Εργασία 2 - Προγραμματισμός με OpenMP

Ονοματεπώνυμο: Μάριος Γιαννόπουλος
Α.Μ.: 1115200000032

Γενικές Πληροφορίες

Υπολογιστικό Σύστημα

Όλο το έργο υλοποιήθηκε στο ίδιο υπολογιστικό περιβάλλον:

- **Όνομα Υπολογιστικού Συστήματος:** Linux12
- **Επεξεργαστής:** Intel(R) Core(TM) i5-6500 CPU @ 3.20GHz
- **Αριθμός Πυρήνων:** 4
- **Λειτουργικό Σύστημα:** Linux Ubuntu 20.04.2 LTS
- **Έκδοση Μεταγλωττιστή:** gcc (Ubuntu 9.4.0-1ubuntu1 20.04.2) 9.4.0

Οδηγίες Εκτέλεσης Python Scripts

Για την εκτέλεση των Python scripts που επεξεργάζονται τα αποτελέσματα, ακολουθήστε τα εξής βήματα:

1. Μεταβείτε στον φάκελο `scripts`.
2. Εγκαταστήστε τις απαραίτητες βιβλιοθήκες:

   ```
   pip install -r requirements.txt
   ```

3. Εκτελέστε το script που σας ενδιαφέρει:

   ```
   python <test_script>.py
   ```

Σημείωση: Όλα τα αποτελέσματα στα γραφήματα είναι από την εκτέλεση των πειραμάτων στο εργαστήριο Linux. Κάθε πείραμα εκτελέστηκε 5 φορές και τα αποτελέσματα αναφέρονται στο μέσο όρο των επαναλήψεων.

Άσκηση 2.1

Εισαγωγή

Σκοπός της παρούσας εργασίας είναι η παραλληλοποίηση του Παιχνιδιού της Ζωής (Game of Life) με χρήση της βιβλιοθήκης OpenMP. Το παιχνίδι, που σχεδιάστηκε από τον John Conway το 1970, είναι ένα μαθηματικό μοντέλο τοπικών κανόνων που παράγει πολύπλοκα μοτίβα. Στο πλαίσιο της εργασίας, υλοποιήθηκε τόσο σειριακή όσο και παράλληλη έκδοση του αλγορίθμου, ενώ τα πειράματα εκτελέστηκαν σε διαφορετικά μεγέθη πλεγμάτων και αριθμούς νημάτων.

Συγχρονισμός

Για την υλοποίηση της παράλληλης έκδοσης χρησιμοποιήθηκαν οι οδηγίες του OpenMP για την κατανομή του έργου στα διαθέσιμα νήματα. Λόγω της ανεξαρτησίας των υπολογισμών για κάθε κελί (εκτός από την ανάγνωση των γειτονικών κυψελών), δεν απαιτείται πρόσθετος συγχρονισμός μεταξύ των νημάτων. Οι γενιές ενημερώνονται ταυτόχρονα χωρίς αλληλεξάρτηση, οπότε η παραλληλία εφαρμόζεται χωρίς την ανάγκη συγχρονισμού ή κλειδώματος των δεδομένων.

