ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΩΝ ΚΑΙ ΥΛΙΚΟΥ

ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΠΑΡΑΛΛΗΛΩΝ ΚΑΙ ΚΑΤΑΝΕΜΗΜΕΝΩΝ ΥΠΟΛΟΓΙΣΤΩΝ (ΗΡΥ 418)

ΕΑΡΙΝΟ ΕΞΑΜΗΝΟ 2018 - 2019
ΑΣΚΗΣΗ 1 : Χρήση OpenMP και pthreads
Ν. Αλαχιώτης

31 Μαρτίου 2019

ΟΜΑΔΑ ΕΡΓΑΣΙΑΣ: LAB41839834
ΚΑΤΑΡΑ ΣΩΤΗΡΙΑ ΜΑΡΙΑ Α.Μ.: 2015030040
ΜΑΡΑΓΚΑΚΗ ΜΑΡΙΑ Α.Μ.: 2015030153

ΑΣΚΗΣΗ 1 : Χρήση OpenMP και pthreads

ΣΚΟΠΟΣ ΑΣΚΗΣΗΣ

Σκοπός της πρώτης άσκησης ήταν η επιτάχυνση του αλγορίθμου Smith-Waterman, ο οποίος χρησιμοποιείται για τοπική ευθυγράμμιση ακολουθιών, μέσω της παραλληλοποίησης του. Η παραλληλοποίηση του αλγορίθμου έγινε με δύο τρόπους. Ο πρώτος τρόπος αφορά την χρήση του OpenMP Application Protocol Interface (API) ενώ ο δεύτερος τη χρήση του υποσυνόλου των συναρτήσεων POSIX threads (pthreads).

ΑΛΓΟΡΙΘΜΟΣ SMITH-WATERMAN

Ο συγκεκριμένος αλγόριθμος χρησιμοποιείται στη βιολογία για την ανάλυση αλληλουχιών γονιδίων και υλοποιεί τοπική ευθυγράμμιση ακολουθίας. Η ευθυγράμμιση γίνεται ώς εξής: οι υπο σύγκριση ακολουθίες τοποθετούνται σε έναν πίνακα (m+1)(n+1) όπου m και n οι διαστάσεις των ακολουθιών. Οι διαστάσεις έχουν το συγκεκριμένο μήκος καθώς στην πρώτη στήλη και πρώτη γραμμή του πίνακα τοποθετούνται μηδενικά. Στη συνέχεια, αφου έχουν δοθεί οι τιμές των παραμέτρων match, mismatch, gap, σύμφωνα με τον παρακάτω τύπο, με δύο εμφωλευμένες επαναλήψεις υπολογίζεται ο πίνακας.

$$H_{ij} = \max \begin{cases} H_{i-1,j-1} + s(a_i, b_j), \\ \max_{k \geq 1}\{H_{i-k,j} - W_k\}, \\ \max_{l \geq 1}\{H_{i,j-l} - W_l\}, \\ 0 \end{cases} \qquad (1 \leq i \leq n, 1 \leq j \leq m)$$

