Αλγόριθμοι και Πολυπλοκότητα

Γαβαλάς Νίκος, ΑΜ 03113121

Ιανουάριος 2019

4η Γραπτή Σειρά Ασκήσεων

Ασκηση 1

Ζητάμε όλα τα μονοπάτια p που ξεκινούν από το i και καταλήγουν σε κάποιο j, με $t(p) = \prod_{q=0}^{q=l-1} t(q, q+1) \geq b_l$. Το πρόβλημα μπορεί να λυθεί κατασκευάζοντας αρχικά έναν γράφο με κόμβους τους χρήστες και ακμές τις σχέσεις μεταξύ τους (δηλαδή αν ο χρήστης i έχει για τον χρήστη j τιμή εμπιστοσύνης $t(i, j)$ τότε αναπαριστούμε τη σχέση αυτή με μία κατευθυνόμενη ακμή (i, j) και βάρος $t(i, j)$), και ύστερα αναζητώντας τα μονοπάτια από το i προς κάθε j με μήκος $l \leq k$ και συνολικό βάρος $\geq b_l$ στον γράφο αυτόν.

Για να μπορέσουμε να τρέξουμε Bellman-Ford και να βρούμε τα ζητούμενα μονοπάτια, κάνουμε πρώτα τα εξής: Πρώτον, αντί για το $p' = \arg\max_p t(p)$, αναζητούμε το $p' = \arg\min_p \{-\log t(p)\}$ και αντί για βάρη ακμών $t(i, j)$ χρησιμοποιούμε τις τιμές $\log \frac{1}{t(i,j)}$.

Με αυτές τις μετατροπές, μπορούμε να τρέξουμε τον Bellman-Ford που δίνει το συντομότερο μονοπάτι από τον i στον j με το πολύ m ακμές με την γνωστή αναδρομική σχέση:

$$D[j, m] = \min\{D[j, m-1], \min_{v:(v,j)\in E}\{D[v, m-1] + w(v, j)\}\}$$

Κάθε φορά που ανανεώνεται μια $D[j, m]$ ελέγχουμε αν είναι $\leq \log \frac{1}{b_m}$, και αν είναι, τότε ο j αποτελεί καλή πρόταση φίλου. Η πολυπλοκότητα είναι $O(|V| + k|E|)$, αφού θέλουμε $|V|$ χρόνο για να αρχικοποιήσουμε κάθε απόσταση D στο 0 και ύστερα $k|E|$ χρόνο για να κάνουμε k επαναλήψεις που η καθεμία εξετάζει όλες τις ακμές. Αρκούν k γιατί τα μονοπάτια που αναζητούμε έχουν μέγιστο μήκος k.

Ασκηση 2

α)

Αν $w(e) \in \{0, 1, ..., C\}$ με C σχετικά μικρό, μπορούμε να κάνουμε κάτι εμπνευσμένο από την Counting Sort. Κάνουμε allocate πίνακα A μεγέθους nC (επειδή αυτή είναι η μέγιστη δυνατή απόσταση) με κάθε στοιχείο του πίνακα αυτού να είναι δείκτης σε λίστα κόμβων, και αρχικοποιούμε και έναν δείκτη i στην αρχή του πίνακα.

Κάθε θέση του πίνακα αντιπροσωπεύει την απόσταση μεταξύ του κόμβου s και καθενός εκ των στοιχείων της λίστας που δείχνει. Εκτελούμε κανονικά Dijkstra, με τη διαφορά ότι αντί να τοποθετούμε τους κόμβους σε λίστα προτεραιότητας, τους τοποθετούμε στην κατάλληλη λίστα από αυτές που δείχνουν τα στοιχεία του A, βάσει απόστασης από τον s, και σε κάθε βήμα προχωράμε τον i προς τα δεξιά, μέχρι το τέλος του πίνακα. Η τελική μορφή του A περιέχει τις συντομότερες αποστάσεις από τον s προς οποιονδήποτε άλλον κόμβο.