Πειραματική Διαδικασία

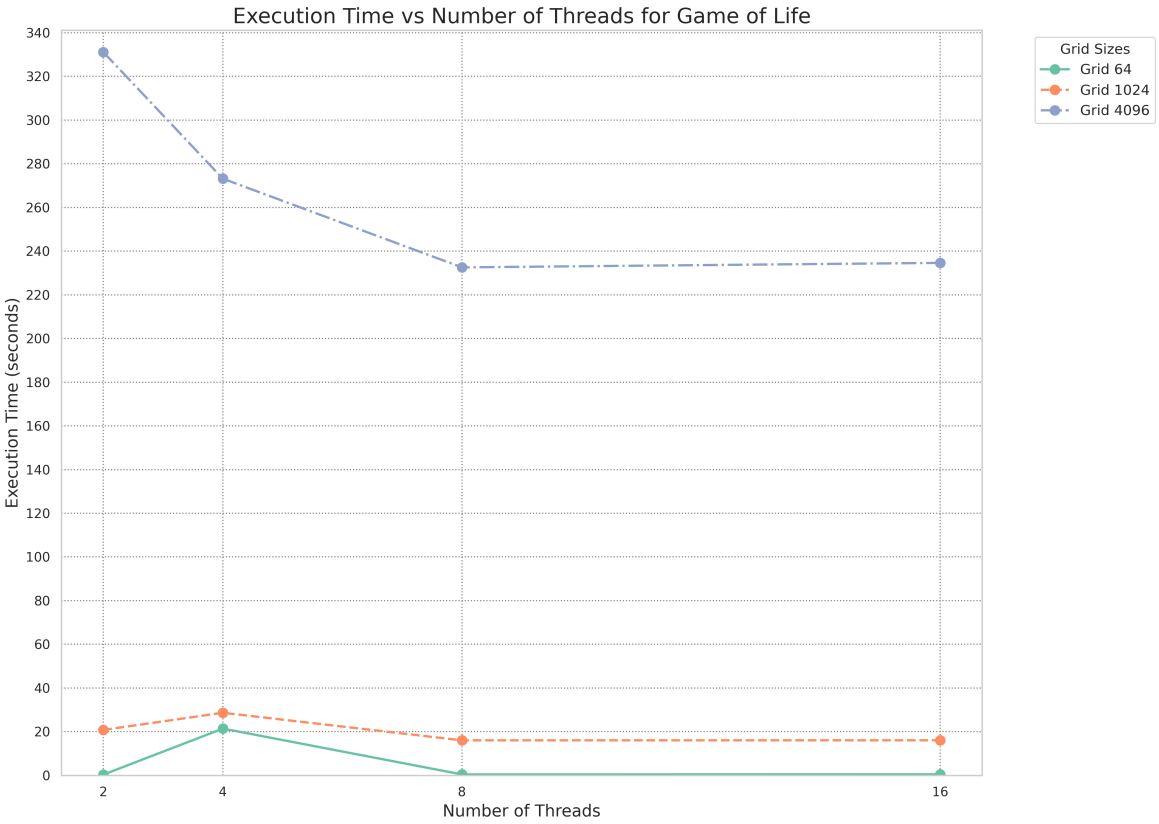
- **Παραμετροποίηση:**
  - Μέγεθος πλέγματος: 64×64, 1024×1024, 4096×4096.
  - Αριθμός γενιών: 1000.
  - Αριθμός νημάτων: 2, 4, 8, 16.
- **Εκτέλεση:**
  - Τα πειράματα εκτελέστηκαν 5 φορές για κάθε συνδυασμό παραμέτρων.
  - Καταγράφηκε ο χρόνος εκτέλεσης για κάθε πείραμα.
  - Τα δεδομένα αποθηκεύτηκαν σε CSV αρχείο.
- **Αυτοματοποίηση:**
  - Αναπτύχθηκαν Python scripts για την εκτέλεση των πειραμάτων και την καταγραφή των δεδομένων.
  - Χρησιμοποιήθηκαν Python scripts για την επεξεργασία των αποτελεσμάτων και τη δημιουργία γραφημάτων.

Αποτελέσματα

- **Σύγκριση Σειριακού και Παράλληλου Αλγορίθμου:**
  - Για το πλέγμα 64x64, η σειριακή εκτέλεση έχει χρόνο εκτέλεσης γύρω από 0.38s. Ωστόσο, η παράλληλη εκτέλεση με 2 νήματα αποδίδει καλύτερα με μέσο χρόνο περίπου 0.32s, ενώ για περισσότερα νήματα (4, 8, 16) παρατηρείται αύξηση του χρόνου εκτέλεσης λόγω του overhead της παραλληλοποίησης. Αυτό δείχνει ότι για μικρά πλέγματα η παραλληλοποίηση μπορεί να έχει μικρότερο όφελος.
  - Για το πλέγμα 1024x1024, η σειριακή εκτέλεση έχει μέσο χρόνο περίπου 40.3s. Η παράλληλη εκτέλεση επιταχύνει τη διαδικασία, με 2 νήματα να χρειάζονται περίπου 20.8s και 16 νήματα περίπου 16s, υποδεικνύοντας σημαντική βελτίωση στους χρόνους εκτέλεσης με την αύξηση του αριθμού των νημάτων.
  - Για το πλέγμα 4096x4096, η σειριακή εκτέλεση απαιτεί περίπου 643.3s. Με την παράλληλη εκτέλεση, οι χρόνοι μειώνονται σημαντικά: 331.2s με 2 νήματα και 234.6s με 16 νήματα, επιβεβαιώνοντας τη μεγάλη αποδοτικότητα της παραλληλοποίησης για μεγάλα πλέγματα.

