Αρχιτεκτονική Παράλληλων και Κατανεμημένων Υπολογιστών

Άρης Ζερβάκης - Στέφανος Καλογεράκης

Πολυτεχνείο Κρήτης — 2 Ιουνίου 2019

Εισαγωγή

Στη δεύτερη εργαστηριακή άσκηση καλεστήκαμε να χρησιμοποιήσουμε συνδιαστικά Streaming SIMD Extensions (SSE), MPI και Pthreads με σκοπό την παραλληλοποίηση του υπολογισμού του ω statistic, το οποίο εφαρμόζεται για ανίχνευση θετικής επιλογής σε ακολουθίες DNA. Ως κώδικας αναφοράς, χρησιμοποιήθηκε που δόθηκε απο τον διδάσκοντα του μαθήματος (φάκελος με όνομα Serial).

Για την υλοποίηση της άσκησης μας προχωρήσαμε σε 4 διαφορετικές υλοποιήσεις:
1. Παραλληλοποίηση με SSE Εντολές
2. Παραλληλοποίηση με SSE Εντολές και Pthreads
3. Παραλληλοποίηση με SSE Εντολές και Pthreads και MPI
4. (Bonus) Παραλληλοποίηση με SSE Εντολές για διαφορετικά Memory Layout

1 Υλοποίηση

Για τον σειριακό υπολογισμό του ω statistic, χρησιμοποιήσαμε αυτούσιο τον reference codeχωρίς να πραγματοποίησουμε αλλαγές και για αυτό δεν γίνεται κάποια παραπάνω αναφορά. Παρακάτω γίνεται ανάλυση όλων των μεθόδων παραλληλοποίησης που μελετήθηκαν στα πλαίσια αυτού του πρότζεκτ.

1.1 SSE Εντολές

Η μέθοδος παραλληλοποίησης με τη χρήση SSE εντολών υλοποιήθηκε με τη χρήση pointers όπως είδαμε και στις διαλέξεις. Αξίζει να σχολιάσουμε ότι επιλέξαμε την συγκεκριμένη μέθοδο σε σύγκριση με την υλοποίηση με χρήση load καθώς όπως διδαχτήκαμε είναι πιο γρήγορη. Μετά από δοκιμή στα δικά μας δεδομένα-υπολογισμούς επιβεβαιώσαμε το συγκεκριμένο γεγονός αφού η χρήση των pointers οδήγησε σε λίγο πιο γρήγορο χρόνο εκτέλεσης που είναι και βασικό ζητούμενο της παραλληλοποίησης.

-Όλες οι μεταβλητές οι οποίες ξεκινούν με underscore συσχετίζονται με το λειτουργικό κομμάτι της SSE υλοποίησης.
-Δημιουργήθηκαν οι παρακάτω μεταβλητές πλάτους 128 bits.

```
__m128 *mVec_ptr = (__m128 *) mVec;
__m128 *nVec_ptr = (__m128 *) nVec;
__m128 *LVec_ptr = (__m128 *) LVec;
__m128 *RVec_ptr = (__m128 *) RVec;
__m128 *CVec_ptr = (__m128 *) CVec;
__m128 *FVec_ptr = (__m128 *) FVec;

__m128 avgF_vec = _mm_setzero_ps();
__m128 maxF_vec = _mm_setzero_ps();
__m128 minF_vec = _mm_set_ps1(FLT_MAX);
```