Ο αλγόριθμος αυτός έχει πολυπλοκότητα που υπολογίζεται ως εξής: Κάνουμε $O(m)$ επισκεπτόμενοι κάθε ακμή από μία φορά κατά την εκτέλεση, και παράλληλα τον i τον μετακινούμε σε όλο το μέγεθος του A, άρα $O(nC)$, οπότε σύνολο $O(m + nC)$.

β)

Ο Dijkstra γνωρίζουμε ότι κάνει χρόνο $O((n + m) \log n)$, αν η ουρά προτεραιότητας είναι υλοποιημένη με Binary Heap, όπου ο όρος $\log n$ προκύπτει από το χρόνο αναδιοργάνωσης του Heap. Αν έχουμε βάρη $w(e) \in \{0, 1, ..., 2^C\}$, τότε το βήμα αναδιοργάνωσης του σωρού δεν θα κάνει χρόνο n, αλλά $\log 2^C = C$ γιατί δεν γίνεται να έχει ο σωρός ύψος μεγαλύτερο του $\log 2^C$ (αποδεικνύεται επαγωγικά). Επομένως έχουμε συνολικά εκτέλεση σε χρόνο $O((n + m)C)$.

Άσκηση 3

α)

Υποθέτουμε ότι $c(e) \in \{0, 1\}$, και ζητάμε το μήκος του συντομότερου μονοπατιού με ακμές συνολικού κόστους το πολύ $k \geq 1$. Το πρόβλημα αντιμετωπίζεται με τη κατασκευή ενός γράφου G' που προκύπτει από το αρχικό G ως εξής:

Αρχικά φτιάχνουμε ένα γράφο G_0 που έχει αντίγραφα όλων των κόμβων u του G ως u_0 και αντίγραφα όλων των ακμών του G με κόστος 0. Μετά φτιάχνουμε έναν γράφο G_1 που έχει επίσης αντίγραφα όλων των κόμβων του G αλλά οι ακμές του που έχουν κόστος 1, ξεκινάνε από αυτόν και καταλήγουν στους κόμβους του G_0, και επίσης ο κόμβος εκκίνησης s_1 ταυτίζεται με αυτόν του G_0 ($s_0 \equiv s_1$).

Με αυτόν τον τρόπο δημιουργούμε k επιπλέον του G_0 γράφους, μέχρι τον G_k, και τους ενώνουμε με ακμές κόστους 1 από τους κόμβους του G_i στους κόμβους του $G_{i-1}, i = \{1, ..., k\}$. Σε αυτό το μεγάλο ενωμένο γράφημα λοιπόν, που αποτελεί το G', τρέχουμε Dijkstra, ο οποίος για κάθε κόμβο $u_i, i = \{0, 1, ..., k\}$ θα υπολογίσει την ελάχιστου μήκους απόσταση με ακριβώς k κόστος. Επειδή ζητάμε την ελάχιστου μήκους απόσταση με το πολύ k κόστος, η τελική απάντηση δίνεται παίρνοντας το ελάχιστο εξ' αυτών για κάθε k, δηλαδή το $\min_{i \leq k} u_i$.

Η ορθότητα του αλγορίθμου αυτού βασίζεται στο ότι άπαξ και κατά την εκτέλεση από ένα γράφημα G_i οδηγηθούμε στο G_j με $i > j$, δεν γίνεται να «γυρίσουμε πίσω» λόγω της τοποθέτησης των ακμών. Κάθε φορά δηλαδή που χρησιμοποιούμε ακμή κόστους 1, αλλάζουμε γράφο με αποτέλεσμα τελικά να εξασφαλίζεται ότι οι τελική απάντηση θα δοθεί με το πολύ k χρήσεις ακμών με κόστος 1. Οι ακμές κόστους 0 δεν επηρεάζουν σε κάτι.

Η πολυπλοκότητα υπολογίζεται ως εξής: $O(km)$ για την κατασκευή του G' και $O(km + kn \log(kn))$ για την εκτέλεση του Dijkstra δίνουν συνολικό χρόνο $O(km + kn \log(kn))$, η οποία είναι πολυωνυμική (της τάξης του $O(mn + 2n^2 \log n)$) αφού $k < n$.

