**Παράλληλη Επεξεργασία**

**Τμήμα Μηχανικών Η/Υ & Πληροφορικής**

**Πανεπιστήμιο Πατρών**

**2021-2022**



**Ομάδα:**

**Ονοματεπώνυμο:** Χατζή Βασιλική

**Α.Μ.:**1064359

**Ονοματεπώνυμο:** Πιερή Χρυσοβαλάντης

**Α.Μ.:** 1056652

**Ονοματεπώνυμο:** Χαριλάου Εφραίμ

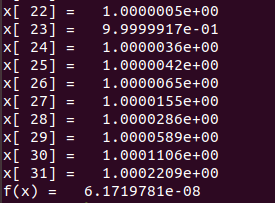
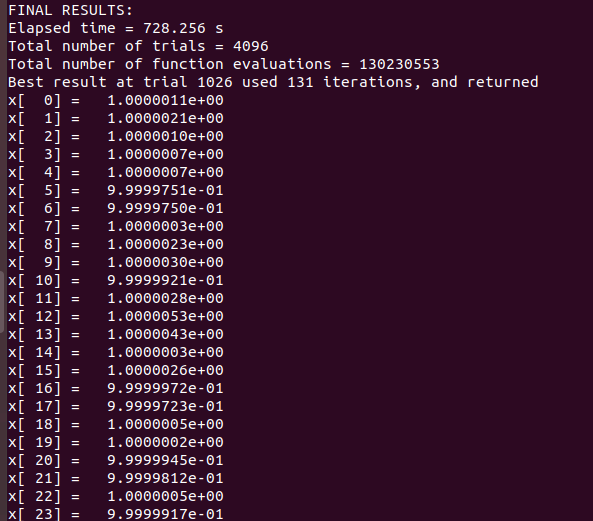
**Α.Μ.:** 1056638

**Ονοματεπώνυμο:** Δημητρίου Χρίστος

**Α.Μ.:** 1056643

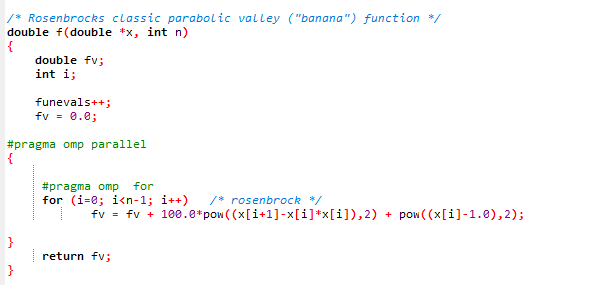
**🖂Email επικοινονίας:** **up1064359@upnet.gr**

ΧΡΌΝΟΣ ΠΡΟΓΡΑΜΜΑΤΟΣ ΧΩΡΙΣ ΠΑΡΑΛΛΗΛΟΠΟΙΗΣΗ:

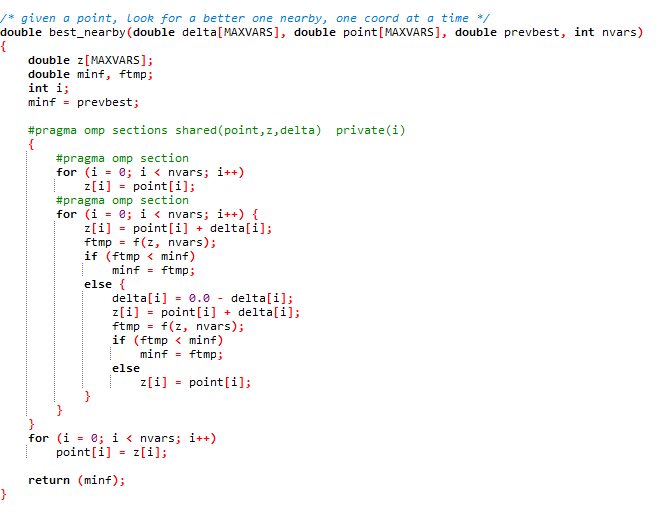


**# Παραλληλοποίηση Openmp:**

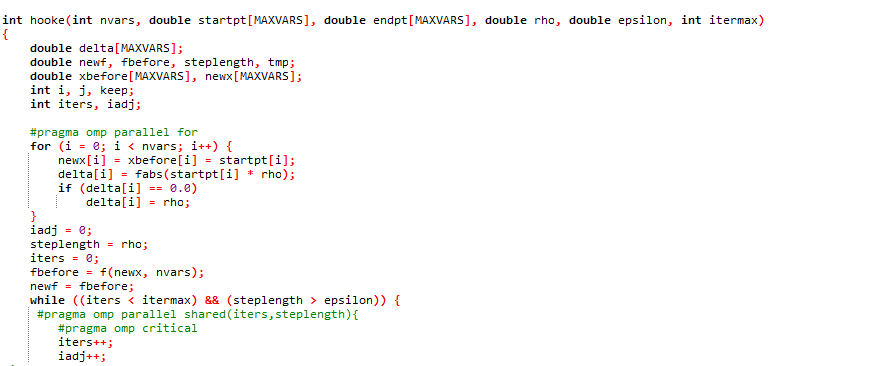
***Στιγμιότυπα από τον κώδικα μας***



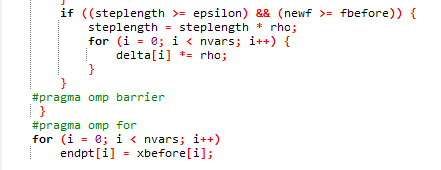
Για να αρχίσουμε την παραλληλοποιήση του κώδικα μας χρησιμοποιήσαμε την εντολή **#pragma omp parallel.** Στην συνέχεια για να θέσουμε σε λειτουργία παραλληλισμού τα **for** μας πρέπει να εισάγουμε την εντολή **#pragma omp for.**



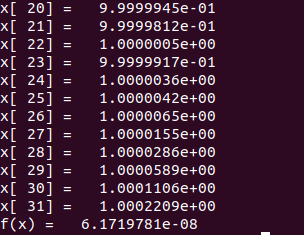
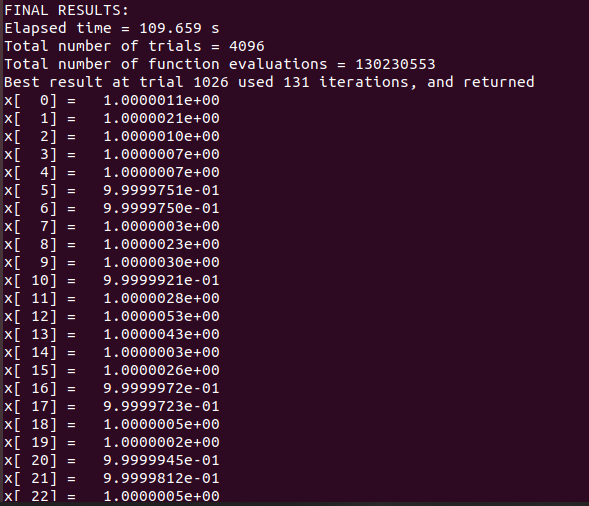
Για να αρχίσουμε την παραλληλοποιήση του κώδικα μας χρησιμοποιήσαμε την εντολή **#pragma omp sections shared(point,z,delta)** όπου σε αυτή την εντολή καθορίζουμε ότι τα ***loop*** μας θα έχουν τη (δική) τους τοπική μεταβλητή αλλά και άλλες κοινές μεταβλητές τιμές. Στην συνέχεια για να θέσουμε σε λειτουργία παραλληλισμού τα **for** μας πρέπει να εισάγουμε την εντολή **#pragma omp section** για να εκτελεστούν ξεχωριστά τα 2 for loop μας αφού δεν εξαρτάται η εκτέλεση τους ενός από το άλλο.

******

Το **pragma omp critical**  το θέσαμε επειδή όταν υπάρχουν μεταβλητές που αυξάνονται πρέπει να εκτελεστούν από 1 νήμα κάθε φορά.

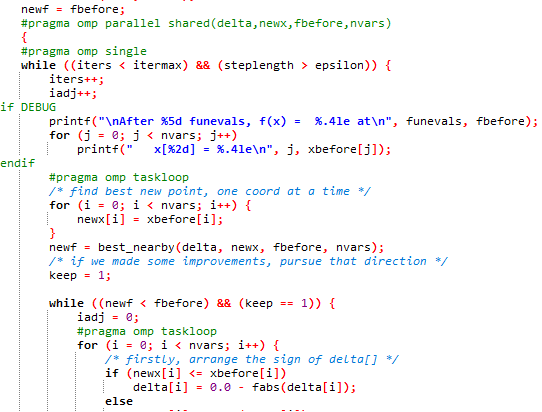


Το **pragma omp barrier** το βάλαμε για τη μεταβλητή **xbefore[i]** που χρησιμοποιείται στο ακολουθώ for loop. Με το **pragma omp barrier** πρέπει να τελειώσουν όλα τα νήματα για να συνεχίσει το πρόγραμμα μας.