Γραφήματα

Το παρακάτω γράφημα παρουσιάζει τη σχέση μεταξύ του μεγέθους του πλέγματος, του αριθμού των νημάτων και του χρόνου εκτέλεσης:



Γράφημα 1: Χρόνος Εκτέλεσης ανά Μέγεθος Πλέγματος και Αριθμό Νημάτων

Συμπεράσματα

1. Η παραλληλοποίηση είναι αποδοτική για μεγάλα μεγέθη πλεγμάτων, όπως φαίνεται από τη σημαντική μείωση του χρόνου εκτέλεσης στα πλέγματα 1024x1024 και 4096x4096, με την αύξηση του αριθμού των νημάτων.

2. Για μικρά πλέγματα (όπως το 64x64), η παραλληλοποίηση με 2 νήματα προσφέρει κάποιες βελτιώσεις, αλλά για περισσότερα νήματα το overhead της παραλληλοποίησης επηρεάζει αρνητικά την απόδοση.

3. Η απόδοση του αλγορίθμου περιορίζεται από τον αριθμό των διαθέσιμων πυρήνων του επεξεργαστή και το μέγεθος του πλέγματος, με την παραλληλοποίηση να αποδίδει καλύτερα σε μεγαλύτερους υπολογιστικούς πόρους και μεγαλύτερα δεδομένα.

4. Για μικρά πλέγματα, η σειριακή έκδοση παραμένει αποδοτικότερη λόγω του overhead της παραλληλοποίησης.

Η εργασία κατέδειξε τη σημασία της παραλληλοποίησης για μεγάλα προβλήματα, αλλά και τους περιορισμούς της λόγω του hardware.

Άσκηση 2.2

Εισαγωγή

Η παρούσα εργασία εξετάζει την παραλληλοποίηση της διαδικασίας επίλυσης γραμμικών συστημάτων με τη χρήση της απαλοιφής Gauss και της αντικατάστασης προς τα πίσω. Η απαλοιφή Gauss μετατρέπει ένα σύστημα σε άνω τριγωνική μορφή, ενώ η αντικατάσταση προς τα πίσω υπολογίζει τις λύσεις. Η εργασία περιλαμβάνει σειριακές και παράλληλες υλοποιήσεις τόσο για τον *"κατά γραμμή"* όσο και για τον *"κατά στήλη"* αλγόριθμο. Η αξιολόγηση των υλοποιήσεων γίνεται μέσω πειραμάτων σε διάφορα μεγέθη συστημάτων και αριθμούς νημάτων.

Προσεγγίσεις Παραλληλοποίησης

1. **Αλγόριθμος "Κατά Γραμμή":**
   - Ο εξωτερικός βρόχος που διατρέχει τις γραμμές μπορεί να παραλληλοποιηθεί.
   - Ο εσωτερικός βρόχος παρουσιάζει εξαρτήσεις δεδομένων που καθιστούν την παραλληλοποίησή του πιο περίπλοκη.
2. **Αλγόριθμος "Κατά Στήλη":**
   - Ο εξωτερικός βρόχος δεν παρουσιάζει εξαρτήσεις δεδομένων και παραλληλοποιείται εύκολα.
   - Ο εσωτερικός βρόχος επηρεάζεται από την εξάρτηση μεταξύ των στηλών, γεγονός που απαιτεί προσεκτικό χειρισμό.

Για την παραλληλοποίηση, χρησιμοποιήθηκε το OpenMP, αξιοποιώντας οδηγίες όπως `#pragma omp parallel for` για τον έλεγχο των βρόχων.

