4η Γραπτή Άσκηση

Αλγόριθμοι και Πολυπλοκότητα

CoReLab

ΣΗΜΜΥ

7 Φεβρουαρίου 2017

Άσκηση 1

- Πρέπει να βρούμε όλες τις καλές προτάσεις φίλων για τον i ανάμεσα σε όλους του υπόλοιπους χρήστες.
- Για κάθε χρήστη j, αν βρούμε ένα μονοπάτι μήκους $2 \leq \ell \leq k$ για το οποίο η συνολική εκτίμηση εμπιστοσύνης είναι μεγαλύτερη από β_ℓ, πρέπει να τον προσθέσουμε στις καλές προτάσεις.
- Αρκεί να βρούμε για κάθε j και για κάθε μήκος μονοπατιού (=πλήθος ακμών) το μονοπάτι με τη μεγαλύτερη συνολική εκτίμηση και αν αυτό πληροί τις προϋποθέσεις της παραμέτρου που αντιστοιχεί στο μήκος του, τότε προφανώς είναι καλή πρόταση. Διαφορετικά, αν δεν τις πληροί για κανένα μήκος μονοπατιού, δεν είναι.
- Το πρόβλημα παραπέμπει στο πρόβλημα Υπολογισμού Συντομότερων Μονοπατιών από μία κορυφή προς όλες τις άλλες. Αλλά πώς πρέπει να το διαμορφώσουμε;

Άσκηση 1

- Για κάθε j και κάθε μονοπάτι p από το i σε αυτό, θέλουμε να βρούμε το μέγιστο $t(p) = \Pi_{q=0}^{q=\ell-1} t(q, q+1)$. Δηλαδή το μέγιστο $\log t(p) = \sum_{q=0}^{q=\ell-1} \log t(q, q+1)$, δηλαδή το ελάχιστο $-\log t(p) = \sum_{q=0}^{q=\ell-1} \log \frac{1}{t(q,q+1)}$.
- Επομένως, αν σχηματίσουμε το γράφημα όλων των n ατόμων και θέσουμε βάρος κάθε ακμής $e = (u, v)$ το $w(u, v) = \log \frac{1}{t(u,v)}$ (θετικό αφού $t(u, v) \in (0, 1)$) και δεν έχουμε καμία ακμή αν $t(u, v) = 0$, μετασχηματίζουμε το πρόβλημα έτσι ώστε να αναζητάμε συντομότερα μονοπάτια από το i προς κάθε j μήκους $\ell \leq k$, που να έχουν συνολικό βάρος $\leq \log \frac{1}{\beta_\ell}$.

Άσκηση 1

- Στις διαφάνειες του μαθήματος έχουμε δει τον αλγόριθμο Bellman-Ford σε μορφή δυναμικού προγραμματισμού που υπολογίζει το συντομότερο μονοπάτι από την αρχική στον u με το πολύ r ακμές $D[u, r] = min\{D[u, r-1], min_{v:(v,u)\in E} D[v, r-1] + w(v, u)\}$.
- Τρέχουμε τον αλγόριθμο αυτό για k επαναλήψεις. Αυτό που μας επιτρέπει να πάρουμε σωστά αποτελέσματα είναι ότι κάθε επανάληψη βασίζεται στις προηγούμενες τιμές ώστε να εξασφαλίζουμε ότι το μονοπάτι που υπολογίζουμε δεν ξεπερνά το δείκτη της επανάληψης.

Άσκηση 1

- Μένει να τροποποιήσουμε λίγο ακόμα τον αλγόριθμο ώστε να μας δίνει τις καλές προτάσεις φίλων. Κάθε φορά που ανανεώνεται μία απόσταση $D[u, r]$ εξετάζουμε αν είναι $\leq \log \frac{1}{\beta_r}$ και μόνο τότε την αποθηκεύουμε ως καλή πρόταση. Όταν ανανεωθεί, τότε πράγματι το μήκος του μονοπατιού θα είναι r γιατί αν ήταν μικρότερο θα το είχαμε βρει σε προηγούμενη επανάληψη και δε θα το ανανεώναμε.
- Στο τέλος έχουμε ελέγξει όλα τα δυνατά συντομότερα μονοπάτια με $\ell \leq k$ μήκος προς κάθε κόμβο j και έχουμε βρει τις καλές προτάσεις φίλων.

Άσκηση 1

Η πολυπλοκότητα υπολογίζεται ως εξής:

- Η αρχικοποίηση όλων των αποστάσεων $D[u, 0]$ γίνεται σε $O(n)$ χρόνο.
- Σε κάθε επανάληψη, εξετάζουμε όλες τις ακμές από μία φορά (για κάθε κόμβο όλες τις γειτονικές του) και κάνουμε σταθερό αριθμό συγκρίσεων σε $O(m)$. Οι επαναλήψεις είναι συνολικά k άρα το δεύτερο μέρος γίνεται σε $O(km)$.

