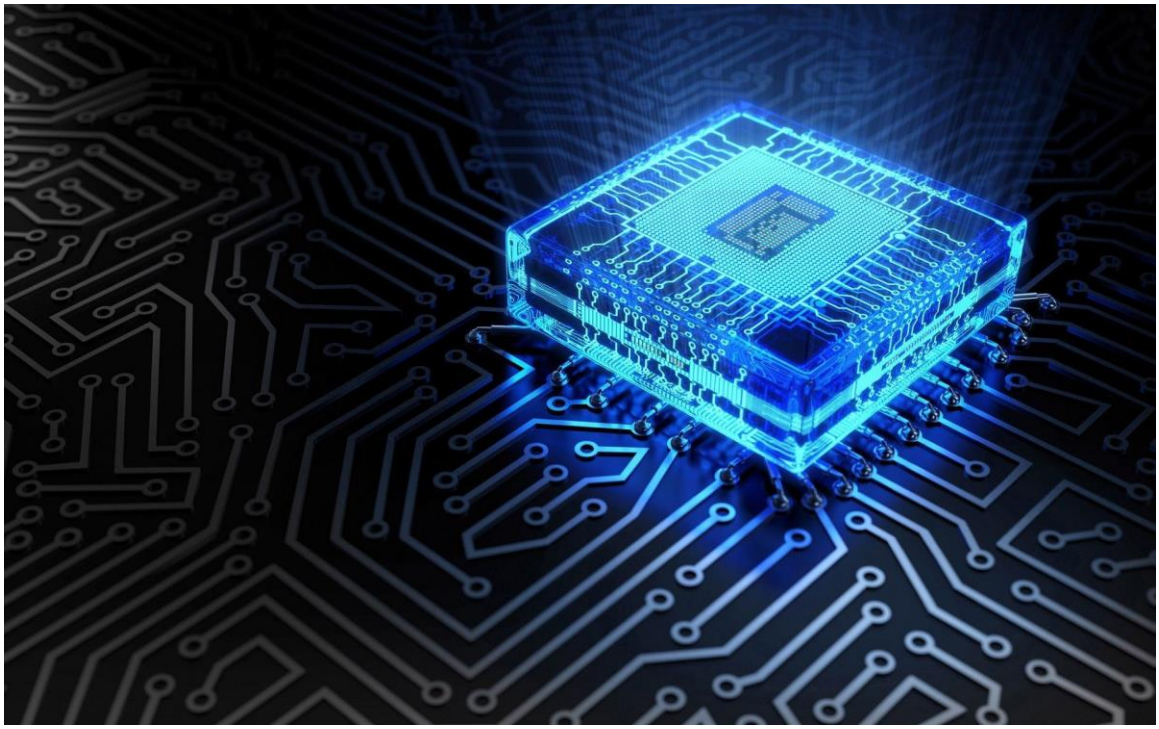
ΠΡΟΗΓΜΕΝΑ ΘΕΜΑΤΑ ΑΡΧΙΤΕΚΤΟΝΙΚΗΣ ΥΠΟΛΟΓΙΣΤΩΝ



ΑΣΚΗΣΗ 3: ΑΝΑΦΟΡΑ

10 ΙΟΥΛΙΟΥ, 2022

ΘΟΔΩΡΗΣ ΑΡΑΠΗΣ – EL18028

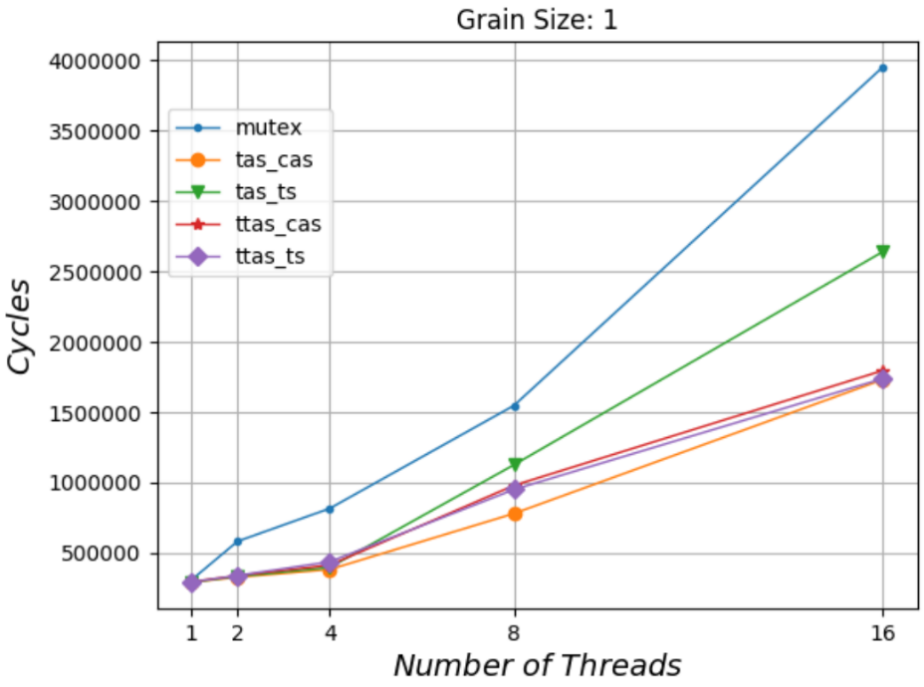
Αξιολόγηση Επίδοσης

4.1 Σύγκριση υλοποιήσεων

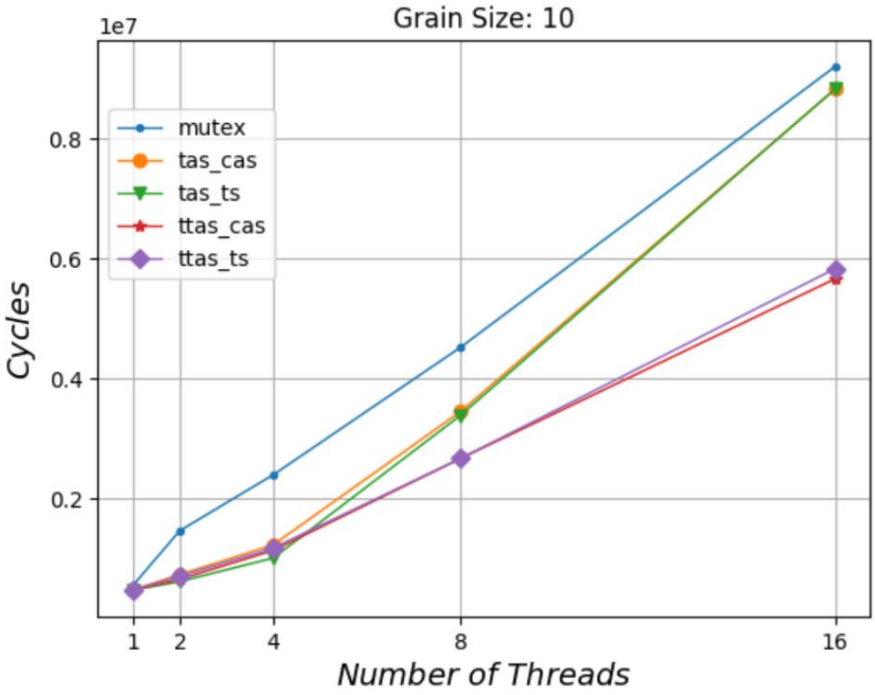
4.1.1

Τα ζητούμενα διαγράμματα είναι τα ακόλουθα:

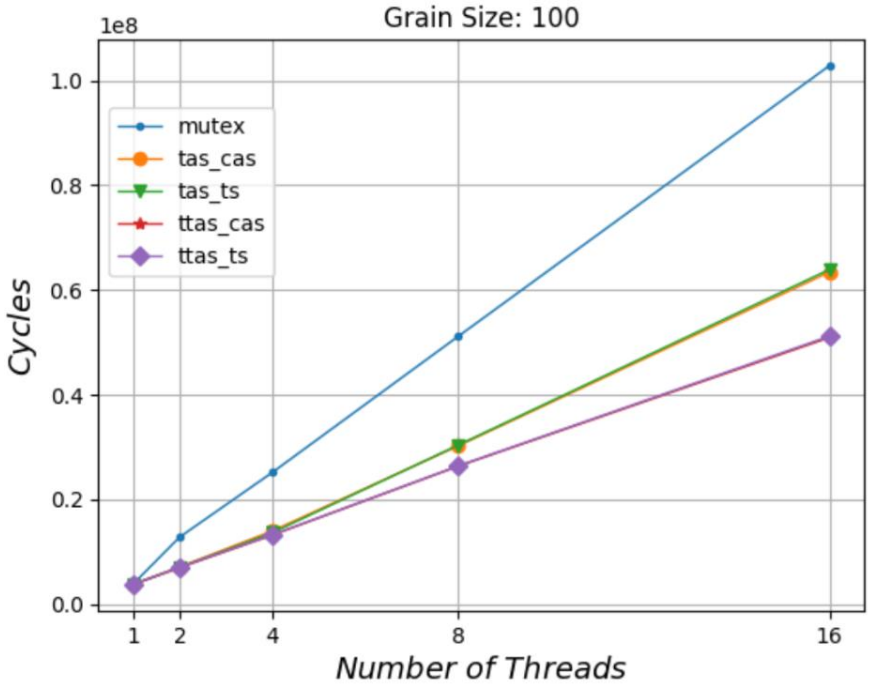
Grain size = 1



Grain size = 10



Grain size = 100



4.1.2

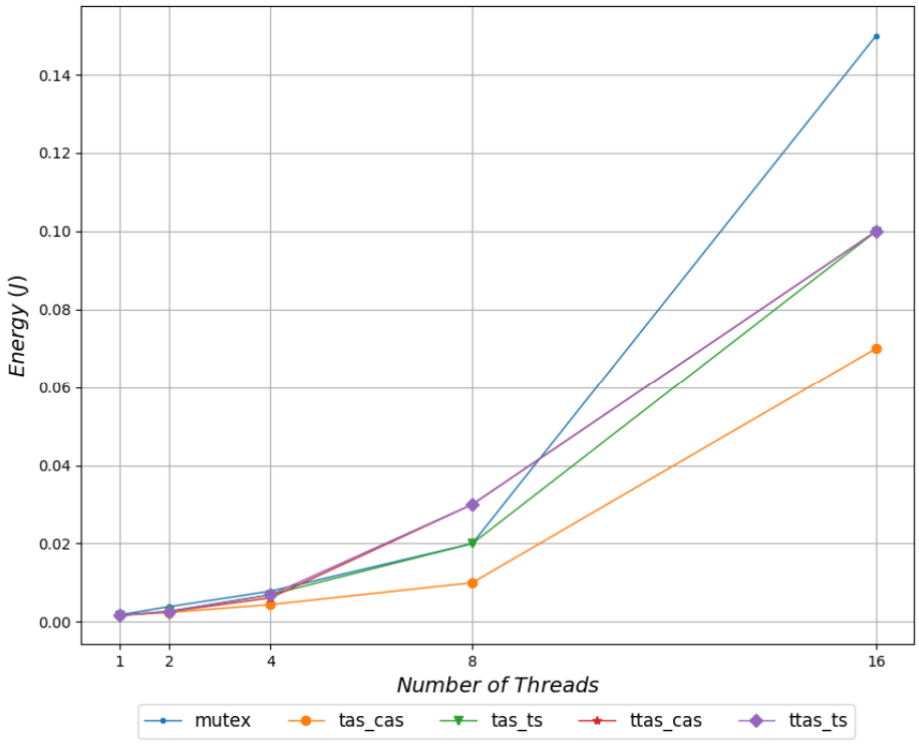
Παρατηρούμε ότι οι μηχανισμοί TAS_CAS και TAS_TS έχουν παρόμοιες επιδόσεις (με εξαίρεση για Grain size = 1), γεγονός που οφείλεται στο ότι χρησιμοποιούν και οι δύο το πρωτόκολλο Test-and-set. Όμοια, οι μηχανισμοί TTAS_CAS και TTAS_TS υλοποιούν το πρωτόκολλο Test-and-Test-and-set, με αποτέλεσμα να παρουσιάζουν κ αυτοί όμοια συμπεριφορά. Η διαφορά τους με τις Test-and-set υλοποιήσεις είναι ότι δεν γράφουν συνεχώς πάνω στο lock. Συγκεκριμένα, στις Test-and-set υλοποιήσεις, όταν 2 ή περισσότεροι επεξεργαστές προσπαθούν να πάρουν το lock έχουμε ως αποτέλεσμα κυκλικές εναλλαγές καταστάσεων (Modified – Invalid) της cache line που περιέχει το lock, πράγμα που προκαλεί αυξημένη περιττή κίνηση πάνω στο bus. Από την άλλη, οι Test-and-Test-and-set υλοποιήσεις εκτελούν πρώτα ένα load για να ελέγξουν αν είναι ελεύθερο το lock και αν ναι τότε δοκιμάζουν το atomic instruction το οποίο κάνει write το lock και προσπαθεί να το δεσμεύσει. Εν ολίγοις, πρώτα τσεκάρουν το lock και