Πειραματική Διαδικασία

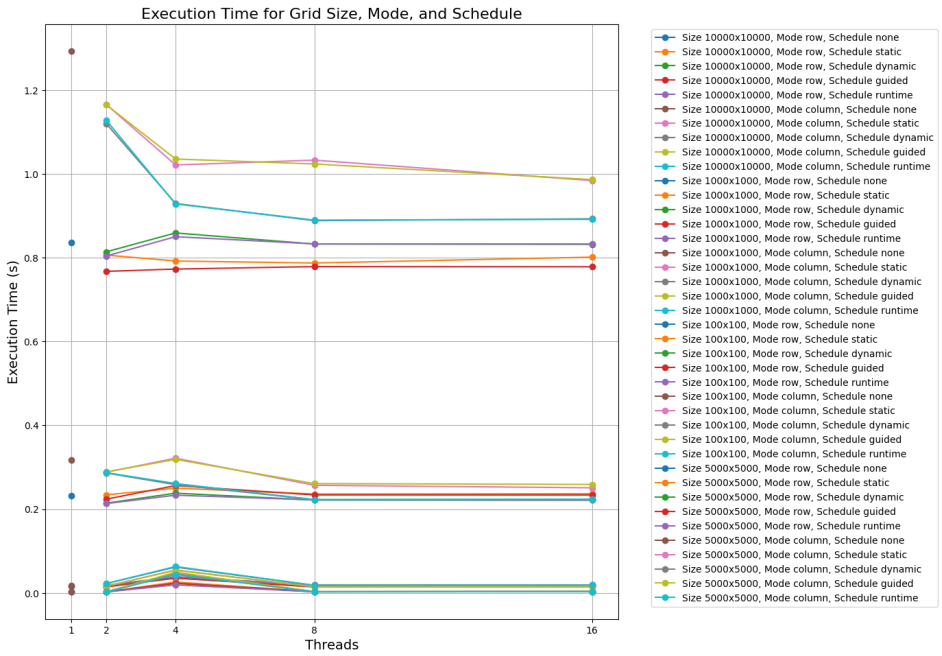
- **Παραμετροποίηση:**
  - Μέγεθος συστήματος (n): 100, 1000, 5000, 10000.
  - Αριθμός νημάτων: 2, 4, 8, 16.
  - Τύποι Αλγορίθμων:
    - "Κατά Γραμμή" και "Κατά Στήλη".
    - Σειριακή και Παράλληλη Υλοποίηση.
  - Χρονοπρογραμματισμός (Schedule): static, dynamic, guided, runtime.
- **Διαδικασία Εκτέλεσης:**
  - Κάθε πείραμα εκτελέστηκε 5 φορές και μετρήθηκε ο μέσος χρόνος εκτέλεσης.
  - Τα δεδομένα αποθηκεύτηκαν σε CSV για ανάλυση.
- **Αυτοματοποίηση:**
  - Αναπτύχθηκαν scripts σε Python για την αυτοματοποίηση των πειραμάτων, την επεξεργασία των αποτελεσμάτων και τη δημιουργία γραφημάτων.

