

Συστήματα Παράλληλης Επεξεργασίας

Σειρά 2^η - Τελική Αναφορά



Ομάδα: parlab30

Τσαγκαράκης Στυλιανός - 03115180

Τσάκας Νικόλαος - 03115433

Δεδομένα

Ως δεδομένα στην Άσκηση δόθηκαν οι σειριακές υλοποιήσεις των αλγορίθμων Jacobi (1), Gauss-Seidel-SOR (2) και Red-Black-SOR (3).

Ζητούμενα

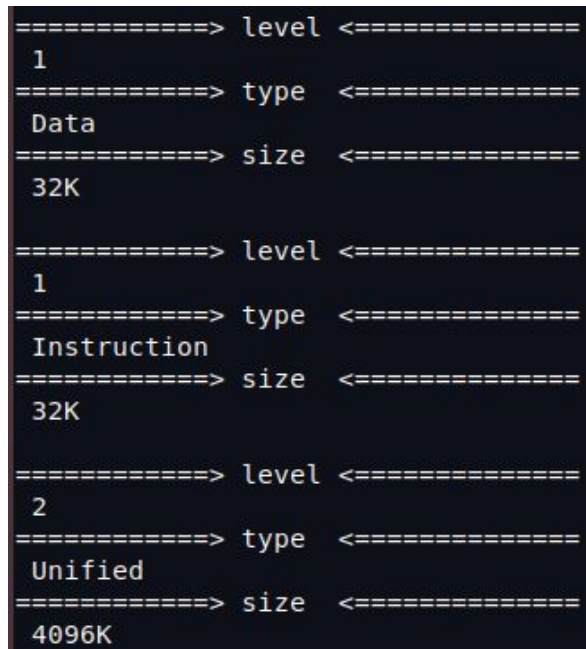
Ζητήθηκε η παράλληλη υλοποίηση των αλγορίθμων (1), (2), (3)-προαιρετικά με τη χρήση MPI. Πραγματοποιήθηκαν μετρήσεις για την παράλληλη υλοποίηση, και τα αποτελέσματα αυτών φαίνονται στα παρακάτω διαγράμματα.

Εδώ πέρα αξίζει να παραθέσουμε μια παρατήρηση από την ενδιάμεση αναφορά του πρώτου εργαστηρίου σχετικά με το μέγεθος της cache σε κάθε μηχανήμα στην ουρά parlab. Με το παρακάτω script τυπώνουμε τα επιθυμητά μεγέθη:

```
module load openmp

for d in /sys/devices/system/cpu/cpu0/cache/index*;
do tail -c+1 $d/{level,type,size}
echo
done
```

Και έχουμε τα αποτελέσματα:



```
=====> level <=====
1
=====> type  <=====
Data
=====> size  <=====
32K

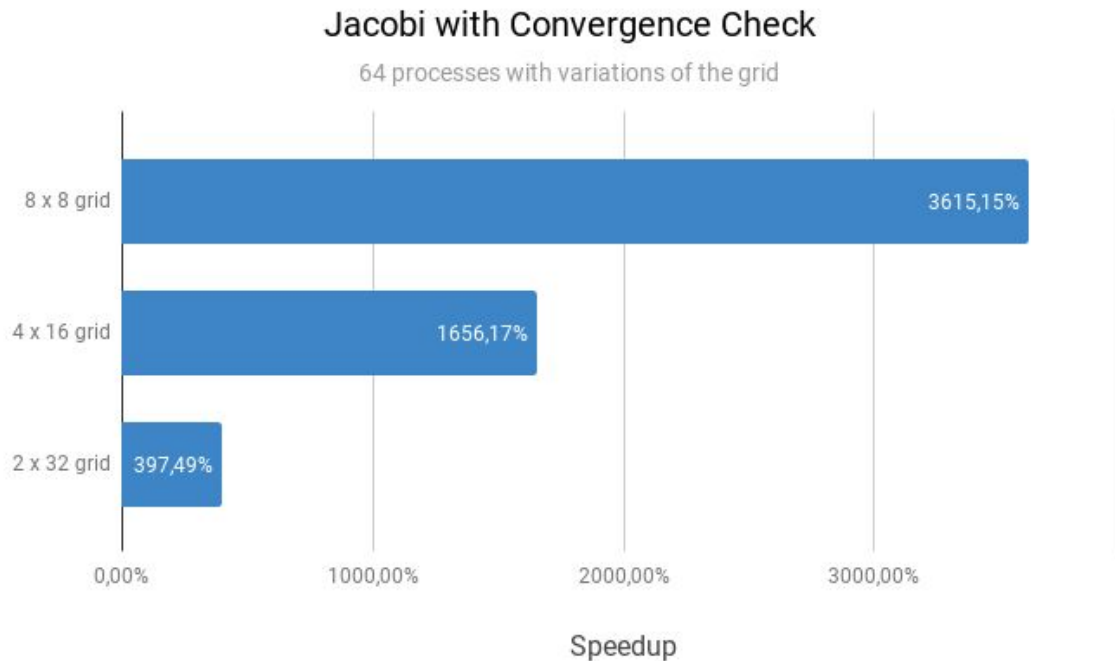
=====> level <=====
1
=====> type  <=====
Instruction
=====> size  <=====
32K

=====> level <=====
2
=====> type  <=====
Unified
=====> size  <=====
4096K
```

Συνεπώς θα μπορούσε κανείς να αποδώσει και κάποιες καθυστερήσεις του συστήματος στο μέγεθος της cache κάθε μηχανήματος. Καθώς ανεβαίνει ο αριθμός των διεργασιών, αναμένουμε ότι η cache θα ανανεώνεται συνεχώς έχοντας πάρα πολύ συχνά misses, συνεπώς θα επηρεάσει άμεσα την επίδοση του συστήματος.