μετά κάνουν test-and-set. Συνεπώς, όσο το lock δεν είναι διαθέσιμο οι επεξεργαστές κάνουν spinning τοπικά στην cache τους με αποτέλεσμα να μην γίνεται αχρείαστο broadcasting στο bus. Γι αυτό αναμένουμε οι μηχανισμοί Test and-Test-and-set να πετυχαίνουν καλύτερη κλιμακωσιμότητα και επίδοση σε σχέση με τους υπόλοιπους.

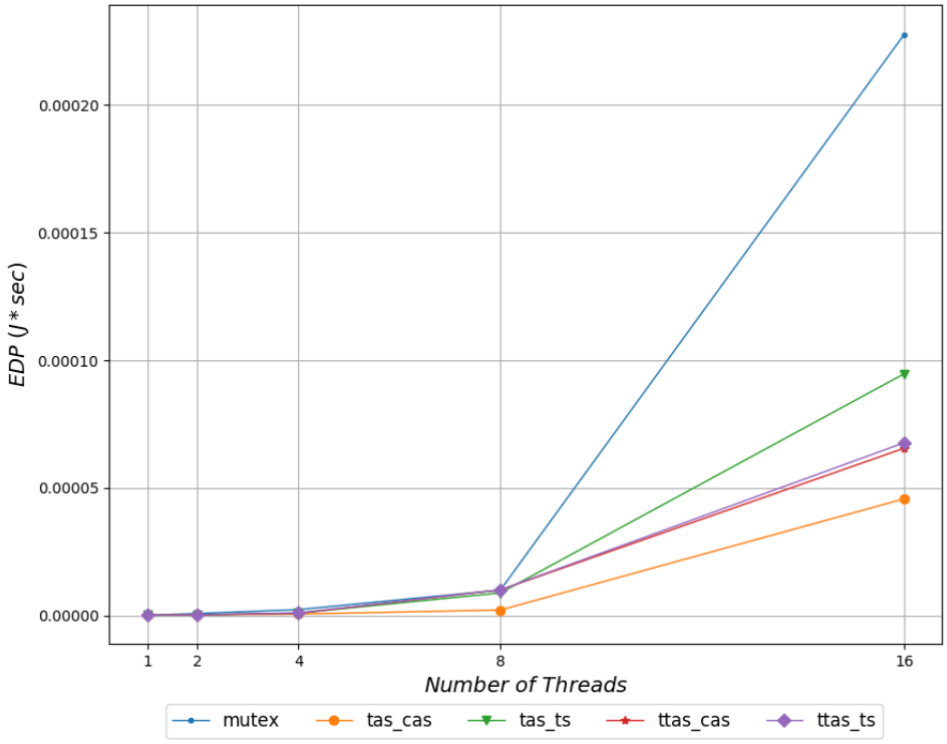
Επιπρόσθετα, συμπεραίνουμε ότι ο χρόνος εκτέλεσης (σε κύκλους) αυξάνεται με την αύξηση του πλήθους των νημάτων. Επιπλέον, παρατηρούμε ότι για grain size 1 και 100 την μεγαλύτερη κλίση στην γραφική παράσταση, δηλαδή τη χειρότερη κλιμακωσιμότητα, την έχει ο μηχανισμός MUTEX, γεγονός που υποδεικνύει ότι αυξάνεται αρκετά πιο γρήγορα ο χρόνος εκτέλεσης σε σύγκριση με τους άλλους μηχανισμούς. Αυτό συμβαίνει διότι όποτε προσπαθούμε να πάρουμε ένα lock σε ένα ήδη κλειδωμένο mutex, προκαλείται ένα context-switch μεταξύ νημάτων. Από όλα τα διαγράμματα γίνεται αντιληπτό ότι την βέλτιστη κλιμακωσιμότητα σε όλες τις περιπτώσεις επιτυγχάνεται με το πρωτόκολλο Test-and-Test-and-SET (TTAS) υλοποιημένος είτε με τις ατομικές εντολές test-and-set ή compare-and-swap.

