Παράλληλα και Διανεμημένα Συστήματα

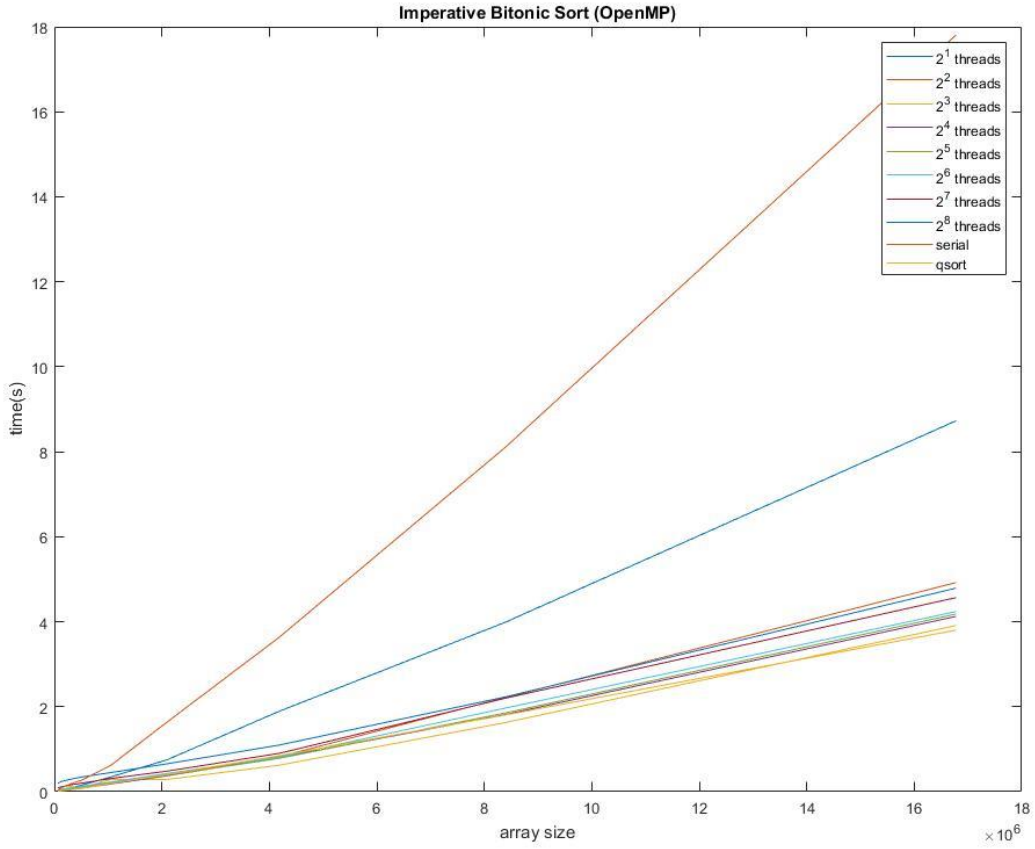
1η Εργασία 2017

Γανωτάκης Στέφανος ΑΕΜ:7664 sganotak@auth.gr

Θεοδωρίδης Ιορδάνης ΑΕΜ:6922 ttiordan@auth.gr

1.Υλοποίηση σε OpenMP

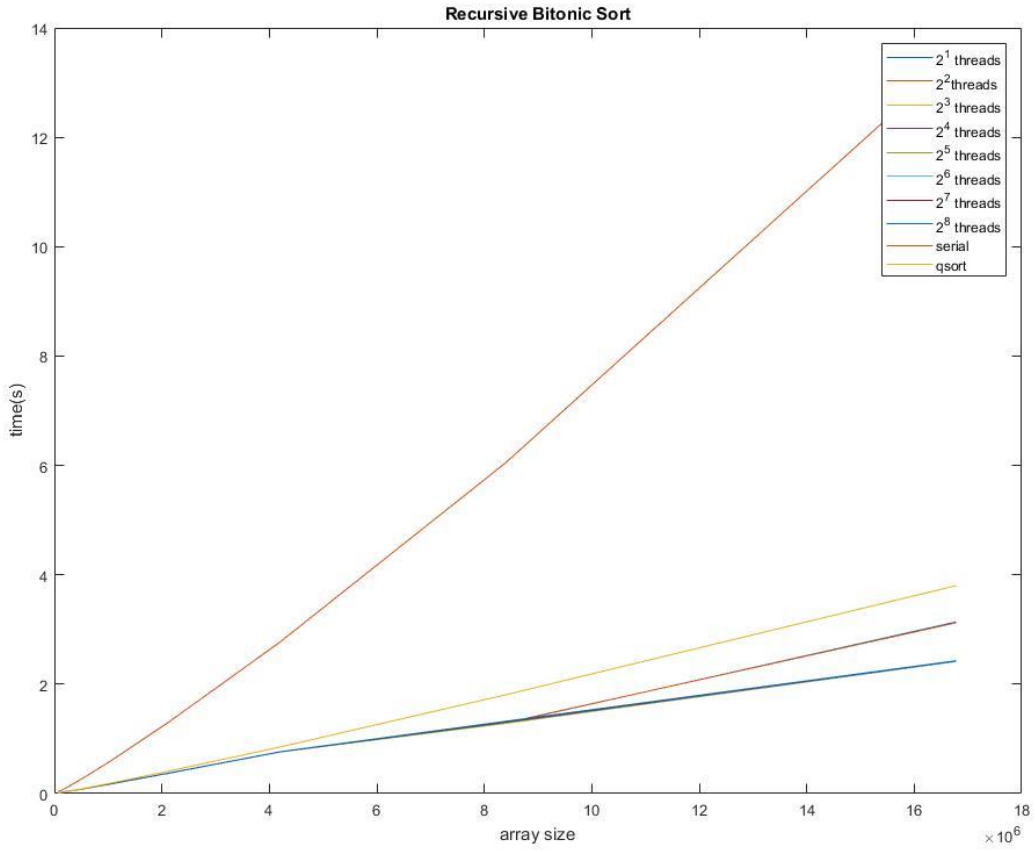
Για την Imperative εκδοχή του αλγορίθμου επιλέξαμε να κάνουμε παραλληλοποίηση του πιο εσωτερικού for loop της συνάρτηση ImpBitonicSort. Τα αποτελέσματα που πείραμε στο σύστημα diades για p=[1:8] και q=[16:24] σε σύγκριση με την σειριακή έκδοση του αλγορίθμου και την qsort παρουσιάζονται στο παρακάτω διάγραμμα