Μετρήσεις με έλεγχο σύγκλισης

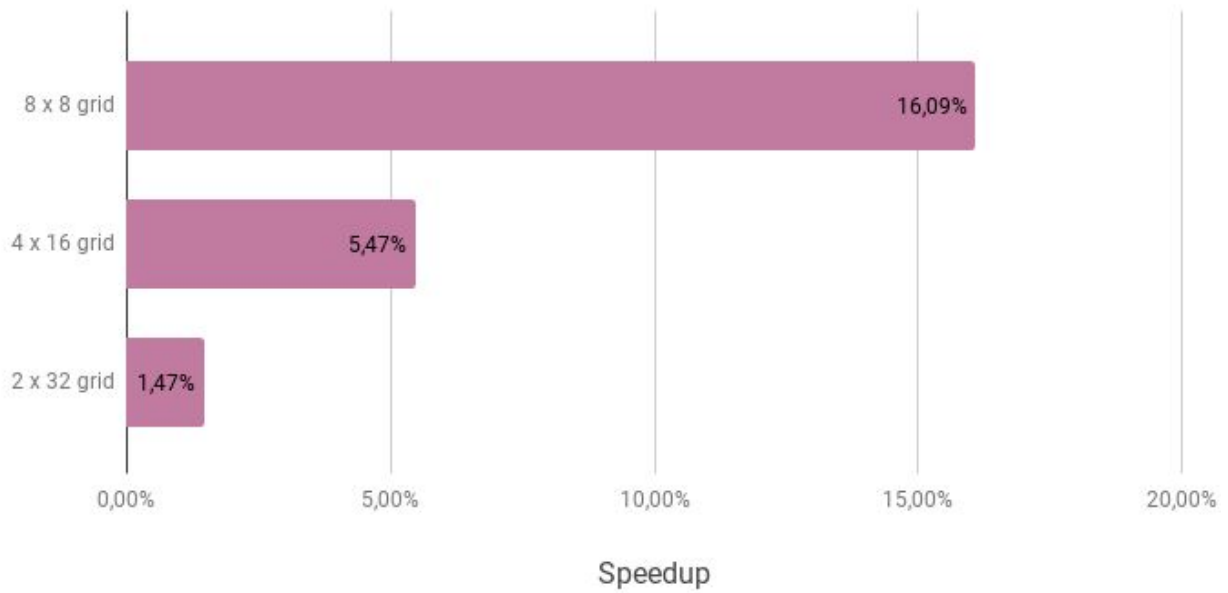
Στα 3 επόμενα διαγράμματα παρουσιάζεται το speedup που έχει κάθε μέθοδος με σταθερό μέγεθος πίνακα ($N=1024$) και μεταβλητό το πλέγμα των διεργασιών, όπως φαίνεται στον κατακόρυφο άξονα.



Στα διαγράμματα αυτά βλέπουμε πως το speedup που έχει το πρόγραμμά μας μειώνεται όσο αλλάζει το grid. Η καθυστέρηση αυτή οφείλεται στο γεγονός πως όσο μειώνουμε τις σειρές (ή τις στήλες αντίστοιχα) τόσο πιο λίγη επικοινωνία (άρα και μικρότερη επιτάχυνση) έχουμε με τις north ή και south διεργασίες της κάθε διεργασίας.

