Παράλληλα και διανεμημένα συστήματα

Κουργιαντάκης Γεώργιος 8592 : https://github.com/Gekourgia/Parallel-and-Distributed-Systems

Τσομλεκτσόγλου Βασίλειος 8259 : https://github.com/tsomilios/Parallel-Ergasia.git

Εργασία 1

Αρχικα οι πρώτες μου ενέργειες για την δημιουργία του κώδικα ήταν η ανάγνωση των πολλών αρχείων examples που που παρέχονταν για την βοήθεια μας. Τόσο ο πίνακας coo όσο και τα Market Matrix ήταν αναγκαία για να λυθούν τα προβλήματα που ζητήθηκαν.

Triangle_V3:

Για τον αλγόριθμο triangle_v3 χρησιμοποιήθηκαν οι συναρτήσεις από τα αρχεία mmio,read.mtx καθως το αρχείο coo2csc. Μέσω αυτών ων αρχείων δίνεται η δυνατότητα μέσω του terminal να επιλέξει ο χρήστης 0,1 για μη δυαδικό και δυαδικό πίνακα αντίστοιχα. Στο συγκεκριμένο ερώτημα για να μην γίνονται πολλαπλές καταμετρήσεις των ιδίων τρίγωνων μετατρέπεται ο πίνακας σε άνω ή κάτω τριγωνικός.

Αφού μετατραπεί ο πίνακας από coo σε csc μορφή μπορεί να ξεκινήσει η αναζήτηση η οποία θα συμβαίνει ανά στήλη βρίσκοντας τις μη μηδενικές γραμμές. Αφού τις βρει παίρνει τις συντεταγμένες οι οποίες είναι η πρώτη κορυφή τρίγωνου.

Αφού οι πίνακες είναι συμμετρικοί ισχύει ότι (row=col) , ψάχνω σε ποια άλλη στήλη έχω το row1.Και παίρνω την επόμενη κορυφή (col2 ,row1). Τέλος ελέγχω την 3 κορυφή η οποία είναι (col1, col2) και αν και αυτή περιέχει μη μηδενική τιμή αυξάνω τον αριθμό των τρίγωνων και τον πίνακα c3 με της ανάλογες κορυφές.

Triangle_v3_cilk : Cilk: Για την υλοποίηση του Cilk στην v3 χρησιμοποίησα παραλληλισμό στο εξωτερικό for loop.

Triangle_v3_openmp: Ομοίως έπραξα και για το Open MP με χρήση του pragma omp.

Triangle_V4:

Για τον κώδικα του v4 έγινε χρίση των οδηγιών της 3 παραγράφου και του πολλαπλασιασμού στο σημείο για τους πίνακες C=A⊙(AA). Ουσιαστικά ένα κομμάτι του κώδικα παραμένει ίδιο με το v3 και χρησιμοποιούνται οι συντεταγμένες των μη μηδενικών τιμών ώστε να βρεθεί το πλήθος των τρίγωνων.

Triangle_v4_calk: Με χρίση του cilk_for παραλληλοποιηθηκε το εξωτερικό loop και στην συνέχεια και το εσωτερικό . Αυτό έδειξε κάποια διαφορά στους χρόνους χωρίς όμως κάτι το σημαντικό.'

Triangle_v4_openmp: Με χρήση του pragma στο εξωτερικό loop κυρίως παρατηρήθηκαν μικρές διάφορες.

Triangle_v4_pthreads: Για την υλοποίηση των pthreads σκοπός και πάλι ήταν η παραλληλοποιηση του for loop σε κομμάτια. Δηλαδή κάθε worker αναλάμβανε από ένα κομμάτι. Βέβαια κάτι τέτοιο