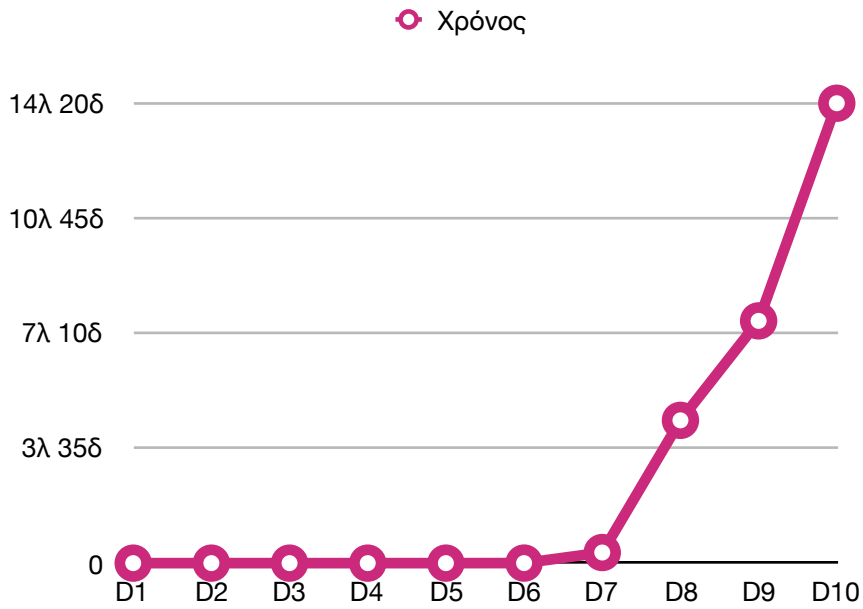
Συγκεκριμένα, όταν δύο στοιχεία των ακολουθιών είναι όμοια τότε η παράμετρος s(ai,bj) θα γίνει ίση με την παράμετρος match(θετικός ακέραιος). Αν είναι διαφορετικά τότε η παράμετρος s(ai,bj) θα γίνει ίση με την παράμετρος mismatch(αρνητικός ακέραιος). Η παράμετρος gap penalty προσδιορίζεται από την τιμή W_k.

ΣΕΙΡΙΑΚΟΣ ΚΩΔΙΚΑΣ

Σε πρώτο επίπεδο υλοποιήθηκε ο σειριακός κώδικας του αλγορίθμου Smith-Waterman στην περίπτωση εφαρμογής linear gap penalty. Αξίζει να σημειωθεί ότι ο πίνακας που περιέχει τα scores συμπληρώθηκε με την μέθοδο της αντιδιαγωνιοποίησης με σκοπό τον σωστό παραλληλισμό του στο επόμενο στάδιο. Από το τερματικό του υπολογιστή δόθηκαν τα εξής flags και arguments: *name, input, match, mismatch, gap*. Όπου *name* το όνομα του αρχείου εξόδου του προγράμματος, *input* το path από το οποίο διαβάστηκαν τα αρχεία εισόδου, match ακέραια τιμή όταν οι ακολουθίες έχουν ίδια στοιχεία, mismatch ακέραια τιμή όταν οι ακολουθίες δεν έχουν ίδια στοιχεία και *gap* το gap penalty του αλγορίθμου. Κατόπιν υλοποιήθηκε ο αλγόριθμος με τον τρόπο που περιγράφηκε παραπάνω. Ανοίγοντας διαδοχικά τα αρχεία που δόθηκαν και εφαρμόζοντας τον αλγόριθμο στις ακολουθίες που περιέχονταν σε αυτά, οι χρόνοι εκτέλεσης προέκυψαν ως εξής:

ΑΣΚΗΣΗ 1 : Χρήση OpenMP και pthreads

Serial	
D1	0m0,001s
D2	0m0,001s
D3	0m0,001s
D4	0m0,002s
D5	0m0,006s
D6	0m0,097s
D7	0m19,85s
D8	4m27s
D9	7m33s
D10	14m20s



Παρατηρήσεις:

Οι ανάλογοι χρόνοι για κάθε αρχείο είναι σχεδόν βέλτιστοι, γεγονός που οφείλεται στην προσπάθεια για όσο το δυνάτον μικρότερη πολυπλοκότητα της υλοποίησης του αλγορίθμου (O(mn)).

ΠΑΡΑΛΛΗΛΟΠΟΙΗΣΗ

Η παραλληλοποίηση του κώδικα είχει ως σκοπό την επιτάχυνση αυτού. Ο κώδικας χωρίζεται σε Tasks με συγκεκριμένο μέγεθος και κάθε νήμα αναλαμβάνει να εκτελέσει κάποιο Task. Έτσι πολλαπλά νήματα εκτελούν ταυτόχρονα ανεξάρτητα κομμάτια του κώδικα. Στην πρώτη άσκηση ζητήθηκε παραλληλισμός με OpenMP και Pthreads. Για κάθε μέθοδο παραλληλισμού εφαρμόστηκαν δύο τρόποι υλοποίησης. Ο πρώτος τρόπος ήταν fine grain παραλληλισμός ενώ ο