Όσον αφορά το grain size, παρατηρείται πως καθώς αυξάνεται το grain size, τόσο περισσότερο βελτιώνεται και γίνεται γραμμική με χαμηλότερη κλίση η κλιμακωσιμότητα των μηχανισμών, κυρίως για το MUTEX, ενώ ακόμα αυξάνονται οι κύκλοι. Αυτό συμβαίνει διότι ξοδεύεται περισσότερος χρόνος για τους υπολογισμούς και λιγότερος χρόνος για τον ανταγωνισμό του lock, οπότε έχουμε καλύτερη επίδοση. Ως εκ τούτου, όμως, αφού αυξάνει το πλήθος των dummy υπολογισμών οι χρόνοι στα αντίστοιχα διαγράμματα μεγαλώνουν κατά μία τάξη μεγέθους (x10) για κάθε δεκαπλασιασμό του grain size.

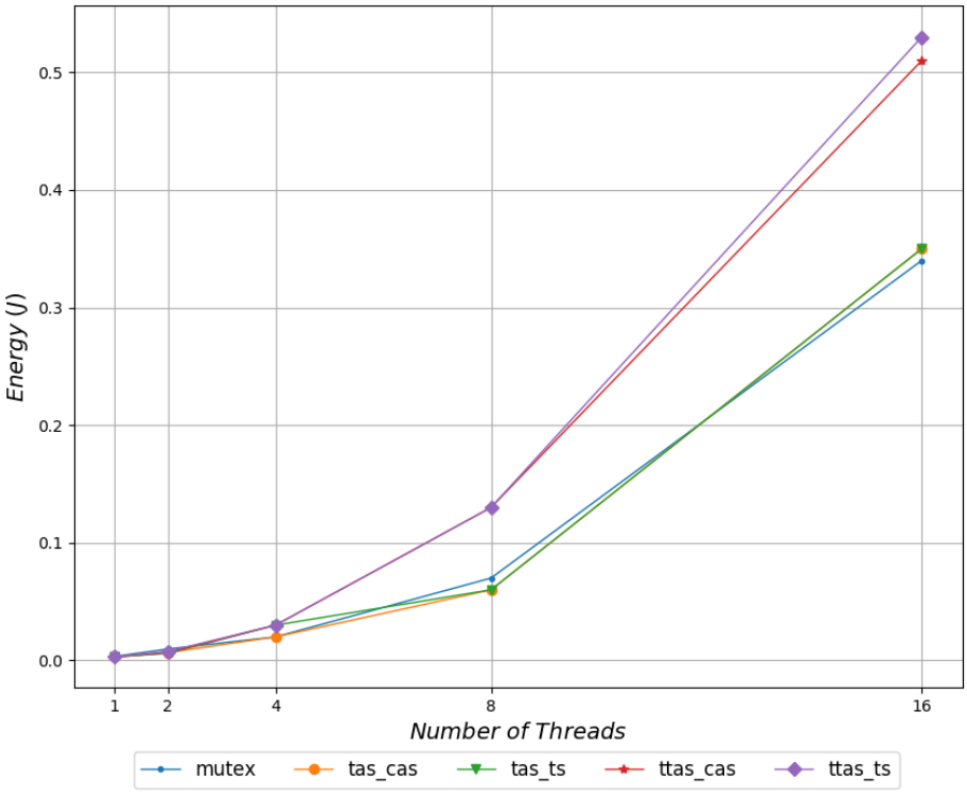
4.1.3

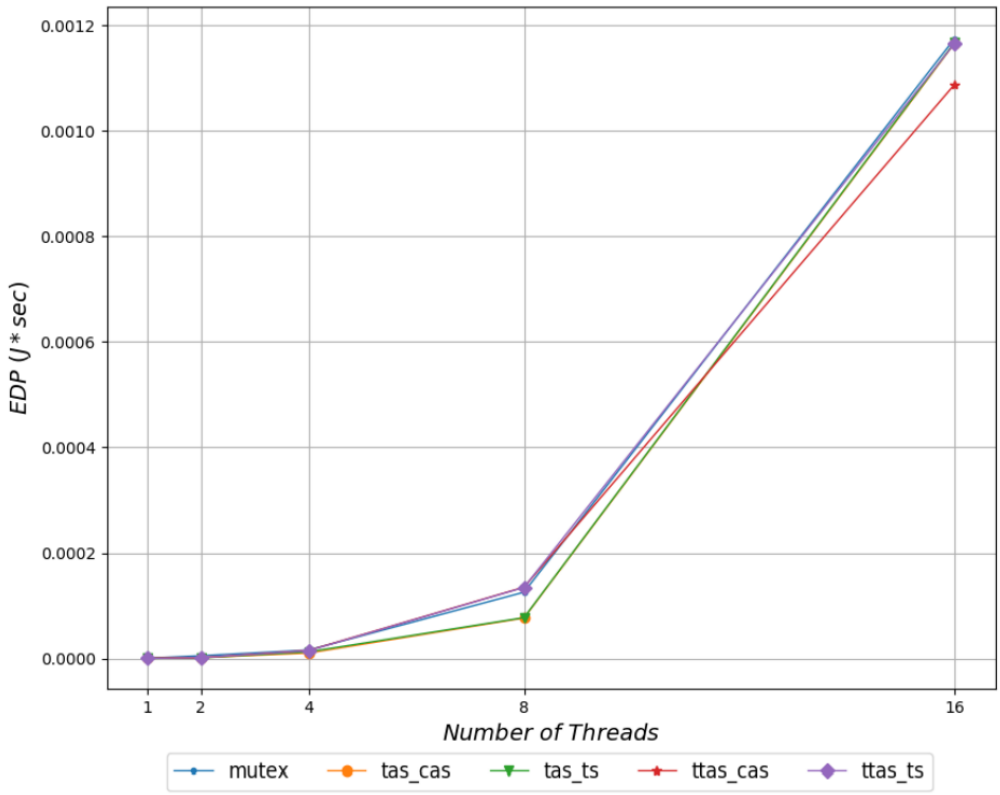
Grain size = 1



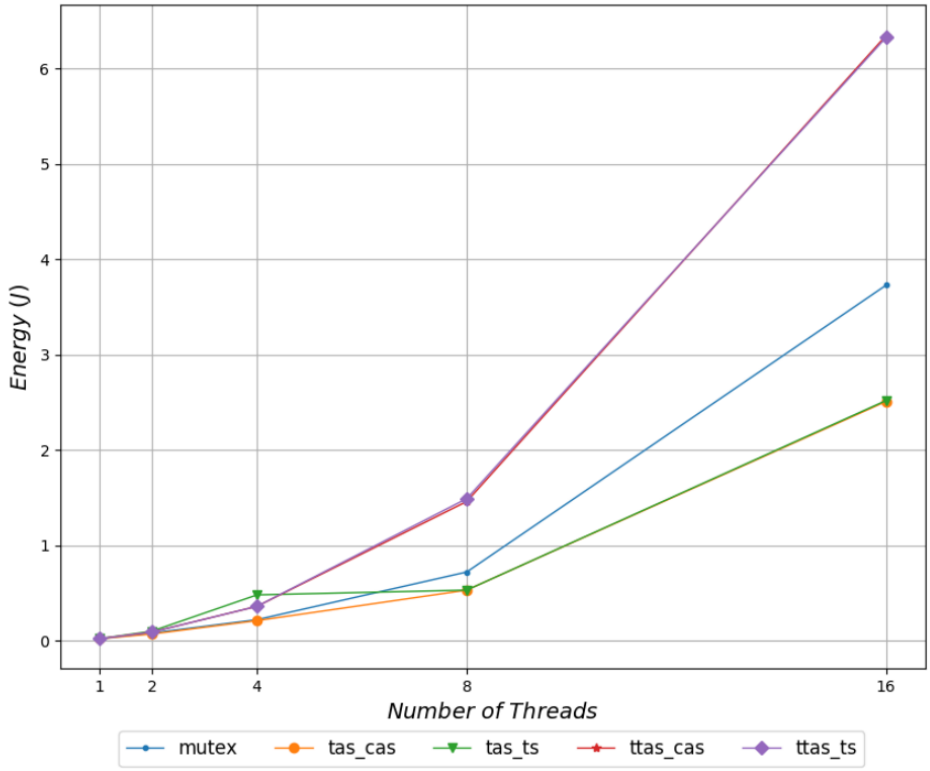


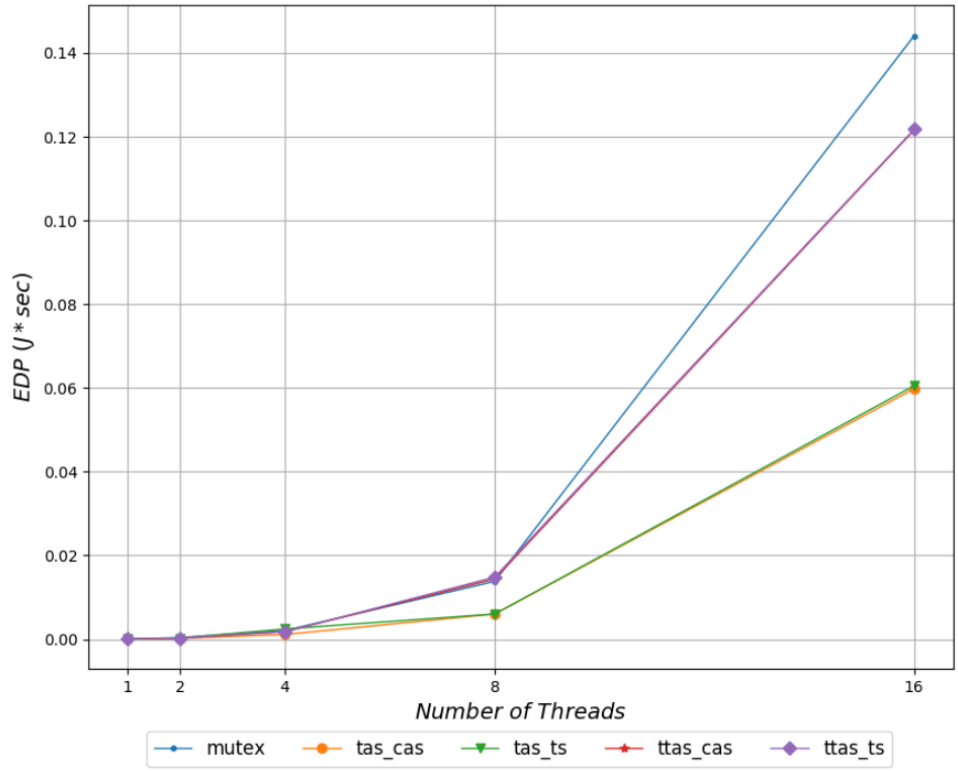
Grain size = 10





Grain size = 100





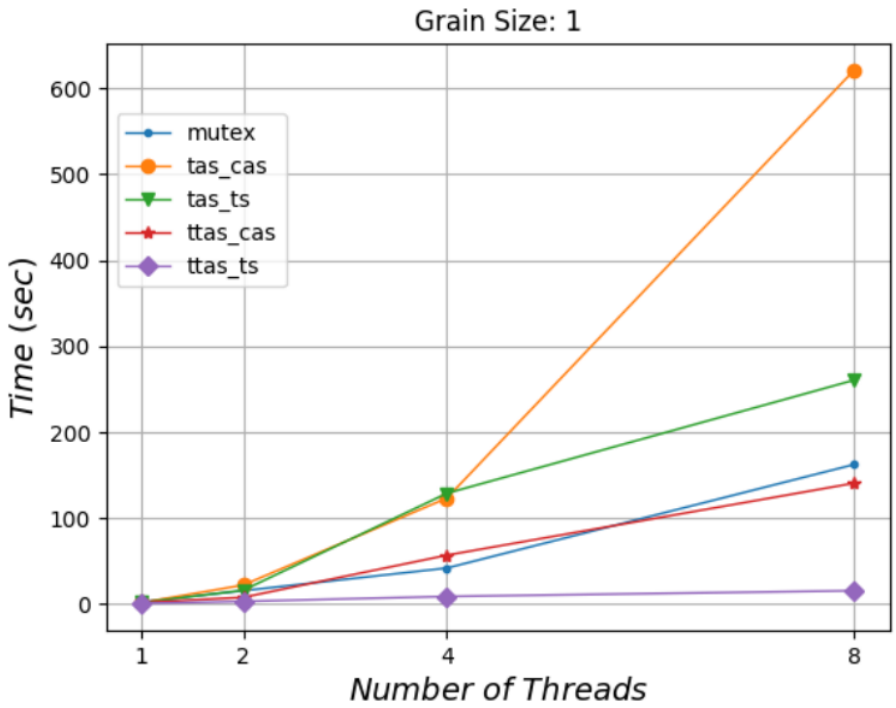
Παρατηρήσεις:

Παρατηρούμε πως η ενέργεια αυξάνεται με την αύξηση του πλήθους των νημάτων. Πιο συγκεκριμένα, όταν μεταβαίνουμε από 8 νήματα σε 16 η ενέργεια που καταναλώνεται είναι σημαντικά μεγαλύτερη. Το κλείδωμα TTAS