Αποτελέσματα

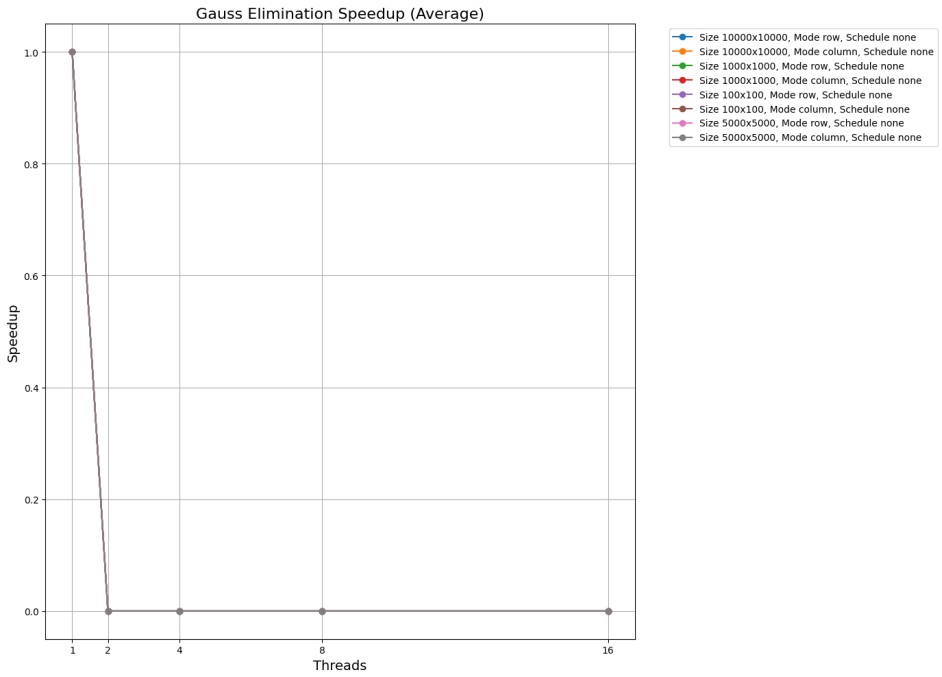
- **Σύγκριση Σειριακού και Παράλληλου Αλγορίθμου:**
  - Για μικρότερα μεγέθη συστημάτων, όπως $100x100$, η παράλληλη εκτέλεση δεν δείχνει σημαντική βελτίωση στον χρόνο εκτέλεσης λόγω του overhead της παραλληλοποίησης. Για παράδειγμα, με 2 νήματα στον "κατά γραμμή" αλγόριθμο και το static schedule, ο χρόνος εκτέλεσης για το $100x100$ πίνακα είναι 0.002366 (average).
  - Για μεγαλύτερα μεγέθη πινάκων, όπως το $10000x10000$, παρατηρείται σημαντική βελτίωση με την παράλληλη εκτέλεση. Στο "κατά γραμμή" αλγόριθμο με guided schedule και 16 νήματα, ο χρόνος εκτέλεσης είναι 0.778388 (average).
- **Επιδόσεις ανά Schedule:**
  - Το *static* schedule παρουσιάζει σταθερή απόδοση για μικρότερα συστήματα, αλλά δεν προσφέρει τη βέλτιστη απόδοση για μεγαλύτερους πίνακες. Στο $10000x10000$ σύστημα, ο χρόνος εκτέλεσης με 16 νήματα και static schedule είναι 0.983888 (average).
  - Το *dynamic* schedule επιτρέπει ευελιξία για ανομοιόμορφα workloads, αλλά με μεγαλύτερο overhead. Στο ίδιο σύστημα, ο χρόνος με 16 νήματα και dynamic schedule είναι 0.892785 (average).
  - Το *guided* schedule παρέχει καλή ισορροπία, με καλύτερες επιδόσεις σε μεγάλους πίνακες. Στο $10000x10000$, ο χρόνος εκτέλεσης με guided schedule είναι 0.986361 (average).
  - Το *runtime* προσφέρει δυναμική ρύθμιση αλλά με κάποια αστάθεια, παρουσιάζοντας χρόνο 0.891547 (average) για το $10000x10000$ με 16 νήματα.
- **Γενικές Παρατηρήσεις:**
  - Όσο αυξάνεται ο αριθμός των νημάτων, παρατηρείται μια σταθερή μείωση στον χρόνο εκτέλεσης για μεγάλους πίνακες. Ωστόσο, η βελτίωση είναι πιο εμφανής όταν τα μεγέθη των πινάκων είναι μεγαλύτερα (π.χ. $10000x10000$).

Γραφήματα

Τα παρακάτω γραφήματα παρουσιάζουν την επίδραση του μεγέθους του πίνακα, του αριθμού των νημάτων, του τύπου του αλγορίθμου και του schedule στον χρόνο εκτέλεσης:



Γράφημα 2: Χρόνος Εκτέλεσης κατά Μέγεθος Πίνακα, Αλγόριθμο και Τύπο Εκτέλεσης



Γράφημα 3: Επίδραση του Schedule στον Χρόνο Εκτέλεσης

Συμπεράσματα

1. Η παραλληλοποίηση έχει σημαντική επίδραση στον χρόνο εκτέλεσης για μεγάλους πίνακες, όπως το $10000x10000$, όπου η χρήση 16 νημάτων μειώνει το χρόνο εκτέλεσης σημαντικά.

2. Ο "κατά γραμμή" αλγόριθμος εμφανίζει καλύτερη απόδοση σε σχέση με τον "κατά στήλη", λόγω της μικρότερης πολυπλοκότητας και των λιγότερων εξαρτήσεων δεδομένων.

3. Οι παράμετροι προγραμματισμού, όπως το schedule, έχουν σημαντική επίδραση στον χρόνο εκτέλεσης. Το guided και το static schedule προσφέρουν καλύτερη απόδοση σε μεγάλες διαστάσεις πινάκων, ενώ το dynamic και το runtime έχουν μεγαλύτερο overhead, ειδικά με μικρότερα συστήματα.

4. Το μέγεθος του πίνακα και ο αριθμός των νημάτων επηρεάζουν σημαντικά την απόδοση. Για μεγαλύτερα συστήματα, η παραλληλοποίηση με υψηλό αριθμό νημάτων είναι ιδιαίτερα αποτελεσματική.

