PROJECT: “DATA STRUCTURES 2019”

Συμμετέχοντες:

Νίκος Μπούσης ΑΜ: 1062661

Γιάννης Χριστοδουλάκος ΑΜ: 1062664

Γλώσσα που χρησιμοποιήθηκε για τη γραφή τον προγραμμάτων: C

# PART I: “Sorting and Searching Algorithms”

# (1)

Χρησιμοποιώντας το ίδιο data set, για τους τρεις αλγόριθμους, **Insertion Sort**, **Selection Sort** και **Bubble Sort** παρατηρούμε πως έντονες διαφορές φαίνονται κυρίως μεταξύ του τελευταίου με τους άλλους δύο. Όπως φαίνεται και στο παρακάτω γράφημα για την ταξινόμηση πίνακα 1.000 στοιχείων οι χρόνοι εκτέλεσης των δύο πρώτων αλγορίθμων πλησιάζουν τα 0,002 sec ενώ η διαφορά τους με τον τελευταίο ανέρχεται στα 0,007 sec. Σημαντική αύξηση στους χρόνους παρατηρούμε για ταξινόμηση πινάκων με πάνω από 30.000 στοιχεία όπου εκεί ξεπερνάμε ακόμα και το 1 sec με τον τελευταίο και πιο αργό αλγόριθμο να φτάνει τα 6 sec. Τέλος είναι δύσκολο να μετρήσουμε τους χρόνους ταξινόμησης σε πίνακες με λιγότερα από 500 στοιχεία καθώς οι αλγόριθμοι εκτελούνται σχεδόν ακαριαία.

# (2)

Για τους αλγόριθμους **Merge Sort** και **Quick Sort** χρησιμοποιόντας το ίδιο data set δεν παρατηρούμε μεγάλη διαφορά ως πρός τον χρόνο εκτέλεσης. Για παράδειγμα , σε Data set με μέγεθος < 1000 στοιχεία, παρατηρούμε ότι οι χρόνοι εκτέλεσης παραμένουν ίδιοι και σχεδόν μηδενικοί . Ωστόσο αν αυξήσουμε το μέγεθος στα 10000 νούμερα αρχίζουμε να παρατηρούμε μια μικρή διαφορά, της τάξεως του 1 msec (0.001) . Από ότι βλέπουμε, με μια πρώτη ματία μπορούμε να θεωρήσουμε ότι δεν είναι σημαντική η συγκεκριμένη διαφορά . Παρόλα αυτά αν έχουμε να κάνουμε με πιο «βαριά» Data ή πολύ περισσότερα νούμερα και τρέξουμε και τους δύο αλγόριθμους, θα διαπιστώσουμε σημαντική ανομοιομορφία στους χρόνους εκτέλεσης, των δύο αλγορίθμων. Παρακάτω δίνεται το γράφημα με τους χρόνους :

΄

Αντίθετα με τα παραπάνω δείγματα, γνωρίζουμε από την θεωρία ότι η **Quick Sort** για ανομοιγενή και πολλά δειγματα παρουσιάζει αστάθεια, με τον χρόνο της χειρότερης περίπτωσης να φτάνει σε n2. Τέλος, η **Merge Sort** με μεγαλύτερη ευστάθεια και αυστηρα όρια της τάξεως Θ(nlogn ), θεωρείται ένας από τους πιο γρήγορους και αξιόπιστους αλγόριθμους ταξινόμησης .

# (3)

Η ταξινόμηση Heap Sort είναι ένας από τους πιο γνωστους αλγόριθμους . Παρουσιάζει μεγάλη ευστάθεια στους χρόνους της, με αυστηρό ασυμπτωτικό όριο Θ(n logn) . Συγκρίνοντας τους χρόνους με τις υπόλοιπες ταξινομήσεις είναι εύκολο να επιβεβαιώθεί ο παραπάνω ισχυρισμός . Πιο αναλυτικα στο παρακάτω γράφημα τα παρουσιάζονται αποτελέσματα των χρόνων εκτέλεσης αλγορίθμων .

Με λίγα λόγια επαληθεύουμε την θεώρια και ταξινομούμε τους αλγόριθμους με βάση τους χρόνους εκτέλεσης κατά φθίνουσα σειρά, με τους τρεις τελευταίους να έχουν ως χρόνο εκτέλεσης λιγότερο από 0,01sec:

* Bubble Sort
* Selection Sort
* Insertion Sort
* Heap Sort
* Merge Sort
* Quick Sort

# (4)

Χρησιμοποιώντας το ίδιο Data Set για τους αλγόριθμους αναζήτησης παρατηρήσαμε πως μικρότερο χρόνο εκτέλεσης έχει η **Δυαδική αναζήτηση** στη συνέχεια ακολουθεί η **Αναζήτηση με Παρεμβολή** και τέλος έρχεται η **Γραμμική αναζήτηση.** Βέβαια σημαντικό ρόλο παίζει η κατανομή των στοιχείων στον πίνακα καθώς ο χρόνος εκτέλεσης τους εξαρτάται και από το κατά πόσο πυκνά είναι τα δεδομένα γύρω από το στοιχείο που ψάχνουμε γι αυτό το λόγο για τη μέση περίπτωση θα θεωρούμε πως παρουσιάζεται ομοιόμορφη κατανομή στα στοιχεία του πίνακα.

# (5)

Για τον αλγόριθμο **Δυικής Αναζήτησης Παρεμβολής (BIS)** παρατηρούμε πως μπορεί να γίνει βελτίωση της χειρότερης περίπτωσης αλλάζοντας το βήμα του αλγορίθμου από i+1 σε 2\*i. Κατά αυτόν τον τρόπο γίνονται μεγαλύτερα άλματα κρατώντας το αριστερό άκρο σταθερό με αποτέλεσμα το τελευταίο υποδιάστημα να είναι πολύ πιο μεγάλο από √n. Μόλις βρεθεί αυτό το υποδιάστημα εφαρμόζουμε τότε αναζήτηση στα στοιχεία μέσα σε αυτό που απέχουν κατά √n και έτσι προκύπτει το ζητούμενο υποδιάστημα μεγέθους √n.