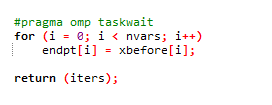


**# Παραλληλοποίηση Openmp\_Tasks:**

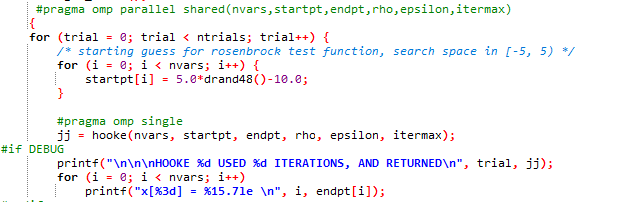
***Στιγμιότυπα από τον κώδικα μας***

****

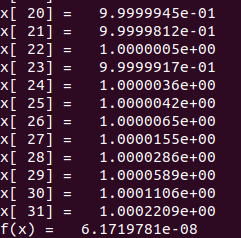
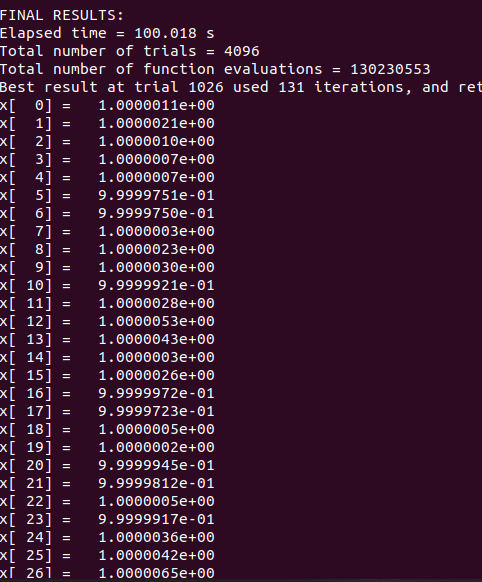
Για να αρχίσουμε την παραλληλοποίηση του κώδικα μας χρησιμοποιήσαμε την εντολή **#pragma omp parallel shared(delta, newx, fbefore, navrs)** όπου σε αυτή την εντολή καθορίζουμε ότι τα ***loop*** μας θα έχουν κάποιες κοινές μεταβλητές τιμές. Το **#pragma omp single** δηλώνει μια περιοχή που πρέπει να εκτελεστή μόνο από 1 νήμα**.** Στησυνέχεια προσθέσαμε το  **#pragma omp taskloop** για να εκτελεστούν παράλληλα τα for loop μας.



Το **pragma omp taskwait** το βάλαμε για τη μεταβλητή **xbefore[i]** που χρησιμοποιείται στο ακολουθώ for loop. Με το **pragma omp taskwait** πρέπει να τελειώσουν όλες οι εργασίες για να συνεχίσει το πρόγραμμα μας.