β)

Το πρόβλημα θυμίζει το 0-1 Knapsack. Ακολουθώντας παρόμοια λογική με πριν, αφού το συνολικό κόστος θέλουμε να είναι $\leq C$, κατασκευάζουμε $C + 1$ γράφους, τους $G_i, i = \{0, 1, ..., C\}$, που έχουν κόμβους αντίγραφα u_i του αρχικού G, και εννόνωνται με ακμές $e_i = (u_i, v_{i+c(e)}), i \leq C - c(e)$, για κάθε ακμή $e = (u, v)$ του G. Τέλος, ταυτίζουμε όλους τους κόμβους εκκίνησης μεταξύ τους. Ο γράφος που προκύπτει αποθηκεύει την πληροφορία του τρέχοντος κόστους στο επίπεδο i.

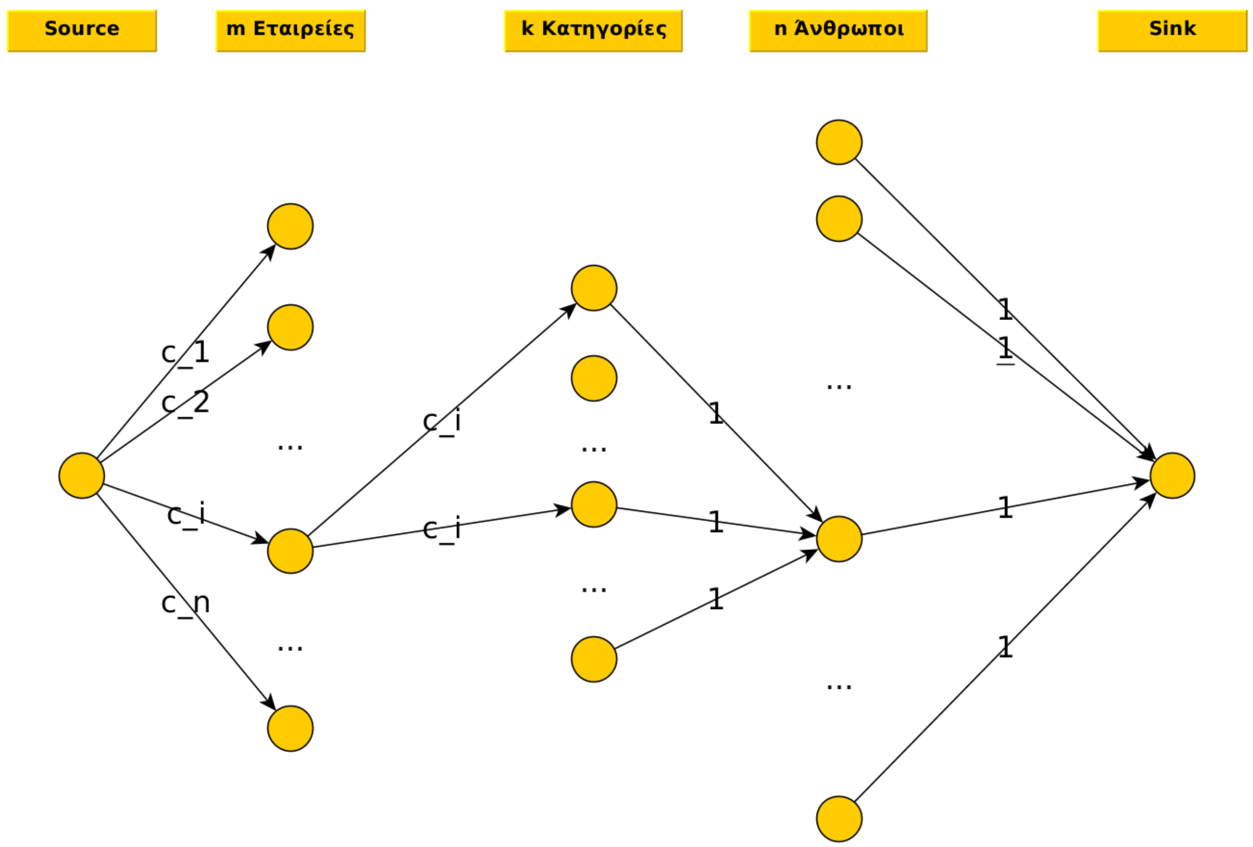
Αφού κατασκευάσουμε αυτόν τον γράφο, τρέχουμε σε αυτόν Dijkstra , και τέλος το συντομότερο μονοπάτι $s - u$ με κόστος το πολύ C είναι το συντομότερο εκ των μονοπατιών $s_0 - u_i, \forall i \leq C$.