δεύτερος ήταν coarse grain. Στον συγκεκριμένο αλγόριθμο, το σημείο που ήταν αναγκαίο να γίνει παράλληλο ήταν ο υπολογισμός του δισδιάστατου πίνακα Score. Ο coarse grane παραλληλισμός υλοποιήθηκε παραλληλοποιόντας την επαναληπτική διαδικασία εφαρμογής του αλγόριθμου στα ζευγάρια του κάθε αρχείου, ενώ ο fine grane παραλληλοποιόντας τον αλγόριθμο ο οποίος πραγματεύοταν τον υπολογισμό των τιμών του δισδιάστατου πίνακα των scores .

ΠΑΡΑΛΛΗΛΟΠΟΙΗΣΗ ΜΕ OpenMP API

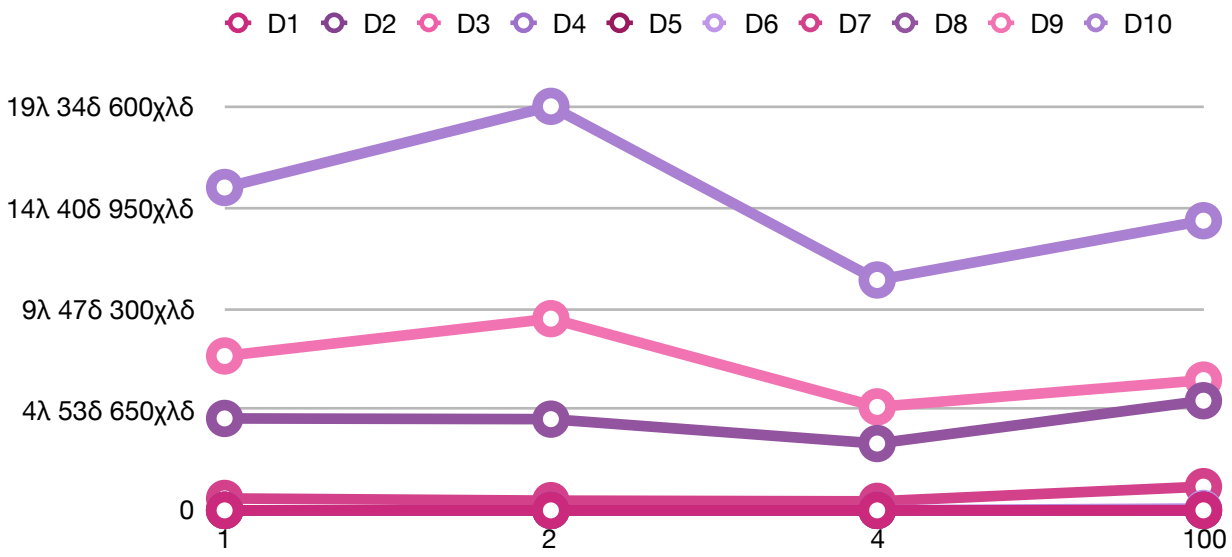
Έπειτα από την υλοποίηση του σειριακού κώδικα έγινε παραλληλοποίηση αυτού με τη χρήση των OpenMp Threads. Τα OpenMp Threads είναι μία διεπαφή προγραμματισμού εφαρμογών (API) που υποστηρίζει πλατφόρμες εργασίας shared memory. Περιέχει compiler directives, ρουτίνες, βιβλιοθήκες και μεταβλητές όπου επηρεάζουν την εκτέλεση του προγράμματος. Τα OpenMp δίνουν στον προγραμματιστή την δυνατότητα ανάπτυξης παράλληλων εφαρμογών με καλή απόδοση. Στη συγκεκριμένη περίπτωση χρησιμοποιήθηκε μια εντολή compiler directive. Η εντολή ήταν η εξής :

- *#pragma omp parallel for private() num_threads()*

Όπου στο πεδίο private δηλώνονται οι μεταβλητές του for loop που θα παραλληλοποιηθεί, και στο πεδίο num_threads δηλώνονται τα νήματα στα οποία θα δοθούν τα παράλληλα Tasks. Στον παρακάτω πίνακα παρουσιάζονται οι χρόνοι εκτέλεσης κάθε αρχείου για συγκεκριμένο αριθμό από threads για την ***fine grain*** υλοποίηση.