Επομένως έχουμε συνολική πολυπλοκότητα:

$$O(n + km)$$

Άσκηση 2

- Πρέπει να βρούμε όλες τις ακμές που ανήκουν σε κάποιο συντομότερο μονοπάτι από τον s στον t και να τις αφαιρέσουμε, ώστε να βρούμε μετά το συντομότερο μονοπάτι χωρίς αυτές.
- Αν έχουμε αφαιρέσει όλες τις *ανεπιθύμητες* ακμές (για παράδειγμα, αν έχουμε θέσει το βάρος τους ίσο με ∞), τότε μπορούμε να τρέξουμε τον αλγόριθμο Dijkstra για να βρούμε το συντομότερο μονοπάτι στο τροποποιημένο γράφημα. Αν σε κάποια επανάληψη πριν φθάσουμε στο t, το $D[u] = min_{v \notin S}\{D[v]\} = \infty$, τότε προφανώς δεν υπάρχει κανένα άλλο μονοπάτι $s - t$ (κι επομένως ούτε συντομότερο) αφού το γράφημα δεν είναι πλέον συνεκτικό.
- Μένει να υλοποιήσουμε τον αλγόριθμο εύρεσης των *ανεπιθύμητων* ακμών.

Άσκηση 2

- Πρώτα, τρέχουμε τον αλγόριθμο Dijkstra στο γράφημά μας με αρχική κορυφή την s ώστε να υπολογίσουμε τις ελάχιστες αποστάσεις από αυτή την κορυφή προς όλες τις υπόλοιπες. Τώρα, όλες οι κορυφές είναι εξοπλισμένες με τις τελικές εκτιμήσεις $D[v] = d(s, v)$.
- Στο δεύτερο βήμα, τρέχουμε έναν τροποποιημένο BFS από το t: Ξεκινώντας από το t εξετάζουμε όλες τις εισερχόμενες σε αυτό ακμές. Ελέγχουμε για ποιές από αυτές ισχύει ότι $D[v] = D[t] + w(v, t)$. Αυτές προφανώς είναι *ανεπιθύμητες* ακμές αφού υπάρχει μονοπάτι $s - v$ το οποίο αν επεκταθεί με την ακμή (v, t) θα αποτελεί μονοπάτι $s - t$ με την ελάχιστη απόσταση. Προσθέτουμε τις κορυφές αυτές στην ουρά του τροποποιημένου BFS και τις σημειώνουμε υπό εξερεύνηση. Συνεχίζουμε με την επόμενη κορυφή της ουράς, διαγράφοντας μία κορυφή όταν εξερευνήσουμε τις εισερχόμενες ακμές της και ελέγχοντας αν μία κορυφή είναι ήδη υπό εξερεύνηση πριν μπει στην ουρά (όπως στον BFS).

Άσκηση 2

Η πολυπλοκότητα υπολογίζεται ως εξής:

- Εκτέλεση Dijkstra με αρχική κορυφή την s: $O(m + n \log n)$
- Υπολογισμός λιστών εισερχόμενων ακμών για κάθε κορυφή από τις λίστες γειτνίασης: $O(m)$
- Εκτέλεση τροποποιημένου BFS για εύρεση *ανεπιθύμητων* ακμών: $O(n + m)$
- Αφαίρεση *ανεπιθύμητων* ακμών: $O(m)$
- Εκτέλεση Dijkstra στο τροποποιημένο γράφημα για εύρεση συντομότερου μονοπατιού χωρίς *ανεπιθύμητες* ακμές: $O(m + n \log n)$

Άρα συνολικά: $O(m + n \log n)$

Άσκηση 3 - α

- Έστω ο αλγόριθμος Dijkstra όπως υπάρχει στις διαφάνειες του μαθήματος.
- Έχουμε απλό πίνακα D, n θέσεων, όπου υπάρχουν οι εκτιμήσεις των αποστάσεων κάθε κόμβου ανά πάσα στιγμή. Τροποποιούμε τον αλγόριθμο ώστε αντί να έχουμε ουρά προτεραιότητας, να έχουμε έναν πίνακα A (επιπλέον του D), μεγέθους nC. Κάθε στοιχείο του πίνακα A είναι μία λίστα κόμβων του γραφήματος.
- Για παράδειγμα αν υπάρχει στη θέση 4 του πίνακα A μία λίστα με τους αριθμούς 2 και 3, σημαίνει ότι οι κόμβοι 2 και 3 έχουν εκτίμηση απόστασης 4 (τη δεδομένη στιγμή) απ' τον αρχικό μας κόμβο. Διαφορετικά, αν υπάρχει NULL, σημαίνει ότι δεν υπάρχει κόμβος με εκτίμηση απόστασης 4 από τον αρχικό.
- Αρχικοποίηση: Ορίζουμε ένα δείκτη (index) του πίνακα A με όνομα *minidx* που αρχικά δείχνει στο $A[0]$. Στη θέση αυτή δημιουργούμε μία λίστα με μοναδικό στοιχείο (αρχικά) τον κόμβο s. Επίσης, υπάρχει λίστα όλων των υπολοίπων κόμβων στο $A[nC-1]$.