Η παράλληλη υλοποίηση είναι πάντα πιο γρήγορη από την αντίστοιχη σειριακή αλλά όχι και από την qsort. Καλύτερη απόδοση έχουμε για 2^3 threads.

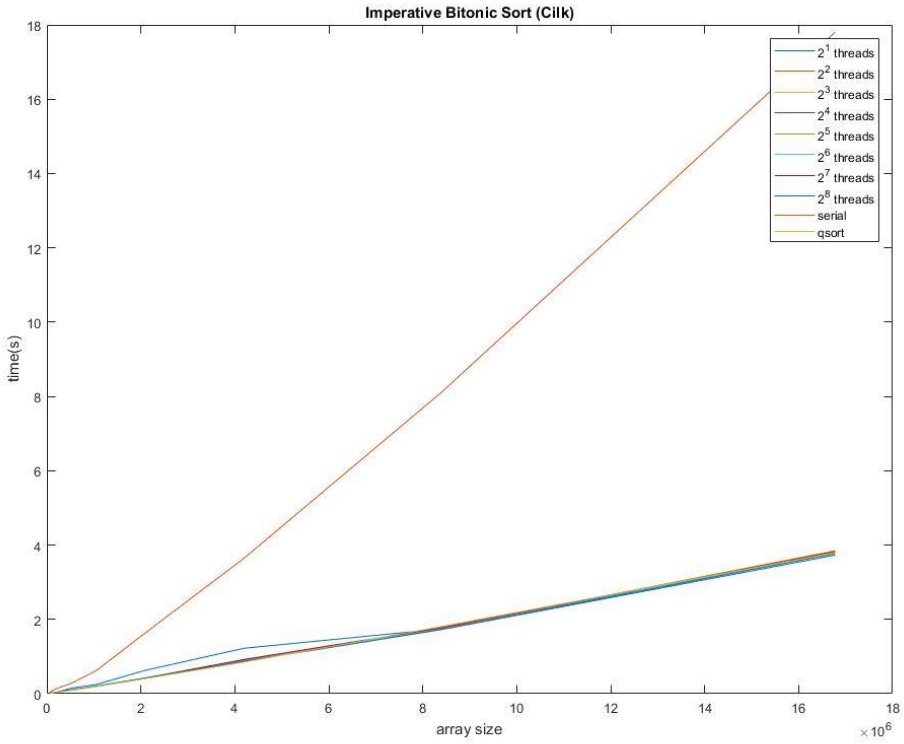
Για την Recursive εκδοχή του αλγορίθμου κάναμε παραλληλοποίση της συνάρτησης RecBitonicSort. Για να πετύχουμε καλύτερους χρόνους εκτέλεσης, κάναμε συνδυασμένη χρήση της qsort για την ταξινόμηση μικρού μεγέθους υποπινάκων. Τα αποτελέσματα που πείραμε στο σύστημα diades για p=[1:8] και q=[16:24] σε σύγκριση με την σειριακή έκδοση του αλγορίθμου και την qsort παρουσιάζονται στο παρακάτω διάγραμμα