OpenMP (fine grain)				
	Threads			
Datasets	**1**	**2**	**4**	**100**
D1	0m0,002s	0m0,002s	0m0,002s	0m0,065s
D2	0m0,002s	0m0,003s	0m0,002s	0m0,085s
D3	0m0,002s	0m0,002s	0m0,004s	0m0,108s
D4	0m0,002s	0m0,003s	0m0,003s	0m,204s
D5	0m0,008s	0m0,010s	0m0,021s	0m0,601s
D6	0m0,244s	0m0,181s	0m0,197s	0m5,654s
D7	0m34,753s	0m28,314s	0m26,248s	1m9,058s
D8	4m27,32s	3m85,21s	3m14,35s	5m18,79s
D9	7m28,89s	9m17,36s	5m0,8932s	6m18,72s
D10	14m98,67s	19m34,58s	11m9,786s	13m62,12s

ΑΣΚΗΣΗ 1 : Χρήση OpenMP και pthreads

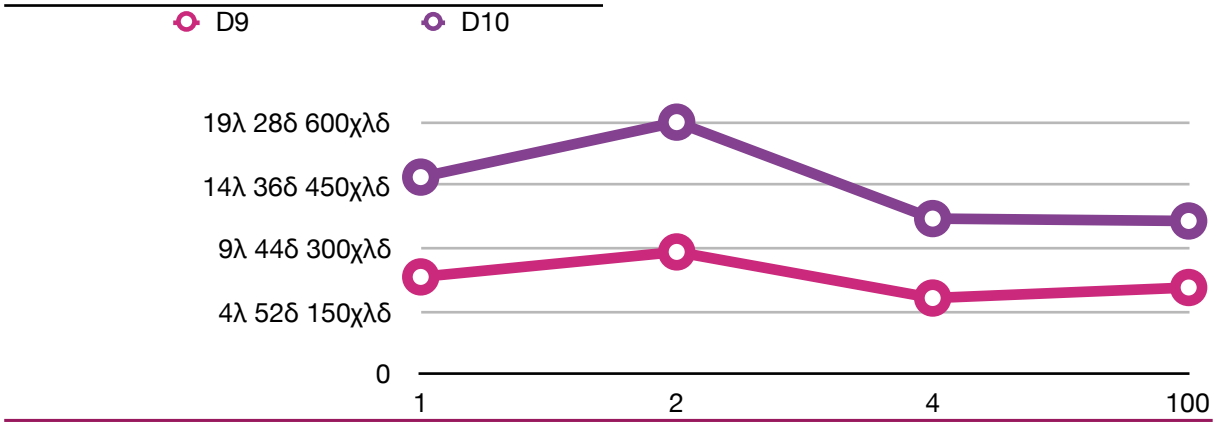


Παρατηρήσεις:

Η βελτιώση των χρόνων εκτέλεσης είναι εμφανής ειδικότερα στα μεγάλα αρχεία , όπου ο αριθμός των ζευγαριών είναι μεγαλύτερος της μονάδας. Για 1 thread ο χρόνος εκτέλεσης είναι σχεδόν όμοιος με αυτόν της σειριακής υλοποίησης. Για 4 (μέγιστος αριθμός πυρήνων επεξεργαστή) threads η βελτίωση είναι εμφανής. Το γεγονός ότι για 2 threads έχουμε μεγαλύτερους χρόνους από τους αναμενόμενους είναι άξιο προβληματισμού και κάτα πάσα πιθανότητα οφείλεται στην εκτέλεση παράλληλης διεργασίας από το υπολογιστικό σύστημα. Για 100 threads ο χρόνος είναι λίγο μεγαλύτερος από εκείνον για 4 threads γεγονός που οφείλεται στην τοποθέτηση των νημάτων στον scheduler και στην αναμονή τους μέχρι να έρθει η σειρά τους να εκτελέσουν κάποιο task.

