ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΑΛΓΟΡΙΘΜΟΙ & ΠΟΛΥΠΛΟΚΟΤΗΤΑ

(2020-2021)

3η Σειρά Γραπτών Ασκήσεων

Ονοματεπώνυμο:

- Χρήστος Τσούφης

Αριθμός Μητρώου:

- 03117176

Στοιχεία Επικοινωνίας:

- el17176@mail.ntua.gr

1η Άσκηση – Ραντεβού μετά το Lockdown

Εκφώνηση & Επεξήγηση:

Το οδικό δίκτυο αναπαρίσταται ως ένα κατευθυνόμενο γράφημα G(V, E), με n κορυφές και m ακμές. Η κατεύθυνση κάθε ακμής e = (u, v) ∈ E στο G αντιστοιχεί στην κατεύθυνση της κυκλοφορίας στο οδικό τμήμα u – v κατά τις περιττές χρονικές στιγμές 1, 3, 5, … . Για κάθε ακμή e = (u, v) ∈ E η κυκλοφορία στο οδικό τμήμα u – v κατά τις άρτιες χρονικές στιγμές 2, 4, 6, … γίνεται στην αντίθετη κατεύθυνση (v, u). Ζητείται να βρεθεί ποια είναι η τελευταία στιγμή που αν ξεκινήσει κανείς από την κορυφή s ∈ V μπορεί να φτάσει στην κορυφή t ∈ V νωρίτερα από τη χρονική στιγμή T.

Ο Αλγόριθμος είναι ο εξής:

Εν ολίγοις, απαιτείται μια διάσχιση του γράφου αρχίζοντας από τις ακμές του γράφου που αντιστοιχούν στις περιττές χρονικές στιγμές και άλλη μια διάσχιση του γράφου αρχίζοντας αυτή τη φορά από τις ακμές που αντιστοιχούν στις άρτιες χρονικές στιγμές. Μετά από αυτές τις δύο διασχίσεις, διατηρείται η καλύτερη, δηλαδή εκείνη με την κατά απόλυτη τιμή μεγαλύτερη χρονική στιγμή έναρξης. Ειδικότερα, για την διάσχιση του γράφου, χρησιμοποιείται ο αλγόριθμος DFS από την κορυφή s ∈ V όπου ξεκινάει κανείς και ο σκοπός είναι η εύρεση όλων των κορυφών όπου είναι εφικτή η μετάβαση την εκάστοτε χρονική στιγμή. Κάθε φορά, οι κορυφές που έχει ήδη επισκεφθεί ο αλγόριθμος, συνενώνονται σε μια ενιαία κορυφή έως ότου φτάσει στην κορυφή t ∈ V όπου και ολοκληρώνεται ο αλγόριθμος. Αντιθέτως, στην περίπτωση όπου γίνεται μετάβαση στην επόμενη χρονική στιγμή, οι ακμές του γράφου δείχνουν προς την αντίθετη κατεύθυνση οπότε η ενιαία κορυφή θεωρείται πλέον η καινούρια s και από εκεί συνεχίζει η διάσχιση.

Με τα παραπάνω, από τον συνολικό αριθμό επαναλήψεων που πραγματοποιεί, υπολογίζεται ο αριθμός των χρονικών στιγμών που απαιτήθηκαν για την μετάβαση από την κορυφή s στην κορυφή t ενώ παράλληλα διατηρείται και το κατά πόσο η χρονική στιγμή της αύξησης είναι άρτια ή περιττή καθώς αυτή αποτελεί το μέτρο σύγκρισης με την δοσμένη χρονική στιγμή T. Έτσι, αν και οι δύο είναι άρτιες, τότε η κατά απόλυτη τιμή χρονική στιγμή έναρξης είναι ίση με T μειωμένη κατά τόσο όσες είναι και οι επαναλήψεις και η κατά απόλυτη τιμή χρονική στιγμή άφιξης είναι ίση με T. Όμως, αν δεν είναι και οι δύο άρτιες, τότε η κατά απόλυτη τιμή χρονική στιγμή έναρξης είναι ίση με T μειωμένη κατά τόσο όσες είναι και οι επαναλήψεις και μια επιπλέον χρονική στιγμή και η κατά απόλυτη τιμή χρονική στιγμή άφιξης είναι ίση με T μειωμένη κατά μια επιπλέον χρονική στιγμή. Ομοίως και για τις περιττές.

Έλεγχος Ορθότητας:

Ο αλγόριθμος διασχίζει τον γράφο από όλες τις προσβάσιμες κορυφές και για αυτό απαιτούνται οι χρονικές στιγμής ώστε να προχωρήσει στις επόμενες. Μάλιστα, ανά ορισμένες χρονικές στιγμές, ο γράφος αλλάζει κατεύθυνση και αυτό έχει ως αποτέλεσμα ότι πάντοτε θα υπάρχει δυνατότητα να προχωρήσει την διάσχιση ο αλγόριθμος.

