034 - Ανάλυση και Σχεδιασμός Αλγορίθμων
Προαιρετική Εργασία

Διδάσκων: Επίκ. Καθ. Παναγιώτης Πετραντωνάκης

Ομάδα 54
Ιωάννης Δημουλιός 10641
Χριστόφορος Μαρινόπουλος ΑΕΜ

Εαρινό εξάμηνο 2024

Πρόβλημα 1

Ερώτημα 1

Θέλουμε να δούμε αν είναι εφικτό ένα δρομολόγιο από την πόλη s στην πόλη t δίχως να χρησιμοποιήσουμε ακμές e με μήκος $l_e > L$.
Επομένως, φτιάχνουμε ένα νέο γράφο $G' = (V, E')$, ο οποίος διαφέρει από τον αρχικό μόνο στις ακμές. Το νέο σύνολο ακμών E' δεν έχει τις ακμές που αναφέρονται παραπάνω, δηλαδή:

$$E' = \{e \in E \mid l_e \leq L\}$$

. Στον γράφο G' τρέχουμε τον αλγόριθμο DFS ξεκινόντας από την κορυφή s και ελέγχουμε έτσι αν υπάρχει μονοπάτι μέχρι την κορυφή t, που είναι και το ζητούμενο. Η χρονική πολυπλοκότητα του αλγορίθμου είναι $O(|V| + |E|)$, αφού προσπελαύνουμε κάθε κορυφή και κάθε ακμή του γράφου G μια φορά. ■

Ερώτημα 2

Θέλουμε να βρούμε το ελάχιστο μήκος του μακρύτερου δρόμου που πρέπει να διασχίσουμε από την πόλη s στην πόλη t. Αυτό το μήκος θα είναι και το ζητούμενο L.
Επομένως θα χρησιμοποιήσουμε τον αλγόριθμο Dijkstra ελαφρώς τροποποιημένο, ώστε το βάρος της κάθε κορυφής v να μην είναι πλέον το ελάχιστο άθροισμα των μηκών των ακμών ενός μονοπατιού από την κορυφή s έως την v, αλλά το ελάχιστο των μέγιστων μηκών των ακμών αυτών των μονοπατιών.
Για να γίνει πιο ξεκάθαρο, αν P το σύνολο των μονοπατιών από την κορυφή s μέχρι την v και, $p \in P$ ένα τέτοιο μονοπάτι, τότε ορίζουμε

$$f(p) = \max_{e \in p} l_e$$

και αν $w(\cdot)$ η συνάρτηση βάρους μια κορυφής, τότε

$$w(v) = \min_{p \in P} f(p)$$

. Ο αλγόριθμος, λοιπόν, βρίσκει και το ζητούμενο $w(t)$.
Το μόνο που χρειάζεται να αλλάξουμε στην υλοποίηση σε σύγκριση με τον κλασικό Dijkstra είναι η συνάρτηση "χαλάρωσης", ώστε να αντικατοπτριστεί η παραπάνω αλλαγή.
Η χρονική πολυπλοκότητα του Dijkstra, εφόσον η ουρά προτεραιότητας που απαιτείται υλοποιηθεί με δυαδικό σωρό (binary heap), είναι $O((|V| + |E|) \log |V|)$. ■

Πρόβλημα 2

Πρόβλημα 3

Ορίζουμε ότι οι δείκτες πινάκων και συμβολοσειρών ξεκινούν από το 0 και υποθέτουμε ότι $i+1$-οστό στοιχείο του πίνακα X είναι το x_i.

Δεν μας νοιάζει η συμβολοσειρά που θα χωρίσουμε αυτούσια, όσο το πλήθος των χαρακτήρων της, έστω n.
Υποθέτουμε ότι οι τομές θα γίνουν στους δείκτες λ_i και προσθέτουμε στον πίνακα των τομών το -1 στην αρχή και το $n-1$ στο τέλος, ώστε η αρχή και το τέλος της συμβολοσειράς να εμφανίζονται ως τομές.
Οπότε ο πίνακας των τομών γράφεται

$$B = (-1, \lambda_1, \ldots, \lambda_m, n-1)$$

. Έστω

$$S = (s_0, \ldots, s_{n-1})$$

η αρχική συμβολοσειρά.
Ορίζουμε

$$S_{i,j} = (s_{b_i+1}, \ldots s_{b_j})$$

με $0 \leq i < j \leq m+1$. Τότε $S = S_{0,m+1}$.
Με το πέρας της διαδικασίας θέλουμε να έχουμε τις $m+1$ συμβολοσειρές

$$S_{0,1}, S_{1,2}, \ldots S_{m,m+1}$$

. Ορίζουμε τον $(m+2) \times (m+2)$ πίνακα D του οποίου το στοιχείο $d_{i,j}$ αναπαριστά τις ελάχιστες μονάδες χρόνου που απαιτούνται, ώστε με χρήση των τομών να λάβουμε από την $S_{i,j}$ τις

$$S_{i,i+1}, \ldots, S_{j-1,j}$$

Σκοπός του προβλήματος, λοιπόν είναι να υπολογίσουμε το $d_{0,m+1}$.
Εφόσον, στην $S_{i,i+1}$ δεν μπορούμε να κάνουμε άλλη τομή έχουμε

$$d_{i,i+1} = 0 \quad \text{για} \quad 0 \leq i \leq m$$

. Τώρα όμως για διαφορά δεικτών $j-i > 1$ προκύπτει

$$\begin{aligned} d_{i,j} &= \min_{i<k<j} \left((d_{i,k} + b_k - b_i) + (d_{k,j} + b_j - b_k) \right) \\ &= \min_{i<k<j} \left(d_{i,k} + d_{k,j} + b_j - b_i \right) \end{aligned}$$

. Υπολογίζουμε, έτσι, με τη χρήση δυναμικού προγραμματισμού τον πίνακα D ξεκινώντας από την κύρια διαγώνιό του, προσπελαύνοντας τις διαγωνίους σταθερής διαφοράς δεικτών, έως ότου φτάσουμε στο στοιχείο $d_{0,m+1}$.

Η χρονική πολυπλοκότητα του αλγορίθμου προκύπτει ως εξής:

- Προσπελαύνουμε τα μισά (όλα πάνω από την κύρια διαγώνιο) στοιχεία του πίνακα D διαστάσεων $(m+2) \times (m+2)$, άρα $O(m^2)$.
- Για τον υπολογισμό κάθε $d_{i,j}$ έχουμε έναν βρόχο με το πολύ m επαναλήψεις, άρα $O(m)$.

Επομένως, τελικά είναι $O(m^3)$. ■