έχει τη μεγαλύτερη ενεργειακή κατανάλωση καθώς αποσχολεί συνεχώς τον επεξεργαστή με busy wait. Όσον αφορά τα διαγράμματα EDP, συμπεραίνουμε ότι το mutex έχει υψηλότερες τιμές ως προς τους άλλους μηχανισμούς εξαιτίας του υψηλότερου χρόνου εκτέλεσής του, ενώ οι μηχανισμοί TAS-CAS και TAS-TS φαίνεται να έχουν την καλύτερη επίδοση.

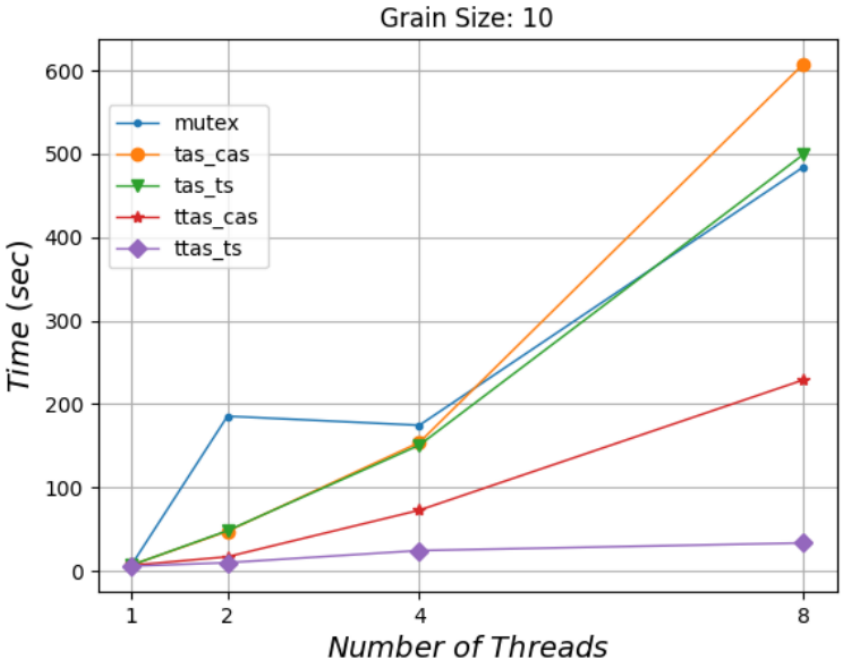
4.1.4

Τα πειράματα εκτελέστηκαν σε τετραπύρηνο πραγματικό σύστημα που υλοποιεί hyper-threading (8 λογικοί πυρήνες) και εκτελέστηκαν 150000000 επαναλήψεις.