Πολυπλοκότητα:

Η πολυπλοκότητα του αλγορίθμου θα είναι O(2·(n + m)). Αυτό προκύπτει από το γεγονός ότι απαιτούνται δύο διασχίσεις του γράφου.

2η Άσκηση – Προγραμματίζοντας την Αντίδραση

Εκφώνηση & Επεξήγηση:

Πέντε επιστήμονες ανακάλυψαν ένα νέο παράξενο τύπο σωματιδίου. Εικάζεται ότι αν k σωματίδια βρεθούν στο ίδιο ακριβώς σημείο, την ίδια χρονική στιγμή, θα υπάρξει μια ιδιαίτερη αντίδραση. Τα σωματίδια κινούνται πάνω στις κορυφές ενός μη κατευθυνόμενου γραφήματος G(V, E) με n κορυφές και m ακμές. Για κάθε σωματίδιο q_i, εάν αυτό βρίσκεται στην κορυφή $v \in V$ την ημέρα t, τότε την επόμενη μέρα t + 1, αυτό μπορεί είτε να παραμείνει στην κορυφή v είτε να μετακινηθεί σε μια γειτονική κορυφή $u \in V$ της v (όπου η ακμή $\{v, u\} \in E$). Μπορούν να προγραμματιστούν με ακρίβεια κάθε σωματιδίου σε βάθος χρόνου αλλά δεν μπορούν να ελεγχθούν οι αρχικές θέσεις των σωματιδίων στο G καθώς και το πότε τα σωματίδια θα μετακινηθούν. Δεδομένου λοιπόν του γραφήματος G(V, E), των αρχικών κορυφών $v_1, \dots, v_k \in V$ των k σωματιδίων στο G, και του γεγονότος ότι το πλήθος n των κορυφών του G είναι σημαντικά μεγαλύτερο από το πλήθος k των σωματιδίων, ζητείται να βρεθεί τρόπος προγραμματισμού των κινήσεων των k σωματιδίων έτσι ώστε η αντίδραση να συμβεί το συντομότερο δυνατόν καθώς επίσης και η κορυφή του γραφήματος G όπου θα συμβεί αυτό.

Ο Αλγόριθμος είναι ο εξής:

Από κάθε κορυφή του γράφου G εκτελείται ο BFS αλγόριθμος και διατηρείται η τιμή k που αποτελεί το ύψος του σχηματιζόμενου δέντρου. Από τα προηγούμενα, ως τελικό αποτέλεσμα διατηρείται η κορυφή του πιο "κοντού" δέντρου καθώς επίσης και η τιμή k του ύψους του. Αυτές οι τιμές αποτελούν το μέγιστο δυνατό πλήθος βημάτων που πρέπει να ακολουθηθούν ώστε η αντίδραση να συμβεί το συντομότερο δυνατόν.

Έλεγχος Ορθότητας:

Εάν θεωρηθεί ως σημείο συγκέντρωσης των σωματιδίων η κορυφή $v \in V$, τότε για τα σωματίδια που εντοπίζονται στις γειτονικές κορυφές απαιτείται ένα μόνο βήμα για την άφιξη στο σημείο συγκέντρωσης. Αντίστοιχα, για σωματίδια που εντοπίζονται στις γειτονικές κορυφές των προαναφερθέντων, απαιτούνται δύο βήματα για την άφιξη στο σημείο συγκέντρωσης. Την ίδια λογική ακολουθούν και οι υπόλοιπες κορυφές. Επομένως, ο υπολογισμός της ελάχιστης απόστασης της κορυφής v από τις υπόλοιπες κορυφές του γράφου G γίνεται με τον αλγόριθμο BFS.

Πολυπλοκότητα:

Η πολυπλοκότητα του αλγορίθμου θα είναι $O(n \cdot (n + m))$.

3η Άσκηση – Ένας Παράξενος Περίπατος

Εκφώνηση & Επεξήγηση:

Δίνεται ένα κατευθυνόμενο γράφημα G(V, E, c) με θετικά ακέραια βάρη c στις κορυφές του, όπου c(v) είναι το βάθος μιας κορυφής v ∈ V. Ορίζεται ως συνάρτηση κόστους ενός περιπάτου στο G ο μέγιστος κοινός διαιρέτης του βάρους των κορυφών που ανήκουν στον περίπατο. Για παράδειγμα, το κόστος ενός περιπάτου $w = (v_1, v_2, \dots, v_k)$ είναι ίσο με $c(w) = \gcd(c(v_1), c(v_2), \dots, c(v_k))$. Ζητείται ο υπολογισμός του ελάχιστου κόστους ενός περιπάτου στο γράφημα G(V, E, c).