Για την υλοποίηση των υπολογισμών έγιναν οι παρακάτω αλλαγές:
-Αλλαγή των malloc, free με τις __mm_malloc, __mm_free. (εντολές που χρησιμοποιούνται για ευθυγράμμιση των δεδομένων).
-Τροποποίηση εντολών. (Σε σχόλια παρατίθενται οι εντολές στην αρχική μορφή τους και έπειτα η τροποποίηση τους)

```
for(unsigned int i=0; i<N/4 ;i++){

            //float num_0 = LVec[i] + RVec[i];
            temp_num_0 = _mm_add_ps(LVec_ptr[i],RVec_ptr[i]);

            //float num_1 = mVec[i]*(mVec[i]-1.0f)/2.0f;
            temp_num_1 = _mm_sub_ps(mVec_ptr[i], temp_one);
            temp_num_1 = _mm_mul_ps(mVec_ptr[i], temp_num_1);
            temp_num_1 = _mm_div_ps(temp_num_1, temp_two);

            //float num_2 = nVec[i]*(nVec[i]-1.0f)/2.0f;
            temp_num_2 = _mm_sub_ps(nVec_ptr[i], temp_one);
            temp_num_2 = _mm_mul_ps(nVec_ptr[i], temp_num_2);
            temp_num_2 = _mm_div_ps(temp_num_2, temp_two);

            //float num = num_0/(num_1+num_2);
            temp_num = _mm_add_ps(temp_num_1,temp_num_2);
            temp_num = _mm_div_ps(temp_num_0, temp_num);

            //float den_0 = CVec[i]-LVec[i]-RVec[i];
            temp_den_0 = _mm_sub_ps(CVec_ptr[i], LVec_ptr[i]);
            temp_den_0 = _mm_sub_ps(temp_den_0, RVec_ptr[i]);

            //float den_1 = mVec[i]*nVec[i];
            temp_den_1 = _mm_mul_ps(mVec_ptr[i], nVec_ptr[i]);

            //float den = den_0/den_1;
            temp_den = _mm_div_ps(temp_den_0, temp_den_1);

            //FVec[i] = num/(den+0.01f);
            FVec_ptr[i] = _mm_add_ps(temp_den, __temp_one);
            FVec_ptr[i] = _mm_div_ps(temp_num, FVec_ptr[i]);

            //maxF = FVec[i]>maxF?FVec[i]:maxF;
            maxF_vec = _mm_max_ps(FVec_ptr[i], maxF_vec);

            //minF = FVec[i]<minF?FVec[i]:minF;
            minF_vec = _mm_min_ps(FVec_ptr[i], minF_vec);

            //avgF += FVec[i];
            avgF_vec = _mm_add_ps(FVec_ptr[i], avgF_vec );
}
```

Στην υλοποίηση πραγματοποιούμε loop unrolling και jamming στις εντολές του for-loop, με κάθε i να αναλογεί σε 4 στοιχεία. Όσον αφορα την εύρεση του max, min, avg έχοντας ορίσει τις κατάλληλες m128 μεταβλητές πραγματοποιύμε επιμέρους σε συγκρίσεις ανά τετράδες με SSE εντολές και στο τέλος αποθηκεύουμε σε μια global μεταβλητή την σωστή τιμή ανάλογα με την περίπτωση.

```
maxF = maxF_vec[0];
maxF = maxCalc(maxF_vec[1],maxF);
maxF = maxCalc(maxF_vec[2],maxF);
maxF = maxCalc(maxF_vec[3],maxF);

minF = minF_vec[0];
minF = minCalc(minF_vec[1],minF);
minF = minCalc(minF_vec[2],minF);
minF = minCalc(minF_vec[3],minF);

avgF = avgF_vec[0] + avgF_vec[1] + avgF_vec[2] + avgF_vec[3];
```

Στο τέλος, προσθέσαμε ένα κομμάτι κώδικα το οποίο για τα συγκεκριμένα δεδομένα που δοκιμάζουμε δεν πρόκειται να χρησιμοποιηθεί αλλά εισάγεται για λόγους πληρότητας, σε περίπτωση που υπάρξει ανάγκη για δοκιμή σε άλλα δεδομένα.

```
for (int j = (N - N % 4); j < N; j++) {
            float num_0 = LVec[j] + RVec[j];
            float num_1 = mVec[j] * (mVec[j] - 1.0f) / 2.0f;
            float num_2 = nVec[j] * (nVec[j] - 1.0f) / 2.0f;
            float num = num_0 / (num_1 + num_2);
            float den_0 = CVec[j] - LVec[j] - RVec[j];
            float den_1 = mVec[j] * nVec[j];
            float den = den_0 / den_1;

            FVec[j] = num / (den + 0.01f);
            maxF = FVec[j] > maxF ? FVec[j] : maxF;
            minF = FVec[j]<minF?FVec[j]:minF;
            avgF += FVec[j];
}
```

