Puha Márk(BZXRI1) Gráf Átmérő Kiszámítása Beadandó feladat 2017-04-23

1. Gráf Ábrázolása

A gráfot az éllistás ábrázoláshoz hasonlóan ábrázoltam, azt leszámítva, hogy az STL vector típusát használtam a szomszédsági csúcsok megkonstruálásához. Létrehoztam egy *GraphVertex* nevezetű osztályt amit a csúcsok ábrázolására használtam itt található a szomszédsági csúcsok *vectora*. Ilyen *GraphVertex* típusú elemekből található egy *vector* a *GraphDiameter* nevezetű osztályban ahol az átmérő kiszámítását végzem.

2. Feladat Megoldásának Ismertetése

Feladat megoldása a *GraphDiameter* nevű osztály implementációjában található. A megoldáshoz a *szélességi bejárást* használom két alkalommal, első alkalommal, hogy megtaláljam a kezdőcsúcstól legtávolabbi csúcsot, majd erre a csúcsra alkalmazom még egyszer az említett algoritmust miközben a *maxDistanceInd* változóban keresem a kezdőcsúcstól a legtávolabbi csúcs indexét így a a végén visszatérek a *maxDistanceInd csúcsához tartozó távolsággal ami a* gráf átmérője. A kezdőcsúcsot véletlenszerűen határozom meg. A *szélességi bejáráshoz* szükséges *Sor* típus implementációja egy külön header fileban található meg.

3. Implementáció Ismertetése

GraphVertex*:*

- distance : a kezdőcsúcstól vett távolság
- Color : adott csúcs színe(felsoroló típussal ábrázolva)
- adjacentVec : adott csúcs szomszédjainak a vectora

GrapDiameter*:*

Változók:

- *maxDistanceInd : az átmérő hossza*
- *graphQueue : széllességi bejárásnál használt sor típus*
- *graphVertexVec : a csúcsok vectora*

Függvények:

- *operator>> : GraphDiameter* típusú elemek beolvasására alkalmas operátor
- *calculateDiameter :* a program fő tevékenysége az átmérő kiszámítása
- *initStartVertexAndQueue :* adatok inicializálása
- *calculateDistanceWithBfsAndDecideDiameter :* sor első csúcsa alapján kiszámolja a legtávolabbi csúcs indexét amit elment a maxDistanceInd nevű változóba
- *decideMaxDistanceInd :* a legtávolabbi csúcs indexének meghatározása
- *reInitVertexesAndQueue : Bfs* újra alkalmazásához újra inicializáljuk a csúcsok távolságát és a színűket
- *incrementActAdjDistanceAndPushItToQueue :* aktuális csúcs szomszédjainak a távolságának növelése és a szomszédok eltárolása a sorban
- *createRandNumber :* egy véletlenszerű szám kreálása

4.Műveletigény

Az előadáson bizonyítottuk, hogy a *BFS* műveletigénye O(n + e) ahol n = |V| és e = |E|, ezt kétszer alkalmazva a konstans szoros nem lesz hatással a műveletigényre, illetve szint úgy nem lesz hatással a csúcsok újra inicializálása és az egyéb műveletek a program futási idejére.