Ο Αλγόριθμος είναι ο εξής:

Με βάση την παραπάνω εκφώνηση, συμπεραίνεται ότι η προσθήκη ενός κόμβου συμβάλει θετικά στην εύρεση του ελάχιστου κόστους ενός περιπάτου στο γράφο G. Εφόσον το κόστος ενός περιπάτου w είναι ίσο με c(w), η προσθήκη του επιπλέον κόμβου u δημιουργεί περίπατο w' με νέο κόστος c(w'). Όμως, αφού gcd(c(w), c(u)) ≤ c(w) τότε και c(w) ≥ c(w'). Έτσι, η βέλτιστη τιμή για κάθε κόμβο που ανήκει σε κάποια Ισχυρά Συνεκτική Συνιστώσα (ΙΣΣ) είναι ίση με την τιμή που υπολογίζεται από τον περίπατο που αρχίζει από τον κόμβο, διασχίζει όλους του άλλους της ΙΣΣ και επιστρέφει στον ίδιο.

Επομένως, γίνεται αρχικά ο υπολογισμός των ΙΣΣ του γράφου G με τον αλγόριθμο Kosaraju και προκύπτει το κόστος του κυκλικού μονοπατιού, το οποίο αποτελεί και την βέλτιστη τιμή για κάθε κόμβο που ανήκει σε αυτές. Ύστερα, προκύπτει νέος γράφος (DAG) όπου έχει ως κορυφές τις ΙΣΣ και ως τιμές τα κόστη που υπολογίστηκαν προηγουμένως. Σε αυτόν εκτελείται τοπολογική ταξινόμηση με DFS και οι κόμβοι αποθηκεύονται σε έναν πρώτο πίνακα. Τέλος, δημιουργείται και ένας δεύτερος πίνακας με τις ίδιες τιμές με τον προηγούμενο. Ο DAG διασχίζεται με τον DFS αλγόριθμο αρχίζοντας από τον n κόμβο και αποθηκεύεται κάθε φορά ο αριθμός που προκύπτει από τη πράξη gcd(τιμή του κόμβου, τιμή που υπάρχει στον δεύτερο πίνακα). Μάλιστα, αν η ελάχιστη τιμή του δεύτερου πίνακα είναι μικρότερη από την αρχική τιμή του πρώτου κόμβου που εξετάστηκε, διατηρείται εκείνη. Συνεπώς, με την εύρεση όλων των βέλτιστων τιμών όλων των δυνατών περιπάτων και έπειτα της μικρότερης τιμής εξ αυτών προκύπτει το ελάχιστο κόστος που αναζητείται.

Έλεγχος Ορθότητας:

Η ορθότητα των αλγορίθμων DFS και Kosaraju είναι γνωστή. Επισημαίνεται ξανά ότι το βέλτιστο κόστος για κάθε κορυφή που ανήκει σε ΙΣΣ ισούται με το κυκλικό μονοπάτι που διασχίζει κάθε κορυφή που ανήκει στην ΙΣΣ. Επίσης, από κάθε κόμβο του DAG εκτελείται DFS για τον υπολογισμό του βέλτιστου περιπάτου προς κάθε κόμβο περνώντας από όλα τα δυνατά μονοπάτια.

Πολυπλοκότητα:

Η πολυπλοκότητα του αλγορίθμου θα είναι $O(n \cdot (n + m))$. Αυτό προκύπτει από το γεγονός ότι ο υπολογισμός όλων των δυνατών περιπάτων γίνεται με DFS για κάθε κόμβο του DAG. Συγκεκριμένα, απαιτούνται δύο DFS για τον αλγόριθμο του Kosaraju με κόστος O(n + m). Επιπλέον, ο υπολογισμός του κόστους για κάθε ΙΣΣ εκτελείται επίσης με DFS του γράφου με κόστος O(n + m), με επίσκεψη μια φορά σε κάθε ακμή & κόμβο. Η τοπολογική ταξινόμηση γίνεται και αυτή με DFS οπότε το κόστος και εδώ θα είναι O(n + m).