Σχετικά με την πολυπλοκότητα, θέλουμε $O(Cn+Cm)$ για τη κατασκευή του γράφου και $O(Cm+Cn\log(Cn))$ και για την εκτέλεση του Dijkstra, άρα συνολικά $O(Cm + Cn\log(Cn))$, όμως επειδή το C δεν φράσσεται από παράμετρο του προβλήματος, ο αλγόριθμος είναι όπως και το 0-1 Knapsack ψευδοπολυωνυμικού χρόνου.

Άσκηση 4

Στο πρόβλημα αυτό έχουμε m εταιρείες, η κάθε μία από τις οποίες (έστω η i-οστή) πληρώνει για c_i διαφημίσεις την ημέρα, που στοχεύουν σε ανθρώπους που ανήκουν σε κάποιες από k δημογραφικές κατηγορίες. Οι άνθρωποι αυτοί είναι στο σύνολό τους n και ο καθένας ανήκει σε το πολύ k δημογραφικές κατηγορίες.

Θέλουμε να ξέρουμε αν υπάρχει τρόπος να προβάλλονται οι διαφημίσεις έτσι ώστε ο κάθε άνθρωπος να βλέπει μία διαφήμιση. Μπορούμε να λύσουμε το πρόβλημα με Max Flow. Αρχικά κατασκευάζουμε τον ακόλουθο γράφο:



, όπου πάνω στις ακμές είναι σημειωμένες οι χωρητικότητες. Βρίσκουμε ύστερα τη μέγιστη ροή με κάποιον αλγόριθμο Max Flow, και αν είναι ίση με n, τότε μπορούμε να είμαστε σίγουροι ότι κάθε άνθρωπος είδε ακριβώς μία διαφήμιση. Για να βρούμε ποιοι ακριβώς βλέπουν για κάθε εταιρεία τις διαφημίσεις της, αρκεί να ακολουθήσουμε τις ροές από τον γράφο αφού τρέξουμε τον αλγόριθμο.

Η πολυπλοκότητα αυτής της λύσης είναι αυτή του αλγορίθμου που θα χρησιμοποιήσουμε για το Max Flow, για παράδειγμα $O(|E|n)$ αν χρησιμοποιήσουμε Ford-Fulkerson, συν όσο χρειάζεται για να κατασκευαστεί ο γράφος.

Άσκηση 5

3-Partition

Είσοδος: $A = \{w_1, ..., w_n\}, w_i \in \mathbb{N}^+, w(A) = \sum_{i \in A} w_i = 3k, k = \{0, 1, ...\}$
Ερώτηση: $\exists$ διαμέριση σε $A_1, A_2, A_3 : w(A_1) = w(A_2) = w(A_3)$;

Το πρόβλημα ανήκει στη κλάση NP γιατί αν μας δοθεί λύση μπορούμε να επιβεβαιώσουμε ότι είναι λύση σε γραμμικό χρόνο. Θα δείξουμε ότι είναι και NP-hard ως αναγωγή από το Partition.

Έστω το σύνολο $A' = \{w_1, ..., w_n, \frac{\sum_{i=1}^n w_i}{2}\}$
Ευθύ: Από το 3-Partition παίρνουμε τρία υποσύνολα από το A' , ένα εκ των οποίων έχει μόνο το στοιχείο $\frac{\sum_{i=1}^n w_i}{2}$ και άλλα δύο που αποτελούνται από στοιχεία του A και δίνουν το 2-Partition αυτού.
Αντίστροφο: Αν υπάρχει 2-Partition, προφανώς είναι και 3-Partition.