Η συγκεκριμένη υλοποίηση είναι πιο γρήγορη απ την σειριακή αλλά και απ την qsort, καλύτερη απόδοση έχουμε για 2^5 threads



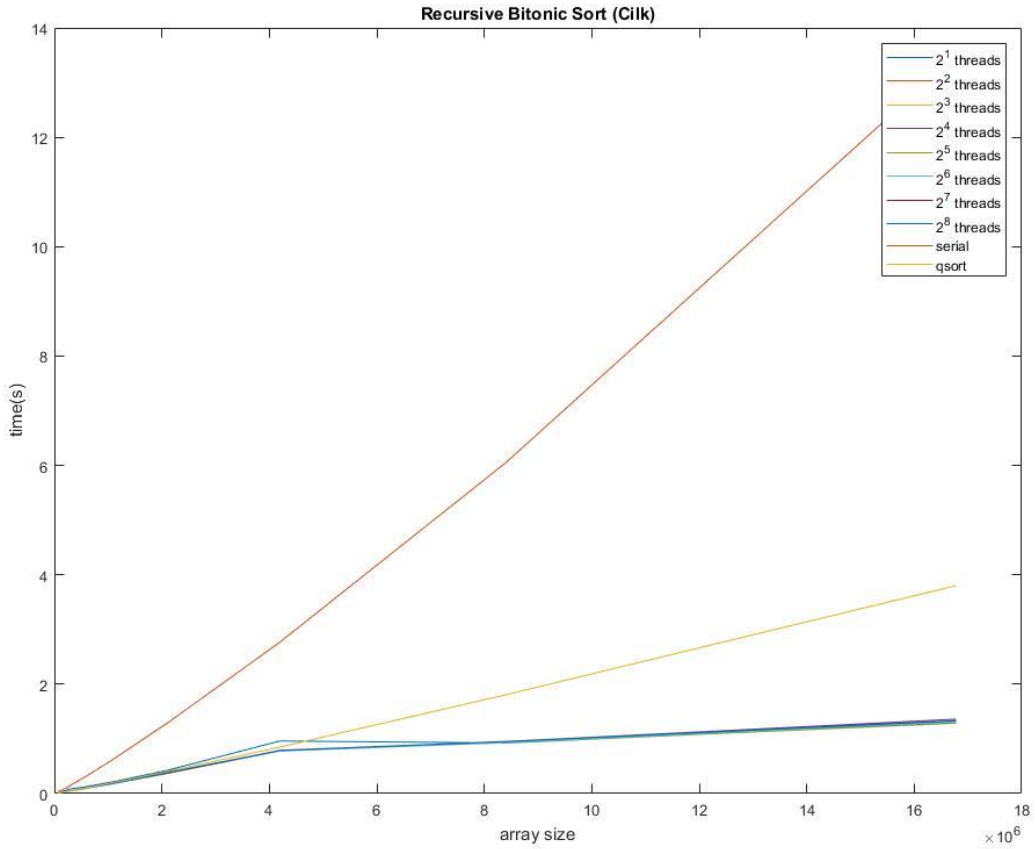
2.Υλοποίηση σε CilkPlus

Για την Imperative εκδοχή του αλγορίθμου επιλέξαμε να κάνουμε παραλληλοποίηση του πιο εσωτερικού for loop της συνάρτηση ImpBitonicSort. Τα αποτελέσματα που πείραμε στο σύστημα diades για p=[1:8] και q=[16:24] σε σύγκριση με την σειριακή έκδοση του αλγορίθμου και την qsort παρουσιάζονται στο παρακάτω διάγραμμα



Η συγκεκριμένη υλοποίηση είναι αρκετά πιο γρήγορη απ την σειριακή και σχεδόν ισοδύναμη με την qsort. Επιπλέον έχουμε περίπου ίδια απόδοση για τις διαφορετικές τιμές των threads.