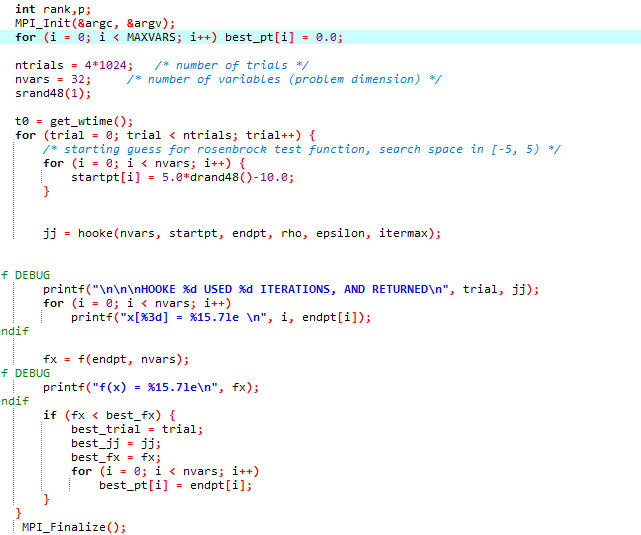


Για να αρχίσουμε την παραλληλοποίηση του κώδικα μας χρησιμοποιήσαμε την εντολή **#pragma omp parallel shared(navrs, startpt, endpt, rho, epsilon, itermax)** όπου σε αυτή την εντολή καθορίζουμε ότι τα ***loop*** μας θα έχουν κάποιες κοινές μεταβλητές τιμές. Το **#pragma omp single** δηλώνει μια περιοχή που πρέπει να εκτελεστή μόνο από 1 νήμα**.**

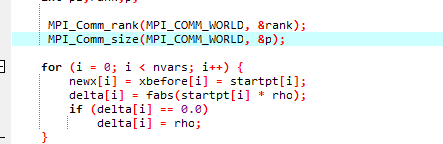


**# Παραλληλοποίηση Mpi:**

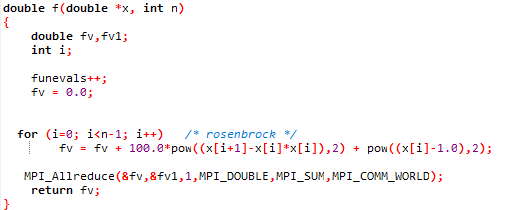
***Στιγμιότυπα από τον κώδικα μας***

******

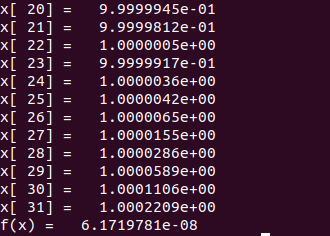
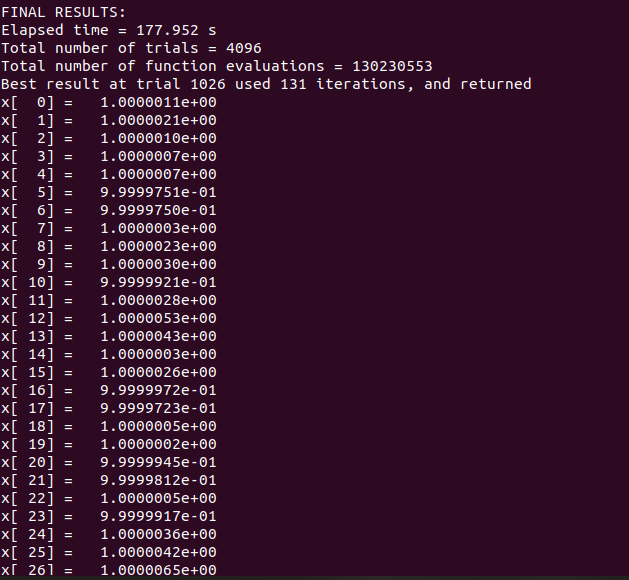
Για να αρχίσουμε την παραλληλοποίηση πρώτα βάλαμε την εντολή MPI\_Init για να θέσουμε από θα ξεκινήσει το παράλληλο πρόγραμμα.



Έπειτα θέσαμε τις εντολές MPI\_Comm\_rank και MPI\_Comm\_size για να προσδιορίσουμε το αναγνωριστικό της διεργασίας μας και τον αριθμό τον τους.

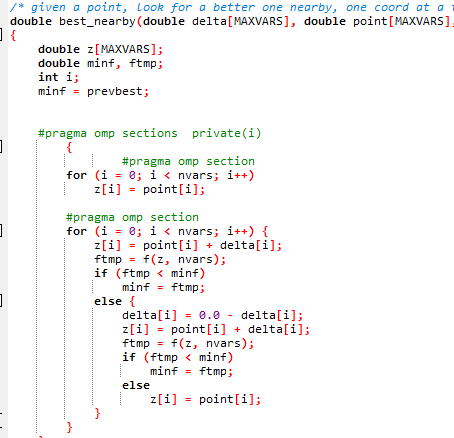


Τέλος προσθέσαμε την εντολή MPI\_Allreduce για να εκτελούνται παράλληλα οι πράξεις μας.