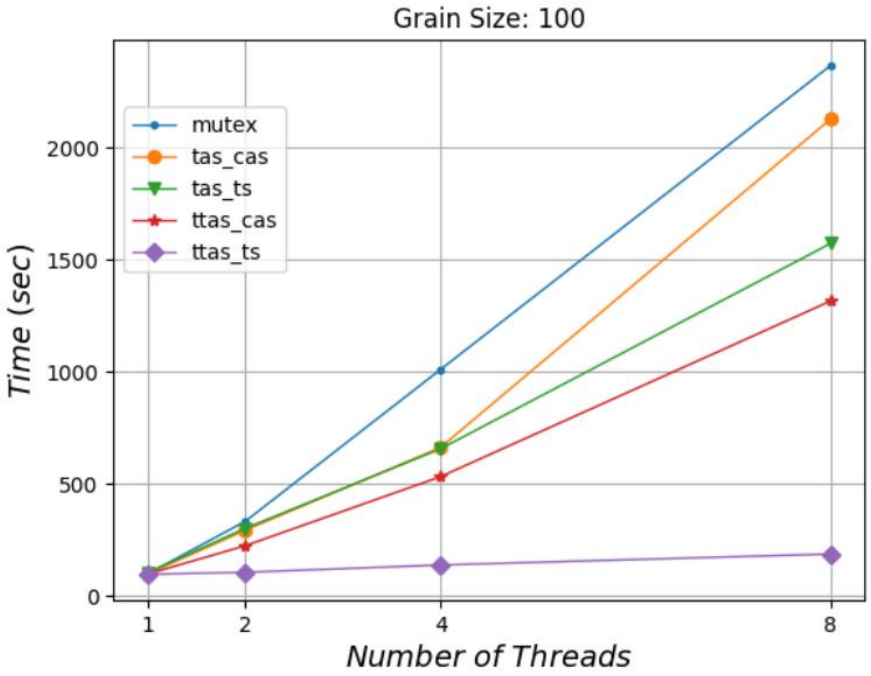
Grain size = 1



Grain size = 10



Grain size = 100



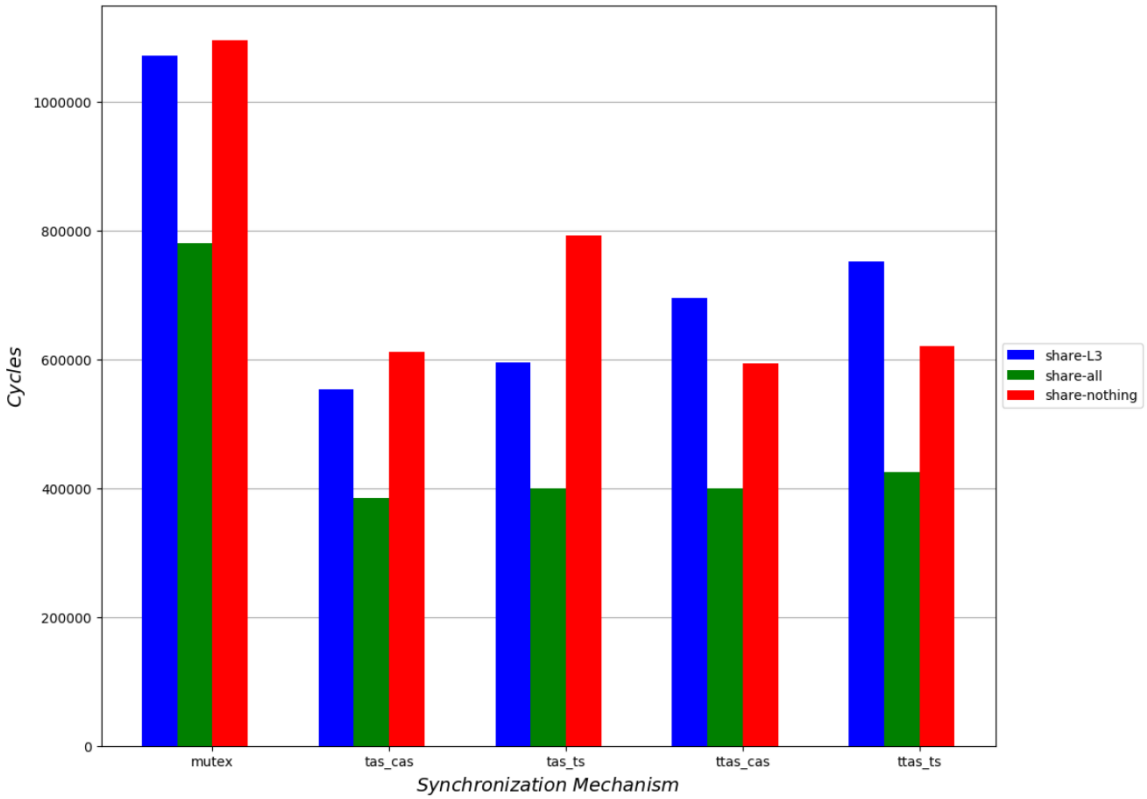
Παρατηρήσεις:

Παρατηρούμε πως έχουμε και σε αυτήν την περίπτωση γραμμική κλιμάκωση, με τον μηχανισμό TTAS_TS να υπερτερεί χρονικά από τους υπόλοιπους, επιτυγχάνοντας την καλύτερη επίδοση. Μάλιστα, φαίνεται ότι η αύξηση των threads προκαλεί ελάχιστη μεταβολή του χρόνου εκτέλεσης για το συγκεκριμένο μηχανισμό. Επιπλέον, αρκετά καλή επίδοση έχει και ο μηχανισμός TTAS_CAS. Τέλος, Ο TAS_CAS φαίνεται να έχει τις χειρότερες επιδόσεις, ειδικά όταν μεταβαίνουμε από τα 4 στα 8 threads, όπου ο χρόνος εκτέλεσής του αυξάνεται δραστικά.