Άσκηση 3 - α

- Η εύρεση ελαχίστου γίνεται γραμμικά απ' τη θέση που δείχνει ο *minidx* και προχωρώντας τον, μέχρι να βρούμε μία μη κενή λίστα κόμβων στον A. Θέτουμε ως u έναν απ' τους κόμβους της λίστας αυτής και τον αφαιρούμε από τη λίστα. Στην επόμενη επανάληψη ο *minidx* ξεκινάει από εκεί που έμεινε. Αν κάπου ο *minidx* ξεπεράσει το στοιχείο $A[nC-1]$, τερματίζει ο αλγόριθμος.
- Στο στάδιο της αναπροσαρμογής μεταφέρουμε τον κόμβο απ'τη λίστα που αντιστοιχεί στην προηγούμενη εκτίμηση, στη λίστα με τη νέα εκτίμηση, στον πίνακα A. Συνολικά, $O(m)$.
- Για την εύρεση του ελαχίστου χρησιμοποιούμε το δείκτη *minidx* ο οποίος ποτέ δεν επιστρέφει πίσω γιατί ο Dijkstra εξετάζει σε όλο και μεγαλύτερες αποστάσεις. Έτσι, σε όλη τη διάρκεια εκτέλεσης του αλγορίθμου, ο *minidx* διατρέχει ολόκληρο τον πίνακα A ακριβώς μία φορά (δηλαδή nC βήματα).
- Έτσι, συνολική πολυπλοκότητα: $O(nC+m)$.

Άσκηση 3 - β

- Δεν αλλάζουμε καθόλου την υλοποίηση που υπάρχει στις διαφάνειες και χρησιμοποιούμε Δυαδικό Σωρό (Binary Heap).
- Ισχυριζόμαστε ότι η ουρά προτεραιότητας έχει μέγιστη διαφορά εκτιμώμενης απόστασης $d_{max} - d_{min} = 2^C$. Άρα η εύρεση μεγίστου στο σωρό και η αναδιοργάνωσή του δε χρειάζονται $\log n$ στη χειρότερη περίπτωση, αλλά

$$\log(d_{max} - d_{min}) = \log 2^C = C.$$

- Έτσι, η συνολική πολυπλοκότητα θα είναι $O((n+m)C)$.
- Αποδεικνύουμε τον ισχυρισμό με επαγωγή.

Άσκηση 3 - β

- **Βάση επαγωγής**: Στην πρώτη επανάληψη, μπαίνουν οι γείτονες του s, οι οποίοι έχουν προφανώς απόσταση απ' το s το πολύ 2^C.
- **Επαγωγική Υπόθεση**: Έστω ότι στην i-οστή επανάληψη έχουμε ότι $d^i_{max} - d^i_{min} \leq 2^C$.
- **Επαγωγικό Βήμα**:

$$d^{i+1}_{max} = max\{d^i_{min} + 2^C, d^i_{max}\}$$

$$d^{i+1}_{min} \leq d^i_{min}$$

Άρα, είτε ισχύει ότι

$$d^{i+1}_{max} - d^{i+1}_{min} \leq d^i_{min} + 2^C - d^i_{min} = 2^C$$

είτε ισχύει ότι

$$d^{i+1}_{max} - d^{i+1}_{min} \leq d^i_{max} - d^i_{min} \leq 2^C$$

Άσκηση 4 - α

Αλγόριθμος Για να εξηγήσουμε την ιδέα, θα θεωρήσουμε την περίπτωση $k = 1$ που λύνεται ως εξής:

- Απ' το γράφημα G, δημιουργούμε ένα αντίγραφο G' το οποίο περιέχει μόνο τις 0-ακμές του G, σε χρόνο $O(m)$.
- Απ' το γράφημα G, δημιουργούμε ένα αντίγραφο G'' στο οποίο κάθε 1-ακμή του G (u, v) αντικαθίσταται με άλλη 1-ακμή (u'', v'), όπου v' η αντίστοιχη κορυφή της v στο αντίγραφο G'. Χρειαζόμαστε χρόνο $O(m)$.
- Θεωρούμε το συνολικό γράφημα $\hat{G}$ με μοναδικό αντίγραφο της αρχικής κορυφής s.
- Για κάθε κορυφή u του G, υπολογίζουμε τις αποστάσεις (με Dijkstra) $s - u'$ και $s - u''$ στο νέο γράφημα $\hat{G}$. Η μικρότερη των δύο αποστάσεων είναι η ζητούμενη για την κορυφή u. Εφόσον το γράφημα είναι ασυμπτωτικά ίδιου μεγέθους με το αρχικό, χρειαζόμαστε χρόνο $O(m + n \log n)$.