Άθροισμα Υποσυνόλου κατά Προσέγγιση

Είσοδος: $A = \{w_1, ..., w_n\}, w_i \in \mathbb{N}$ και $x, B \in \mathbb{N}$ με $B > x \geq 1$
Ερώτηση: $\exists S \subseteq A : B - x \leq w(S) \leq B$;

Το πρόβλημα ανήκει στη κλάση NP γιατί αν μας δοθεί λύση μπορούμε να επιβεβαιώσουμε ότι είναι λύση σε γραμμικό χρόνο. Θα δείξουμε ότι είναι και NP-hard ως αναγωγή από το Subset Sum.

Έστω το $A' = \{2w_1, 2w_2, ..., 2w_n\}$. Θέτουμε $B = 2W$ και $x = 1$, οπότε γνωρίζουμε αν υπάρχει $S \subseteq A : 2W - 1 \leq w(S) \leq 2W$, δηλαδή αν υπάρχει $S : w(S) = 2W - 1$ ή $w(S) = 2W$, όμως $2W - 1$ δεν γίνεται να υπάρχει γιατί το $w(S)$ θα είναι άρτιος, οπότε γνωρίζουμε για ισότητα με $2W$. Αυτό συνεπάγεται ότι γνωρίζουμε και τη λύση του Subset Sum, για τον πίνακα A και παράμετρο W.

Κύκλος Hamilton κατά Προσέγγιση

Είσοδος: $G(V, E)$ μη κατευθυνόμενο.
Ερώτηση: $\exists$ κυκλική διαδρομή που διέρχεται από κάθε κορυφή τουλάχιστον μία και το πολύ δύο φορές;

Το πρόβλημα ανήκει στη κλάση NP γιατί αν μας δοθεί λύση μπορούμε να επιβεβαιώσουμε ότι είναι λύση σε γραμμικό χρόνο (ότι δηλαδή είναι κύκλος και περιλαμβάνει κάθε κορυφή από 1 ή 2 φορές). Θα δείξουμε ότι είναι και NP-hard ως αναγωγή από το Hamilton Cycle.

Ευθύ: Κατασκευάζουμε τον γράφο G' με ίδιους κόμβους u με το G και ίδιες ακμές, και ύστερα του προσθέτουμε σε κάθε κόμβο u έναν κόμβο u' που συνδέεται με μία μοναδική ακμή. Τρέχοντας το Approximation Hamilton Cycle, «υποχρεωνόμαστε» να περάσουμε δύο φορές από κάθε κόμβο αφού για να συμπεριληφθεί στον τελικό κύκλο κάθε κόμβος u' πρέπει να περάσουμε δύο φορές από τον αντίστοιχο u ($u \to u' \to u$). Τέλος παίρνουμε τη λύση για το Hamiltonian Path κόβοντας τους έξτρα κόμβους u'.
Αντίστροφο: Αν υπάρχει κύκλος Hamilton, υπάρχει και κύκλος κατά προσέγγιση.

Ικανοποιησιμότητα με Περιορισμούς

Είσοδος: $\phi = \bigwedge_{j=1}^{m} (\ell_{j1} \vee \ell_{j2} \vee \ell_{j3} \vee \ell_{j4})$ λογική πρόταση σε 4-CNF
Ερώτηση: $\exists$ ανάθεση λογικών τιμών ώστε κάθε όρος $\ell_{j1} \vee \ell_{j2} \vee \ell_{j3} \vee \ell_{j4}$ να περιλαμβάνει τουλάχιστον ένα αληθές και ένα ψευδές literal;

Το πρόβλημα ανήκει στη κλάση NP γιατί αν μας δοθεί λύση μπορούμε να επιβεβαιώσουμε ότι είναι λύση σε γραμμικό χρόνο. Θα δείξουμε ότι είναι και NP-hard ως αναγωγή από το 3-SAT.