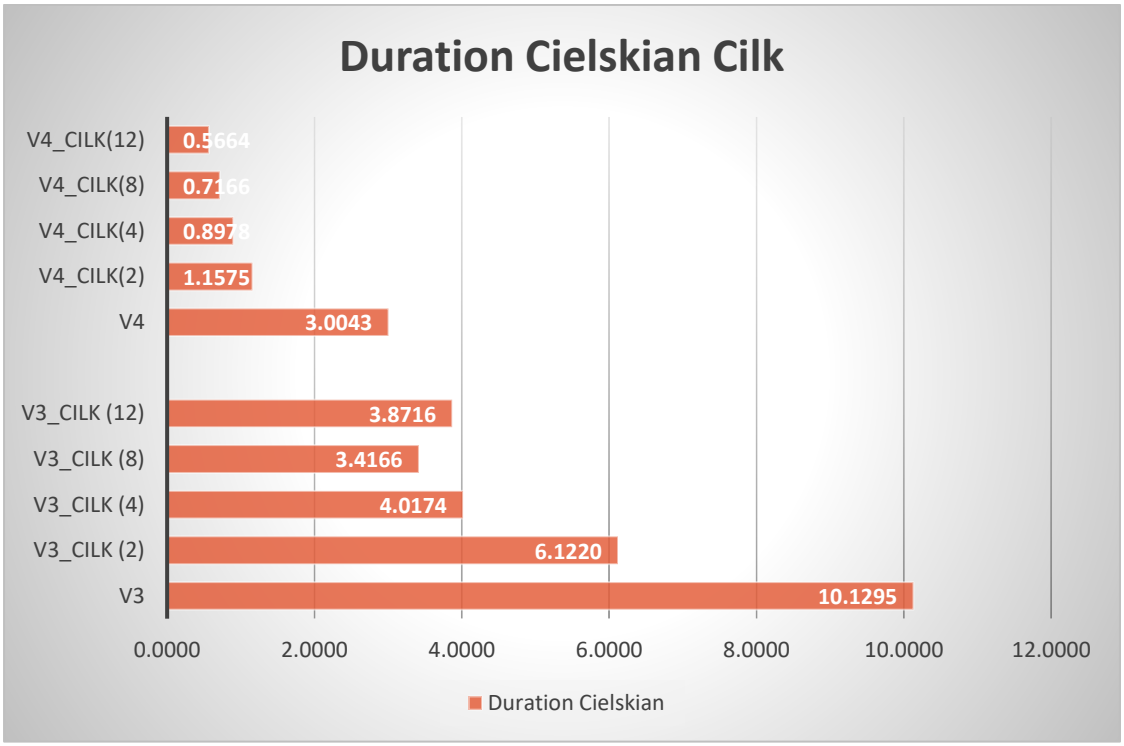
αποδείχθηκε να μην λειτουργεί πλήρως καθώς οι αποδόσεις σε χρόνο έδειχναν όλο και να μεγαλώνουν. Αρά με την αύξηση των workers αργούσε όλο και περισσότερο.