Άσκηση 4 - α

Ορθότητα

- Αρκεί να δείξουμε ότι η συντομότερη διαδρομή (με το πολύ μία 1-ακμή) $s - u$ μπορεί να είναι είτε η $s - u'$, είτε η $s - u''$ στο γράφημα $\hat{G}$.
- Υπάρχουν δύο περιπτώσεις: να χρησιμοποιούμε ή να μη χρησιμοποιούμε 1-ακμή.
- Έστω ότι δεν έχουμε 1-ακμές στο συντομότερο μονοπάτι. Τότε η συντομότερη διαδρομή περιέχει μόνο ακμές του G'' γιατί το γράφημα δεν περιέχει 1-ακμές. Έτσι, η ελάχιστη διαδρομή είναι η $s - u''$ (Σημ: θα έχουμε ακόμα ένα μονοπάτι $s - u'$ ίδιου μήκους με ακμές μόνο απ' το G'.)
- Έστω ότι έχουμε ακριβώς μία 1-ακμή στο συντομότερο μονοπάτι: την (u_1, u_2). Αυτή η διαδρομή υπάρχει στο $\hat{G}$. Η διαδρομή $s - u_1''$ ανήκει στο G''. Μετά ακολουθούμε την ακμή (u_1'', u_2') και μεταφερόμαστε στο G'. Στο G' υπάρχει η διαδρομή $u_2' - u'$ που αποτελείται μόνο από 0-ακμές (και δε μπορεί να επιστρέψει στο G').

Άσκηση 4 - α

- Ο γενικευμένος αλγόριθμος περιλαμβάνει την κατασκευή του G' όπως πριν αλλά και την κατασκευή άλλων k αντιγράφων του G,

$$G^i, 1 \leq i \leq k$$

- Για κάθε κορυφή u στο αρχικό γράφημα G, συμβολίζουμε με u^i την αντίστοιχη κορυφή στο γράφημα G^i.
- Για $1 \leq i \leq k-1$, στη θέση κάθε 1-ακμής (u, v) του G, υπάρχει 1-ακμή (u^i, v^{i+1})
- Για $i = k$, η 1-ακμή (u, v) αντικαθίσταται με (u^k, v').
- Για τη δημιουργία του γραφήματος χρειαζόμαστε $O(k \cdot m)$. Για τον υπολογισμό των συντομότερων διαδρομών χρειαζόμαστε $O(km + kn \log(kn))$. Εφόσον $k < n$, η τελική χρονική πολυπλοκότητα είναι $O(mn + 2n^2 \log(n)) = O(n^3)$. Έτσι, ο αλγόριθμος είναι πολυωνυμικού χρόνου.

Άσκηση 4 - β

Γενικεύουμε το πρόβλημα ακόμα περισσότερο. Κάθε ακμή πλέον έχει κόστος $c(e) \in \mathbb{N}$.

- Δημιουργούμε γραφήματα G^i με $0 \leq i \leq C$. Κάθε τέτοιο γράφημα G^i έχει για κάθε u κορυφή του G, μία κορυφή (u, i). Χρόνος κατασκευής ως εδώ $O(Cn)$.
- Για κάθε ακμή $e = (u, v)$ του G, προσθέτουμε όλες τις ακμές $\Big((u, i), (v, i + c(e))\Big)$ με $0 \leq i \leq C - c(e)$. Χρόνος προσθήκης ακμών $O(Cm)$.
- Έτσι στο συνολικό αυτό γράφημα, κινούμενοι από μία κορυφή στην επόμενη, κρατάμε στην κατάστασή μας και το πόσο κόστος έχουμε ήδη συγκεντρώσει.

Άσκηση 4 - β

Όμοια με προηγουμένως, τρέχουμε τον αλγόριθμο Dijkstra στο γράφημα που κατασκευάσαμε (σε χρόνο $O(Cm + Cn\log(Cn))$) και για να βρούμε το συντομότερο $s - u$ μονοπάτι με κόστος το πολύ C, αναζητάμε τη συντομότερη από τις αποστάσεις $(s, 0) - (u, i)$ για κάθε $0 \leq i \leq C$ (σε χρόνο $O(C)$).
Η πολυπλοκότητα του αλγορίθμου είναι συνολικά

$$O(Cm + Cn\log(Cn))$$

αλλά αφού το C δε φράσσεται πλέον από παράμετρο του προβλήματος (παρά μόνο για παράδειγμα από το $nmax_{e \in E}\{c(e)\}$ το οποίο είναι και πάλι συνάρτηση του κόστους), ο αλγόριθμος θεωρείται ψευδοπολυωνυμικός και όχι πολυωνυμικός.

Άσκηση 5

Δεδομένα:

- n άνθρωποι
- k κατηγορίες
- m εταιρίες
- Κάθε άνθρωπος ανήκει σε μία ή περισσότερες κατηγορίες.

Περιορισμοί

- Για κάθε εταιρία i θέλουμε το πολύ c_i διαφημίσεις τη μέρα.
- Οι διαφημίσεις μίας εταιρίας i πρέπει να προβάλλονται σε ανθρώπους που να ανήκουν σε συγκεκριμένο υποσύνολο των κατηγοριών $S_i \subseteq \{1, \ldots, k\}$.
- Κάθε άνθρωπος πρέπει να βλέπει το πολύ μία διαφήμιση.

Άσκηση 5

Ζητούμενο:

- Υπάρχει τρόπος προβολής διαφημίσεων έτσι ώστε να τηρούνται οι περιορισμοί και κάθε άνθρωπος να βλέπει **ακριβώς** μία διαφήμιση;
- Αν ναι, βρες την ανάθεση.