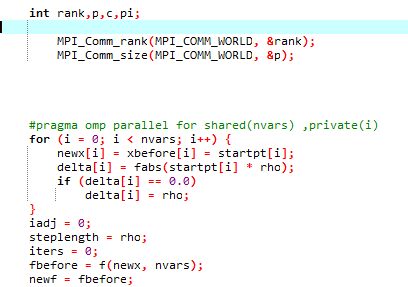


**# Παραλληλοποίηση Mpi-Openmp:**

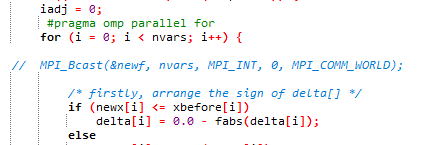
***Στιγμιότυπα από τον κώδικα μας***



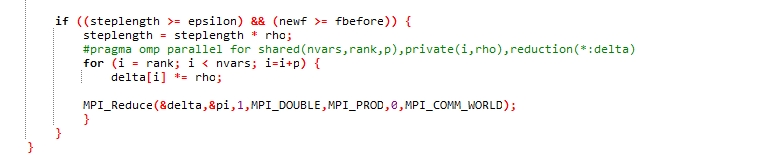
Για να αρχίσουμε την παραλληλοποιήση του κώδικα μας χρησιμοποιήσαμε την εντολή **#pragma omp sections private(i)** όπου σε αυτή την εντολή καθορίζουμε ότι τα ***loop*** μας θα έχουν τη (δική) τους τοπική μεταβλητή . Στην συνέχεια για να θέσουμε σε λειτουργία παραλληλισμού τα **for** μας πρέπει να εισάγουμε την εντολή **#pragma omp section** για να εκτελεστούν ξεχωριστά τα 2 for loop μας αφού δεν εξαρτάται η εκτέλεση τους ενός από το άλλο



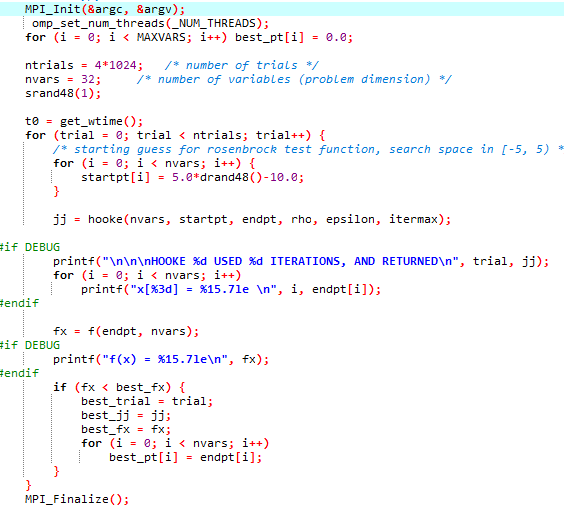
Αρχικά δηλωνουμε τις εντολές MPI\_Comm\_rank και MPI\_Comm\_size για να προσδιορίσουμε το αναγνωριστικό της διεργασίας μας και τον αριθμό τον τους .Επιπρόσθετα για να αρχίσουμε την παραλληλοποιήση του κώδικα μας χρησιμοποιήσαμε την εντολή **#pragma omp parallel for shared(nvars),private(i)**



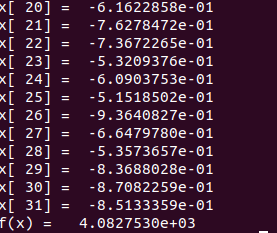
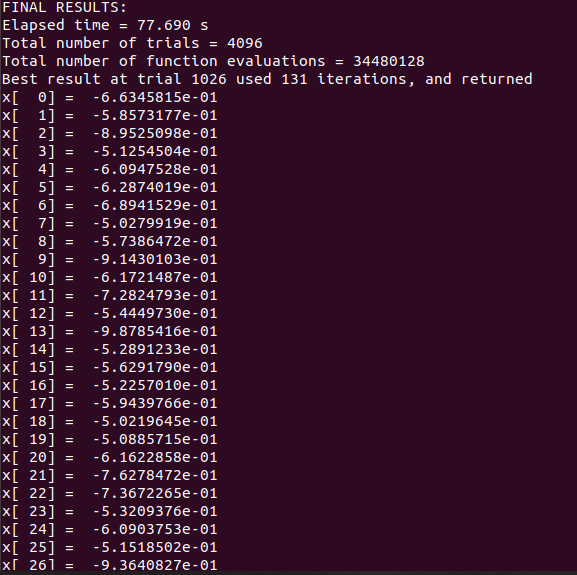
Για να θέσουμε σε λειτουργία παραλληλισμού τα **for** μας πρέπει να εισάγουμε την εντολή **#pragma omp for.**