Έστω η λογική πρόταση $\psi = \bigwedge_{j=1}^{m} (\ell_{j1} \vee \ell_{j2} \vee \ell_{j3})$ και η $\psi' = \bigwedge_{j=1}^{m} (\ell_{j1} \vee \ell_{j2} \vee \ell_{j3} \vee x)$, όπου x λογική μεταβλητή που δεν εμφανίζεται αλλού.
Ευθύ: Αν η ψ' ικανοποιείται για κάποια ανάθεση A, τότε η ίδια ανάθεση ικανοποιεί και την ψ, για $x = false$, διαφορετικά αν $x = true$, τότε την ικανοποιεί η ανάθεση που σχηματίζεται από τις λογικές αρνήσεις της A, άρα λύνεται το 3-SAT.
Αντίστροφο: Αν υπάρχει ανάθεση που να ικανοποιεί την ψ, τότε σημαίνει ότι τουλάχιστον ένας όρος από κάθε clause είναι $true$, οπότε θέτοντας $x = false$, ικανοποιείται και ο περιορισμός για την ψ'.

Επιλογή Ανεξάρτητων Υποσυνόλων

Είσοδος: $S = \{S_1, ..., S_m\}, S_i \subset U, U$ σύνολο με n στοιχεία, $k \in \mathbb{N}, 2 \leq k \leq m$
Ερώτηση: Υπάρχουν k υποσύνολα από την S που να είναι ανά δύο ξένα μεταξύ τους;

Το πρόβλημα ανήκει στη κλάση NP γιατί αν μας δοθεί λύση μπορούμε να επιβεβαιώσουμε ότι είναι λύση σε πολυωνυμικό χρόνο. Θα δείξουμε ότι είναι και NP-hard ως αναγωγή από το MIS (Max Independent Set).

Έστω γράφος G με m κορυφές, κατ' αντιστοιχία με τα σύνολα S_m. Αν ένα από τα S_i τέμνεται με ένα S_j, εισάγουμε ακμή (i, j) στο G.
Ευθύ: Αν υπάρχουν k υποσύνολα του S που να είναι ανά δύο ξένα μεταξύ τους, τότε στον G υπάρχουν k κορυφές που δεν έχουν κοινές ακμές μεταξύ τους, άρα αποτελούν Independent Set με παράμετρο k.
Αντίστροφο: Αν ο G έχει k Independent Sets, δηλαδή k κόμβους που δεν έχουν κοινές ακμές μεταξύ τους, τότε τα αντίστοιχα S_i δεν τέμνονται, και άρα υπάρχουν k ανεξάρτητα εξ' αυτών.

Συντομότερο Μονοπάτι με Περιορισμούς

Είσοδος: Κατευθυνόμενο $G(V, E, w, c)$, s, t δύο κορυφές, $w(e), c(e)$ μη-αρνητικοί ακέραιοι και W, C δύο επίσης μη-αρνητικοί ακέραιοι.
Ερώτηση: Υπάρχει $s - t$ μονοπάτι στο G με συνολικό μήκος $\leq W$ και συνολικό κόστος $\leq C$;

Το πρόβλημα ανήκει στη κλάση NP γιατί αν μας δοθεί λύση μπορούμε να επιβεβαιώσουμε ότι είναι λύση σε γραμμικό χρόνο. Θα δείξουμε ότι είναι και NP-hard ως αναγωγή από το Knapsack.

Έστω n αντικείμενα με β_i, p_i το καθένα, και θέλουμε $\sum p_i \geq P$, με τον περιορισμό $\sum \beta_i \leq B$, και έστω γράφος G με n κορυφές, τις u_i. Προσθέτουμε στο G ακμές ως εξής: για κάθε κορυφή i προσθέτουμε δύο ακμές προς την $i+1$, η μία με κόστος $c_i = \sum_{i=0}^{n} (p_i) - p_i$ και μήκος $w_i = \beta_i$ και η άλλη με κόστος $\sum_{i=0}^{n} (p_i)$ και μήκος 0. Είναι εμφανές ότι η λύση του ενός προβλήματος για $s = u_0, t = u_n, W = B, C = n \sum_{i=0}^{n} (p_i) - P$ συνεπάγεται τη λύση του άλλου και αντίστροφα.