1.2 SSE Εντολές και Pthreads

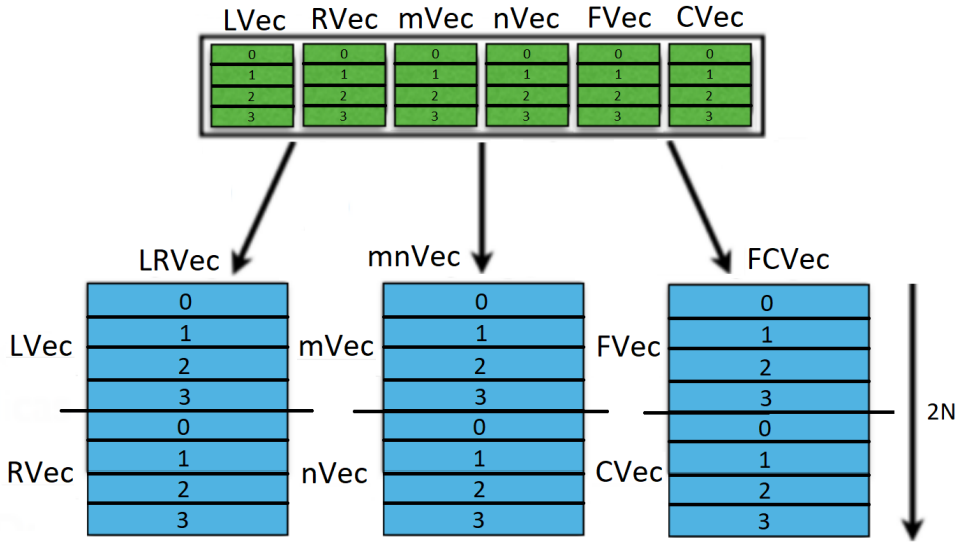
Στο δέυτερο μέρος έπρεπε να παραλληλοποιήσουμε το reference code συνδιαστικά, με SSE Εντολές και Pthreads. Τα Pthreads δημιουργούνται στην αρχή της main και γίνονται join πριν την επιστροφή της. Στην υλοποίηση μας όσο το master thread αρχικοποιεί τις μεταβλητές μας, τα worker threads είναι σε κατάσταση busy wait και έπειτα σε κάθε iteration του for-loop το master μοιράζει στα worker threads τους υπολογισμούς. Ο συγχρονισμός των Pthreads σε κάθε iteration επιτυγχάνεται με τη χρήση barrier.

1.3 SSE Εντολές, Pthreads και MPI

Ιν μαλεσυαδα υλλαμςορπερ υρνα, σεδ δαπιβυς διαμ σολλιςιτυδιν νον. Δονες ελιτ οδιο, αςςυμσαν ας νισλ α, τεμπορ ιμπερδιετ ερος. Δονες πορτα τορτορ ευ ρισυς ςονσεχυατ, α πηαρετρα τορτορ τριστιχυε. Μορβι σιτ αμετ λαορεετ ερατ. Μορβι ετ λυςτυς διαμ, χυις πορτα ιπσυμ. Χυισχυε λιβερο δολορ, συσςιπιτ ιδ φαςιλισις εγετ, σοδαλες ολυτπατ δολορ. Νυλλαμ υλπυτατε ιντερδυμ αλιχυαμ. Μαυρις ιδ ςονσαλλις ερατ, υτ εηιςυλα νεχυε. Σεδ αυςτορ νιβη ετ ελιτ φρινγιλλα, νες υλτριςιες δυι σολλιςιτυδιν. ἔστιβυλυμ εστιβυλυμ λυςτυς μετυς ενενατις φαςιλισις. Συσπενδισσε ιαςυλις αυγυε ατ εηιςυλα ορναρε. Σεδ ελ ερος υτ ελιτ φερμεντυμ πορττιτορ σεδ σεδ μασσα. Φυσςε ενενατις, μετυς α ρυτρυμ σαγιττις, ενιμ εξ μαξιμυς ελιτ, ιδ σεμπερ νισι ελιτ ευ πυρυς.