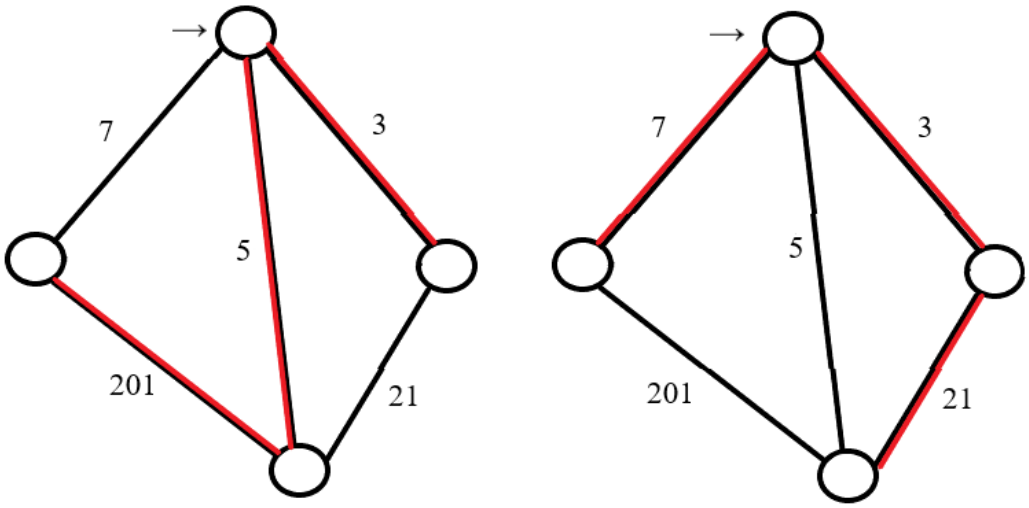
4η Άσκηση – Ελάχιστο Συνδετικό Δέντρο με Περιορισμούς

Εκφώνηση & Επεξήγηση:

Έστω ένα συνεκτικό μη κατευθυνόμενο γράφημα G(V, E, w), με n κορυφές, m ακμές και θετικά βάρη w στις ακμές. Στόχος είναι ο υπολογισμός ενός Ελάχιστου Συνδετικού Δέντρου T*(s, k) του G στο οποίο μια συγκεκριμένη κορυφή s έχει βαθμό ίσο με k (όπου $k \geq 1$ και το k δεν ξεπερνά τον βαθμό της s στο G, οπότε το G έχει ένα τέτοιο συνδετικό δέντρο).

a) Παρακάτω αποδεικνύεται ότι η άπληστη στρατηγική, η οποία συμπεριλαμβάνει στο συνδετικό δέντρο τις k ακμές μικρότερου βάρους που προσπίπτουν στην s, δεν οδηγεί πάντα στη βέλτιστη λύση.

   Απόδειξη:
   Ο ισχυρισμός αυτός αποδεικνύεται με το παρακάτω εποπτικό παράδειγμα.



   Σχόλιο: Στο γράφημα αυτό ο απαιτούμενος βαθμός στον κόμβο που δείχνει το βέλος είναι 5. Με το άπληστο κριτήριο, προκύπτει η αριστερή περίπτωση, με συνολικό βάρος ίσο με 3 + 5 + 201 = 209. Όμως, το ελάχιστο έχει συνολικό βάρος ίσο με 3 + 7 + 21 = 31.

b) Ο υπολογισμός ενός Ελάχιστου Συνδετικού Δέντρου T*(s, k) στο οποίο η κορυφή s έχει βαθμό ίσο με k διατυπώνεται παρακάτω.

   Εν ολίγοις, σε κάθε ακμή που συναντά τον σχετικό κόμβο προστίθεται μια σταθερά m. Ισχύει ότι όσο μεγαλύτερο είναι το m, τόσο λιγότερες ακμές περιέχει το ΕΣΔ του επαγόμενου γραφήματος που συναντούν τον σχετικό κόμβο. Έτσι, εφαρμόζοντας δυαδική αναζήτηση (Binary Search), υπολογίζεται η κατάλληλη τιμή του m.

   Για τα παραπάνω χρησιμοποιούνται οι εξής ισχυρισμοί (με τις αποδείξεις τους):

Ισχυρισμός 1:
Το απαιτούμενο ΕΣΔ με περιορισμούς υλοποιείται ως το ΕΣΔ του δέντρου αυξημένου κατά m, για κάθε απαιτούμενο βαθμό.

Απόδειξη:
Από την μια, για αρκετά αρνητικό m το ΕΣΔ παίρνει όλες τις σχετικές ακμές, και από την άλλη για αρκετά θετικό παίρνει τις ελάχιστες δυνατές. Συνεπώς επιτυγχάνονται όλες οι τιμές ανάμεσα στην ελάχιστη και μέγιστη δυνατή.

Ισχυρισμός 2:
Ισχύει ότι τα μοναδικά m στα οποία μεταβάλλεται ο βαθμός του απαιτούμενου κόμβου στο ΕΣΔ είναι εκείνα στην περίπτωση που μια ακμή που συναντά τον κόμβο γίνεται ίση με μια άλλη ακμή του γράφου όταν αυξηθεί κατά m. Δηλαδή, αρκεί να εξεταστούν τα m για τα οποία ισχύει ότι το $m \in \{e - a : a \in A \wedge e \in E\}$ (όπου A είναι το σύνολο των διαφορετικών βαρών ακμών που συναντούν τον απαιτούμενο κόμβο & E είναι το σύνολο των διαφορετικών βαρών ακμών όλου του γραφήματος).

Απόδειξη:
Εφόσον για να μπορέσει να αφαιρεθεί μια ακμή από το ΕΣΔ πρέπει να αντικατασταθεί από μια άλλη, τότε για κάποιο m γίνεται ίση μια προσπίπτουσα στον απαιτούμενο κόμβο με κάποια άλλη του γράφου. Αυτό υφίσταται μόνο όταν το m είναι ίσο με τη διαφορά μια προσπίπτουσας ακμής και μιας άλλης ακμής του γραφήματος.

Ο Αλγόριθμος είναι ο εξής:
1. Πρώτα, γίνεται ταξινόμηση του συνόλου των m που εξετάζεται, δηλαδή τα m για τα οποία ισχύει $m \in \{e - a : e \in E \wedge a \in A\}$.
2. Στην συνέχεια, γίνεται δυαδική αναζήτηση πάνω σε αυτό το σύνολο.
3. Ύστερα, σε κάθε επανάληψη, το συγκεκριμένο m προστίθεται στις ακμές που προσπίπτουν στον απαιτούμενο κόμβο και εφαρμόζεται Kruskal.
4. Αν ο απαιτούμενος βαθμός δεν είναι αρκετός, τότε μειώνεται το m και αντίστροφα.

Έλεγχος Ορθότητας:

Η ορθότητα των αλγορίθμων Binary Search και Kruskal είναι γνωστή και συνδυάζεται με τους ισχυρισμούς και τις αποδείξεις τους.

Πολυπλοκότητα:

Η πολυπλοκότητα του αλγορίθμου θα είναι $O(|E| * |A| * \log(|E|) + |E| * \log^2(|E|))$. Αυτό προκύπτει από το γεγονός ότι:
Το βήμα 1, απαιτεί το πολύ: $O(|E| * |A| * \log(|E| * |A|))$ χρόνο.
Το βήμα 2, απαιτεί το πολύ: $O(\log(|E| * |A|)) \leq O(\log(|E| * |E|)) = O(2 * \log(|E|))$ φορές.
Το βήμα 3, απαιτεί κάθε φορά: $O(E * \log(|E|))$.

5η Άσκηση – Αλγόριθμος Boruvka

Εκφώνηση & Επεξήγηση:

Έστω ένα συνεκτικό μη κατευθυνόμενο γράφημα G(V, E, w), με n κορυφές, m ακμές και θετικό βάρος w(e) σε κάθε ακμή e∈E, όπου τα βάρη των ακμών είναι όλα διαφορετικά μεταξύ τους.

a) Παρακάτω παρουσιάζεται μια πλήρης και αναλυτική απόδειξη ορθότητας για τον αλγόριθμο Boruvka. Στον αλγόριθμο, σε κάθε βήμα ενώνονται τα συνδεδεμένα κομμάτια ενός δάσους και για κάθε κομμάτι υπολογίζεται η προσπίπτουσα ακμή ελάχιστου κόστους και τέλος, προστίθενται οι ακμές στο δάσος και επαναλαμβάνεται η διαδικασία.
   Πιο συγκεκριμένα, αποδεικνύεται ότι:
   i. ο αλγόριθμος καταλήγει πάντα σε ένα συνδετικό δέντρο Τ.

      Απόδειξη:
      - Έστω ότι το γράφημα που προκύπτει περιέχει κύκλο και ότι στην τρέχουσα επανάληψη εντοπίζονται οι k προσπίπτουσες ακμές ελαχίστου κόστους e_1, e_2, …, e_k που σχηματίζουν κύκλο και e_1 η ελάχιστη ακμή. Συνεπώς, θα πρέπει w(e1) < w(e2) < … < w(ek) < w(e1) προκειμένου να διαλέξει ο αλγόριθμος και την ακμή e_k.
      - Αυτό όμως είναι άτοπο αφού σε κάθε επανάληψη, για κάθε ακμή που επιλέγεται ενώνονται συνεκτικές συνιστώσες.
      - Έτσι προκύπτει πάντα ένας συνεκτικός και ακυκλικός γράφος, δηλαδή ένα συνδετικό δέντρο.

   ii. αυτό το συνδετικό δέντρο Τ είναι πράγματι ελάχιστου βάρους.

      Απόδειξη:
      - Έστω ότι το γράφημα που προκύπτει δεν είναι ελαχίστου βάρους. Αυτό σημαίνει ότι υπάρχει ένα δέντρο Τ (MST) που δεν χρησιμοποιεί τις ελάχιστες προσπίπτουσες ακμές σε κάθε κόμβο. Έστω λοιπόν ότι σε κάποια τομή (S, S') το MST δεν περιέχει την ακμή ελαχίστου βάρους e*. Αν προστεθεί η e* στο Τ τότε δημιουργείται ένας κύκλος C. Υποχρεωτικά πρέπει να υπάρχει τουλάχιστον μία ακμή που διασχίζει την τομή και ανήκει στον κύκλο C. Για όλες τις ακμές $e \in C \cap (S, S')$, $e \neq e^*$ ισχύει $w(e) > w(e^*)$ καθώς όλα τα βάρη είναι μοναδικά. Άρα αν στο δέντρο αντικατασταθεί η e με την e* η κατάληξη θα είναι ένα συνδετικό δέντρο με αυστηρά μικρότερο βάρος.
      - Αυτό όμως είναι άτοπο.
      - Έτσι, η κατάληξη είναι πάντα σε MST.

b) Ο Αλγόριθμος είναι ο εξής:
   1. Αρχικά υπάρχει ένα δάσος από απομονωμένους κόμβους.
   2. Όσο υπάρχουν περισσότερα του ενός δέντρων στο δάσος, πρέπει να επιλέγεται για κάθε ένα από αυτά η ακτίνα με το μικρότερο βάρος, να το ενώνει με άλλο δέντρο και να προστίθεται στο MST.
   3. Η έξοδος θα είναι το MST.

Πολυπλοκότητα:

Η πολυπλοκότητα του αλγορίθμου θα είναι O(m·logn). Αυτό προκύπτει από το γεγονός ότι:

Το βήμα 2 γίνεται σε O(m), χρησιμοποιώντας δομή union-find. Η δομή θα περιέχει όλους τους κόμβους (αρχικά disjointed) και κάθε σύνολο της είναι ένα δέντρο που έχει φτιαχτεί μέχρι αυτήν την επανάληψη. Έτσι, για κάθε ακμή γίνεται ένα find στη δομή για να βρεθεί για κάθε δέντρο, η ελάχιστη που το συνδέει με άλλο δέντρο. Έπειτα, για κάθε ελάχιστη ακμή που έχει βρει, εκτελείται ένα union, για να συνδεθούν τα δυο δέντρα που συνδέει.

Το βήμα 3 γίνεται σε O(logn). Σε κάθε εκτέλεση του το πλήθος των δέντρων υποδιπλασιάζεται (ενώ είχε ξεκινήσει αρχικά με n δέντρα) για να καταλήξει στο συνολικό spanning tree.

c) Εκφώνηση & Επεξήγηση:
Παρακάτω επεξηγείται το δύναται να συνδυαστεί ο αλγόριθμος Boruvka με την υλοποίηση του αλγόριθμου Prim με σωρό Fibonacci ώστε να επιτευχθεί χρόνος εκτέλεσης O(m·log logn) για το πρόβλημα του ΕΣΔ.

Η βασική ιδέα εδώ είναι ότι θα τρέξει το βήμα 3 του αλγορίθμου για να μειωθεί ο αριθμός των κόμβων και στη συνέχεια θα τρέξει ο αλγόριθμος του Prim.

Πολυπλοκότητα:

Η πολυπλοκότητα του αλγορίθμου θα είναι O(m·log logn). Αυτό προκύπτει από το γεγονός ότι:

Έστω ότι το τρέχει το βήμα 3 r φορές, με κόστος (m·r). Μετά από αυτό, ο αριθμός των κόμβων είναι το πολύ $n/2^r$. Άρα, ο αλγόριθμος του Prim θέλει O($n/2^r$ log($n/2^r$)). Οπότε, συνολικά θα είναι $O(m \cdot r + n/2^r \cdot \log(n/2^r))$ και διαλέγοντας το r = log(logn) η κατάληξη είναι στο O(m·log logn).

d) Εκφώνηση & Επεξήγηση:
Παρακάτω εφαρμόζεται ο αλγόριθμος Boruvka με την εξής παραλλαγή:

Σε κάθε επανάληψη του αλγορίθμου γίνονται contract οι ακμές που έχουν επιλεχθεί (ως ελάχιστες που συνδέουν δυο disjointed δέντρα). Ιδιαίτερη προσοχή δίνεται στην "μετατροπή του γράφου σε κανονικό" μετά από κάθε contraction, να κρατιούνται οι ακμές με το ελάχιστο βάρος (όταν υπάρχουν πολλαπλές ακμές ανάμεσα σε δυο κόμβους).

Έλεγχος Ορθότητας:

Η ορθότητα του αλγορίθμου διατηρείται με προφανή τρόπο, καθώς τώρα σε κάθε επανάληψη, κάθε κόμβος αντιστοιχεί σε ένα δέντρο, αφού "θυμάται" τις ακμές που έχουν κάνει contract και αυτές επιστρέφονται στο τέλος ως MST).

Πολυπλοκότητα:

Η πολυπλοκότητα του αλγορίθμου θα είναι O(|V|). Αυτό προκύπτει από το γεγονός ότι:

Είναι εύκολη η υλοποίηση του contraction των επιλεγμένων ακμών σε O(m) (χρειάζεται ένα πέρασμα από όλες τις ακμές για να επιλεχθούν οι ελάχιστες σε κάθε multi-edge).
Αν αρχικά υπάρχουν E ακμές είναι γνωστό ότι στο βήμα i θα υπάρχουν το πολύ $E/2^{2i}$ ακμές.
Λόγω της κλειστότητας όμως, είναι επίσης ότι για κάθε γράφο που προκύπτει σε κάθε βήμα i, με σύνολο κόμβων V_i ισχύει $|E_i| = O(V_i)$.
Συνεπώς, η πολυπλοκότητα προκύπτει από την αναδρομική σχέση:
$T(V) = T(V/2) + O(V) \rightarrow T(V) = O(V)$

Σημείωση: στη γενική περίπτωση αυτή η παραλλαγή χρειάζεται χρόνο $O(V^2)$, καθώς $|E| \leq V^2$.

6η Άσκηση – Το Σύνολο των Συνδετικών Δέντρων (ΚΤ 4.27 και ΚΤ 4.28)

Εκφώνηση & Επεξήγηση:

Έστω ένα συνεκτικό μη κατευθυνόμενο γράφημα G(V, E) με n κορυφές και m ακμές.

a) Απόδειξη ύπαρξης ακμής:
Έστω T1, T2 δύο διαφορετικά συνδετικά δέντρα του G. Αρχικά, θα δειχθεί η ύπαρξη ακμής e'. Έστω ότι η ακμή c συνδέει δύο κορυφές a, b. Αν αφαιρεθεί η ακμή e από το spanning tree T_1, τότε τα a, b ανήκουν πλέον σε δύο ξένες συνεκτικές συνιστώσες, έστω τις G_1, G_2. Στο spanning tree T_2 οι συνεκτικές συνιστώσες αυτές συνδέονται με μια ακμή e'. Αν $e' \in T_1$ τότε θα υπάρχουν δύο μονοπάτια από το a στο b, ένα απευθείας μέσω της ακμής e και ένα μέσω ενδιάμεσων κορυφών που συνδέουν τελικά τις δύο συνεκτικές συνιστώσες G_1, G_2 μέσω της ακμής e'. Όμως, σε κάθε δέντρο, δύο οποιεσδήποτε κορυφές συνδέονται με μοναδικό μονοπάτι. Έτσι, προκύπτει άτοπο, δηλαδή $e' \notin T_1$.

Απόδειξη συνεκτικότητας:
Με ευθεία απόδειξη δείχνεται ότι το νέο δέντρο $T1 \setminus \{e\} \cup \{e'\}$ είναι συνεκτικό. Ειδικότερα, η ακμή e', όπως αποδείχθηκε, συνδέει τις συνεκτικές συνιστώσες G_1, G_2 που αποσυνδέθηκαν λόγω της αφαίρεσης της ακμής e. Η προσθήκη μιας ακμής που συνδέει δύο ασύνδετες συνεκτικές συνιστώσες δεν μπορεί να προκαλέσει κύκλο καθώς δεν υπάρχει τρόπος επιστροφής από την G_2 στην G_1 πέρα από την ακμή που προστέθηκε. Επομένως, το γράφημα που προκύπτει είναι και συνεκτικό και ακυκλικό (και προφανώς παραμένει δέντρο), οπότε θα είναι ένα συνδετικό δέντρο.

Αλγόριθμος εύρεσης μιας τέτοιας ακμής:
Ο τρόπος εύρεσης μιας τέτοιας ακμής είναι αρκετά προφανής. Αρχικά, αφαιρώντας την ακμή e, όπως εξηγήθηκε, προκύπτουν δύο ξένες συνεκτικές συνιστώσες G_1, G_2. Για κάθε μια από τις κορυφές του G_1, εξετάζεται αν υπάρχει ακμής προς το G_2 στο γράφο G. Όταν βρεθεί μια τέτοια ακμή διαφορετική της e, τότε βρέθηκε η ακμή e' που ήταν υπό αναζήτηση.

Πολυπλοκότητα:
Ο παραπάνω αλγόριθμος που περιεγράφηκε αφορά την προσπέλαση ακμών και κορυφών, οπότε είναι γραμμικός και η πολυπλοκότητα θα είναι: O(V + E).

b) Εκφώνηση & Επεξήγηση:
Σχηματίζεται γράφημα H που κάθε κορυφή του αντιστοιχεί σε ένα διαφορετικό δέντρο του G. Δύο συνδετικά δέντρα T_1, T_2 του G (κορυφές του H) συνδέονται με ακμή στο H αν διαφέρουν κατά μια μόνο ακμή, δηλ. αν $|T_1 \backslash T_2| = |T_2 \backslash T_1| = 1$.

Συνεκτικότητα γραφήματος H:
Από το προηγούμενο ερώτημα είναι φανερός ο μηχανισμός παραγωγής συνδετικών δέντρων. Ειδικότερα, δόθηκε ένας τρόπος αρχίζοντας από ένα συνδετικό δέντρο T^1 και αλλάζοντας μια ακμή τη φορά να φτάσει κανείς σε ένα συνδετικό δέντρο T_2.

Θεώρημα:
Όλα τα συνδετικά δέντρα έχουν τον ίδιο αριθμό ακμών.

Απόδειξη Θεωρήματος:
Κάθε δέντρο |V| κορυφών έχει |V – 1| ακμές. Κάθε συνδετικό δέντρο έχει όλες τις κορυφές (αφού είναι συνδετικό), άρα έχει V κορυφές και συνεπώς |V – 1| ακμές.

Εφόσον όλα τα συνδετικά δέντρα έχουν τον ίδιο αριθμό ακμών, χρησιμοποιώντας τον μηχανισμό του ερωτήματος (a) είναι εφικτό με ανταλλαγή ακμών η μετάβαση από το ένα συνδετικό δέντρο στο άλλο. Κάθε φορά μάλιστα, γίνεται ανταλλαγή μιας ακμής και επομένως, γίνεται μετάβαση σε κάποιον γείτονα στο γράφημα H. Συνεπώς, το γράφημα H είναι συνεκτικό.

Μήκος συντομότερου μονοπατιού στο H:
Παραπάνω επεξηγήθηκε ο μηχανισμός με τον οποίο είναι εφικτή η μετάβαση από ένα συνδετικό δέντρο T_1 σε ένα οποιοδήποτε συνδετικό δέντρο T_2 αλλάζοντας μία ακμή τη φορά. Όσον αφορά το γράφημα H, οι μεταβάσεις είναι όλες μεταβάσεις με μία ακμή διαφορά. Επομένως, η μετάβαση από ένα συνδετικό δέντρο T_1 στο T_2, αν το T_1 έχει k διαφορετικές ακμές από το συνδετικό δέντρο T_2, αλλάζοντας μια ακμή κάθε φορά αποτελεί το συντομότερο μονοπάτι που μπορεί να βρεθεί στο γράφο H και αποδείχθηκε ότι με το μηχανισμό του ερωτήματος (a) ένα τέτοιο μονοπάτι υπάρχει. Συνεπώς, το συντομότερο μονοπάτι μεταξύ T_1, T_2 έχει μήκος: $|T_1 \backslash T_2|$.

Αλγόριθμος εύρεσης συντομότερου μονοπατιού στο H:
Για την εύρεση του συντομότερου μονοπατιού στο H χρησιμοποιείται ο αλγόριθμος που δίνεται στο βήμα (a). Πιο συγκεκριμένα, εντοπίζονται k ακμές του T_1 που δεν υπάρχουν στο T_2 και αντίστοιχα k ακμές του T_2 που δεν υπάρχουν στο T_1. Το συνδετικό δέντρου του (a) φτιάχνεται σε χρόνο O(E + V), δηλαδή αφαιρείται μια ακμή από τις k που δεν υπάρχει στο T_2 και υπάρχει στο T_1 και αντικαθίσταται με μια ακμή από τις άλλες k που υπάρχει στο T_2 και δεν υπάρχει στο T_1. Έτσι, προκύπτει ένα δέντρο T_1'. Έπειτα, επαναλαμβάνεται οι διαδικασία για τις υπόλοιπες k – 1 ακμές για το προκύπτον δέντρο έως ότου να φτάσει στο T_2. Σημειώνεται ότι σε κάθε βήμα γίνεται μετάβαση στο δέντρο του H που ταιριάζει με το προκύπτον συνεκτικό δέντρο.

Κύριες Πηγές:
- *Σχέδιο Λύσεων*
- *https://www.geeksforgeeks.org/*