Στον παρακάτω πίνακα παρουσιάζονται οι χρόνοι εκτέλεσης κάθε αρχείου για συγκεκριμένο αριθμό από threads για την ***coarse grain*** υλοποίηση.

OpenMP (coarse grain)				
	Threads			
Datasets	**1**	**2**	**4**	**100**
D9	7m30,9s	9m26,78s	5m52,8932s	6m41,72s
D10	15m13,647s	19m28,54s	12m1,343s	11m50,231s



Παρατηρήσεις:

Όμοια με πριν για 1 thread ο χρόνος εκτέλεσης είναι σχεδόν όμοιος με αυτόν της σειριακής υλοποίησης. Για 4 (μέγιστος αριθμός πυρήνων επεξεργαστή) threads η βελτίωση είναι εμφανής. Το γεγονός ότι για 2 threads έχουμε μεγαλύτερους χρόνους από τους αναμενόμενους είναι άξιο προβληματισμού και κάτα πάσα πιθανότητα οφείλεται στην εκτέλεση παράλληλης διεργασίας από το υπολογιστικό σύστημα. Για 100 threads ο χρόνος είναι λίγο μεγαλύτερος από εκείνον για 4 threads γεγονός που οφείλεται στην τοποθέτηση των νημάτων στον scheduler και στην αναμονή τους μέχρι να έρθει η σειρά τους να εκτελέσουν κάποιο task.

ΠΑΡΑΛΛΗΛΟΠΟΙΗΣΗ ΜΕ Pthreads

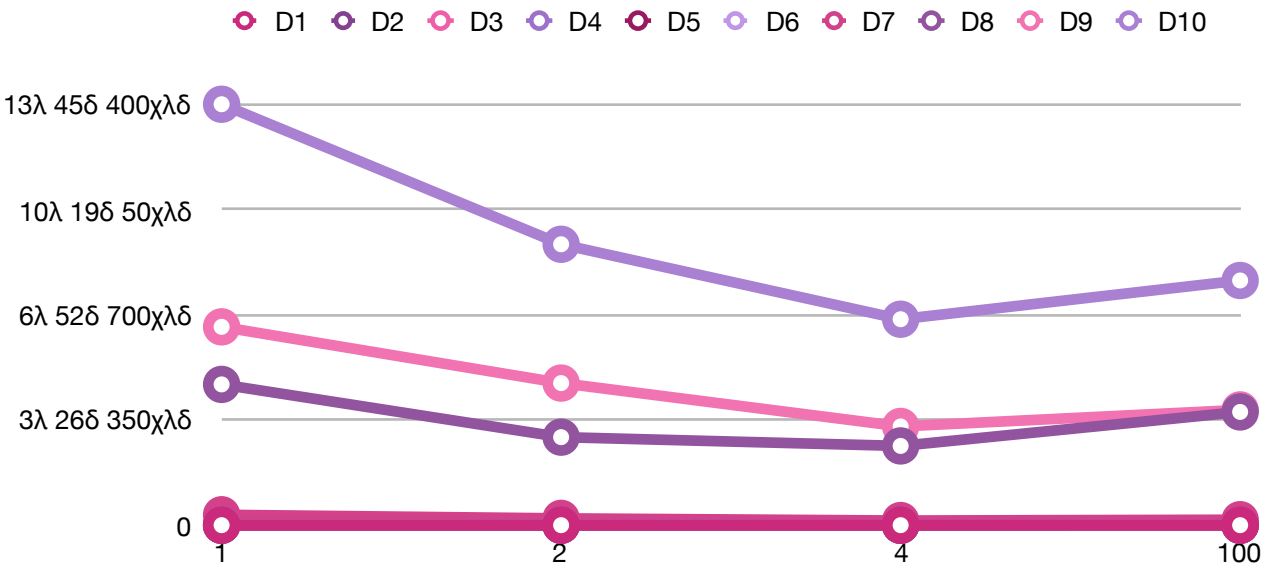
Τα POSIX threads είναι ένα μοντέλο εκτέλεσης που υπάρχει ανεξάρτητα από μια γλώσσα, καθώς και ένα μοντέλο παράλληλης εκτέλεσης. Επιτρέπει σε ένα πρόγραμμα να ελέγχει πολλαπλές ροές εργασία που επικαλύπτονται στο χρόνο. Κάθε ροή εργασίας αναφέρεται ως νήμα και η δημιουργία και ο έλεγχος αυτών των ροών επιτυγχάνεται με κλήσεις προς το API των POSIX threads.