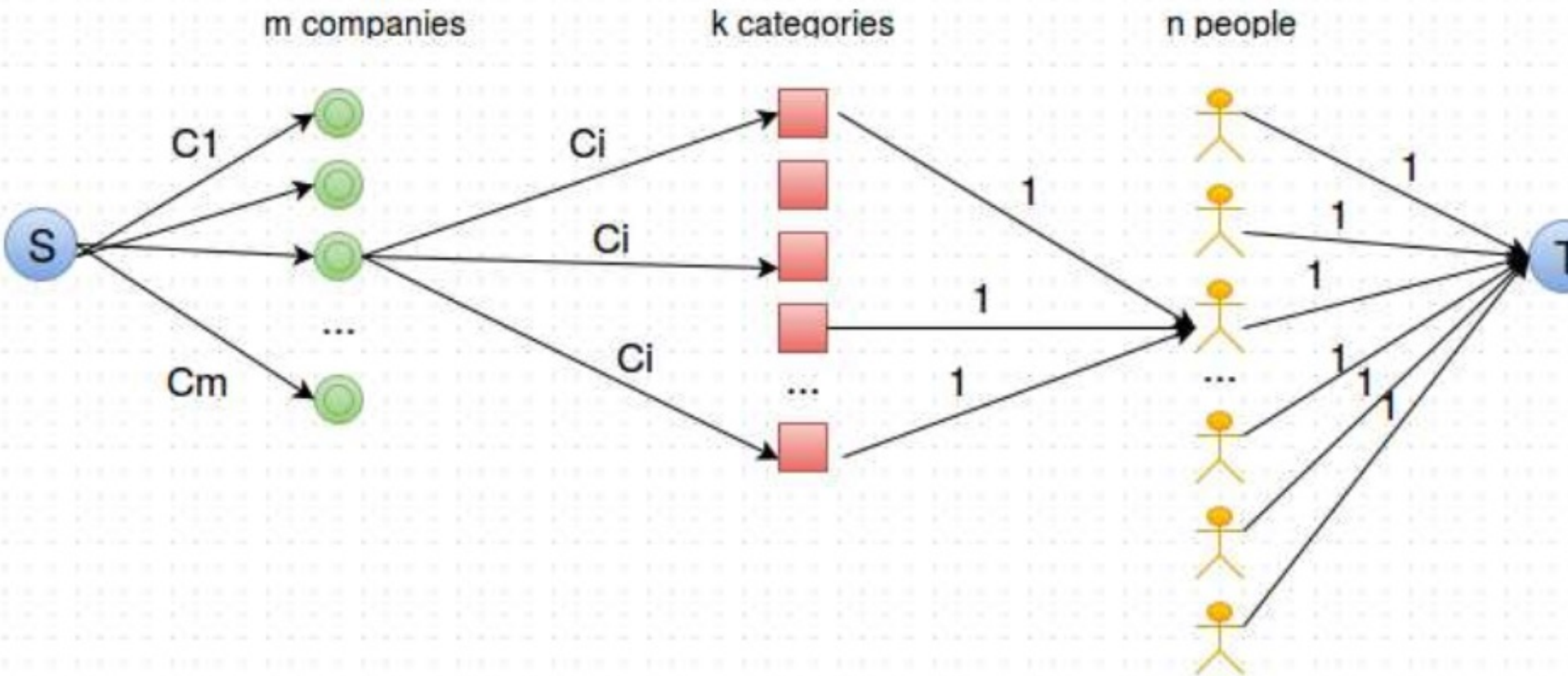
Λύση:

Flows!!!

Άσκηση 5

- Θεωρούμε μία κορυφή για κάθε άνθρωπο, κάθε κατηγορία και κάθε εταιρία.
- Συνολικά $n + k + m$ κορυφές.
- Προσθέτουμε επιπλέον μια κορυφή s για source καθώς και μία κορυφή t για sink.
- Συνδέουμε την κορυφή s με κάθε κορυφή-εταιρία i με ακμή χωρητικότητας c_i.
- Συνδέουμε με ακμή κάθε κορυφή-εταιρία i με κάθε κορυφή-κατηγορία που ανήκει στο S_i. Θέτουμε επίσης c_i χωρητικότητα σε κάθε τέτοια ακμή.
- Συνδέουμε κάθε κορυφή-άνρθωπο με κάθε κατηγορία στην οποία ανήκει με ακμές χωρητικότητας 1.
- Συνδέουμε κάθε κορυφή-άνθρωπο με την κορυφή t, με ακμές χωρητικότητας 1.

Άσκηση 5


m companies
k categories
n people
S
C1
Cm
Ci
Ci
Ci
1
1
1
1
1
1
1
1
1
1
1
T

Άσκηση 5

Αλγόριθμος: Κατασκεύασε το παραπάνω γράφημα και εκτέλεσε κάποιον αλγόριθμο για υπολογισμό $s - t$ Max Flow. Εάν η μέγιστη ροή είναι ίση με n (τον αριθμό των ανθρώπων) τότε απάντησε ΝΑΙ και επίστρεψε για κάθε εταιρία τους ανθρώπους που βλέπουν τις διαφημίσεις της. Διαφορετικά απάντησε ΟΧΙ.