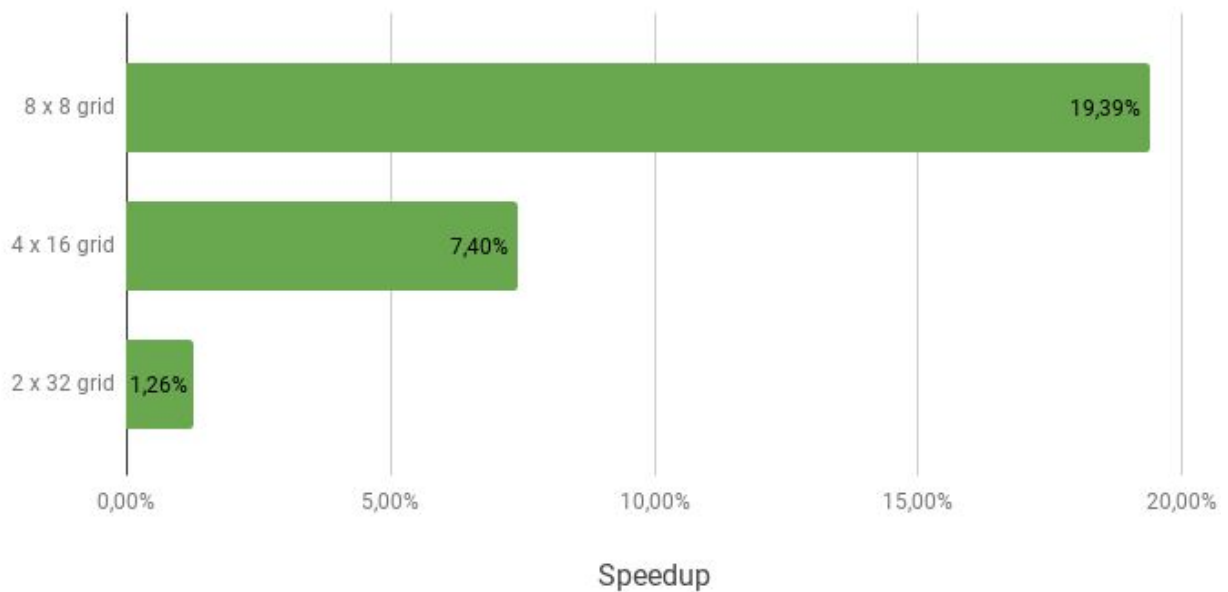
Gauss-Seidel-SOR with Convergence Check

64 processes with variations of the grid

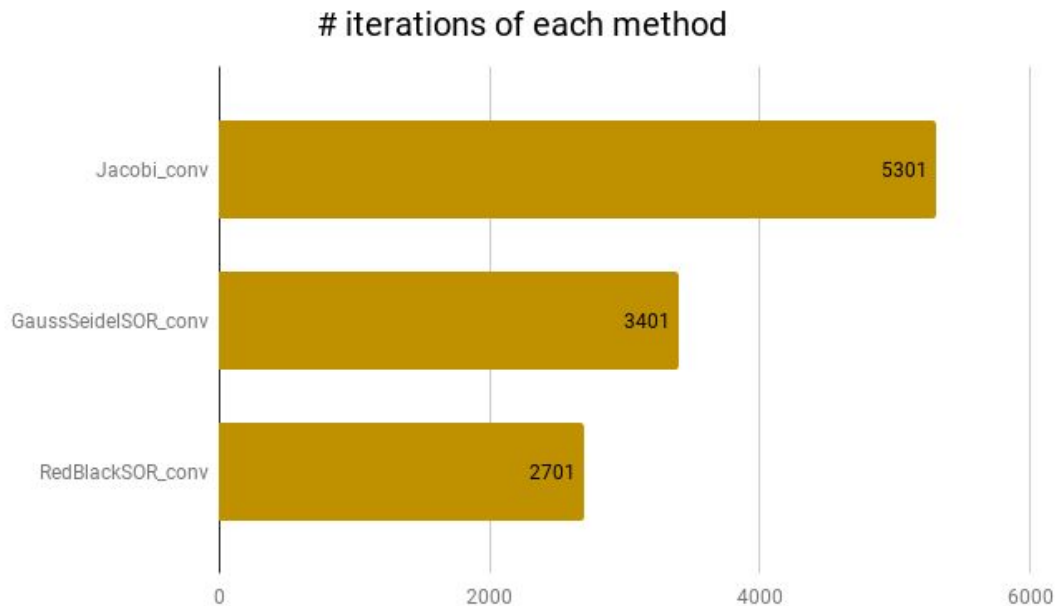


Red-Black-SOR with Convergence Check

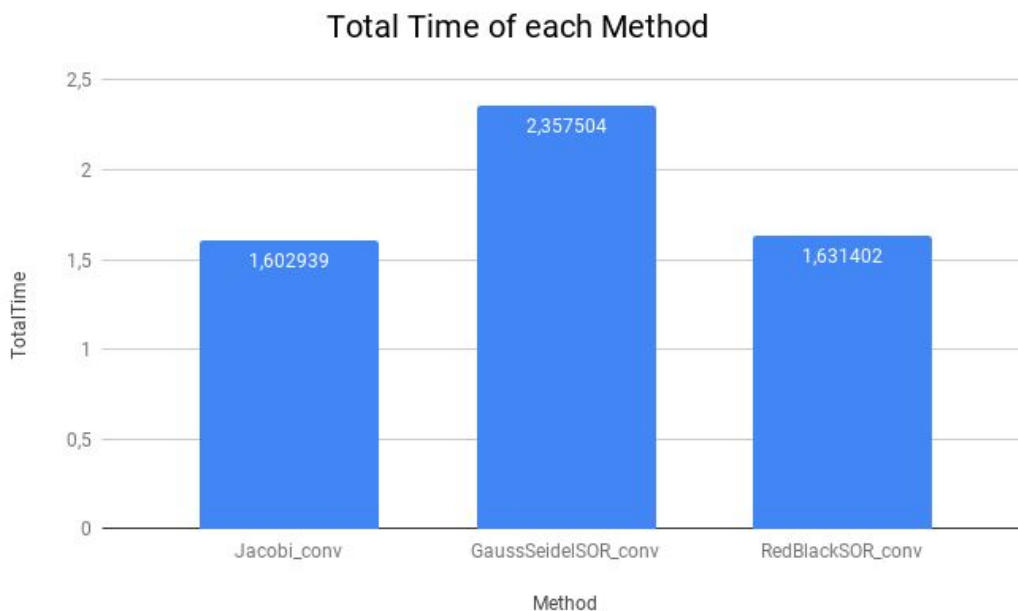
64 processes with variations of the grid



Στο παρακάτω διάγραμμα βλέπουμε τον αριθμό των επαναλήψεων για κάθε μέθοδο. Η μέθοδος Jacobi φαίνεται ότι χρειάζεται περισσότερες επαναλήψεις για να ελέγξει τη σύγκλιση και αυτό πιθανότατα κοστίζει σε μνήμη.



Στο επόμενο διάγραμμα φαίνεται πως η πιο γρήγορη μέθοδος είναι η Jacobi, όμως με μικρή διαφορά σε σχέση με την Red-Black.

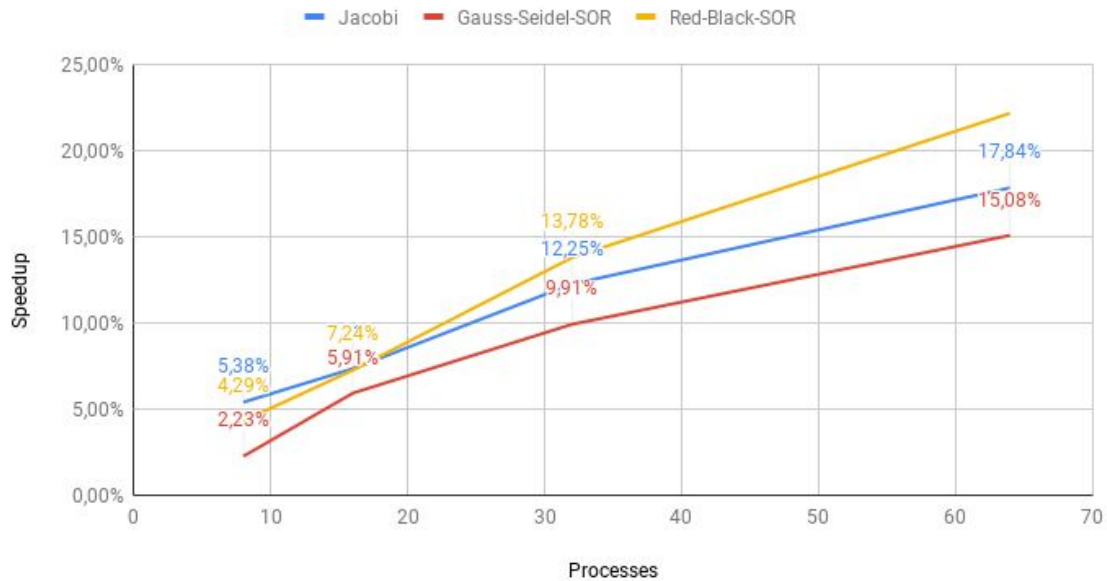


Επειδή η **Red-Black** έχει κατα πολύ λιγότερες επικοινωνίες μέσω του MPI, αυτό θα βοηθούσε σε ένα σύστημα κατανομημένης μνήμης, οπότε θα επιλέγαμε τη μέθοδο αυτή.

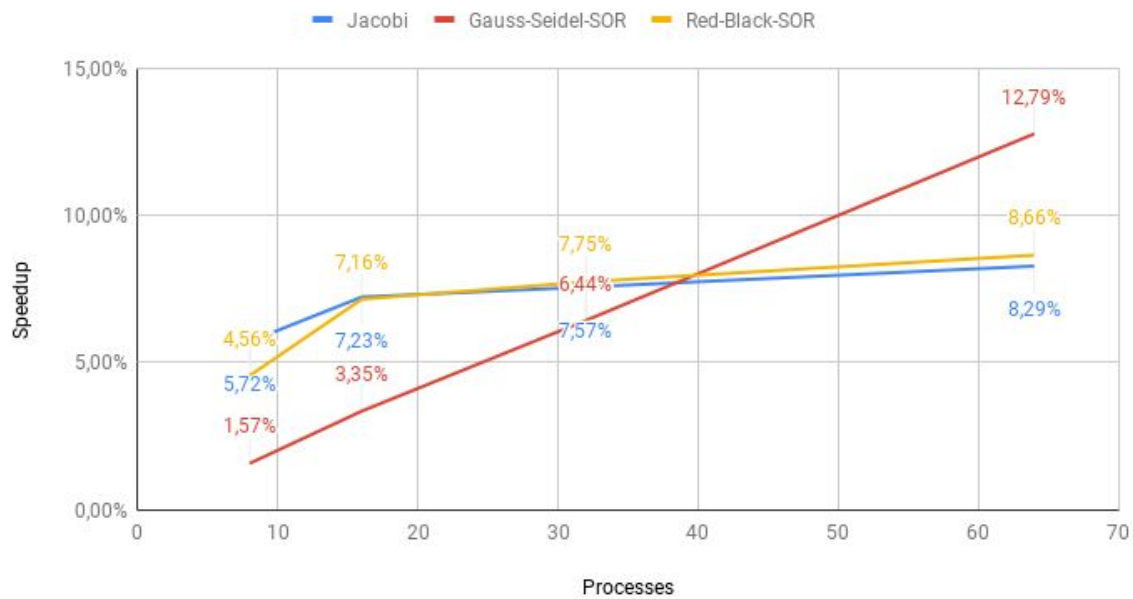
Μετρήσεις χωρίς έλεγχο σύγκλισης

Στα παρακάτω διαγράμματα φαίνεται το speedup κάθε μεθόδου με σταθερό μέγεθος πίνακα. (2048, 4096 και 6144).