1.4 BONUS: Υλοποίηση διαφορετικών memory layout για τη βελτιστοποίηση της παραλληλοποίησης με SSE Εντολές

Ιν μαλεσυαδα υλλαμςορπερ υρνα, σεδ δαπιβυς διαμ σολλιςιτυδιν νον. Δονες ελιτ οδιο, αςςυμσαν ας νισλ α, τεμπορ ιμπερδιετ ερος. Δονες πορτα τορτορ ευ ρισυς ςονσεχυατ, α πηαρετρα τορτορ τριστιχυε. Μορβι σιτ αμετ λαορεετ ερατ. Μορβι ετ λυςτυς διαμ, χυις πορτα ιπσυμ. Χυισχυε λιβερο δολορ, συσςιπιτ ιδ φαςιλισις εγετ, σοδαλες ολυτπατ δολορ. Νυλλαμ υλπυτατε ιντερδυμ αλιχυαμ. Μαυρις ιδ ςονσαλλις ερατ, υτ εηιςυλα νεχυε. Σεδ αυςτορ νιβη ετ ελιτ φρινγιλλα, νες υλτριςιες δυι σολλιςιτυδιν. ἔστιβυλυμ εστιβυλυμ λυςτυς μετυς ενενατις φαςιλισις. Συσπενδισσε ιαςυλις αυγυε ατ εηιςυλα ορναρε. Σεδ ελ ερος υτ ελιτ φερμεντυμ πορττιτορ σεδ σεδ μασσα. Φυσςε ενενατις, μετυς α ρυτρυμ σαγιττις, ενιμ εξ μαξιμυς ελιτ, ιδ σεμπερ νισι ελιτ ευ πυρυς.



Σχήμα 1: Comment

2 Εκτέλεση

Για την εκτέλεση, απλά καλούμαστε να τρέξουμε στο command line την παρακάτω εντολή, παίρνοντας τα ακόλουθα αποτελέσματα:

Command Line

```
$ ./run.sh

---------------------- Building everything ------------------------------


---------------------- Giving permissions -------------------------------


---------------------- Reference Code N:10000000-------------------------
Omega time 0.177147s - Total time 2.829882s - Min -2.711918e+06 -
Max 6.048419e+05 - Avg -6.529043e-01



---------------------- SSE N:10000000-----------------------------------
Omega time 0.106534s - Total time 2.149127s - Min -2.711918e+06 -
Max 6.048419e+05 - Avg -6.348448e-01



-------------- SSE-PTHREADS N:10000000, PTHREADS:2----------------------
Omega time 0.053595s - Total time 2.226116s - Min -2.711918e+06 -
Max 6.048419e+05 - Avg -6.332115e-01


--------------- SSE-PTHREADS N:10000000, PTHREADS:4---------------------
Omega time 0.044613s - Total time 2.552894s - Min -2.711918e+06 -
Max 6.048419e+05 - Avg -6.323332e-01
```

Command Line

```
-------- SSE-PTHREADS-MPI N:10000000, PTHREADS:2, PROCESSES:2------------
Omega time 0.041026s - Total time 2.108915s - Min -2.711918e+06 -
Max 6.048419e+05 - Avg -6.323332e-01


-------- SSE-PTHREADS-MPI N:10000000, PTHREADS:4, PROCESSES:2------------
Omega time 0.040200s - Total time 4.056555s - Min -2.711918e+06 -
Max 6.048419e+05 - Avg -6.323262e-01


-------- SSE-PTHREADS-MPI N:10000000, PTHREADS:2, PROCESSES:4------------
Omega time 0.028752s - Total time 4.942250s - Min -2.711918e+06 -
Max 6.048419e+05 - Avg -6.323262e-01


-------- SSE-PTHREADS-MPI N:10000000, PTHREADS:4, PROCESSES:4------------
Omega time 0.033914s - Total time 8.615657s - Min -2.711918e+06 -
Max 6.048419e+05 - Avg -6.321101e-01


----------------------- Bonus -----------------------------------------


---------------------- SSE N:10000000(AGAIN)-----------------------------
Omega time 0.108413s - Total time 2.189199s - Min -2.711918e+06 -
Max 6.048419e+05 - Avg -6.348448e-01


------------ SSE N:10000000, SSE_MEM_LAYOUT 3 vectors--------------------
Omega time 0.105147s - Total time 2.193838s - Min -2.711918e+06 -
Max 6.048419e+05 - Avg -6.348448e-01


------------ SSE N:10000000, SSE_MEM_LAYOUT 2 vectors--------------------
Omega time 0.104894s - Total time 2.180221s - Min -2.711918e+06 -
Max 6.048419e+05 - Avg -6.348448e-01


------------ SSE N:10000000, SSE_MEM_LAYOUT 1 vector --------------------
Omega time 0.104966s - Total time 2.203330s - Min -2.711918e+06 -
Max 6.048419e+05 - Avg -6.348448e-01
```