Για **fine grane** παραλληλισμό, παραλληλοποιήθηκε η συνάρτηση που υλοποιεί τον αλγόριθμο Smith Waterman (γέμισμα δισδιάστατου πίνακα) και τα αποτελέσματα για τους χρόνους εκτέλεσης ήταν τα εξής:

Pthreads (fine grain)				
	Threads			
Datasets	**1**	**2**	**4**	**100**
D1	0m0,002s	0m0,003s	0m0,004s	0m0,007s
D2	0m0,002s	0m0,002s	0m0,003s	0m0,007s
D3	0m0,002s	0m0,002s	0m0,006s	0m0,017s
D4	0m0,002s	0m0,002s	0m0,005s	0m0,018s
D5	0m0,006s	0m0,005s	0m0,005s	0m0,020s
D6	0m0,124s	0m0,091s	0m0,076s	0m0,119s
D7	0m20,372s	0m13,3255s	0m9,071s	0m10,686s

ΑΣΚΗΣΗ 1 : Χρήση OpenMP και pthreads

Pthreads (fine grain)				
D8	4m36s	2m52,567s	2m35,36s	3m42s
D9	6m28,85s	4m38,577s	3m13,588s	3m46s
D10	13m45,311s	9m10,746s	6m43,779s	7m59,762s

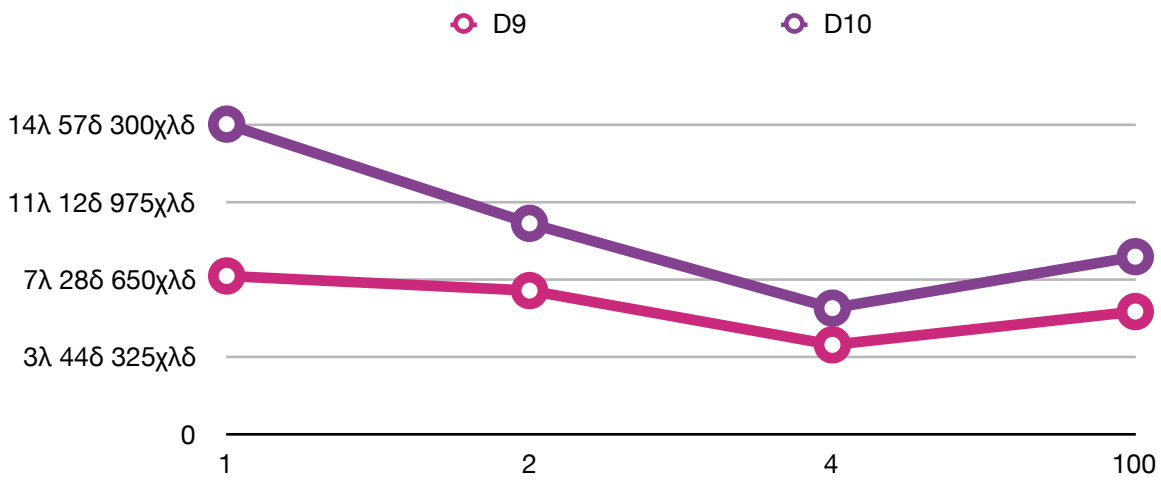


Παρατηρήσεις:

Για 1 thread ο χρόνος εκτέλεσης είναι σχεδόν όμοιος με αυτόν της σειριακής υλοποίησης. Για 2 και ακόμη καλύτερη για 4 (μέγιστος αριθμός πυρήνων επεξεργαστή) threads η βελτίωση είναι εμφανής. Για 100 threads ο χρόνος είναι λίγο μεγαλύτερος από εκείνον για 4 threads γεγονός που οφείλεται στην τοποθέτηση των νημάτων στον scheduler και στην αναμονή τους μέχρι να έρθει η σειρά τους να εκτελέσουν κάποιο task.

