A black background with a black square

Description automatically generated with medium confidenceΕθνικό Μετσόβιο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Εαρινό Εξάμηνο 2023-2024

ΑΛΓΟΡΙΘΜΟΙ ΚΑΙ ΠΟΛΥΠΛΟΚΟΤΗΤΑ

Λύσεις Επαναληπτικής 23 (ημιτελής)

Ιωάννης (Χουάν) Τσαντήλας

03120883

[Github-Repo](https://github.com/ntua-el20883/ece-ntua2020)

# Επαναληπτική 23

Θέμα 1

*(Δεν υπάρχει εκφώνηση, μόνο ότι ζήτησε εκτέλεση Dijkstra)*

Θέμα 2

Θεωρούμε n διαστήματα στην ευθεία των φυσικών αριθμών (έχουμε λοιπόν ότι και ). Θέλουμε να επιλέξουμε κάποια από αυτά n διαστήματα ώστε τα επιλεγμένα διαστήματα να μην επικαλύπτονται μεταξύ τους και να έχουν μέγιστο συνολικό μήκος. Το μήκος ενός διαστήματος είναι ίσο με .

**Παράδειγμα**: θεωρούμε n=5 διαστήματα . Κάποιες εφικτές λύσεις (δηλ. επιλογές διαστημάτων που δεν επικαλύπτονται μεταξύ τους) είναι οι και με συνολικό μήκος επιλεγμένων διαστημάτων 5, και η με συνολικό μήκος επιλεγμένων διαστημάτων 6. Η τελευταία συλλογή αποτελεί και την βέλτιστη λύση για το συγκεκριμένο στιγμιότυπο.

1. Να δείξετε, μέσω αντιπαραδείγματος, ότι ο άπληστος αλγόριθμος που επιλέγει, σε κάθε επανάληψη το διαθέσιμο διάστημα με μέγιστο μήκος δεν οδηγεί απαραίτητα στη βέλτιστη λύση. Να κάνετε το ίδιο για τον άπληστο αλγόριθμο που επιλέγει το διαθέσιμο διάστημα με ελάχιστο χρόνο ολοκλήρωσης .
2. Να διατυπώσετε αποδοτικό αλγόριθμο για το πρόβλημα της επιλογής μη επικαλυπτόμενων διαστημάτων με μέγιστο συνολικό μήκος. Να αιτιολογήσετε την ορθότητα και την υπολογιστική πολυπλοκότητα του αλγορίθμου σας. *Σημείωση: οι 8 από τις 12 μονάδες δίνονται για σωστό αλγόριθμο πολυωνυμικού χρόνου, οι 12 μονάδες δίνονται για σωστό αλγόριθμο με βέλτιστη πολυπλοκότητα.*

Λύση

**Ερώτημα 1**

Θα αποδείξουμε με αντιπαράδειγμα πως ο άπληστος αλγόριθμος που επιλέγει πάντα τα μεγαλύτερο διάστημα είναι λάθος. Εάν έχουμε τα διαστήματα: [0,42), [0,17) και [17,43), ο αλγόριθμος θα διάλεγε το [0,42), ενώ ο μεγαλύτερος δυνατός συνδυασμός είναι ο [0,17) και [17,43).

Επίσης, ο άπληστος αλγόριθμος που επιλέγει πάντα το διάστημα με μικρότερο fi είναι λάθος. Εάν έχουμε τα διαστήματα: [0,1), [1,17) και [16,42), ο αλγόριθμος θα διάλεγε τα [0,1) και [1,17), ενώ ο μεγαλύτερος δυνατός συνδυασμός είναι ο [0,1) και [16,42).

