VERIFICA DELLA COLLINEARITÀ:

Per verificare che tre vertici siano collineari è necessario verificare che il loro prodotto vettoriale sia nullo:

Dati due vertici P_1, P_2 della Frattura e un estremo della Traccia P_3 la **condizione di collinearità** è:

$$(P_2 - P_1) \times (P_3 - P_1) = 0$$

Se la condizione è soddisfatta per entrambi gli estremi della Traccia, e il punto è all'interno del segmento, la sua tipologia è passante altrimenti non-passante.

d- Algoritmo di ordinamento MergeSort:

L'algoritmo di ordinamento utilizzato per ordinare in senso decrescente le Tracce è **MergeSort**, il quale è un algoritmo di ordinamento efficiente basato sulla tecnica divide et impera.

Le sue caratteristiche principali sono:

1- **Divide et Impera**: Il MergeSort divide ricorsivamente il vettore da ordinare in due metà finché ogni sottosequenza non è lunga uno o zero elementi. Successivamente, le sottosequenze vengono riunite in ordine nel nostro caso decrescente.
2- **Complessità Temporale**: Ha una complessità temporale di $\boldsymbol{O(nlog(n))}$ sia nel caso medio che nel caso peggiore, il che lo rende più efficiente di algoritmi come il bubblesort o l'insertion sort, che hanno complessità $O(n^2)$.
3- **Performance Costante**: La sua performance non dipende dalla distribuzione dei dati in ingresso rendendolo una scelta robusta per un'ampia varietà di scenari.

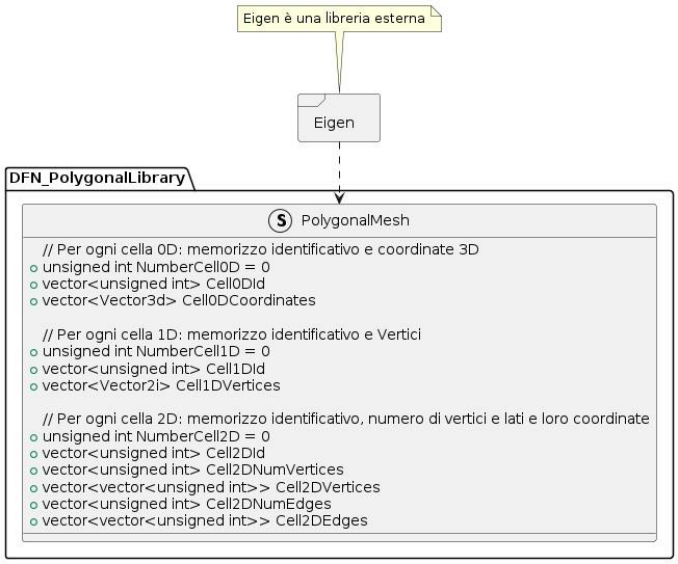
4- STRUTTURA DELLA MESH POLIGONALE

Una **mesh poligonale** è una rappresentazione geometrica tridimensionale di un oggetto utilizzata principalmente in computer grafica, modellazione 3D e simulazioni. Essa è costituita da una rete di poligoni, caratterizzati da forme elementari come triangoli, quadrilateri o poligoni.

Caratteristiche principali:

- **Celle 0D**: Punti nello spazio tridimensionale che definiscono i vertici dei poligoni.
- **Celle 1D**: Segmenti che collegano due vertici consecutivi, formando i lati dei poligoni.
- **Celle 2D**: Superfici piane delimitate da un insieme di spigoli, che caratterizzano il poligono stesso.

UML della struttura:



Cenni sulle dimensioni delle unità logiche della struttura:

- Cell0DId è un vettore di dimensione (1 x NumberCell0D).
- Cell0DCoordinates ha dimensione (3 x NumberCell0D) dove 3 rappresenta le coordinate spaziali (x, y, z).
- Cell1DId è un vettore di dimensione (1 x NumberCell1D).
- Cell1DVertices ha dimensione (2 x NumberCell1D) dove 2 indica che i vertici che generano un lato sono sempre e solo due.
- Cell2DId è un vettore di dimensione (1 x NumberCell2D).
- Cell2DVertices ha dimensione (Cell2DNumVertices x NumberCell2D)
- Cell2DEdges ha dimensione (Cell2DNumeDGES x NumberCell2D)

5- STRUTTURE MATEMATICHE PER COSTRUZIONE DELLA MESH

a- Suddivisione delle Fratture per ricavare le Mesh

Il meccanismo di suddivisione della Frattura in sotto-poligoni generati dalle sue Tracce viene descritto da seguente algoritmo:

È necessario innanzitutto controllare che la Traccia sia appartenente anche solo in parte ad un poligono.