Άσκηση 5

Ορθότητα:

- Ακέραιες χωρητικότητες $\rightarrow$ Ακέραιες ροές
- Από την κατασκευή του γραφήματος γνωρίζουμε ότι η μέγιστη ροή είναι ίση με n.
- Σε περίπτωση λοιπόν που υπολογίσουμε ροή n αυτό θα σημαίνει ότι κάθε άνθρωπος είδε ακριβώς μία διαφήμιση.
- Σε περίπτωση ΝΑΙ, ο υπολογισμός των ανθρώπων που βλέπουν τις διαφημίσεις κάθε εταιρίας μπορεί να γίνει εύκολα: Ξεκινάμε από μια εταιρία και κοιτάμε προς ποιές κατηγορίες έχει μη μηδενική ροή. Σε κάθε μια από αυτές τις κατηγορίες, κοιτάμε προς ποιούς ανθρώπους (που δεν έχουν "δεθεί" με κάποια εταιρία) υπάρχει ροή. Διαλέγουμε από αυτούς έναν αριθμό ίσο με τη ροή της εταιρίας προς την κατηγορία και τους "δένουμε" με την εταιρία. Επαναλαμβάνουμε για όλες τις εταιρίες.

Άσκηση 5

Πολυπλοκότητα:

- Η πολυπλοκότητα του αλγορίθμου εξαρτάται από την πολυπλοκότητα του αλγορίθμου υπολογισμού μέγιστης ροής που θα χρησιμοποιήσουμε, αφού σε κάθε περίπτωση αποτελεί το πιο "βαρύ" κομμάτι του υπολογισμού.
- Η πολυπλοκότητα της κατασκευής του γράφου καθώς και του υπολογισμού των ανθρώπων που είδαν διαφήμιση κάθε εταιρίας (εάν αυτός υλοποιηθεί με έξυπνο - σχετικά - τρόπο) είναι μικρότερη του υπολογισμού της ροής.

Άσκηση 6 - 3-Διαμέριση

3-PARTITION

- Είσοδος: Σύνολο $A = \{w_1, ..., w_n\}$ με n θετικούς ακεραίους και συνολικό άθροισμα στοιχείων $w(A)$ πολλαπλάσιο του 3.
- Ερώτηση: Υπάρχει διαμέριση του A σε τρία σύνολα ώστε $w(A_1) = w(A_2) = w(A_3)$;

Απόδειξη:

- Ανήκει στο NP: γιατί αν δοθούν τα σύνολα, επαληθεύουμε σε γραμμικό χρόνο ότι αποτελούν διαμέριση και ότι το άθροισμα των στοιχείων του καθενός είναι $w(A)/3$.
- Είναι NP-hard: Υπάρχει πολυωνυμική αναγωγή από το PARTITION.

Άσκηση 6 - 3-Διαμέριση

- Έστω $A = \{w_1, ..., w_n\}$ το σύνολο του γενικού στιγμιοτύπου του PARTITION.
- Θέτουμε $A' = \{w_1, ..., w_n, w' = \frac{w_1+...+w_n}{2}\}$ (κατασκευή σε γραμμικό χρόνο).
- Υπάρχει 2-διαμέριση του συνόλου A αν και μόνο αν υπάρχει 3-διαμέριση του συνόλου A': Κάθε 3-διαμέριση του A' θέτει το στοιχείο w' σε ένα σύνολο μόνο του, και τα υπόλοιπα δύο σύνολα αποτελούν διαμέριση των υπολοίπων στοιχείων με συνολικό βάρος $w(A)/2 = w'$ το καθένα. Αλλά τα δύο αυτά τελευταία σύνολα είναι μία 2-διαμέριση του A. Αντίστροφα, κάθε 2-διαμέριση του A προφανώς αρκεί για να υπάρχει μία 3-διαμέριση του A'.

Άσκηση 6 - Άθροισμα Υποσυνόλου κατά Προσέγγιση

APPROXIMATION SUBSET SUM (APSS)

- Είσοδος: Σύνολο $A = \{w_1, ..., w_n\}$ με n φυσικούς, και φυσικοί B και x με $B > x \geq 1$.
- Ερώτηση: Υπάρχει $S \subseteq A$ τέτοιο ώστε $B - x \leq w(S) \leq B$;

Απόδειξη:

- Ανήκει στο NP: γιατί αν δοθεί το σύνολο, επαληθεύουμε σε γραμμικό χρόνο ότι είναι λύση.
- Είναι NP-hard: Υπάρχει πολυωνυμική αναγωγή από το SUBSET-SUM

Άσκηση 6 - Άθροισμα Υποσυνόλου κατά Προσέγγιση

- Έστω $A_1 = \{w_1, ..., w_n\}$ το σύνολο του γενικού στιγμιοτύπου του SUBSET SUM και W η παράμετρός του.
- Θέτουμε $A = \{2w_1, ..., 2w_n\}$, $B = 2W$, $x = 1$.
- Το να βρεθεί σύνολο $w(A_1) = W$ είναι ισοδύναμο με το να βρεθεί σύνολο $w(A') = 2W$ στο ειδικό στιγμιότυπο του AP-SS.
- Υπάρχει περίπτωση να βρεθεί $w(A') = 2W - 1$ στο πρόβλημά μας; Όχι, γιατί αυτό θα ήταν περιττό ενώ το άθροισμά μας είναι πάντα άρτιο.