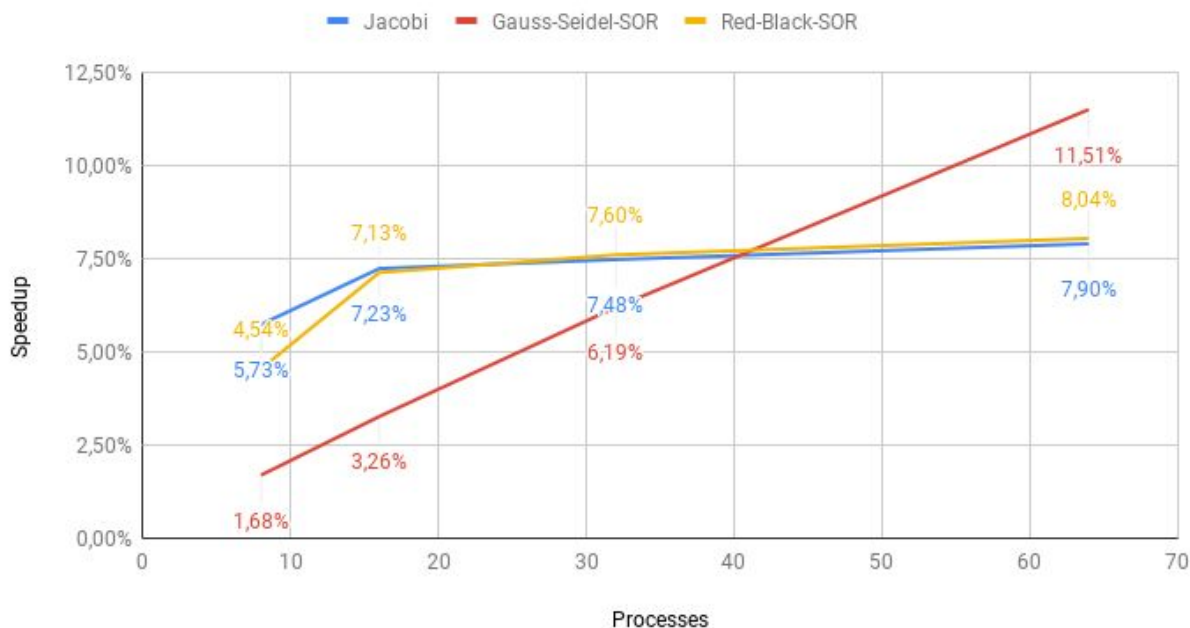
Speedup of Jacobi, Gauss-Seidel-SOR and Red-Black-SOR with N=2048



Speedup of Jacobi, Gauss-Seidel-SOR and Red-Black-SOR with N=4096



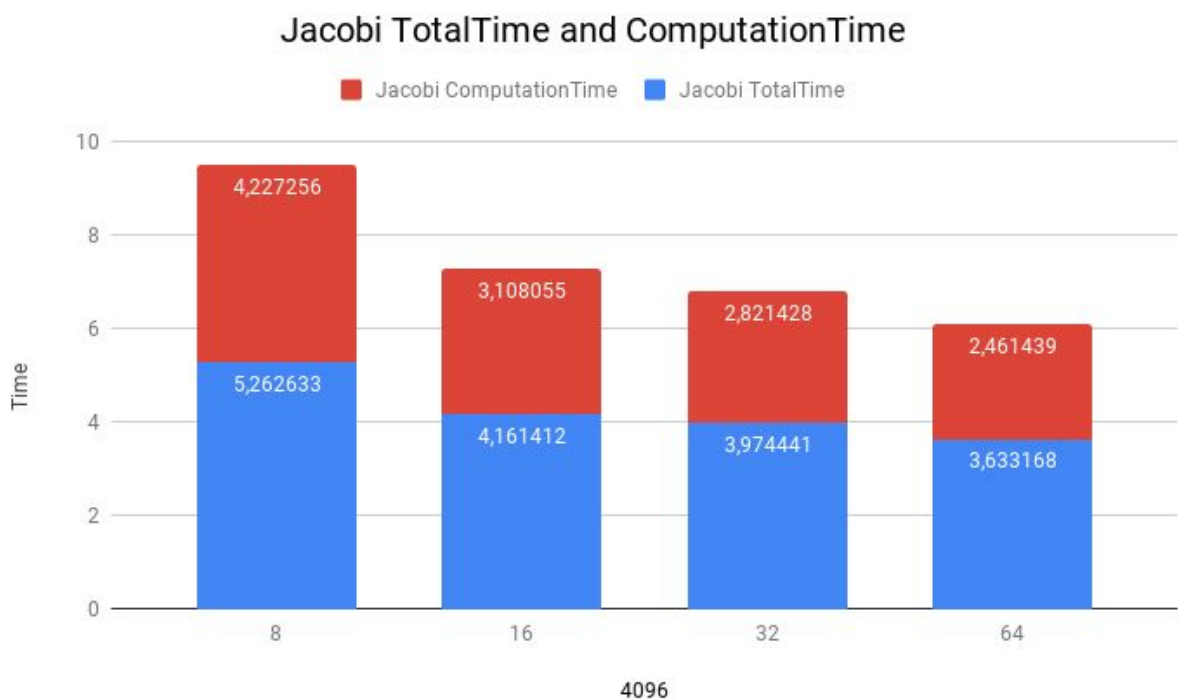
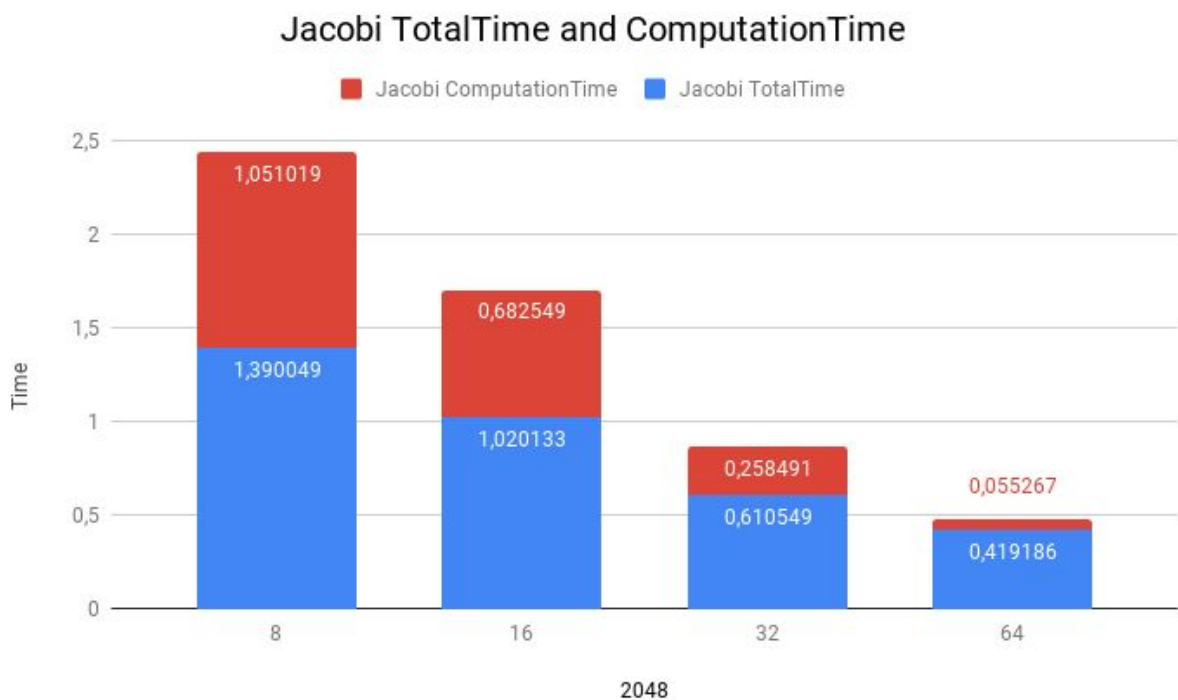
Speedup of Jacobi, Gauss-Seidel-SOR and Red-Black-SOR with N=6144