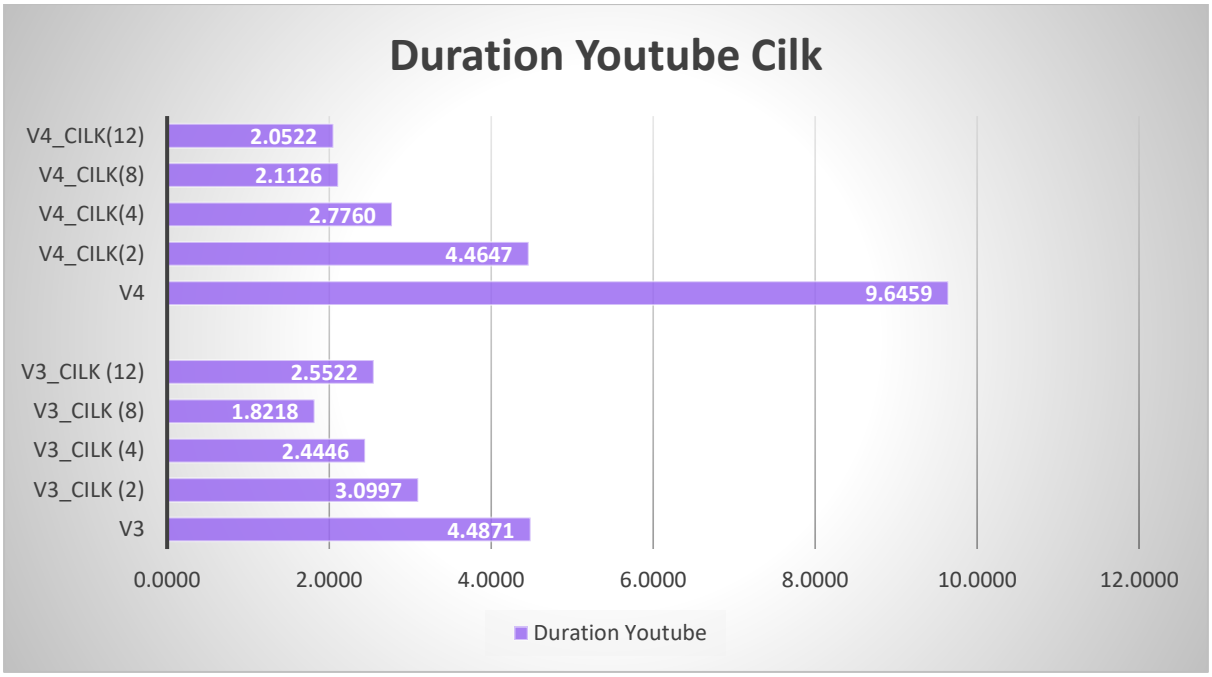
Σχόλια

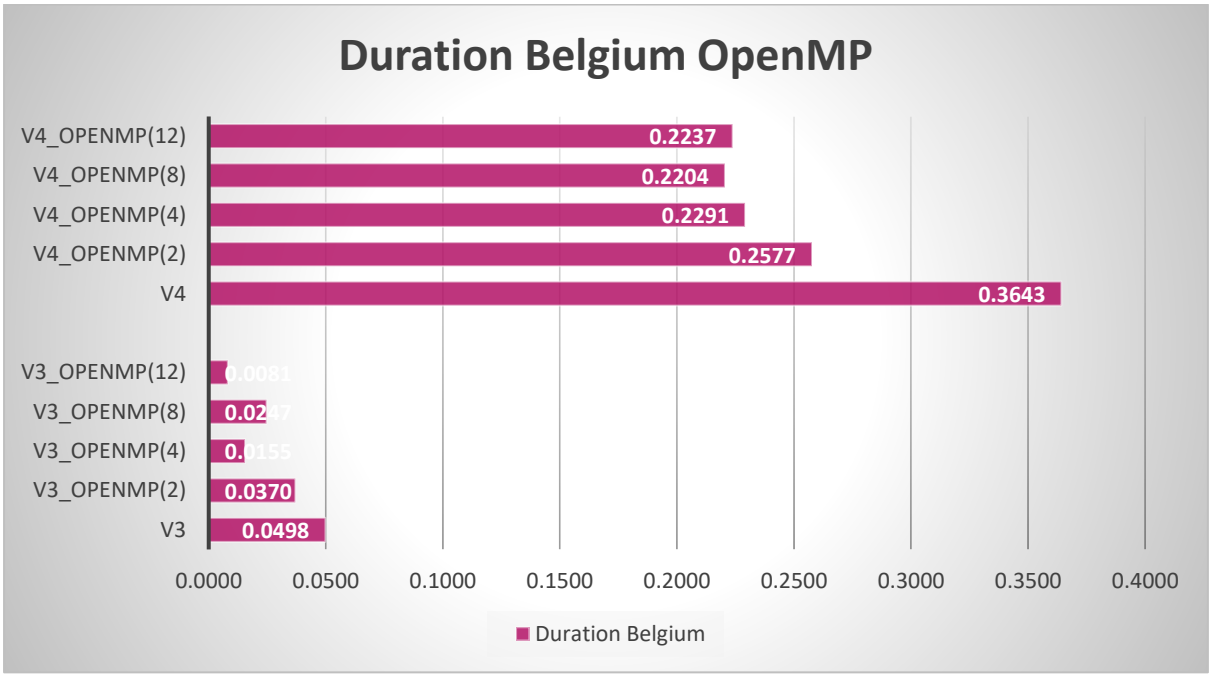
Σε γενικές γραμμές φάνηκε πως ο triangle_v3 ήταν γρηγορότερος από τον v4 πράγμα που βέβαια μπορεί αν οφείλεται στην υλοποίηση και μια διαφορετική να έδειχνε καλυτέρα αποτελέσματα.