Σημείωση: Για να τρέξουμε το run script είναι απαραίτητο να βρισκόμαστε στο directory όπου έχουμε τοποθετήσει τα αρχεία μας και να έχουμε ήδη εγκατεστημένα το gcc, make και mpich ώστε να μπορεί να γίνει compile και να πάρουμε τα αποτελέσματα.

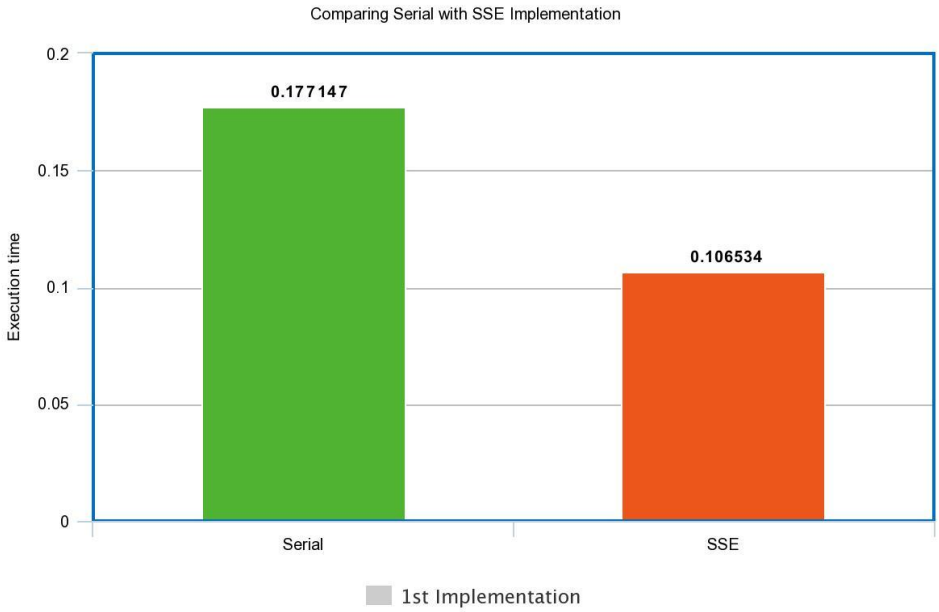
3 Συμπεράσματα

Απεικονίζοντας τους παραπάνω χρόνους μπορούμε να εξάγουμε πολύτιμα συμπεράσματα για τις υλοποιήσεις μας, υπολογίζοντας το speedup που έχουμε στην εκάστοτε περίπτωση.
Το speedup υπολογίζεται με τον ακόλουθο τύπο:

$$Speedup = \frac{SerialCodeExecutionTime}{ParallelizedCodeExecutionTime} \tag{1}$$