- **Traccia che non appartiene al poligono**: se non c'è appartenenza al poligono esso non verrà diviso da quella traccia e si passerà a controllare il poligono successivo
- **Traccia appartenente al poligono**: se la Traccia appartiene interamente o in parte al poligono quest'ultimo verrà suddiviso in due sotto-poligoni, seguendo il seguente algoritmo:

Vengono individuati i due punti di intersezione della traccia, eventualmente allungata, con i lati del poligono che stiamo considerando e la suddivisione di quest'ultimo segue il seguente schema:

Siano P_1, P_2 i punti di intersezione della traccia (eventualmente allungata) con i lati della frattura e considerando la loro posizione nel poligono insieme alla posizione dei suoi vertici è possibile dividere il poligono in due seguendo un senso di orientamento antiorario:

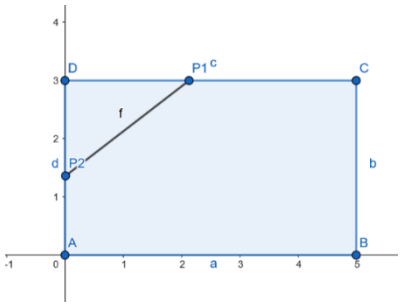
- Poligono A: Il poligono A sarà formato dai vertici

$$P_2, P_2(posizione) + 1, vertici\ poligono\ da\ P_2(posizione)\ a\ P_1(posizione), P_1$$

- Poligono B: Il poligono B sarà formato dai vertici

$$P_1, P_1(posizione) + 1, vertici\ poligono\ da\ P_1(posizione)\ a\ P_2(posizione), P_2$$

Seguendo l'algoritmo si eliminano tutti i casi particolari poiché il poligono viene diviso sempre e solo quando una parte della traccia risulta essergli interna.



Ripetendo l'algoritmo per tutte le Tracce della frattura seguendo l'ordine decrescente prestabilito si ottiene la Frattura divisa da tutte le sue Tracce, ed è dunque possibile costruire la Mesh Poligonale.

b- Intersezione tra Traccia e lato del poligono

Dal punto di vista matematico l'intersezione tra la Traccia e il lato del poligono per vedere l'esistenza o meno di un punto di intersezione si ottiene risolvendo il seguente sistema lineare:

Considerando il sistema: $\begin{cases} P = V_1 + \alpha\,(V_2 - V_1) \\ P = origin + \beta\,(end - origin) \end{cases}$

dove V_1, V_2 sono i vertici che definiscono il lato del poligono e $(origin, end)$ sono gli estremi della Traccia.

Risolvo il sistema:

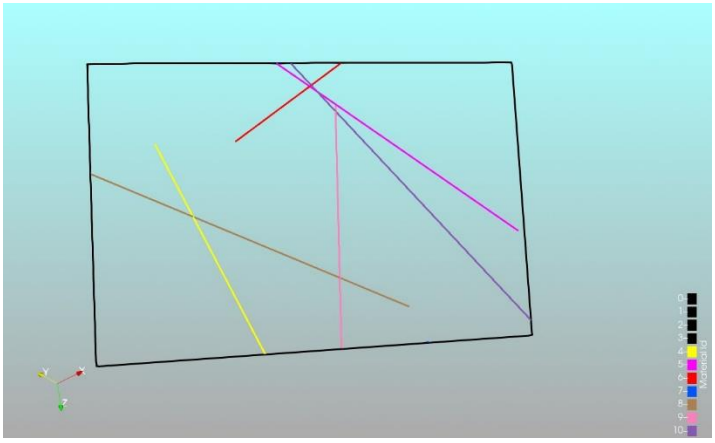
$$\begin{pmatrix} x_1^2 - x_1^1 & end_1 - origin_1 \\ x_2^2 - x_2^1 & end_2 - origin_2 \\ x_3^2 - x_3^1 & end_3 - origin_3 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} origin_1 - x_1^1 \\ origin_2 - x_2^1 \\ origin_3 - x_3^1 \end{pmatrix}$$

Per controllare se il punto di intersezione trovato è originato da una traccia interna al poligono, sottoponiamo il vettore $\binom{\alpha}{\beta}$ ai seguenti controlli, tramite i quali, se si verifica uno dei primi due casi la traccia è effettivamente interna, con il terzo caso trovo eventuali punti allungando la traccia:

- Origine e fine della traccia sono entrambi interni al poligono, rappresenta il caso in cui la traccia è interna al poligono senza avere intersezioni con i suoi lati.
- $\alpha, |\beta| \in [0,1]$, rappresenta il caso in cui la Traccia interseca effettivamente il lato del poligono.
- $\alpha \in [0,1], \beta \in \mathbb{R}$, rappresenta il caso in cui la Traccia interseca il lato del poligono solo quando essa viene allungata.

