Αναφορά 1ης Εργαστηριακής Άσκησης

Κατά την εκτέλεση του κώδικα, πρέπει να εισαχθεί ως είσοδο μία επιλογή από τις παρακάτω καθώς και ο αριθμός των κόμβων που θα περιέχει το γράφημα εισόδου.

- grid
- random
- synthetic

Για τις δεντρικές συνεκτικές συνιστώσες, χρησιμοποιήθηκε η κλάση TreeCohesiveComponent. Κάθε στιγμιότυπο της κλάσης έχει τα παρακάτω attributes:

- Μία λίστα με τους κόμβους που περιέχει η συγκεκριμένη δεντρική συνιστώσα
- Έναν δείκτη στο πρώτο στοιχείο της λίστας κόμβων
- Έναν δείκτη στο τελευταίο στοιχείο της λίστας κόμβων
- Το μήκος της λίστας κόμβων

Κατά την εκτέλεση του αλγορίθμου Kruscal, δημιουργείται μία λίστα με στιγμιότυπα της κλάσης TreeCohesiveComponent, ένα στιγμιότυπο ανά κόμβο του γραφήματος. Όταν απαιτείται η ένωση 2 δεντρικών συνεκτικών συνιστωσών, δημιουργείται ένα νέο στιγμιότυπο της κλάσης TreeCohesiveComponent που περιέχει την ένωση των δύο στιγμιότυπων, εισάγεται στην αρχική λίστα και διαγράφονται τα στιγμιότυπα που ενώθηκαν.

Για τα γραφήματα χρησιμοποιήθηκε η template κλάση UGRAPH<vtype, etype>. Για την συγκεκριμένη υλοποίηση, χρησιμοποιήθηκε το UGRAPH<int, int>, ώστε κάθε ακμή να έχει ακέραιο κόστος. Η ακέραια τιμή που έχει, ο κάθε κόμβος, ως βάρος δεν χρησιμοποιήθηκε στην συγκεκριμένη υλοποίηση.

Για την υλοποίηση του αλγορίθμου ελέγχου, έχουν χρησιμοποιηθεί τα dynamic trees. Για τον έλεγχο των ακμών που περιέχονται στον κύκλο, που δημιουργείται με την είσοδο μίας μη-δεντρικής ακμής, εντοπίζεται ο ελάχιστος κοινός γείτονας των κόμβων της μη-δεντρική ακμής. Έπειτα, για κάθε ένα από τα δύο μονοπάτια, μεταξύ των κόμβων και του ελάχιστου κοινού γείτονα, ελέγχονται διαδοχικά οι ακμές των δύο μονοπατιών με βάση την ανισότητα για τα κόστη των ακμών, με βάση την ιδιότητα του κύκλου.

Οι πέρασμα των ορισμάτων στις συναρτήσεις γίνεται σε μεγάλο βαθμό, με πέρασμα με αναφορά. Η συνάρτηση EdgeCostCompare, χρησιμοποιείται για να ταξινομηθεί η λίστα με τις ακμές, κατά αύξουσα σειρά κόστους.

Προκειμένου να χρησιμοποιηθεί ως είσοδο ένα τυχαίο συνεκτικό γράφημα, δημιουργείται ένα κενό στιγμιότυπο της κλάσης ugraph, το οποίο αρχικοποιείται με την συνάρτηση random_simple_undirected_graph και μετατρέπεται σε συνεκτικό μη προσανατολισμένο γράφημα, με την κλήση της συνάρτησης Make_Biconnected(). Έπειτα, το ugraph αυτό αντιγράφεται, στην template class UGRAPH<int, int>, λόγω αδυναμίας κλήσης της συνάρτησης random_simple_undirected_graph με template παράμετρο. Ώστε να επιτευχθεί η πολυπλοκότητα χειρότερης περίπτωσης του αλγορίθμου Kruscal, τα κόστη των ακμών W(I, J), των συνθετικών γραφημάτων έχουν ορισθεί ως J – 1.

Για την παραγωγή των τυχαίων τιμών, για το κόστος των ακμών στην περίπτωση των grid και random biconnected γραφημάτων, έχει χρησιμοποιηθεί η συνάρτηση rand με seed από την συνάρτηση time που περιέχονται στην iostream, time της cpp.

Αναφορά 1ης Εργαστηριακής Άσκησης

Για την δοκιμή του αλγορίθμου, έχουν χρησιμοποιηθεί τα παρακάτω γραφήματα.

Τυχαία Συνεκτικά Γραφήματα:

Χρόνοι Εκτέλεσης / Γράφημα Εισόδου	User Defined Kruscal	MIN_SPANNING_TREE Leda
random_simple_undirected_graph(undirectedGraph, 4000, 43.864)	8.08 sec	0.0 sec
random_simple_undirected_graph (undirectedGraph, 8000, 207.453)	201.98 sec	0.0 sec
random_simple_undirected_graph (undirectedGraph, 16000, 446.906)	2382.65 sec	0.01 sec

Γραφήματα Τύπου Πλέγματος:

Χρόνοι Εκτέλεσης / Γράφημα Εισόδου	User Defined Kruscal	MIN_SPANNING_TREE Leda
grid_graph(undirectedGraph, 200)	348.67 sec	0.07 sec
grid_graph(undirectedGraph, 300)	2149.0 sec	0.20 sec
grid_graph(undirectedGraph, 400)	7284.37 sec	0.21 sec

Συνθετικά Γραφήματα:

Χρόνοι Εκτέλεσης / Γράφημα Εισόδου	User Defined Kruscal	MIN_SPANNING_TREE Leda
complete_ugraph(undirectedGraph, 2000)	529.55 sec	0.98 sec
complete_ugraph(undirectedGraph, 3000)	1575.5 sec	2.28 sec
complete_ugraph(undirectedGraph, 4000)	3831.76 sec	3.04 sec

Για τα συνθετικά γραφήματα, χρησιμοποιήθηκαν οι παρακάτω αρχικοποιήσεις και για το κόστος της κάθε ακμής w(I, J), χρησιμοποιήθηκε η τιμή J – 1.