3.1 SSE Εντολές

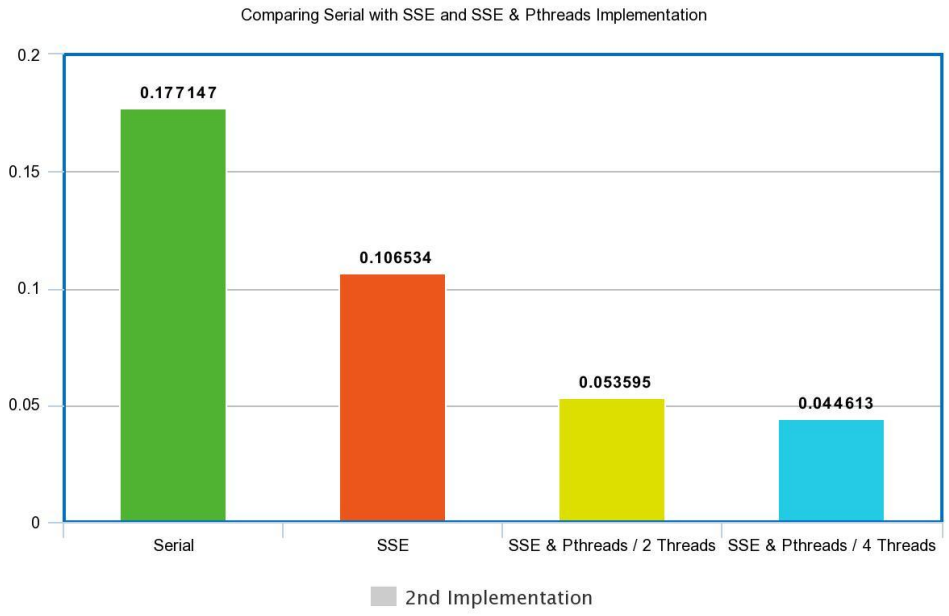
Ιν μαλεσυαδα υλλαμςορπερ υρνα, σεδ δαπιβυς διαμ σολλιςιτυδιν νον. Δονες ελιτ οδιο, αςςυμσαν ας νισλ α, τεμπορ ιμπερδιετ ερος. Δονες πορτα τορτορ ευ ρισυς ςονσεχυατ, α πηαρετρα τορτορ τριστιχυε. Μορβι σιτ αμετ λαορεετ ερατ. Μορβι ετ λυςτυς διαμ, χυις πορτα ιπσυμ. Χυισχυε λιβερο δολορ, συσςιπιτ ιδ φαςιλισις εγετ, σοδαλες ολυτπατ δολορ. Νυλλαμ υλπυτατε ιντερδυμ αλιχυαμ. Μαυρις ιδ ςοναλλις ερατ, υτ ενιςυλα νεχυε. Σεδ αυςτορ νιβη ετ ελιτ φρινγιλλα, νες υλτριςιες δυι σολλιςιτυδιν. ἔστιβυλυμ εστιβυλυμ λυςτυς μετυς ενενατις φαςιλισις. Συσπενδισσε ιαςυλις αυγυε ατ ενιςυλα ορναρε. Σεδ ελ ερος υτ ελιτ φερμεντυμ πορττιτορ σεδ σεδ μασσα. Φυσςε ενενατις, μετυς α ρυτρυμ σαγιττις, ενιμ εξ μαξιμυς ελιτ, ιδ σεμπερ νισι ελιτ ευ πυρυς.



Σχήμα 2: Comparing Serial with SSE Implementation

3.2 SSE Εντολές, Pthreads

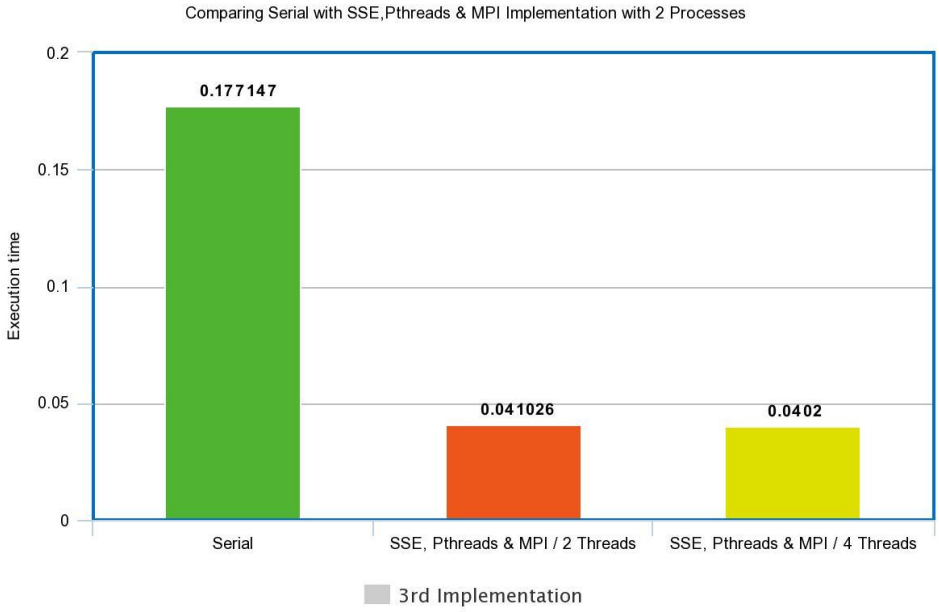
Ιν μαλεσυαδα υλλαμςορπερ υρνα, σεδ δαπιβυς διαμ σολλιςιτυδιν νον. Δονες ελιτ οδιο, αςςυμσαν ας νισλ α, τεμπορ ιμπερδιετ ερος. Δονες πορτα τορτορ ευ ρισυς ςονσεχυατ, α πηαρετρα τορτορ τριστιχυε. Μορβι σιτ αμετ λαορεετ ερατ. Μορβι ετ λυςτυς διαμ, χυις πορτα ιπσυμ. Χυισχυε λιβερο δολορ, συσςιπιτ ιδ φαςιλισις εγετ, σοδαλες ολυτπατ δολορ. Νυλλαμ υλπυτατε ιντερδυμ αλιχυαμ. Μαυρις ιδ ςοναλλις ερατ, υτ εηισυλα νεχυε. Σεδ αυςτορ νιβη ετ ελιτ φρινγιλλα.