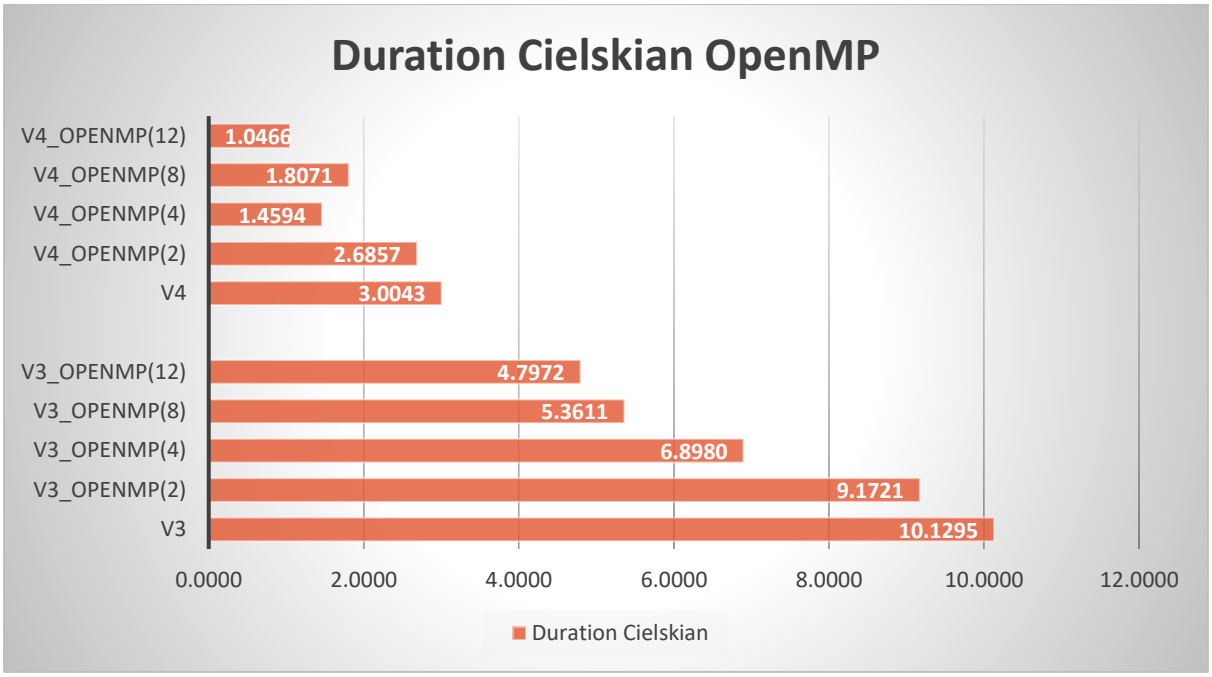
Επίσης περίεργο ήταν στην threads που ο μεγαλύτερος αριθμός worker φαίνεται να αύξανε την διάρκεια εκτέλεσης της εργασίας. Ίσως προβληματικό ήταν και το σημείο από το οποίο ξεκινούσε να μετράει χρόνο το πρόγραμμα.

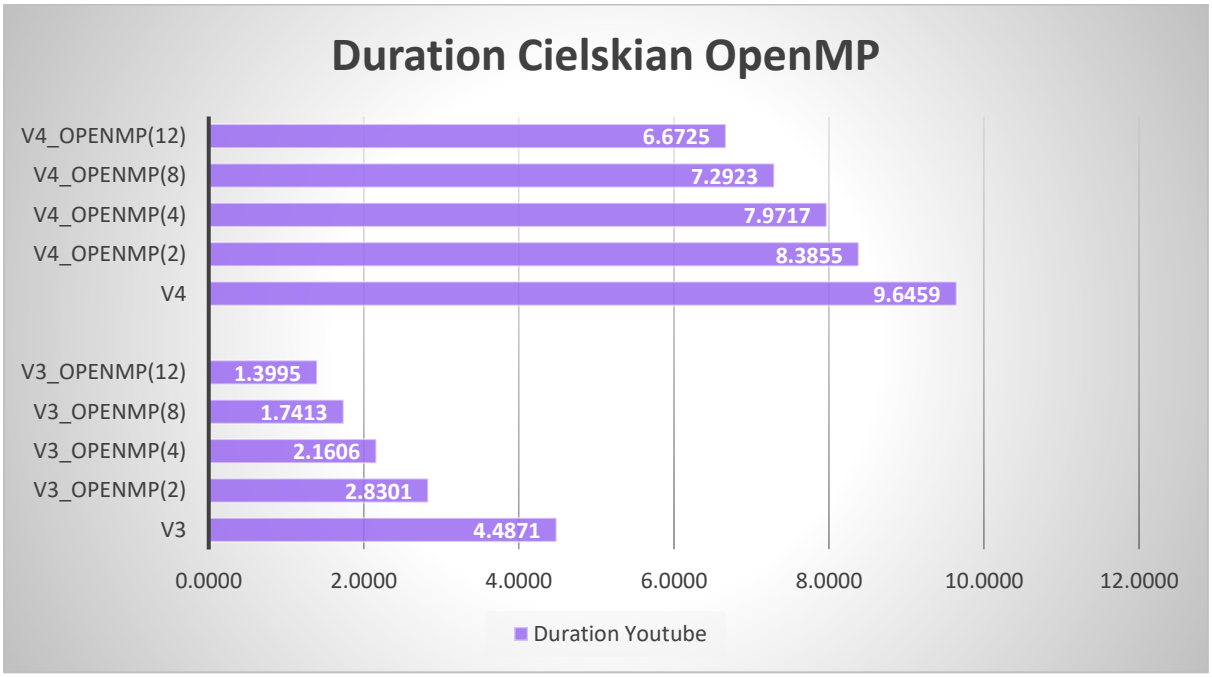
ΔΙΑΓΡΑΜΜΑΤΑ:

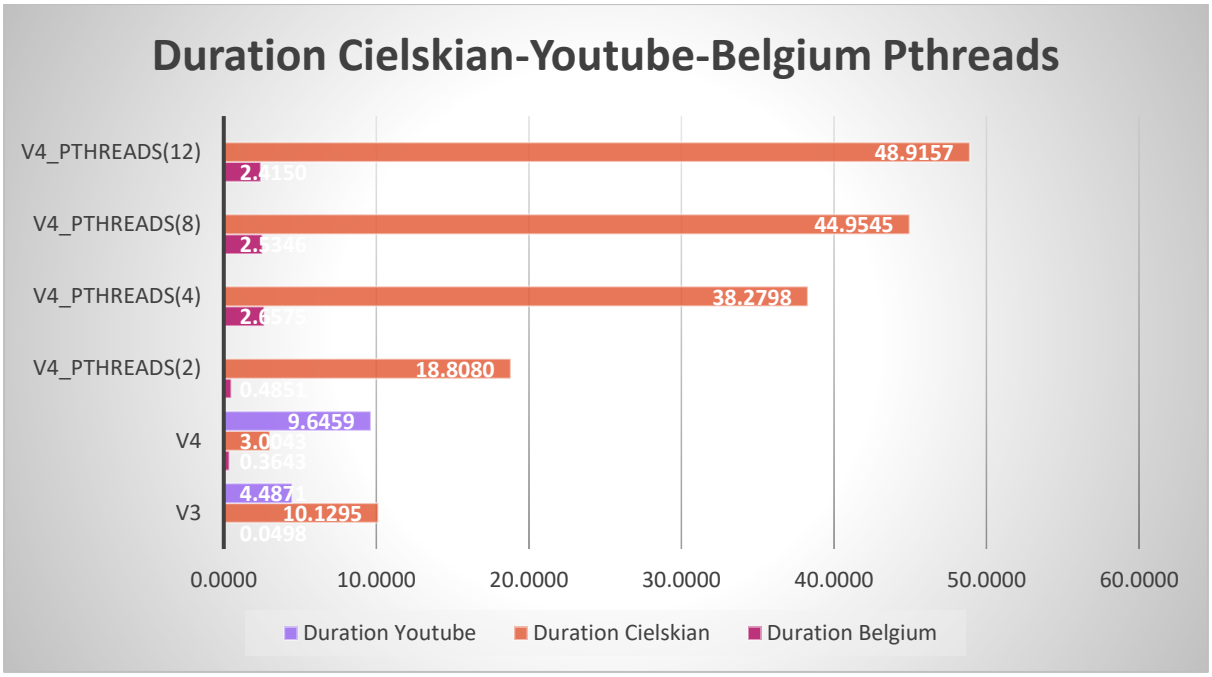


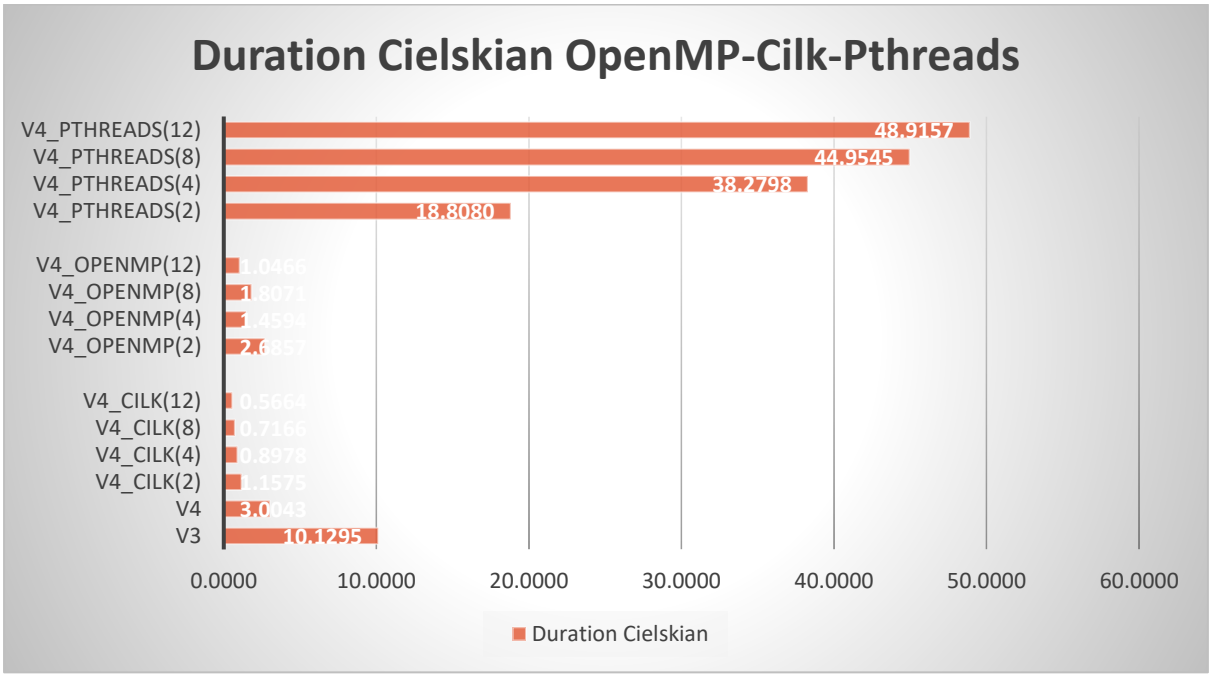

Duration Youtube Cilk
V4_CILK(12)
V4_CILK(8)
V4_CILK(4)
V4_CILK(2)
V4
V3_CILK (12)
V3_CILK (8)
V3_CILK (4)
V3_CILK (2)
V3
2.0522
2.1126
2.7760
4.4647
9.6459
2.5522
1.8218
2.4446
3.0997
4.4871
0.0000
2.0000
4.0000
6.0000
8.0000
10.0000
12.0000
Duration Youtube

