Αλγόριθμοι - Κανονική 2015

1) Σ/Λ

- Αν T(n) = 2T(n/2) + n(logn)^3, T(1)=Θ(1), τότε T(n) = Ω(n(logn)^3).

Σωστό γιατί είναι $\Theta(n\log^4 n)$ άρα και $\Omega(n\log^3 n)$

- Αν T(n) = T(n/2) + T(n/3) + n, T(1)=Θ(1), τότε T(n) = Θ(nlogn).

Λάθος Θα είναι $\Theta(n)$

- Αν T(n) = 3T(n/3) + nlogn, T(1)=Θ(1), τότε T(n) = O(nlogn).

Λάθος είναι Θ(n(logn)^2)

- Το max flow διατυπωμένο ως πρόβλημα απόφασης ανήκει στην κλάση NP.

Σωστό, το max flow επαληθεύεται σε γραμμικο χρονο

- Στο max flow αν αυξήσω τη χωρητικότητα 2 ακμών της ελάχιστης τομής κατά k την καθεμία αυξάνεται η μέγιστη ροή κατά 2k.

Λάθος (ευκολα αντιπαράδειγμα, αλλάζει η ελ.τομή με τις νέες χωρητικότητες άρα όχι απαραίτητα αύξηση της maxflow κατα 2k)

- Ο γρηγορότερος συγκριτικός αλγόριθμος για κατασκευή σωρού είναι $\mathrm{O}(n\log n)$.

Λάθος, $O(n)$ απο διαφάνειες

- Το DFS με n κορυφές και n^3/2 ακμές θέλει O(n^2) αν αναπαρίσταται με πίνακα γειτνίασης.

Σωστό, $O(n^2)$ Αν είχα ΛΙΣΤΑ γειτνίασης θα ήθελα O(κορυφες + ακμες) = $\mathrm{O}(n+n^{3/2})$ = $O(n^{3/2})$

- Στον Dijkstra αν διαλέγω κάθε φορά αντί για το μικρότερο το μεγαλύτερο βρίσκω longest paths.

Λάθος, παίζει αντιπαράδειγμα εύκολα οπου το άπληστο κρητίριο δέν δουλεύει.

- Στον Kruskal αν έχω ταξινομημένες τις ακμές σε φθίνουσα σειρά, βρίσκω το μέγιστο συνδετικό δέντρο.

Σωστό (το γκουγκλάραμε)

- Κατά την εκτέλεση του αλγορίθμου των Ford-Fulkerson είναι πιθανό σε κάποιο βήμα η ροή κάποιας ακμής να μειωθεί.

Σωστό

- Το πρόβλημα της μέγιστης ροής σε ένα δίκτυο, εκφρασμένο ως πρόβλημα απόφασης ανήκει στο NP.

Σωστό

- Έστω ένα πρόβλημα για το οποίο για κάθε στιγμιότυπο υπάρχει CNF λογική πρόταση φ (η οποία κατασκευάζεται σε πολυωνυμικό χρόνο), για την οποία το στιγμιότυπο είναι αποδεκτό αν και μόνο αν η φ είναι ικανοποιήσιμη. Τότε το πρόβλημα είναι NP-δύσκολο.

Σωστο ??

Σχόλια από Facebook:

Κατά την εκτέλεση του αλγορίθμου των Ford-Fulkerson είναι πιθανό σε κάποιο βήμα η ροή κάποιας ακμής να μειωθεί.

Αυτό είναι σωστό. Σε κάθε βήμα του Ford-Fulkerson η συνολική ροή αυξάνεται μέχρι να πετύχουμε το ολικό μέγιστο, αλλά ενδεχομένως σε κάποιο σημείο η ροή μιας ακμής να μειωθεί (ενδέχεται δηλαδή να αυξηθεί η ροή της αντίστροφής της στο residual network).

Έστω ένα πρόβλημα για το οποίο για κάθε στιγμιότυπο υπάρχει CNF λογική πρόταση φ (η οποία κατασκευάζεται σε πολυωνυμικό χρόνο), για την οποία το στιγμιότυπο είναι αποδεκτό αν και μόνο αν η φ είναι ικανοποιήσιμη. Τότε το πρόβλημα είναι NP-δύσκολο.

Λάθος, αυτό μας λέει ότι το πρόβλημα ανάγεται στο SAT. Όλα τα προβλήματα που ανήκουν στο NP (συνεπώς και στο P) ανάγονται σε όλα τα NP-Complete προβλήματα (συνεπώς και στο SAT), αυτό δεν σημαίνει όμως ότι είναι και τα ίδια NP-Complete/Hard. Για να ίσχυε αυτό θα έπρεπε το SAT να ανάγεται στο πρόβλημά μας.