Η εργασία ανέδειξε την αποδοτικότητα της παραλληλοποίησης για μεγάλα συστήματα και τη σημασία της επιλογής του κατάλληλου schedule για τη βελτιστοποίηση του χρόνου εκτέλεσης.

Άσκηση 2.3

Εισαγωγή

Σκοπός της παρούσας εργασίας είναι η παραλληλοποίηση του Παιχνιδιού της Ζωής (Game of Life) χρησιμοποιώντας την οδηγία `task` της βιβλιοθήκης OpenMP. Η παραλληλοποίηση με τη χρήση της οδηγίας `task` επιτρέπει την κατανομή του υπολογιστικού έργου σε επιμέρους μονάδες εργασίας, οι οποίες μπορούν να

εκτελούνται ασύγχρονα. Στο πλαίσιο της εργασίας, υλοποιήθηκε μια παράλληλη έκδοση του αλγορίθμου με την οδηγία `task`, η οποία συγκρίνεται με την προηγούμενη υλοποίηση της Άσκησης 2.1.

Συγχρονισμός

Για την υλοποίηση της παράλληλης έκδοσης με την οδηγία `task`, χρησιμοποιήθηκε η κατάλληλη κατανομή εργασιών μεταξύ των νημάτων, με στόχο την αποτελεσματική παράλληλη εκτέλεση. Κάθε γενιά του παιχνιδιού υποδιαιρείται σε μικρότερα καθήκοντα, τα οποία ανατίθενται σε διαφορετικά νήματα μέσω `task`. Η χρήση των `task` εξασφαλίζει τη σωστή εκτέλεση του αλγορίθμου χωρίς να απαιτείται άλλος συγχρονισμός μεταξύ των νημάτων.

Πειραματική Διαδικασία

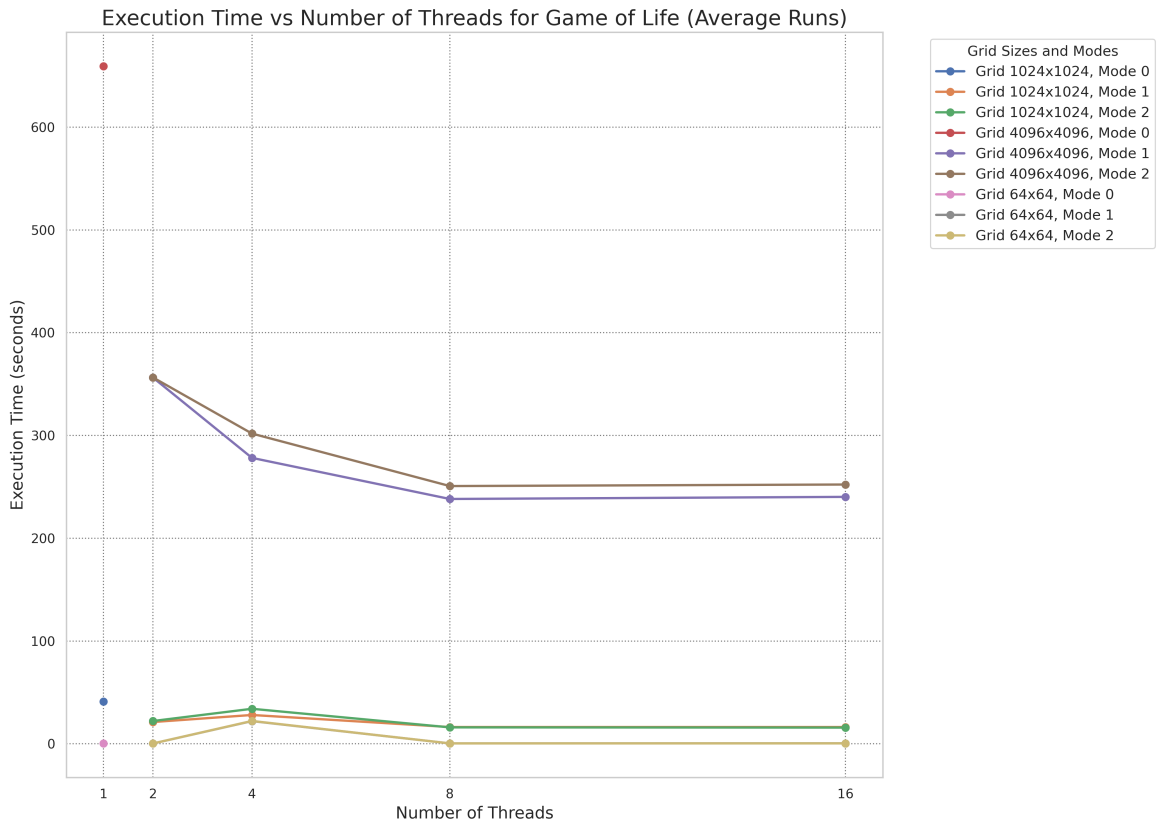
- **Παραμετροποίηση:**
  - Μέγεθος πλέγματος: 64×64, 1024×1024, 4096×4096.
  - Αριθμός γενιών: 1000.
  - Αριθμός νημάτων: 2, 4, 8, 16.
- **Εκτέλεση:**
  - Τα πειράματα εκτελέστηκαν 5 φορές για κάθε συνδυασμό παραμέτρων.
  - Καταγράφηκε ο χρόνος εκτέλεσης για κάθε πείραμα.
  - Τα δεδομένα αποθηκεύτηκαν σε CSV αρχείο.
- **Αυτοματοποίηση:**
  - Αναπτύχθηκαν Python scripts για την εκτέλεση των πειραμάτων και την καταγραφή των δεδομένων.
  - Χρησιμοποιήθηκαν Python scripts για την επεξεργασία των αποτελεσμάτων και τη δημιουργία γραφημάτων.