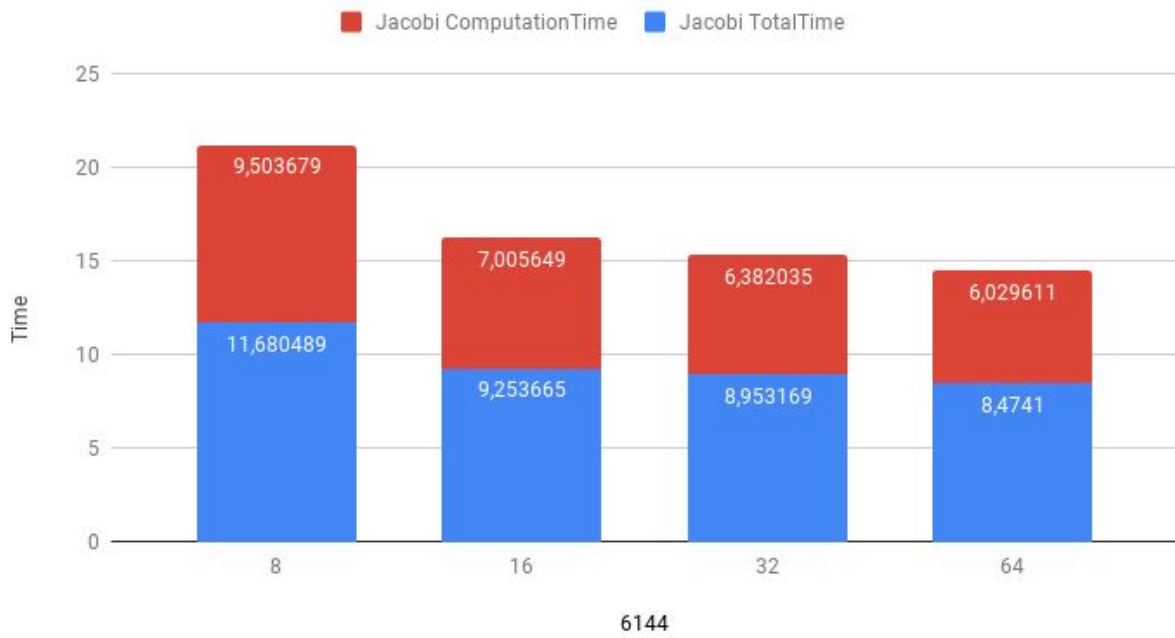
Παρατηρούμε πως η Gauss-Seidel-SOR έχει ένα σταθερό speedup σε κάθε διπλασιασμό διεργασιών σε όλα τα μεγέθη πίνακα. Αντίθετα οι άλλες δύο μέθοδοι έχουν κάνουν ένα “γόνατο” στα μεγάλα μεγέθη (4096, 6144) και δεν επιταχύνονται άλλο ενώ στο $N = 2048$ έχουν σταθερό speedup.

Αυτό οφείλεται στο γεγονός ότι η Gauss-Seidel-SOR περιμένει από το επόμενο iteration πληροφορίες και έτσι προχωράει πιο γρήγορα, καθώς δεν περιμένει όλους τους γείτονές της να τελειώσουν το συγκεκριμένο iteration. Αντίθετα, οι άλλες μέθοδοι περιμένουν να τελειώσει το συγκεκριμένο iteration και μετά προχωρούν στα επόμενα βήματα. Επειδή μετά από ένα σημείο (processes = 16) ο φόρτος επικοινωνίας μεγαλώνει και περιμένουν ακόμα περισσότεροι, το σύστημα καλεί την `MPI_wait()` και περιμένει να τελειώσουν όλοι. Εξού και η καθυστέρηση.