---

2)

α)

Kruskal και Prim

β)

Αν υπάρχει r-περιορισμένο μονοπάτι μεταξύ s-t να δειχθεί ότι κάθε ΕΣΔ έχει r-περιορισμένο μονοπάτι μεταξύ s-t (όπου r-περιορισμένο εννοείται ότι κάθε ακμή του μονοπατιού έχει βάρος το πολύ r)

Λύση: Έστω ότι $\exists\, e : w(e) \leq r \in (s-t)$. Επίσης έστω $(s-u)-(u-v)-(v-t)$ το μονοπάτι $(s-t)$. Και έστω $\exists\, e' : w(e') \geq r \,\&\, e' \in MST \,\&\, e' \in (u-v)$ (e παράλληλη με την e'). Όμως αν αντικαταστήσουμε την e' με την e στο MST, θα έχουμε ένα MST, μικρότερου βάρους συνολικά. Άρα άτοπο η 2^η υπόθεση.

---

3)

Θέλουμε να πάμε ένα ταξίδι το οποίο θα διαρκέσει k >= 2 μέρες, κι έχουμε επιλέξει ένα σύνολο n > k ενδιάμεσων σταθμών με αποστάσεις d1, d2, ..., dn μεταξύ τους (η απόσταση του 1ου σταθμού από την αφετηρία είναι d1, η απόσταση του 2ου σταθμού από τον 1ο είναι d2, κοκ). Οι αποστάσεις είναι θετικοί ακέραιοι αριθμοί.

Θέλουμε να βρούμε αποδοτικό αλγόριθμο που να υπολογίζει ποιες στάσεις θα πρέπει να κάνουμε κάθε μέρα ώστε η μέγιστη απόσταση που θα διανύσουμε για κάθε μέρα να ελαχιστοποιείται. (Οι αποστάσεις μπορούσαν να διανύονται μόνο στο ακέραιο.)

---

4) Ακολουθία p(1), p(2), ..., p(n) και ζητείται

α)

Για κάθε διάστημα [x,y] το $P[x,y] = max(p(j) - p(i))$ με $x \leq i < j \leq y$

Λύση από εδώ.

```
public int solution(int[] A)
{
    int N = A.Length;
    if (N < 1) return 0;

    int max = MIN_INT;
    int result = A[N-1]-A[N-2];

    for(int i = N-2; i >= 0; --i)
    {
        if(A[i] > max)
            max = A[i];

        int tmpResult = max - A[i];
        if(tmpResult > result)
            result = tmpResult;
    }

    return result;
}
```

β)

Η βέλτιστη επιλογή το πολύ k μη επικαλυπτόμενων διαστημάτων $[s_i, b_i]$ έτσι ώστε να μεγιστοποιείται το άθροισμα των $p(b_i) - p(s_i)$.

(Η ακολουθία παρουσιαζόταν ως τιμές μιας μετοχής την εκάστοτε μέρα και το $p[j] - p[i]$ (χτύπημα) αντιστοιχεί στην αγορά της τάδε μετοχής την i-οστή μέρα και στην πώληση της τη j-οστή μέρα, οπότε ζητούταν η μεγιστοποίηση του συνολικού κέρδους με το πολύ k χτυπήματα.)

Λύση: DP - FTW.

$$cost(n, k) = \max\left\{cost(j, k-1) + \sum\left(dif\ from\ (a)\right)\right\}$$

---

5)

Δίνεται γράφος G(V, E, w) όπου κάθε ακμή έχει μοναδιαίο μήκος και κάποιο βάρος w(e) και ζητείται

α)

Για κάθε κόμβο u η ελαχίστου βάρους με το μικρότερο μήκος διαδρομή. (Δηλαδή αν δύο μονοπάτια είναι ισοβαρή, αυτό με τις λιγότερες ακμές.)

Λύση: Shortest-path tree of minimum total weight.

β)

Το ίδιο, αλλά μόνο για τα μονοπάτια περιττού μήκους. Δηλαδή από κάθε s-u μονοπάτι περιττού μήκους αυτό που έχει το ελάχιστο βάρος και σε περίπτωση ισοπαλίας αυτό με τις λιγότερες ακμές.

6)

Αν $P \neq NP$ να ελεγχθεί αν τα παρακάτω είναι NP-complete ή ανήκουν στο P: α) Δοθέντος γράφου G(V, E) αν υπάρχει υποσύνολο του V μεγέθους 2015 το οποίο να είναι ανεξάρτητο.

β) 3-partition

γ) Δοθέντος γράφου όπου οι ακμές έχουν αξία και βάρος, υπάρχει συνδετικό δέντρο με συνολικά αξία τουλάχιστον P και συνολικό βάρος το πολύ W;