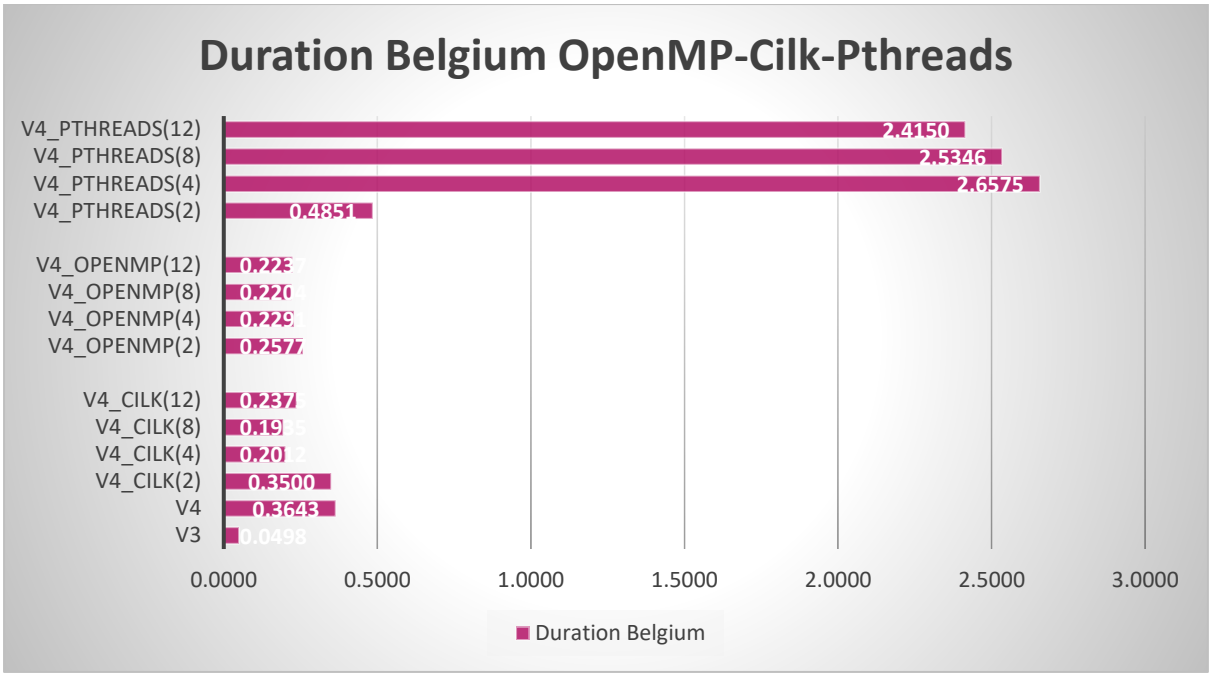

Duration Belgium OpenMP
V4_OPENMP(12)
V4_OPENMP(8)
V4_OPENMP(4)
V4_OPENMP(2)
V4
V3_OPENMP(12)
V3_OPENMP(8)
V3_OPENMP(4)
V3_OPENMP(2)
V3
0.2237
0.2204
0.2291
0.2577
0.3643
0.0370
0.0498
0.0000
0.0500
0.1000
0.1500
0.2000
0.2500
0.3000
0.3500
0.4000
Duration Belgium


Duration Cielskian OpenMP
V4_OPENMP(12)
V4_OPENMP(8)
V4_OPENMP(4)
V4_OPENMP(2)
V4
V3_OPENMP(12)
V3_OPENMP(8)
V3_OPENMP(4)
V3_OPENMP(2)
V3
1.0466
1.8071
1.4594
2.6857
3.0043
4.7972
5.3611
6.8980
9.1721
10.1295
0.0000
2.0000
4.0000
6.0000
8.0000
10.0000
12.0000
Duration Cielskian


Duration Cielskian OpenMP
V4_OPENMP(12)
V4_OPENMP(8)
V4_OPENMP(4)
V4_OPENMP(2)
V4
V3_OPENMP(12)
V3_OPENMP(8)
V3_OPENMP(4)
V3_OPENMP(2)
V3
6.6725
7.2923
7.9717
8.3855
9.6459
1.3995
1.7413
2.1606
2.8301
4.4871
0.0000
2.0000
4.0000
6.0000
8.0000
10.0000
12.0000
Duration Youtube


Duration Cielskian-Youtube-Belgium Pthreads
V4_PTHREADS(12)
V4_PTHREADS(8)
V4_PTHREADS(4)
V4_PTHREADS(2)
V4
V3
48.9157
44.9545
38.2798
18.8080
9.6459
10.1295
0.0000
10.0000
20.0000
30.0000
40.0000
50.0000
60.0000
Duration Youtube
Duration Cielskian
Duration Belgium


Duration Cielskian OpenMP-Cilk-Pthreads
V4_PTHREADS(12)
V4_PTHREADS(8)
V4_PTHREADS(4)
V4_PTHREADS(2)
V4_OPENMP(12)
V4_OPENMP(8)
V4_OPENMP(4)
V4_OPENMP(2)
V4_CILK(12)
V4_CILK(8)
V4_CILK(4)
V4_CILK(2)
V4
V3
48.9157
44.9545
38.2798
18.8080
10.1295
0.0000
10.0000
20.0000
30.0000
40.0000
50.0000
60.0000
Duration Cielskian


Duration Belgium OpenMP-Cilk-Pthreads
V4_PTHREADS(12)
V4_PTHREADS(8)
V4_PTHREADS(4)
V4_PTHREADS(2)
V4_OPENMP(12)
V4_OPENMP(8)
V4_OPENMP(4)
V4_OPENMP(2)
V4_CILK(12)
V4_CILK(8)
V4_CILK(4)
V4_CILK(2)
V4
V3
2.4150
2.5346
2.6575
0.4851
0.3500
0.3643
0.0498
0.0000
0.5000
1.0000
1.5000
2.0000
2.5000
3.0000
Duration Belgium