Αποτελέσματα

- **Σύγκριση Σειριακού και Παράλληλου Αλγορίθμου (OpenMP task):**
  - Για μικρά πλέγματα (64×64), ο σειριακός αλγόριθμος ήταν ταχύτερος λόγω του overhead της παραλληλοποίησης.
  - Για μεγαλύτερα πλέγματα (1024×1024 και 4096×4096), ο παράλληλος αλγόριθμος παρουσίασε σημαντική βελτίωση στον χρόνο εκτέλεσης.
- **Επιτάχυνση:**
  - Η απόδοση βελτιώθηκε με την αύξηση του αριθμού των νημάτων μέχρι να εξαντληθούν οι φυσικοί πόροι του επεξεργαστή.
  - Το μέγεθος του πλέγματος επηρέασε άμεσα την απόδοση, καθώς μεγαλύτερα πλέγματα επέτρεψαν την πλήρη αξιοποίηση της παραλληλίας.
- **Γραμμικότητα της Κλιμάκωσης:**
  - Παρατηρήθηκε γραμμική βελτίωση στην επιτάχυνση για μικρό αριθμό νημάτων.
  - Σε μεγαλύτερους αριθμούς νημάτων, η γραμμικότητα μειώθηκε λόγω αυξημένου synchronization overhead.

Γραφήματα

Το παρακάτω γράφημα παρουσιάζει τη σχέση μεταξύ του μεγέθους του πλέγματος, του αριθμού των νημάτων και του χρόνου εκτέλεσης για την παράλληλη υλοποίηση με την οδηγία `task`:



Γράφημα 4: Χρόνος Εκτέλεσης ανά Μέγεθος Πλέγματος και Αριθμό Νημάτων με OpenMP Task

Συμπεράσματα

1. Η χρήση της οδηγίας `task` στην παραλληλοποίηση του Game of Life προσφέρει σημαντικά πλεονεκτήματα για μεγάλα μεγέθη πλεγμάτων.

2. Η απόδοση της παραλληλοποίησης περιορίζεται από το hardware του συστήματος, συγκεκριμένα από τον αριθμό των διαθέσιμων πυρήνων και το synchronization overhead.

3. Για μικρά μεγέθη πλεγμάτων, η σειριακή έκδοση παραμένει πιο αποδοτική λόγω του υψηλού κόστους διαχείρισης των tasks.

4. Ο παράλληλος αλγόριθμος με `task` προσφέρει ευελιξία στη διαχείριση του υπολογιστικού έργου, αλλά απαιτεί προσεκτική ρύθμιση των παραμέτρων για τη μεγιστοποίηση της απόδοσης.

Η ανάλυση των αποτελεσμάτων κατέδειξε ότι η επιλογή παραλληλοποίησης πρέπει να γίνεται με βάση το μέγεθος του προβλήματος και τις δυνατότητες του συστήματος, επιβεβαιώνοντας την ευελιξία και την αποδοτικότητα της οδηγίας `task`.