Άσκηση 6 - Κύκλος Hamilton κατά Προσέγγιση

APPROXIMATION HAMILTON CYCLE (APHC)

- Είσοδος: Μη κατευθυνόμενο γράφημα G.
- Ερώτηση: Υπάρχει κύκλος Hamilton στο G που να περνά τουλάχιστον μία και το πολύ δύο φορές από κάθε κορυφή του;

Απόδειξη:

- Ανήκει στο NP: γιατί αν μας δοθεί ο κύκλος, επαληθεύουμε σε γραμμικό χρόνο ότι είναι όντως κύκλος και ότι περιλαμβάνει 1-2 φορές κάθε κορυφή.
- Είναι NP-hard: Υπάρχει πολυωνυμική αναγωγή από το HAMILTON CYCLE.

Άσκηση 6 - Κύκλος Hamilton κατά Προσέγγιση

- Έστω G ένα γράφημα στο οποίο αναζητούμε κύκλο Hamilton.
- Κατασκευάζουμε σε πολυωνυμικό χρόνο το G' που αποτελείται από τις ίδιες κορυφές και ακμές με το G και για κάθε κορυφή u, του προσθέτουμε άλλη μία u' η οποία συνδέεται μόνο με τη u.
- Αν υπάρχει Hamilton κύκλος κατά προσέγγιση στο G', θα υπάρχει και Hamilton κύκλος στο G και το αντίστροφο: Για το ευθύ, παρατηρούμε ότι για να περάσουμε τουλάχιστον μία φορά από τις τονούμενες κορυφές, πρέπει να περάσουμε δύο φορές από τις αντίστοιχες του αρχικού μας γραφήματος (πχ $u \to u' \to u$). Μιας και δε μπορούμε να περάσουμε παραπάνω από δύο φορές από καμία κορυφή, ο κύκλος που σχηματίζεται αν αφαιρέσουμε τα "πήγαινε-έλα" στις τονούμενες κορυφές είναι ένας κύκλος στο G που περνά ακριβώς μία φορά από κάθε κορυφή του, δηλαδή Hamilton. Αντίστροφα, αν υπάρχει κύκλος Hamilton στο G, τότε προσθέτοντας ανάμεσα τις διαδρομές $u \to u' \to u$ θα είχαμε προφανώς Hamilton κύκλο κατά προσέγγιση στο G'.

Άσκηση 6 - Ικανοποιησιμότητα με Περιορισμούς

CONSTRAINED SAT(C-SAT)

- Είσοδος: Λογική πρόταση $\phi = \bigwedge_{j=1}^{m} (\ell_{j1} \vee \ell_{j2} \vee \ell_{j3} \vee \ell_{j4})$.
- Ερώτηση: Υπάρχει ανάθεση τιμών αλήθειας ώστε κάθε όρος $(\ell_{j1} \vee \ell_{j2} \vee \ell_{j3} \vee \ell_{j4})$ να περιέχει τουλάχιστον ένα αληθές κι ένα ψευδές literal;

Απόδειξη:

- Ανήκει στο *NP*: γιατί αν μας δοθεί η ανάθεση, επαληθεύουμε σε γραμμικό χρόνο ότι είναι έγκυρη.
- Είναι NP-hard: Υπάρχει πολυωνυμική αναγωγή από το 3-SAT.

Άσκηση 6 - Ικανοποιησιμότητα με Περιορισμούς

- Έστω $\phi = \bigwedge_{j=1}^{m} (\ell_{j1} \vee \ell_{j2} \vee \ell_{j3})$ μία 3-CNF πρόταση για την οποία θέλουμε να ελέγξουμε αν υπάρχει ανάθεση αληθοτιμών που την ικανοποιεί (δηλαδή κάθε όρος της περιλαμβάνει τουλάχιστον ένα αληθές literal).
- Κατασκευάζουμε σε πολυωνυμικό χρόνο τη λογική πρόταση $\phi' = \bigwedge_{j=1}^{m} (\ell_{j1} \vee \ell_{j2} \vee \ell_{j3} \vee z)$ όπου z καινούρια μεταβλητή που δεν εμφανίζεται στη ϕ.
- Αν υπάρχει ανάθεση που να ικανοποιεί τον περιορισμό της ϕ' τότε η ϕ είναι ικανοποιήσιμη και το αντίστροφο: Για το αντίστροφο, αν η ϕ είναι ικανοποιήσιμη, έχει τουλάχιστον ένα literal αληθές σε κάθε όρο άρα η ίδια ανάθεση, αν θέσουμε επιπλέον z =false ικανοποιεί τον περιορισμό μας για τη ϕ'. Για το ευθύ, αν έχουμε μία ανάθεση για τη ϕ' που ικανοποιεί τον περιορισμό τότε προφανώς αν z =false τότε η ίδια ανάθεση ικανοποιεί και την ϕ, διαφορετικά την ικανοποιεί η αντίθετή της (δηλ. όποια μεταβλητή έχει true κάντη false και το αντίθετο).