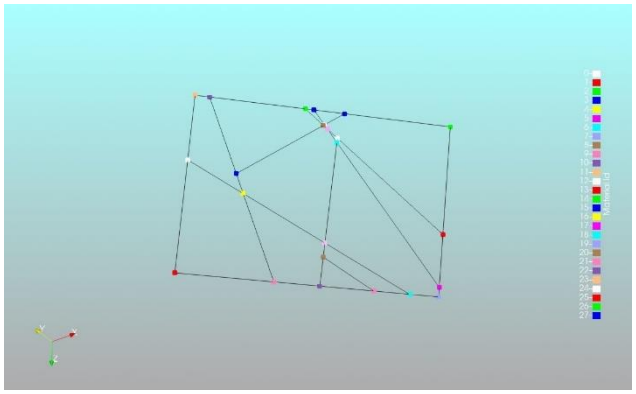
c- **Immagini di alcune Mesh con Paraview:**

Dal file FR10, presa la Frattura 0 con le sue Tracce:

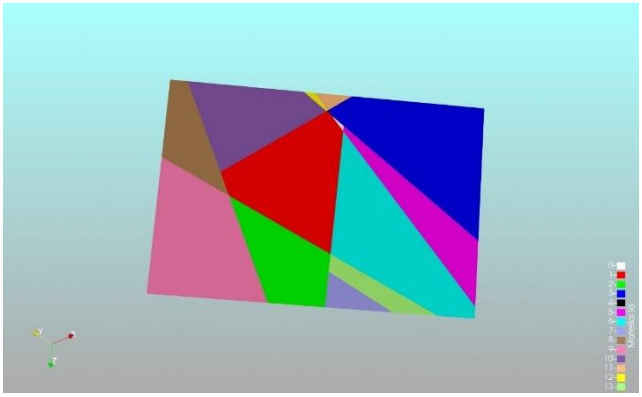


Si genera la seguente Mesh:

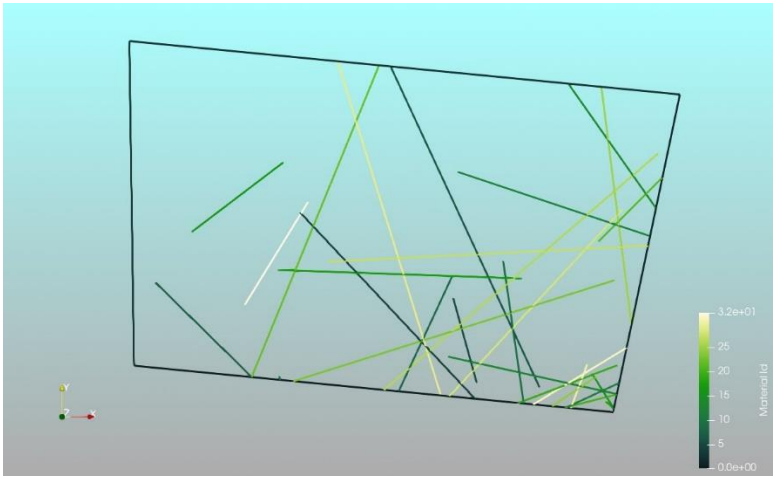
Celle 0D e 1D:



Celle 2D:

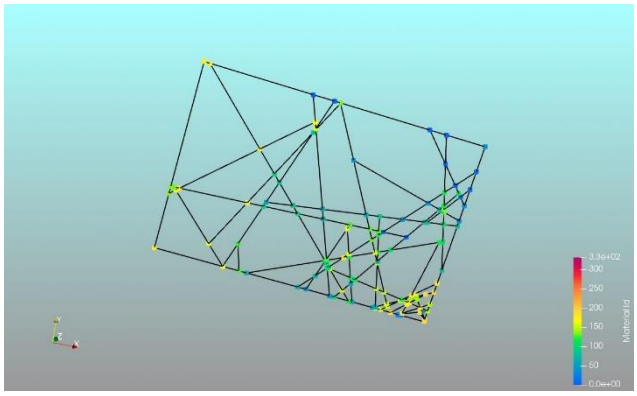


Dal file FR10, presa la Frattura 0 con le sue Tracce:



Si genera la seguente Mesh:

Celle 0D e 1D:



Celle 2D: