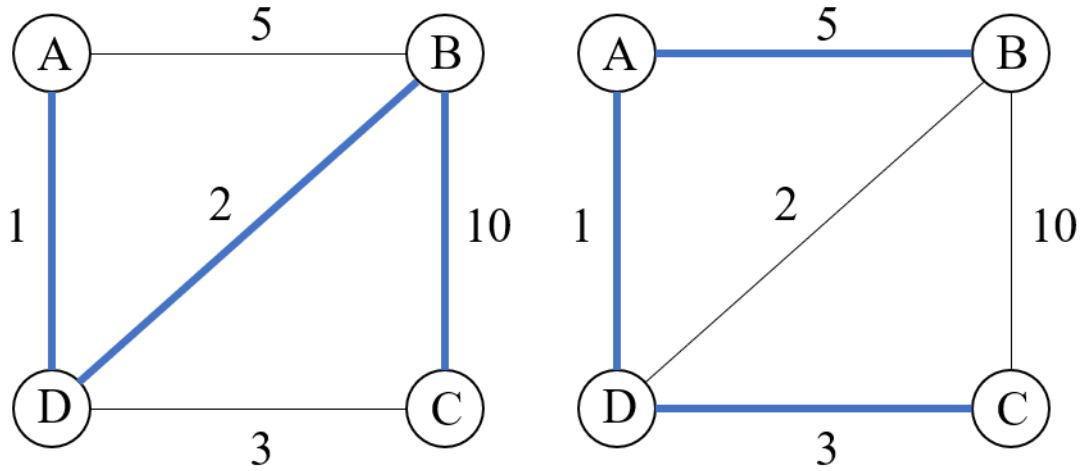
Δημήτριος Κυριακίδης
el17077

Αλγόριθμοι και Πολυπλοκότητα

3η Φυλλάδιο Ασκήσεων

Άσκηση 1:

α.) Αποδεικνύεται μέσω του εξής αντιπαραδείγματος, όπου θεωρούμε k = 2 για την κορυφή D:



Η άπληστη μέθοδος καταλήγει σε συνδετικό δένδρο συνολικού βάρους 13, ενώ το ελάχιστο συνδετικό δένδρο με τον περιορισμό έχει συνολικό βάρος 9.

β.) Προσθέτουμε σταθερά K στο βάρος κάθε ακμής προσπίπτουσας στην κορυφή s. Όσο αυξάνεται η K, θα ελαττώνεται το πλήθος των ακμών αυτών που θα ανήκουν στο επαγόμενο MST. Επομένως, η κατάλληλη τιμή της K μπορεί να εντοπισθεί με δυαδική αναζήτηση. Συγκεκριμένα, ο αλγόριθμος μπορεί να κατασκευαστεί ύστερα από την απόδειξη των ακόλουθων προτάσεων:

«Δεδομένου $G(V, E)$, για κάθε απαιτούμενο βαθμό k της s, το αντίστοιχο MST $T^*(s, k)$ με περιορισμό υλοποιείται ως το MST του γραφήματος $G(V, E')$, όπου το E' περιέχει όλες τις ακμές του E, με μοναδική διαφορά ότι οι προσπίπτουσες στην s έχουν βάρος αυξημένο κατά K».

Απόδειξη: για αρκετά μικρό K (αρνητικό), το MST περιέχει όλες τις προσπίπτουσες στην s ακμές, ενώ για αρκετά μεγάλο K περιέχει τις ελάχιστες δυνατές. Μεταξύ των δύο αυτών τιμών της K, υπάρχουν τιμές αυτής που οδηγούν σε MST με οποιοδήποτε από τα ενδιάμεσα πλήθη ακμών της s.

«Οι μοναδικές τιμές της K για τις οποίες αλλάζει ο βαθμός της s στο MST, είναι αυτές για τις οποίες το βάρος προσπίπτουσας στην s ακμής γίνεται ίσο με το βάρος άλλης ακμής του γραφήματος. Αρκεί, δηλαδή, να εξετασθούν τα $K \in \{w - a : w \in W \wedge a \in A\}$, όπου W το σύνολο των διαφορετικών βαρών

ακμών ολόκληρου του γραφήματος, και A το σύνολο των διαφορετικών βαρών ακμών προσπιπτουσών στην s».
Απόδειξη: Για να αφαιρεθεί μία ακμή από το MST, πρέπει να αντικατασταθεί με άλλη, ίσου βάρους. Άρα, μία προσπίπτουσα στην s ακμή μπορεί να αφαιρεθεί μόνον όταν η τιμή της K είναι τέτοια, ώστε το νέο βάρος της ακμής αυτής να ισούται με το βάρος άλλης ακμής του γραφήματος. Η τιμή αυτή προφανώς είναι η διαφορά των δύο βαρών των ακμών αυτών στο αρχικό γράφημα.
Ο αλγόριθμος συνίσταται στην ταξινόμηση του υπό εξέταση συνόλου $\{w - a: w \in W \wedge a \in A\}$, σε χρόνο $O(|W||A|\log(|W||A|))$, και έπειτα στην εκτέλεση δυαδικής αναζήτησης στο σύνολο αυτό. Η αναζήτηση έχει πολυπλοκότητα $O(\log(|W||A|)) \subseteq O(\log(|W||W|)) = O(2\log|W|)$, αφού $A \subseteq W$. Σε κάθε επανάληψη της δυαδικής αναζήτησης, βρίσκουμε MST για την αντίστοιχη τιμή της K με τον αλγόριθμο Kruskal, σε χρόνο $O(|E|\log|E|)$. Αν ο απαιτούμενος βαθμός δεν είναι αρκετός, συνεχίζουμε την αναζήτηση προς μικρότερα K, και αντιστρόφως. Αν όλες οι ακμές έχουν διαφορετικά βάρη, τότε $|W| = |E|$. Συνολικά, απαιτείται $O(|W||A|\log(|W||A|) + 2|E|\log^2|E|)$. Η μέθοδος αυτή αποτελεί βελτίωση επί της αναζήτησης σε ολόκληρο το διάστημα $[-\max W, \max W]$.

Άσκηση 5:

Sparse Subgraph:
Πρώτα θα περιγράψουμε αναγωγή από το πρόβλημα Clique (CL) στο Dense Sub-graph (DSG). Κατ' αρχάς, σε μια κλίκα εντός ενός μη κατευθυνόμενου γραφήματος, αποτελούμενη από n κορυφές, υπάρχουν ακριβώς n(n-1)/2 ακμές, και αντιστρόφως, αν μεταξύ n κορυφών υπάρχουν τόσες ακμές, τότε αυτές οι κορυφές δημιουργούν κλίκα.

Δοθέντος ενός στιγμιοτύπου του προβλήματος CL, μπορούμε να το αναγάγουμε σε στιγμιότυπο του DSG, θεωρώντας τους δύο θετικούς ακεραίους που απαιτούνται για την περιγραφή του DSG, a και b=a(a-1)/2. Η απάντηση για το αρχικό στιγμιότυπο του CL είναι θετική (δηλαδή υπάρχει ένα υποσύνολο κορυφών του γραφήματος, μεγέθους a, τέτοιο, ώστε να υπάρχουν a(a-1)/2 ακμές μεταξύ των κορυφών), αν και μόνο αν η απάντηση για το νέο στιγμιότυπο (του DSG) είναι επίσης θετική, δηλαδή υπάρχει σύνολο κορυφών στο γράφημα, μεγέθους a, τέτοιο, ώστε να υπάρχουν τουλάχιστον (εν προκειμένω ακριβώς) b=a(a-1)/2 ακμές μεταξύ τους. Επομένως, αφού γνωρίζουμε ότι το πρόβλημα CL είναι NP-πλήρες, συμπεραίνουμε ότι και το DSG είναι NP-πλήρες.

Το δοθέν πρόβλημα, Sparse Sub-graph (SSG), μπορεί να αναχθεί στο DSG ως εξής:
Έστω $G(V, E)$, $k \in \mathbb{N}$. Αν υπάρχει $S \subset V$ τέτοιο, ώστε το επαγόμενο υπο-γράφημα του G που ορίζεται από τις κορυφές του S να έχει τουλάχιστον $|E| - k$ ακμές, τότε το υπο-γράφημα του G που ορίζεται από τις κορυφές του $V \setminus S$ θα έχει το πολύ k ακμές (ακριβώς k αν το S έχει ακριβώς $|E| - k$, και το G δεν είναι συνεκτικό). Αν, λοιπόν, υπάρχει λύση στο DSG, τότε βρίσκεται εύκολα λύση και στο SSG, επομένως και αυτό είναι NP-πλήρες.

Άσκηση 6:

α.) Στον αλγόριθμο σε κάθε βήμα ενώνουμε τα συνδεδεμένα κομμάτια ενός δάσους. Για κάθε κομμάτι βρίσκουμε την προσπίπτουσα ακμή ελάχιστου κόστους. Προσθέτουμε τις ακμές στο δάσος και επαναλαμβάνουμε την διαδικασία.

Έστω ότι το γράφημα που προκύπτει περιέχει κύκλο. Έστω ότι στην τρέχουσα επανάληψη έχουμε εντοπίσει τις k προσπίπτουσες ακμές ελαχίστου κόστους $e_1, e_2, \dots, e_k$ που σχηματίζουν κύκλο και e_1 η ελάχιστη ακμή. Επομένως πρέπει να έχουμε $w(e_1) < w(e_2) < \dots < w(e_k) < w(e_1)$ προκειμένου να διαλέξει ο αλγόριθμος και την ακμή e_k → άτοπο.
Σε κάθε επανάληψη, για κάθε ακμή που επιλέγεται ενώνονται συνεκτικές συνιστώσες. Έτσι καταλήγουμε πάντα σε ένα συνεκτικό και ακυκλικό γράφημα, δηλαδή σε ένα συνδετικό δένδρο.

Έστω ότι το γράφημα που προκύπτει δεν είναι είναι ελαχίστου βάρους. Αυτό σημαίνει ότι υπάρχει ένα δέντρο T (MST) που δεν χρησιμοποιεί τις ελάχιστες προσπίπτουσες ακμές σε κάθε κόμβο. Έστω λοιπόν ότι σε κάποια τομή $(S, \bar{S})$ το MST δεν περιέχει την ακμή ελαχίστου βάρους e^*. Αν προσθέσουμε την e^* στο T τότε δημιουργούμε ένα κύκλο C. Υποχρεωτικά πρέπει να υπάρχει τουλάχιστον μία ακμή που διασχίζει την τομή και ανήκει στον κύκλο C. Για όλες τις ακμές $e \in C \cap (S, \bar{S}), e \neq e^*$ ισχύει $w(e) > w(e^*)$ καθώς όλα τα βάρη είναι μοναδικά. Άρα, αν στο δένδρο αντικαταστήσουμε την e με την e^* καταλήγουμε σε ένα συνδετικό δέντρο με αυστηρά μικρότερο βάρος → άτοπο.
Έτσι καταλήγουμε πάντα σε MST.
β.) Αλγόριθμος:
1. Ξεκινάμε με ένα δάσος από απομονωμένες κορυφές.
2. Όσο υπάρχουν περισσότερα του ενός δένδρων στο δάσος, διαλέγουμε για κάθε ένα από αυτά την ακτίνα με το μικρότερο βάρος το ενώνει με άλλο δένδρο και την προσθέτουμε στο MST.
3. Η έξοδος είναι το MST.
Πολυπλοκότητα: 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 δένδρο που έχουμε φτιάξει μέχρι αυτήν την επανάληψη. Έτσι, κάνουμε για κάθε ακμή ένα find στην δομή για να βρούμε για κάθε δένδρο, την ελάχιστη που το συνδέει με άλλο δένδρο. Έπειτα, για κάθε ελάχιστη ακμή που έχουμε βρει, εκτελούμε ένα union, για να συνδέσουμε τα δυο δένδρα που συνδέει. Τέλος, σε κάθε εκτέλεση του βήματος 3 το πλήθος των δένδρων υποδιπλασιάζεται (ενώ έχουμε ξεκινήσει αρχικά με n δένδρα). Έτσι, ο αλγόριθμος θα κάνει το πολύ log n επαναλήψεις για να καταλήξει στο συνολικό spanning tree. Επομένως, συνολικά απαιτείται χρόνος O(mlogn).
γ.) Η βασική ιδέα είναι να τρέξουμε το βήμα 3 του αλγορίθμου για να μειώσουμε τον αριθμό των κορυφών, και στην συνέχεια να τρέξουμε τον αλγόριθμο του Prim. Έστω ότι το τρέχουμε r φορές το βήμα 3, με κόστος (mr). Μετά από αυτό ο αριθμός των κορυφών είναι το πολύ $n/2^r$. Άρα ο αλγόριθμος του Prim θέλει $O(n/2^r \log n/2^r)$. Συνολικά έχουμε $O(mr + n/2^r \log n/2^r)$. Διαλέγουμε το $r = \log\log n$ και καταλήγουμε στο $O(m \log\log n)$.
δ.) Μπορούμε να εφαρμόσουμε τον αλγόριθμο Boruvka, με την εξής παραλλαγή:

Σε κάθε επανάληψη του αλγορίθμου κάνουμε contract τις ακμές που έχουμε επιλέξει (ως ελάχιστες που συνδέουν δυο disjoint δέντρα). Φροντίζουμε στην ”μετατροπή του γράφου σε κανονικό” μετά από κάθε contraction, να κρατάμε τις ακμές με το ελάχιστο βάρος (όταν έχουμε πολλαπλές ακμές ανάμεσα σε δυο κόμβους). Η ορθότητα του αλγορίθμου διατηρείται με προφανή τρόπο, καθώς τώρα σε κάθε επανάληψη, κάθε κόμβος αντιστοιχεί σε ένα δέντρο. (Θυμόμαστε τις ακμές που έχουμε κάνει contract και τις επιστρέφουμε στο τέλος ως MST). Είναι εύκολο να υλοποιήσουμε το contraction των επιλεγμένων ακμών σε O(m) (χρεάζεται ένα πέρασμα από όλες τις ακμές για να διαλέξουμε τις ελάχιστες σε κάθε multiedge). Αν αρχικά έχουμε m ακμές ξέρουμε ότι στο βήμα i θα υπάρχουν το πολύ $m/2^{2i}$ ακμές. Λόγω της κλειστότητας όμως, γνωρίζουμε ότι για κάθε γράφο που προκύπτει σε κάθε βήμα i, με πλήθος κόμβων n_i ισχύει $m_i = O(n_i)$. Συνεπώς, η πολυπλοκότητα προκύπτει από την αναδρομική σχέση: $T(V) = T(n/2) + O(n) \Rightarrow T(n) = O(n)$ Στην γενική περίπτωση αυτήν, η παραλλαγή χρειάζεται χρόνο $O(n^2)$, καθώς το m θα μπορούσε να φθάνει μέχρι και n^2.

Άσκηση 8:

α.) Suppose that we have a flow network $G = (V, E)$ with source s and sink t. Let *f* be a flow in G, and consider a pair of vertices $u, v \in V$. We define the residual capacity $c_f(u, v)$ by:

$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v), & if\ (u,v) \in E \\ f(v,u), & if (v,u) \in E \\ 0, & otherwise \end{cases}$$

Given a flow network $G = (V, E)$, and a flow *f*, the residual network of G induced by *f* is $G_f = (V, E_f)$, where $E_f = \{(u, v) \in V \times V : c_f(u, v) > 0\}$.

Πηγή:

Thomas, Cormen H., et al. *Introduction to Algorithms*. 3rd ed., Cambridge, MIT Press, 2009.

Επομένως, εάν στο υπολειμματικό δίκτυο δεν υπάρχει διαδρομή από την s στην t, τότε η ροή είναι μέγιστη. Η κατασκευή του υπολειμματικού δικτύου και η εξερεύνηση αυτού (DFS) μπορούν να επιτευχθούν σε χρόνο γραμμικό ως προς το πλήθος των κορυφών και ακμών του γραφήματος.