Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών
Αλγόριθμοι και Πολυπλοκότητα
Διδάσκοντες: Αριστείδης Παγουρτζής, Δώρα Σουλίου, Ευστάθιος Ζάχος, Δημήτρης Σακαβάλας

Λύσεις 2ης Σειράς Προγραμματιστικών Ασκήσεων

Άσκηση 1: Διαδίδοντας την γνώση

Για την λύση αυτού τουθ προβλήματος, θα χρησιμοποιήσουμε δυναμικό προγραμματισμό. Οι μεταβλητές που θα χρησιμοποιήσουμε είναι ο αριθμός i των χωριών που έχουμε επισκεφτεί και μέχρι πόσες αγοραπωλησίες k μπορούμε να κάνουμε. Από την θεωρία του δυναμικού προγραμματισμού, ξέρουμε ότι η καλύτερη πολυπλοκότητα που μπορούμε να πετύχουμε είναι $O(NK)$, αφού τόσο μεγάλος είναι το πεδίο ορισμού του προβλήματος. Ο αναδρομικός τύπος που προκύπτει είναι:

$$dp[i,k] = \max_{j<i}(dp[j,k-1] - A_j + A_i, 0) \quad \forall i,k \tag{1}$$

Η τελική λύση που ζητάμε είναι το $dp[N,K]$Αυτή η λύση έχει πολυπλοκότητα $O(N^2K)$. Σίγουρα μπορούμε και καλύτερα. Παρατηρούμε, ότι αφού ψάχνουμε για το μέγιστο, μπορούμε αντί για το max, να ανζητήσουμε το $argmax$. Άρα αυτό που μας ενδιαφέρει σε κάθε σημείο, μας μένει:

$$arg\max_j dp[j,k-1] - A_j + A_i = arg\max_j dp[j,k-1] - A_j \tag{2}$$

Η παραπάνω αποτελεί μια παράσταση, την οποία μπορούμε να υπολογίσουμε απευθείας αφού υπολογίσουμε την τιμή κάθε καινούργιου $dp[i,k]$. Άρα πλέον έχουμε την επιθυμητή πολυπλοκότητα $O(NK)$. Το μόνο που απαιτείται, είναι να κρατάμε μια μεταβλητή με την ελάχιστη τιμή $dp[j,k-1] - A_j$. Μετά τον υπολογισμό κάθε καινούργιου στοιχείου, αρκεί να πάρουμε υπόψιν μας, το καινούργιο στοιχείο της προηγούμενης στήλης που πλέον μας είναι διαθέσιμο.

Άσκηση 2: Κύριος Κρίοζοτ

Συμβολίζουμε με $C[i]$ τοο μέγιστο δυνατό συντελεστή MIAM χρησιμοποιώντας μόνο τα πιάτα $1, 2, ..., i$. Για τον υπολογισμό του πίνακα C, φιξάρουμε για κάθε i κάθε δυνατό σπάσιμο $(j, j+1, ..., i)\forall 0 < j < i$και υπολογίζουμε την λύση επιλέγοντας αυτό που μας μεγιστοποιεί τον συντελεστή MIAM. Έχουμε λοιπόν:

$$C[i] = \max_{0<j<i}\{C[j-1] + a(\sum_{k=j}^{i} x_k)^2 + b(\sum_{k=j}^{i} x_k) + c\} \tag{3}$$

Αν χρησιμοποιήσουμε τον πίνακα μερικών αθροισμάτων S με $S[i] = \sum_{k=1}^{i} x_i$ η σχέση μετασχηματίζεται σε:

$$C[i] = \max_{0<j<i}\{C[j-1] + a(S[i] - S[j-1])^2 + b(S[i] - S[j-1]) + c\} \tag{4}$$

Η απάντηση δίνεται από τον όρο $C[N]$, η συνολική πολυπλοκότητα είναι $\theta(N^2)$

Bonus : Για να ρίξουμε την πολυπλοκότητα σε γραμμική ή $\Theta(NlogN)$, χρησιμοποιούμε το Convex-Hull Optimization. Μετά από πράξεις η αναδρομική σχέση του παραπάνω ερωτήματος γίνεται:

$$C[i] = aS^2[i] + bS[i] + c + \max_{0<j<i}\{(C[j] + aS^2[j] - bS[j]) - 2aS[j]S[i]\} \tag{5}$$

Όπου οι όροι που έχουν βγει εκτός παρένθεσης είναι όροι που εξαρτώνται μόνο από το i και συνεπώς δεν παίζουν ρόλο στην μεγιστοποίηση της συνάρτησης.

Αν θέσουμε τώρα $a = -2aS[j]$, $b = C[j] + aS^2[j] - bS[j]$, $y = C[i]$ και $x = S[i]$ παρατηρούμε πως η συνάρτηση είναι της μορφής $y = ax + b$, όπου οι συντελεστές a, b αλλάζουν ανάλογα με το ποιό j θα επιλέξουμε. Συνεπώς ενδιαφερόμαστε για την ευθεία που μεγιστοποιεί το y στην θέση $x = S[i]$. Προκειμένου να το κάνουμε αυτό γρήγορα χρησιμοποιούμε μια δομή γνωστή ως Convex-Hull Trick ή Convex-Hull Optimization .

Με χρήση του εν λόγω optimisation μπορούμε αρχικά να ρίξουμε την πολυπλοκότητα σε $\Theta(NlogN)$ και μετά, κάνοντας χρήση της μονοτονίας των x σε γραμμικό χρόνο. Για την περιγραφή της δομής, ανατρέξτε στα References του πρώτου σχέδιου λύσεων.