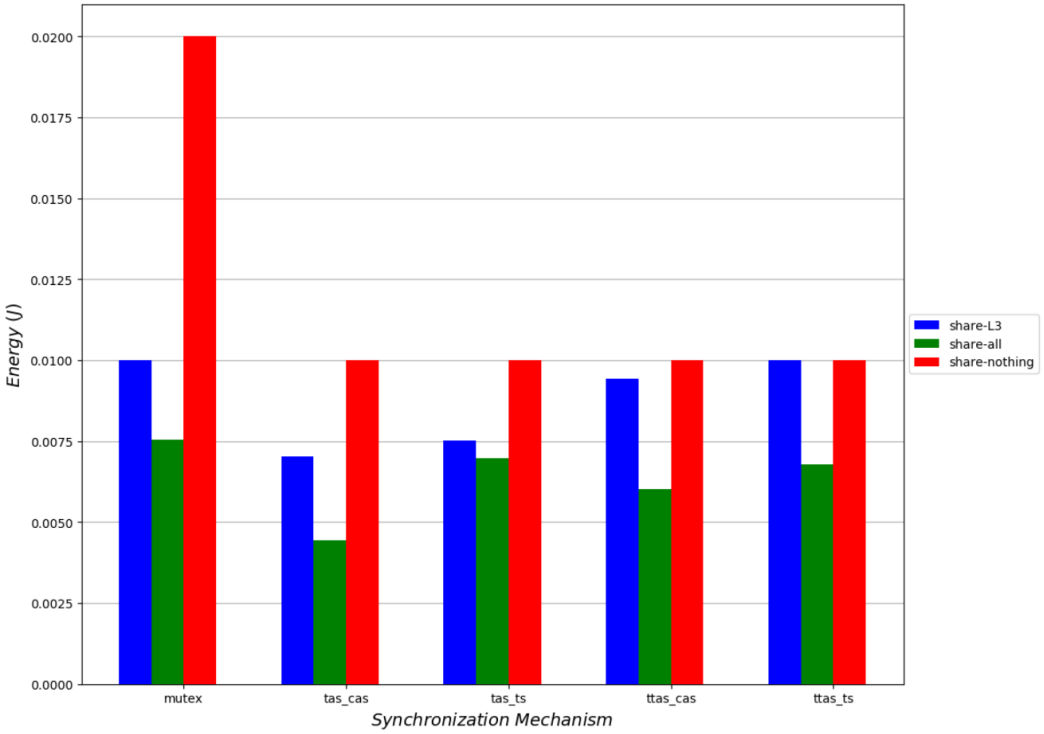
4.1 Τοπολογία Νημάτων

4.2.1

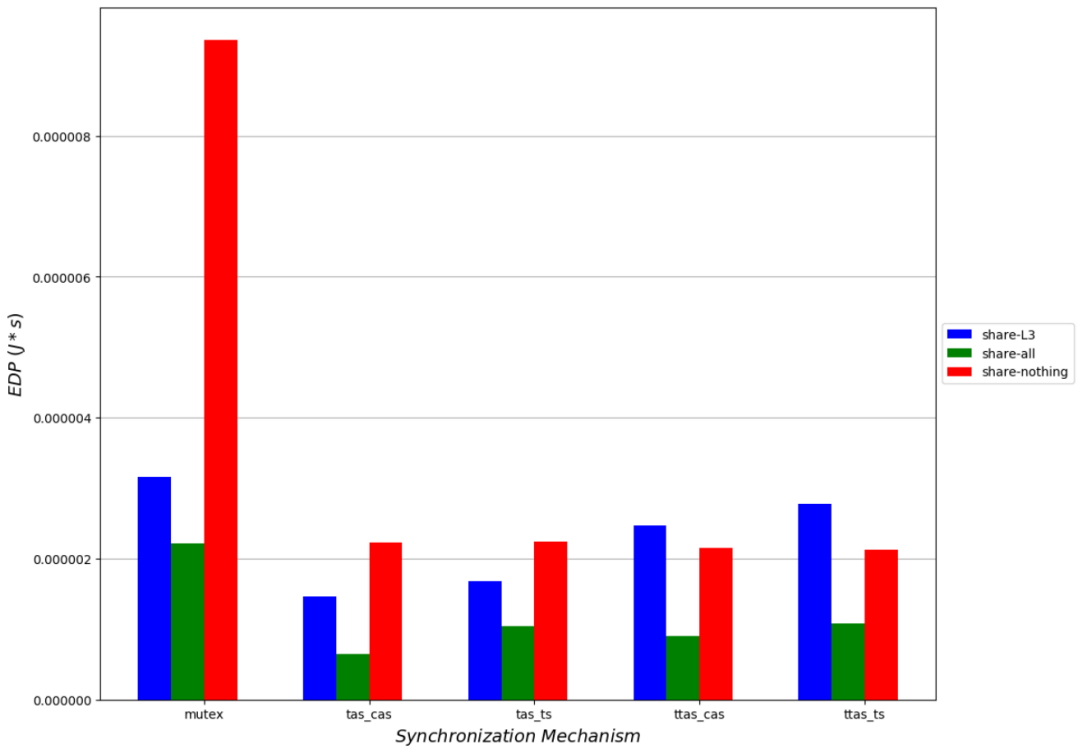
Time Analysis



Energy Analysis



EDP Analysis



Παρατηρήσεις:

Παρατηρούμε πως για την τοπολογία share-all έχουμε λιγότερους κύκλους εκτέλεσης για όλους τους μηχανισμούς. Αυτό συμβαίνει διότι, αν ο επεξεργαστής ζητήσει ένα Invalid cache line, θα ενημερωθεί η ιεραρχία μνήμης μέχρι την L2 και θα το διαβάσει από εκεί. Από την άλλη, στο share-L3, η invalid cache line θα ζητηθεί από την L3, ενώ για share-nothing θα χρειαστεί να φτάσει μέχρι και την κύρια μνήμη για να το διαβάσει, γεγονός που προκαλεί σημαντική αύξηση στον χρόνο εκτέλεσης, μιας και όσο πιο ”μακριά” ιεραρχικά βρίσκεται μία μνήμη από τον επεξεργαστή τόσο πιο αργή είναι.

Ως προς την κατανάλωση ενέργειας, βλέπουμε ότι την μεγαλύτερη κατανάλωση την έχει η τοπολογία share-nothing, ενώ την μικρότερη κατανάλωση η share-all.

Τέλος, με βάση την EDP, η τοπολογία share-all παρουσιάζει και πάλι την καλύτερη επίδοση.