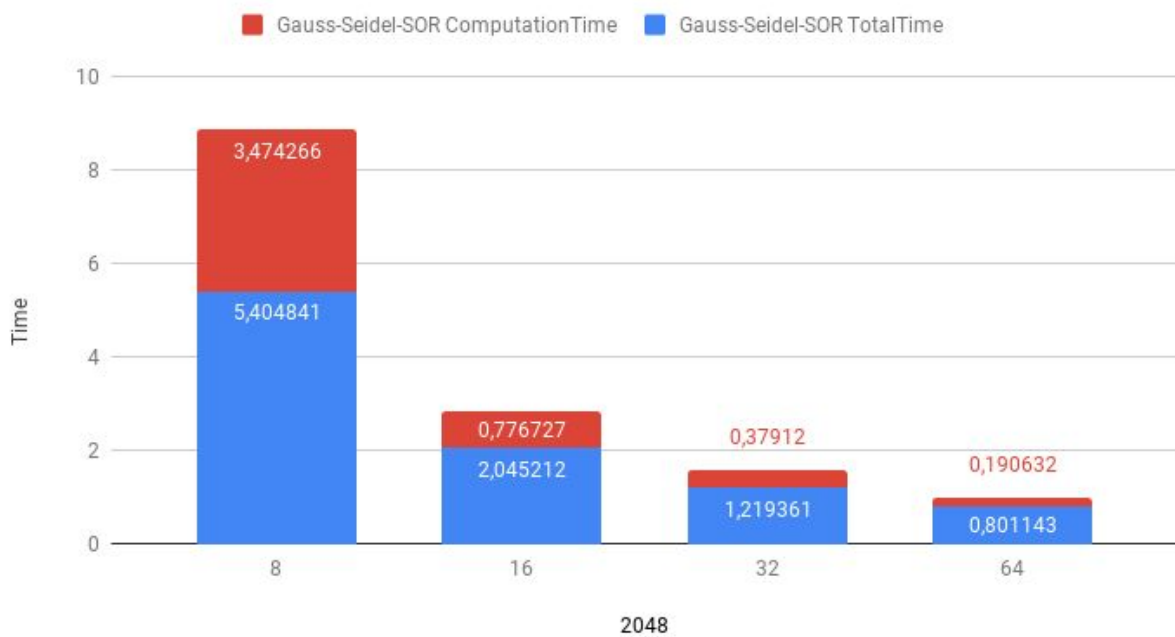
Στα παρακάτω διαγράμματα (διαγράμματα μπαρών στοίβας) φαίνεται ο χρόνος υπολογισμού και ο συνολικός χρόνος για κάθε μέθοδο για τιμές διεργασιών = {8, 16, 32, 64}. Στο τέλος αυτών των διαγραμμάτων υπάρχουν και τρία συγκεντρωτικά διαγράμματα, ένα για κάθε μέθοδο.



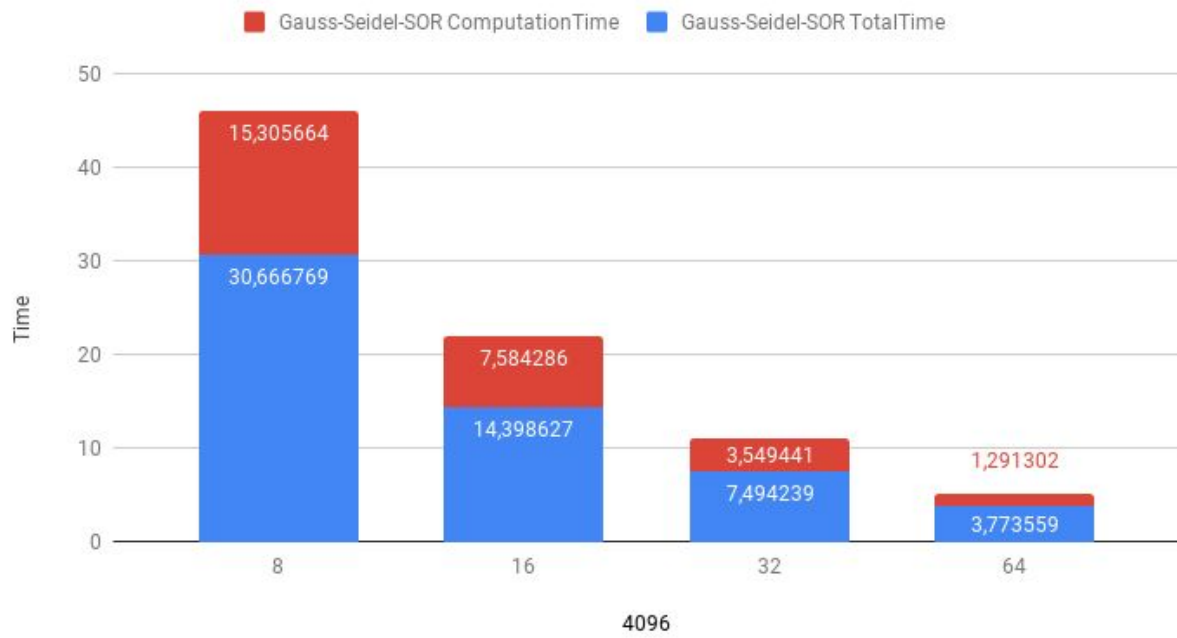
Jacobi TotalTime and ComputationTime



Gauss-Seidel-SOR TotalTime and ComputationTime



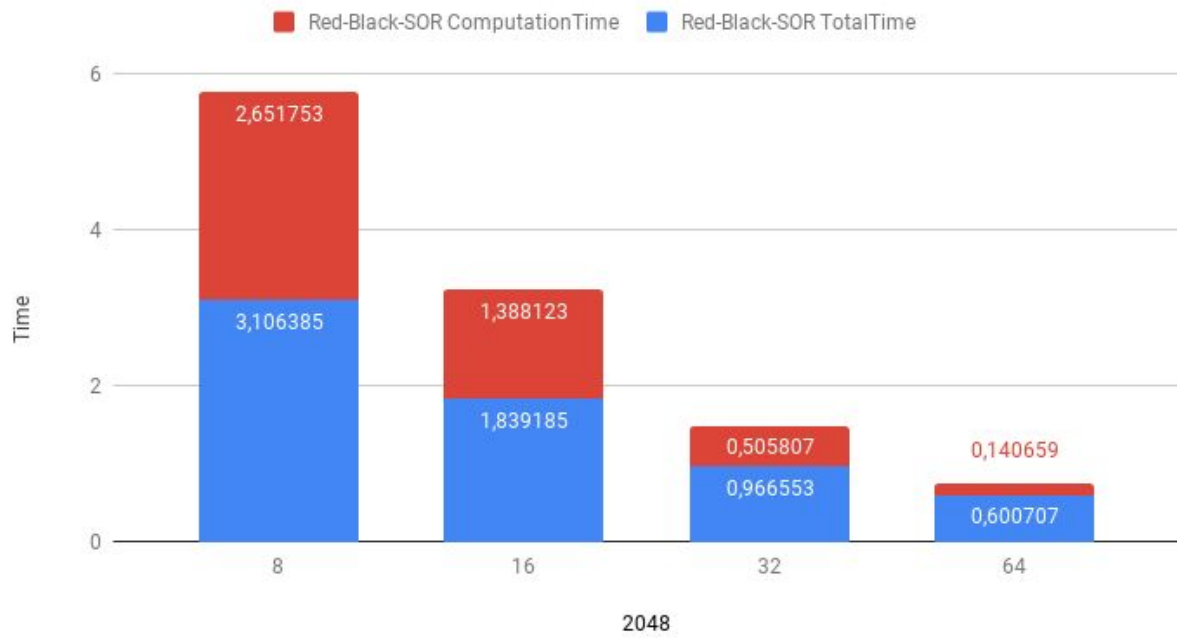
Gauss-Seidel-SOR TotalTime and ComputationTime