**Ερώτημα 2**

Ταξινομούμε τα διαστήματα σε αύξουσα σειρά του . Χωρίς βλάβη της γενικότητας, θεωρούμε πως και δημιουργούμε δύο πίνακες μεγέθους , τους και . Στο θα αποθηκεύουμε το μέγιστο συνολικό μήκος των διαστημάτων που τελειώνει το πολύ στην χρονική στιγμή , ενώ . Ξεκινούμε με το . Οι θέσεις έως και γεμίζουν με 0, ενώ και

Συνεχίζουμε με τα υπόλοιπα διαστήματα και έστω πως είμαστε στο . Γεμίζουμε τις θέσεις έως και με την ίδια τιμή της θέσης , αφού δεν υπάρχει κάποιο άλλο διάστημα που μπορεί να προστεθεί και να οδηγήσει σε μεγαλύτερο μήκος.

Στην θέση υπάρχουν δύο ενδεχόμενα, το διάστημα . να μπει ή να μην μπει στην ακολουθία. Αρκεί λοιπόν να συγκρίνουμε την τιμή – η οποία είναι το μέγιστο μήκος χωρίς το – με το μήκος , το οποίο είναι το άθροισμα του μήκους του και του μέγιστου δυνατού μήκους των διαστημάτων που τελειώνουν έως στο πολύ .

Θέτουμε , και εάν είναι το – δηλαδή επιλέξαμε το νέο διάστημα, θέτουμε . Εάν περισσότερα διαστήματα τελειώνουν στην ίδια θέση, τα ελέγχουμε με τη σειρά και κρατάμε την βέλτιστη επιλογή. Τελικά, το θα έχει το μέγιστο δυνατό μήκος.

Για να βρούμε ποια διαστήματα αποτελούν αυτό το μήκος, ξεκινούμε από το και προχωράμε προς την αρχή του πίνακα, ελέγχοντας πότε αλλάζει η τιμή του. Στην τελευταία θέση προτού αλλάξει η τιμή, βρίσκεται το τέλος του προηγούμενου διαστήματος, η αρχή του οποίου μπορεί να βρεθεί από τον πίνακα Β. Συνεχίζουμε ομοίως.

Για να ταξινομήσουμε τα διαστήματα (π.χ. με quicksort) χρειαζόμαστε επαναλήψεις. Για να βρούμε το συνολικό μήκος και τα σωστά διαστήματα χρειαζόμαστε επαναλήψεις αντίστοιχα, ενώ χρειαζόμαστε επιπλέον επαναλήψεις για να ελέγξουμε κάθε διάνυσμα πριν βάλουμε τις τιμές στον πίνακα Α. Συνολικά:

Θέμα 3

1. Θεωρούμε συνεκτικό κατευθυνόμενο γράφημα με n κορυφές, m ακμές, μήκος για κάθε ακμή , και αρχική κορυφή s. Για δεδομένη κορυφή , συμβολίζουμε με το μήκος του συντομότερου s-v μονοπατιού στο G που μπορεί να περιέχει **το πολύ μία ακμή αρνητικού βάρους**. Να διατυπώσετε αποδοτικό αλγόριθμο που υπολογίζει τα μήκη , για κάθε . Να αιτιολογήσετε την ορθότητα και την υπολογιστική πολυπλοκότητα του αλγορίθμου σας.
2. Θεωρούμε συνεκτικό κατευθυνόμενο γράφημα με n κορυφές, m ακμές, (μη αρνητικό) ακέραιο μήκος για κάθε ακμή , και αρχική κορυφή s.
   1. Έστω (σχετικά μικρός) φυσικός C τέτοιος ώστε για κάθε ακμή . Να περιγράψετε μια υλοποίηση του αλγορίθμου Dijkstra με χρόνο εκτέλεσης .
   2. Υποθέτουμε τώρα ότι για κάποιον (σχετικά μικρό) φυσικό C, για κάθε ακμή . Να περιγράψετε μια υλοποίηση του αλγορίθμου Dijkstra με χρόνο εκτέλεσης . Να αιτιολογήσετε την ορθότητα και την υπολογιστική πολυπλοκότητα της υλοποίησης που προτείνετε.