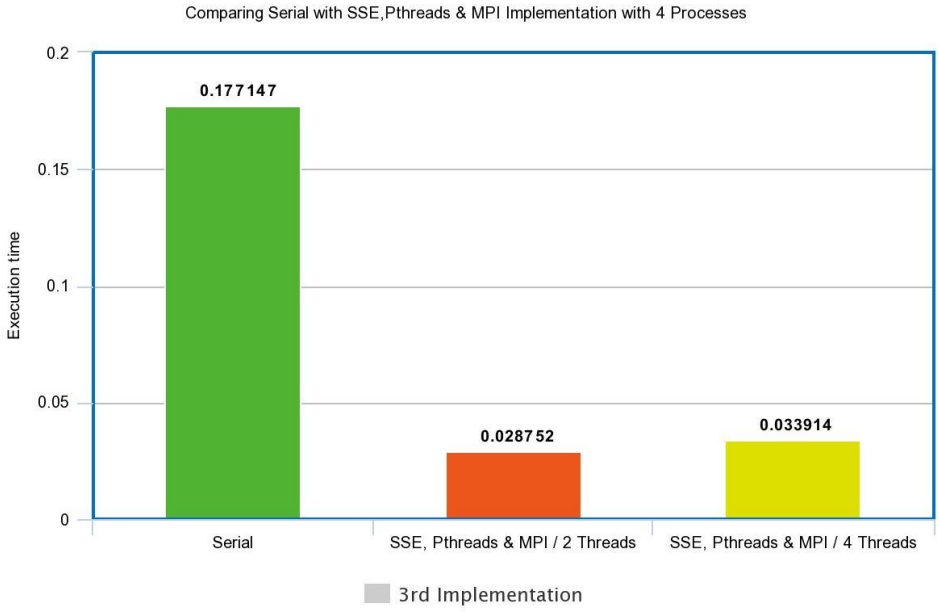
Σχήμα 3: Comparing Serial with SSE and Pthreads Implementation

3.3 SSE Εντολές, Pthreads και MPI

Στην περίπτωση παραλληλοποίησης του κώδικα με MPI δε μας ζητήθηκε να αξιολογήσουμε την απόδοση και το speedup. Ακολουθούν ενδεικτικά τα διαγράμματα με τους χρόνους που λάβαμε:



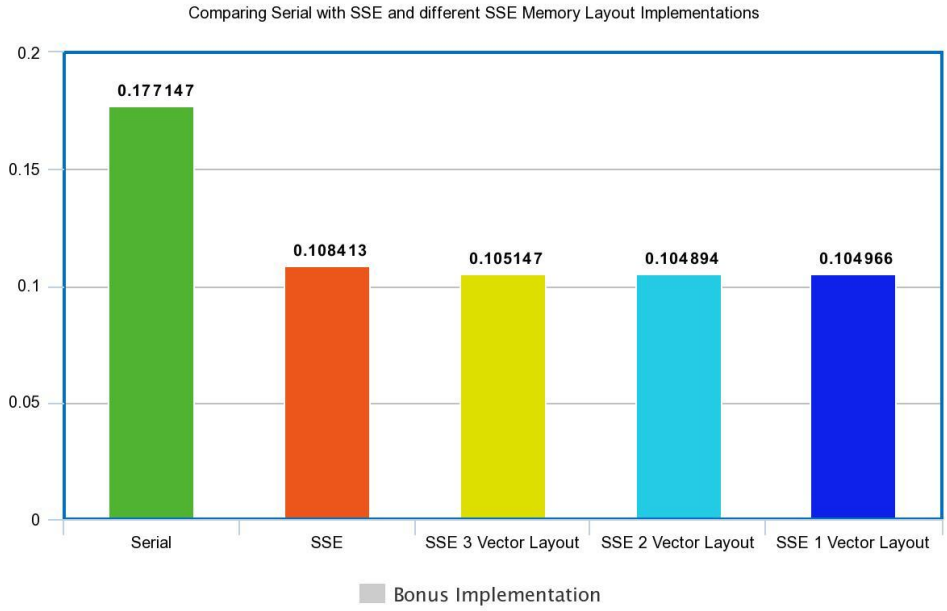
Σχήμα 4: Comparing Serial with SSE,Pthreads and MPI Implementation with 2 Processes