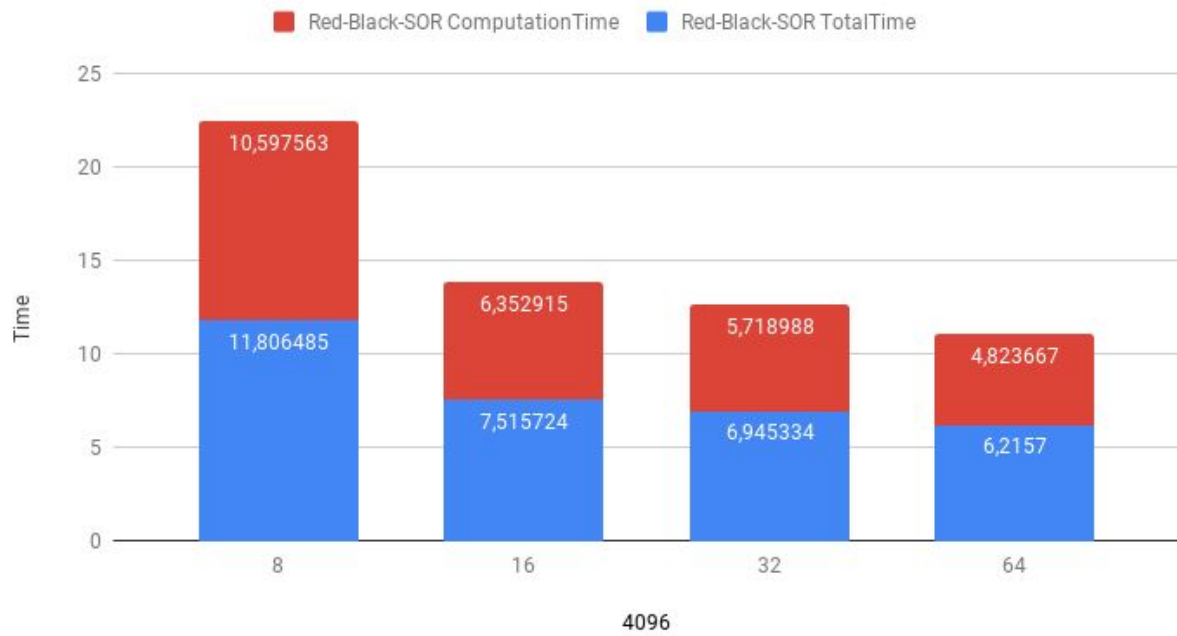
Gauss-Seidel-SOR TotalTime and ComputationTime



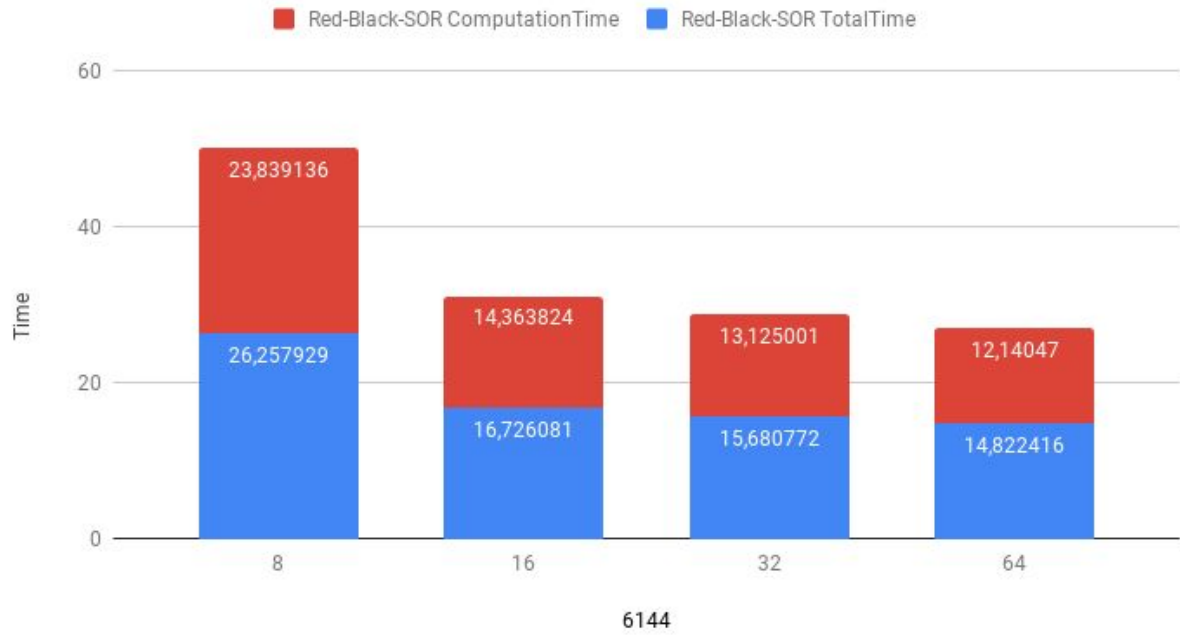
Red-Black-SOR TotalTime and ComputationTime



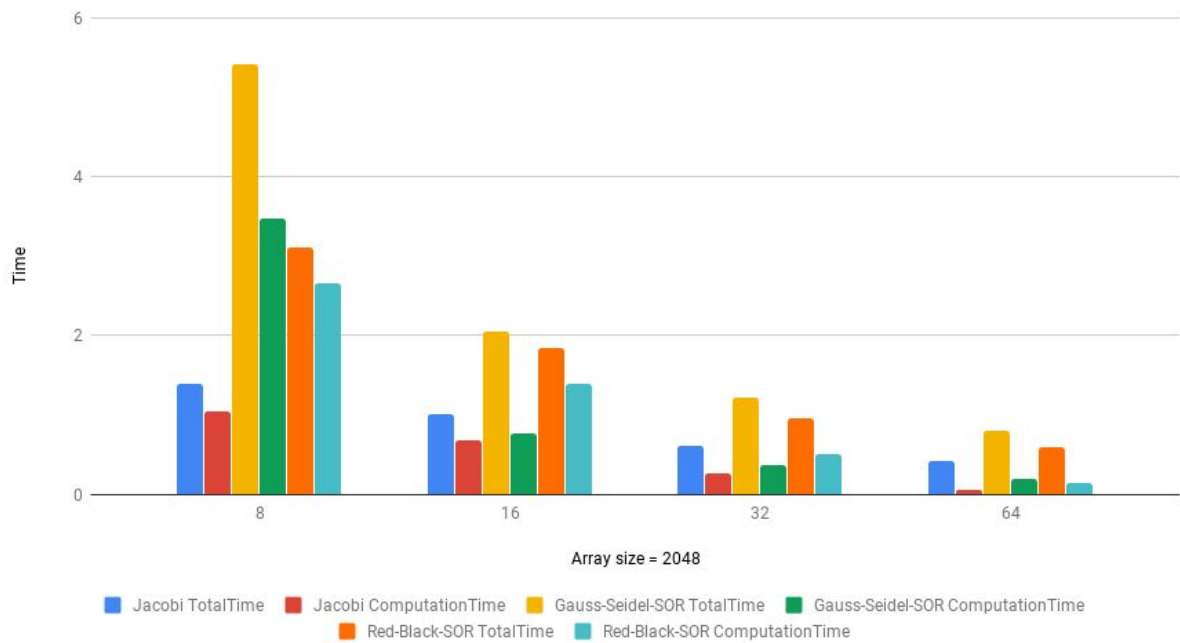
Red-Black-SOR TotalTime and ComputationTime