Λύση

**Ερώτημα 1**

Θέμα 4

1. Θεωρήστε το πρόβλημα **Subgraph Non-Isomorphism**: δίνονται γραφήματα και ζητείται να διαπιστωθεί εάν ισχύει ότι κανένα υπογραφήματα του δεν είναι ισομορφικό με το . Αποδείξτε ότι το **Subgraph Non-Isomorphism** είναι coNP-πλήρες (ως προς την αναγωγή πολυωνυμικού χρόνου ).
2. Θεωρήστε το πρόβλημα **Blue-Red Clique**: δίνεται ένα γράφημα του οποίου κάθε κορυφή είναι χρωματισμένη με μπλε ή κόκκινο χρώμα, και ακέραιος k>0, και ζητείται αν υπάρχει στο G κλίκα μεγέθους τουλάχιστον k στις κορυφές της οποίας να εμφανίζονται και τα δύο χρώματα. Αποδείξτε ότι το πρόβλημα **Blue-Red Clique** είναι NP-πλήρες (θεωρώντας γνωστό ότι το **Clique** είναι).

Θέμα 5

1. Θεωρούμε προβλήματα απόφασης. Ένας πιθανοτικός αλγόριθμος είναι (τύπου) *Las Vegas* αν υπολογίζει πάντα τη σωστή απάντηση, αλλά ο χρόνος εκτέλεσης του Τ(n) είναι τυχαία μεταβλητή (π.χ. quicksort). Ένας πιθανοτικός αλγόριθμος είναι (τύπου) *Monte Carlo* αν ο χρόνος εκτέλεσης του φράσσεται ντετερμινιστικά, αλλά υπάρχει πιθανότητα λάθος απάντησης (π.χ. πιθανοτικός αλγόριθμος Min-Cut). Έστω Las Vegas αλγόριθμος Α για πρόβλημα απόφασης Π με αναμενόμενο χρόνο εκτέλεσης . Με βάση τον Α, να διατυπώσετε αλγόριθμο Monte Carlo (για το ίδιο πρόβλημα Π) με χρόνο εκτέλεσης το πολύ και πιθανότητα λάθους το πολύ .
2. Θυμηθείτε τον 2-προσεγγιστικό αλγόριθμο για το **TSP** που είδαμε στο μάθημα, ο οποίος ξεκινάει βρίσκοντας ένα ελάχιστο συνδετικό δέντρο Τ και στη συνέχεια διπλασιάζει τις ακμές του Τ. Συμπληρώστε (συνοπτικά) την περιγραφή του αλγορίθμου και δώστε μια οικογένεια γραφημάτων για την οποία η λύση που περιγράφει ο αλγόριθμος αυτός να είναι όσο γίνεται πιο κοντά στο διπλάσιο της βέλτιστης. Εξηγήστε την απάντηση σας.
3. Περιγράψτε τα βήματα του αλγορίθμου Kruth-Morris-Patt στο πρόβλημα του ταιριάσματος συμβολοσειρών (string-matching). Εφαρμόστε τον αλγόριθμο στον εντοπισμό του pattern «ababaca» στο text «bacbabababacaca». Περιγράψτε μία τροποποίηση του αλγορίθμου ταιριάσματος συμβολοσειρών Kruth-Morris-Patt στην οποία το pattern μπορεί να περιέχει οποιοδήποτε αριθμό 3 διαδοχικών ίδιων συμβόλων μπαλαντέρ =, καθένα από τα οποία ταιριάζει με τους ίδιους αυθαίρετους μεμονωμένους χαρακτήρες. Για παράδειγμα, το pattern ===HOC===SPOC===S εμφανίζεται στο κείμενο WHBR**UUU**HOC**UUU**SPOC**UUU**SOT (μεμονωμένοι χαρακτήρες **UUU**) και στο κείμενο ABR**AAA**HOC**AAA**SPOC**AAA**SCADAABRA (μεμονωμένοι χαρακτήρες **AAA**) αλλά όχι στο κείμενο FRISABAHOCAAASPOCAABSTIX. Υπολογίστε την πολυπλοκότητα του αλγορίθμου σας σε σχέση με το και το, όπου το μήκος του και το μήκος του κειμένου.