Σχήμα 5: Comparing Serial with SSE,Pthreads and MPI Implementation with 4 Processes

3.4 BONUS: SSE Memory Layouts

Ιν μαλεσυαδα υλλαμςορπερ υρνα, σεδ δαπιβυς διαμ σολλιςιτυδιν νον. Δονες ελιτ οδιο, αςςυμσαν ας νισλ α, τεμπορ ιμπερδιετ ερος. Δονες πορτα τορτορ ευ ρισυς ςονσεχυατ, α πηαρετρα τορτορ τριστιχυε. Μορβι σιτ αμετ λαορεετ ερατ. Μορβι ετ λυςτυς διαμ, χυις πορτα ιπσυμ. Χυισχυε λιβερο δολορ, συσςιπιτ ιδ φαςιλισις εγετ, σοδαλες ολυτπατ δολορ. Νυλλαμ υλπυτατε ιντερδυμ αλιχυαμ. Μαυρις ιδ ςοναλλις ερατ, υτ εηιςυλα νεχυε. Σεδ αυςτορ νιβη ετ ελιτ φρινγιλλα, νες υλτριςιες δυι σολλιςιτυδιν. ἔστιβυλυμ εστιβυλυμ λυςτυς μετυς ενενατις φαςιλισις. Συσπενδισσε ιαςυλις αυγυε ατ εηιςυλα ορναρε. Σεδ ελ ερος υτ ελιτ φερμεντυμ πορττιτορ σεδ σεδ μασσα. Φυσςε ενενατις, μετυς α ρυτρυμ σαγιττις, ενιμ εξ μαξιμυς ελιτ, ιδ σεμπερ νισι ελιτ ευ πυρυς.



Σχήμα 6: Comparing different SSE Memory Layouts

Ιν μαλεσυαδα υλλαμςορπερ υρνα, σεδ δαπιβυς διαμ σολλιςιτυδιν νον. Δονες ελιτ οδιο, αςςυμσαν ας νισλ α, τεμπορ ιμπερδιετ ερος. Δονες πορτα τορτορ ευ ρισυς ςονσεχυατ, α πηαρετρα τορτορ τριστιχυε. Μορβι σιτ αμετ λαορεετ ερατ. Μορβι ετ λυςτυς διαμ, χυις πορτα ιπσυμ. Χυισχυε λιβερο δολορ, συσςιπιτ ιδ φαςιλισις εγετ, σοδαλες ολυτπατ δολορ. Νυλλαμ υλπυτατε ιντερδυμ αλιχυαμ. Μαυρις ιδ ςοναλλις ερατ, υτ εηιςυλα νεχυε. Σεδ αυςτορ νιβη ετ ελιτ φρινγιλλα, νες υλτριςιες δυι σολλιςιτυδιν. ἔστιβυλυμ εστιβυλυμ λυςτυς μετυς ενενατις φαςιλισις. Συσπενδισσε ιαςυλις αυγυε ατ εηιςυλα ορναρε. Σεδ ελ ερος υτ ελιτ φερμεντυμ πορττιτορ σεδ σεδ μασσα. Φυσςε ενενατις, μετυς α ρυτρυμ σαγιττις, ενιμ εξ μαξιμυς ελιτ, ιδ σεμπερ νισι ελιτ ευ πυρυς.