Άσκηση 6 - Επιλογή Ανεξάρτητων Υποσυνόλων

SET PACKING

- Είσοδος: Συλλογή $S = \{S_1, \ldots, S_m\}$ υποσυνόλων ενός συνόλου U με n στοιχεία και φυσικός αριθμός k με $2 \leq k \leq m$.
- Ερώτηση: Υπάρχουν k υποσύνολα στη συλλογή S που να είναι ανά δύο, ξένα μεταξύ τους;

Απόδειξη:

- Ανήκει στο NP: Ελέγχουμε τα υποσύνολα που πρέπει να ικανοποιούν το ζητούμενο σε πολυωνυμικό χρόνο.
- Είναι NP-hard: Υπάρχει πολυωνυμικό αναγωγή από το Max Independent Set (MIS).

Άσκηση 6 - Επιλογή Ανεξάρτητων Υποσυνόλων

- Για κάθε κόμβο του γραφήματος G, δημιουργούμε ένα σύνολο στο S με όλες τις προσπίπτουσες ακμές του κόμβου αυτού στο γράφημα. Η σταθερά k του Set Packing είναι ίση με την αντίστοιχη του MIS. Το U είναι το σύνολο των ακμών E.
- Έστω ότι έχουμε ανεξάρτητο σύνολο κορυφών με πληθάριθμο τουλάχιστον k. Κανένα ζεύγος κορυφών αυτού του συνόλου δε θα έχει κοινή ακμή. Άρα και κανένα ζεύγος των αντίστοιχων υποσυνόλων του Set Packing δε θα έχει κοινό στοιχείο. Δηλαδή, όλα τα υποσύνολα που θα δημιουργήσουμε θα είναι ξένα μεταξύ τους και τουλάχιστον k στο πλήθος.
- Αντίστροφα, έστω ότι έχουμε τουλάχιστον k ξένα υποσύνολα του U. Εφόσον το καθένα αντιστοιχεί σε κορυφή του αρχικού γραφήματος, θα έχουμε προφανώς στο G ίδιου πλήθους σύνολο κορυφών χωρίς κοινές ακμές, δηλαδή ένα ανεξάρτητο σύνολο με τον κατάλληλο πληθάριθμο.

Άσκηση 6 - Συντομότερο Μονοπάτι με Περιορισμούς

CONSTRAINT SHORTEST PATH

- Είσοδος: Κατευθυνόμενο γράφημα $G(V, E, w, c)$, όπου κάθε ακμή έχει ακέραιο μήκος $w(e) \geq 0$ και ακέραιο κόστος $c(e) \geq 0$, δύο κορυφές s, t και δύο ακέραιοι $W, C \geq 0$.
- Ερώτηση: Υπαρχει s-t μονοπάτι στο G με συνολικό μήκος μικρότερο ή ίσο του W και συνολικό κόστος μικρότερο ή ίσο του C;

Απόδειξη:

- Ανήκει στο NP: Σε γραμμικό χρόνο διασχύουμε το δοθέν s-t μονοπάτι και επιβεβαιώνουμε ότι πληροί τους περιορισμούς.
- Είναι NP-hard: Υπάρχει πολυωνυμικη αναγωγή από το Knapsack.

Άσκηση 6 - Συντομότερο Μονοπάτι με Περιορισμούς

Η αναγωγή γίνεται από το KNAPSACK.
Γενικό στιγμιότυπο: n στοιχεία, το i-οστό με βάρος w_i και κέρδος p_i.
Υπάρχει υποσύνολό τους με συνολικό βάρος μικρότερο ή ίσο του B και συνολικό κέρδος μεγαλύτερο ή ίσο του P;
Θα το μετασχηματίσουμε σε ένα ειδικό στιγμιότυπο του προβλήματός μας.

Άσκηση 6 - Συντομότερο Μονοπάτι με Περιορισμούς

Δημιουργούμε κατευθυνόμενο γράφημα με κορυφές $x_0, ..., x_n$ όπου x_0 η αρχική κορυφή και $x_1, ..., x_n$ αντιστοιχούν στα στοιχεία του σακιδίου. Υπάρχουν δύο κατευθυνόμενες ακμές από κάθε x_{i-1} στο x_i.
Διαισθητικά η μία ακμή αντιστοιχεί στην επιλογή του στοιχείου i για το σακίδιο με $w(e) = w_i$ και $c(e) = P_{sum} - p_i$. Η δεύτερη ακμή αντιστοιχεί στη μη επιλογή του στοιχείου i για το σακίδιο και έχει $w(e) = 0$ και $c(e) = P_{sum}$. Θέτουμε επίσης $a = x_0$, $b = x_n$, $W = B$ και $C = n \cdot P_{sum} - P$.
Βλέπουμε εύκολα πως η εύρεση ενός αποδεκτού σακιδίου είναι ισοδύναμη με την εύρεση ενός μονοπατιού από το x_0 στο x_n που πληροί τους περιορισμούς.