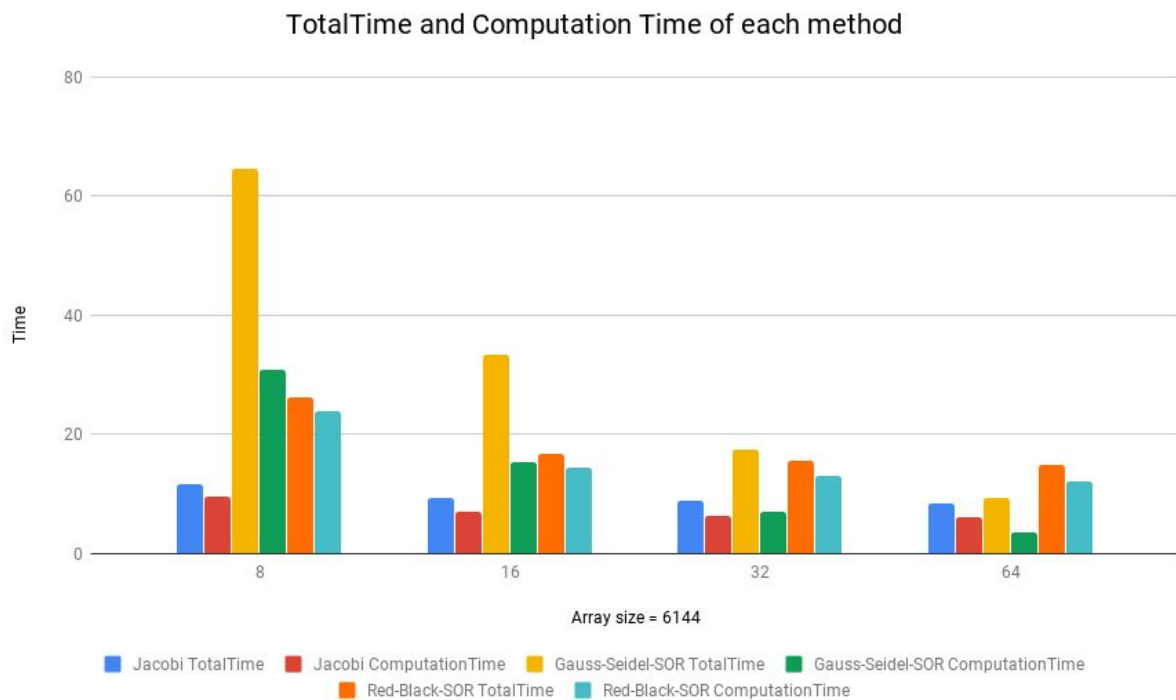
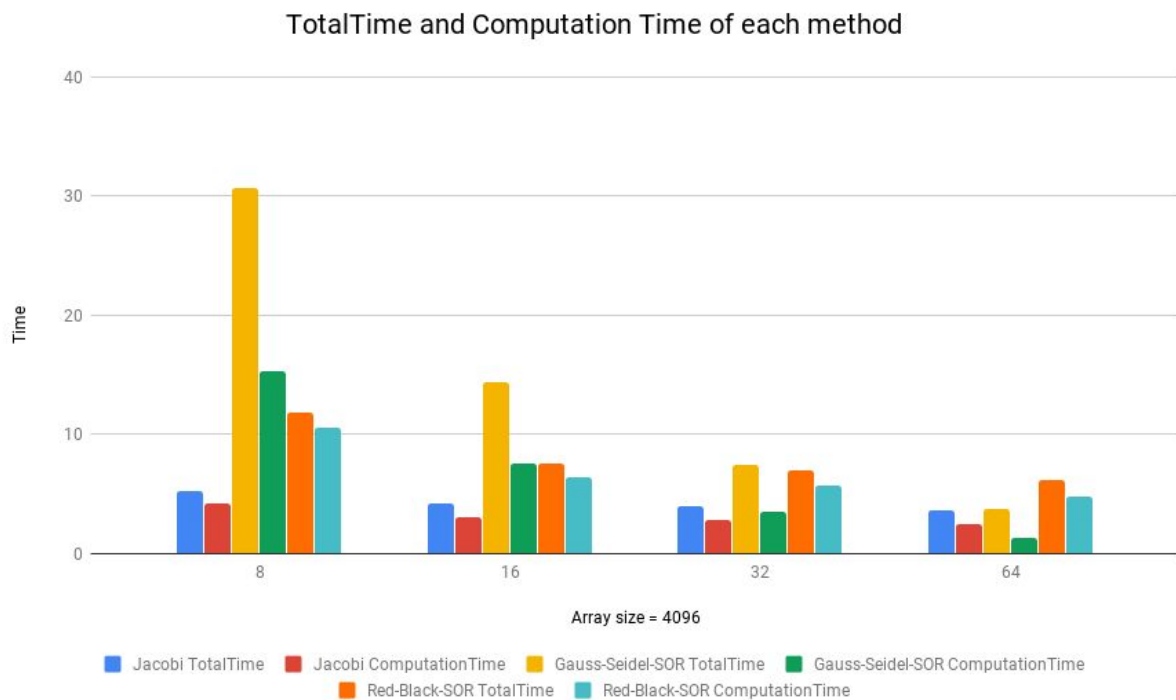


Red-Black-SOR TotalTime and ComputationTime



TotalTime and Computation Time of each method





Όπως βλέπουμε στα διαγράμματα, ο χρόνος υπολογισμού είναι αντιστρόφως ανάλογος του speedup στα αντίστοιχα μεγέθη πινάκων. Στο Jacobi και στο Red-Black-SOR για $N=2048$

έχουμε σταθερή μείωση του χρόνου υπολογισμού ενώ για $N=4096$ και $N=6144$ έχουμε σταθερό περίπου χρόνο υπολογισμού.

Αντίστοιχα στην μέθοδο Gauss-Seidel-SOR, έχουμε σταθερή μείωση του χρόνου υπολογισμού. Επίσης βλέπουμε πως όσο μικρότερος είναι ο χρόνος υπολογισμού τόσο μικρότερος είναι και ο συνολικός χρόνος (αναλογικά).