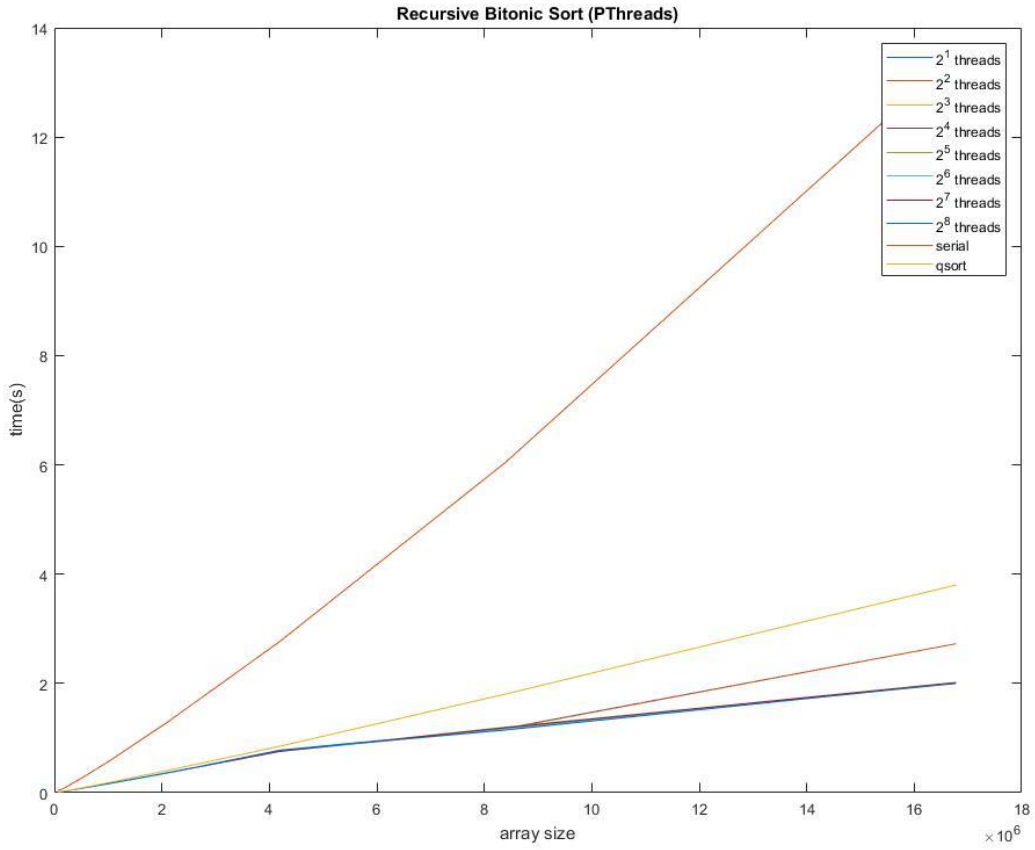
Για την Recursive εκδοχή του αλγορίθμου κάναμε παραλληλοποίση των αναδρομικών κλήσεων στις συναρτήσεις RecBitonicSort και BitonicMege. Και για τις δυο συναρτήσεις θέσαμε ένα ελάχιστο όριο μεγεθους πίνακα για το οποίο ο κώδικας θα τρέχει παράλληλα, έαν οι υποπίνακες είναι μικρότεροι από αυτόν τον αριθμό, χρησιμοποιήσαμε την qsort για την ταξινόμηση. Τα αποτελέσματα που πείραμε στο σύστημα diades για p=[1:8] και q=[16:24] σε σύγκριση με την σειριακή έκδοση του αλγορίθμου και την qsort παρουσιάζονται στο παρακάτω διάγραμμα



Η συγκεκριμένη υλοποίηση είναι αρκετά πιο γρήγορη απ την σειριακή και για μέγεθος πίνακα >4.65*10^6 είναι ταχύτερη και από την qsort. Επιπλέον έχουμε περίπου ίδια απόδοση για τις διαφορετικές τιμές των threads.

3.Υλοποίηση σε PThreads

Επιλέξαμε να παραλληλοποιήσουμε την recursive έκδοση του προγράμματος. Εφαρμόσαμε την ίδια μέθοδο με την υλοποίση σε Cilk, κάνοντας παραλληλοποίηση της BitonicMerge και της RecBitonicSort. Τα αποτελέσματα που πείραμε στο σύστημα diades για p=[1:8] και q=[16:24] σε σύγκριση με την σειριακή έκδοση του αλγορίθμου και την qsort παρουσιάζονται στο παρακάτω διάγραμμα



Η συγκεκριμένη υλοποίηση είναι πιο γρήγορη απ την σειριακή αλλά και απ την qsort, καλύτερη απόδοση έχουμε για 2^6 threads