Το Mpi\_reduce το προσθέσαμε για την πιο γρήγορη εκτέλεση του πολλαπλασιασμού

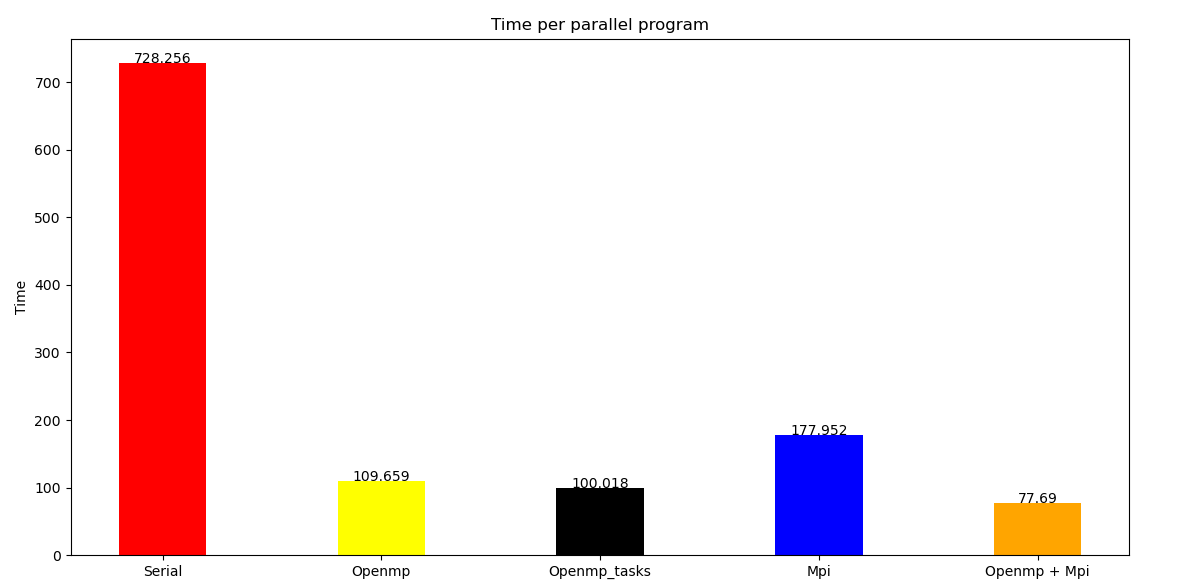


Εδώ γίνεται η αρχικοποίηση του MPI με την εντολή MPI\_Init και επιπλέον συνεπάγεται και η δήλωση του MPI\_Finalize για την λήξη της παραλληλοποιησης.



**------🕙------**

**Διάγραμμα**



**------🕙------**

**Συμπεράσματα:**

* Βλέπουμε, με την βοήθεια της **γραφικής,** το χρόνο του ***μη παραλλοποιημένου*** σαφώς *μεγαλύτερο* και *σταθερό* από τους άλλους χρόνους το οποίο ήταν και αναμενόμενο.
* Παρατηρήσαμε μετά από σκέψη μας για υλοποίηση της επεξεργασίας των κόμβων του γραφήματος με περισσότερα έργα (**Tasks**) ότι οι χρόνοι μας αυξάνονταν (*πιο αργό*). Γι’ αυτό και έγινε η χρήση όσο το δυνατών λιγότερων έργων.
* Παρατηρούμε ότι το πρόγραμμα μας στο mpi είναι πιο αργό από το openmp. Πιστεύουμε ο λόγος είναι επειδή υπάρχει μικρότερη οικειότητα και περιοριστήκαμε στη χρήση των εντολών που υπάρχουν στο mpi.
* Παρατηρούμε ότι το πρόγραμμα μας στο mpi και στο openmp είναι όπως περιμέναμε πιο γρήγορο, αφού με τον συνδυασμό αυτό των δύο η παραλληλοποίηση μας είναι πιο αποδοτική.