Maximum Sub-array problem

Brute force approach

Ο πρώτος αλγόριθμος βρίσκει όλα τα πιθανά sub-arrays, υπολογίζει τα αθροίσματα τους και επιστρέφει το μεγαλύτερο. Είναι ο πιο αργός με πολυπλοκότητα $O(n^3)$, αφού έχει τρία nested loops.

Improved solution

Ο δεύτερος αλγόριθμος δεν υπολογίζει εκ νέου το άθροισμα ενός sub-array όταν αυξάνεται ο δείκτης του τέλους, αλλά ανανεώνει το ήδη υπολογισμένο άθροισμα προσθέτοντας μόνο το επόμενο στοιχείο. Έτσι η πολυπλοκότητα μειώνεται σε $O(n^2)$

Divide and Conquer algorithm

Ο τρίτος αλγόριθμος διαιρεί σε υποπροβλήματα το αρχικό. Χωρίζει στη μέση τον πίνακα και επιστρέφει το μέγιστο από τα εξής: Μέγιστο άθροισμα από το αριστερό μισό (υπολογίζεται αναδρομικά), μέγιστο άθροισμα από το δεξί μισό (υπολογίζεται αναδρομικά), μέγιστο άθροισμα τέτοιο ώστε ο υποπίνακας να διασχίζει το μεσαίο σημείο. Η πολυπλοκότητα μειώνεται σε O(nlogn).

Optimal Solution Kadane's Algorithm

Με αυτόν τον αλγόριθμο διατρέχουμε μία φορα τον πίνακα και σε κάθε βήμα ενημερώνουμε τις μεταβλητές current_sum και maximum_sum και κρατάμε το μέγιστο των δύο. Αν το current_sum γίνει αρνητικό, το μηδενίζουμε και το sub-array ξεκινά από το επόμενο στοιχείο. Έχουμε την ελάχιστη δυνατή πολυπλοκότητα O(n).

Αποτελέσματα

Παρακάτω παρουσιάζονται τα αποτελέσματα από τα τρεξίματα του κάθε αλγορίθμου για δύο πίνακες διαφορετικού μήκους με τυχαίους αριθμούς από

(-100 έως 100) ώστε ο χρόνος εκτέλεσης να είναι της τάξης των δευτερολέπτων. Παρατηρούμε ότι όσο μειώνεται η πολυπλοκότητα το μέγιστο άθροισμα του sub-array υπολογίζεται για όλο και μεγαλύτερο μέγεθος πίνακα σε λιγότερο χρόνο.

Complexity	Length of array n	Time (sec)
$O(n^3)$	1,000	6.20
$O(n^3)$	1,500	21.44
$O(n^2)$	15,000	5.48
$O(n^2)$	30,000	22.21
O(nlogn)	1,000,000	2.06
O(nlogn)	10,000,000	24.49
O(n)	10,000,000	1.43
O(n)	100,000,000	24.39

Table 1: Length of array vs Running Time

Παρακάτω παρουσιάζονται τα αποτελέσματα μετά από δοκιμές που δίνουν το μέγιστο μέγεθος προβλήματος για κάθε αλγόριθμο του οποίου η λύση υπολογίζεται σε 3 seconds περίπου. Σε κάθε τρέξιμο είναι αδύνατο να έχουμε τις ίδιες ακριβώς μετρήσεις γιατί τα στοιχεία του πίνακα παράγονται τυχαία κάθε φορά και ο υπολογιστής διαχειρίζεται και άλλες διεργασίες παράλληλα με το πρόγραμμα.

Complexity	Length of array n	Time (sec)
$O(n^3)$	800	3.06
$O(n^2)$	11,000	2.989
O(nlogn)	2,100,000	3.006
O(n)	20,500,000	3.008

Table 2: Maximum array length for fixed running time 3 seconds

Για μία τελευταία σύγκριση των αλγορίθμων δείχνουμε το χρόνο στον οποίο επιλύουν το πρόβλημα για τον ίδιο τυχαίο πίνακα μήκους n=2500

Complexity	Time (sec)
$O(n^3)$	119.0651
$O(n^2)$	0.1579
O(nlogn)	0.0041
O(n)	0.0009

Table 3: Different running times for the same array