Στον παρακάτω πίνακα παρουσιάζονται οι χρόνοι εκτέλεσης κάθε αρχείου για συγκεκριμένο αριθμό από threads για την ***coarse grain*** υλοποίηση.

Pthreads (coarse grain)				
	Threads			
Datasets	**1**	**2**	**4**	**100**
D9	7m38,85s	6m56,57s	4m19,89s	5m56s
D10	14m57,3s	9m70,76s	6m5,867s	8m34,542s



Παρατηρήσεις:

Για 1 thread ο χρόνος εκτέλεσης είναι σχεδόν όμοιος με αυτόν της σειριακής υλοποίησης. Για 2 και ακόμη καλύτερη για 4 (μέγιστος αριθμός πυρήνων επεξεργαστή) threads η βελτίωση είναι εμφανής. Για 100 threads ο χρόνος είναι λίγο μεγαλύτερος από εκείνον για 4 threads γεγονός που οφείλεται στην τοποθέτηση των νημάτων στον scheduler και στην αναμονή τους μέχρι να έρθει η σειρά τους να εκτελέσουν κάποιο task.

ΣΥΜΠΕΡΑΣΜΑΤΑ

Στον παραλληλισμό με Pthreads βλέπουμε λίγο καλύτερους χρόνους σε σχέσ με τον παραλληλισμό με OpenMP. Αυτό συμβαίνει γιατί τα κομμάτια κώδικα που παραλληλοποιούνται είναι πιο σύνθετα. Κατά γενική ομολογία οι OpenMp υλοποιήσεις είναι πιο αποδοτικές σε σχέση με τις υλοποιήσεις με Pthreads γεγονός που αφορά πρόγραμματα με απλές συναρτήσεις. Γι αυτο στην προκειμένη υλοπολποίση δεν το παρατηρούμε αυτό ,αφού οι συναρτήσεις που παραλληλοποιούνται είναι αρκετά σύνθετες ειδικά στην coarse grain υλοποίηση. Τέλος τα OpenMP threads είναι καλύτερα σε σχέση με τα POSIX αφού δεν κλειδώνουν το λογισμικό σε έναν προκαθορισμένο αριθμό νημάτων, σε αντίθεση με τα POSIX. Έτσι αποφεύγεται η άσκοπη κατανάλωση πόρων

ΠΛΗΡΟΦΟΡΙΕΣ-ΠΑΡΑΠΟΜΠΕΣ

- Ο κώδικας εκτελέστηκε σε υπολογιστικό σύστημα με τα ακόλουθα χαρακτηριστικά: 4-Processor 2,7 GHz Intel Core i7 , Memory 16 GB 2133 MHz LPDDR3
- Για την εκτέλεση του κώδικα από το τερματικό : Στο πεδίο match ορίζεται θετικός ακέραιος , στο πεδίο mismatch αρνητικός ακέραιος και στο πεδίο gap ορίζεται θετικός ακέραιος

Πηγές υλικού:

- https://software.intel.com/en-us/articles/threading-models-for-high-performance-computing-pthreads-or-openmp?fbclid=IwAR0XG5srSVARUoOgLSrY6j5myAjZ-0DF-VLUOcjTBN9u3LZym244zNvlspg
- https://www.ibm.com/developerworks/library/l-posix1/index.html
- https://arch.icte.uowm.gr/docs/OpenMP_v21.GreekTutorialLLNLGOV.pdf
- http://www.bowdoin.edu/~ltoma/teaching/cs3225-GIS/fall16/Lectures/openmp.html