ΠΑΡΑΛΛΗΛΗ ΕΠΕΞΕΡΓΑΣΙΑ

ΑΚΑΔΗΜΑΪΚΟ ΕΤΟΣ 2017/2018

Εισαγωγή

Στα πλαίσια της άσκησης θα ασχοληθούμε με το μοντέλο προσομοίωσης βιολογικών νευρώνων Leaky Integrate-and-Fire (LIF). Το συγκεκριμένο μοντέλο είναι από τα πιο γνωστά και πλέον χρησιμοποιούμενα. Σκοπός της εργασίας είναι να βελτιστοποιήσετε σε πολλαπλά επίπεδα την υλοποίηση του συγκεκριμένου μοντέλου σε υπολογιστή. Φυσικά θα πρέπει να παραλληλοποιήσετε την εφαρμογή σε διάφορα επίπεδα για να μελετήσετε την επίδραση κάθε προσέγγισης στην απόδοση της εφαρμογής. Επιπλέον όμως θα πρέπει να βελτιστοποιήσετε την εφαρμογή αναδιοργανώνοντας τους υπολογισμούς που πρέπει να πραγματοποιηθούν ώστε να μειώσετε τον χρόνο εκτέλεσης, αλλά και να χρησιμοποιήσετε βελτιστοποιημένες βιβλιοθήκες για την πραγματοποίηση βασικών αλγεβρικών πράξεων.

Νευρώνες και συνάψεις

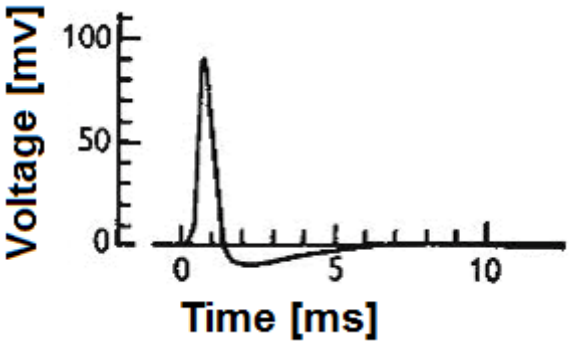
Οι *νευρώνες* είναι κύτταρα που λαμβάνουν, επεξεργάζονται και μεταδίδουν πληροφορίες μέσω ηλεκτρικών και χημικών σημάτων. Αυτά τα σήματα μεταξύ νευρώνων μεταφέρονται μέσω εξειδικευμένων συνδέσεων που ονομάζονται *συνάψεις* (Εικόνα 1). Οι νευρώνες μπορούν να συνδεθούν μεταξύ τους για να σχηματίσουν νευρωνικά δίκτυα. Οι νευρώνες αποτελούν τα βασικά δομικά συστατικά του κεντρικού νευρικού συστήματος, το οποίο περιλαμβάνει τον εγκέφαλο και τον νωτιαίο μυελό, αλλά και του περιφεριακού νευρικού συστήματος, το οποίο περιλαμβάνει το αυτόνομο νευρικό σύστημα και το σωματικό νευρικό σύστημα.

Εικόνα 1

Υπάρχουν πολλών ειδών εξειδικευμένοι νευρώνες. Οι αισθητήριοι νευρώνες ανταποκρίνονται σε ένα συγκεκριμένο τύπο διέγερσης όπως η αφή, ο ήχος ή το φως και άλλα ερεθίσματα που επηρεάζουν τα κύτταρα των αισθητηρίων οργάνων. Οι αισθητήριοι νευρώνες μετατρέπουν τις διεγέρσεις αυτές σε ηλεκτρικό σήμα, το οποίο στη συνέχεια αποστέλλεται στον νωτιαίο μυελό ή τον εγκέφαλο. Οι κινητήριοι νευρώνες λαμβάνουν σήματα από τον εγκέφαλο και το νωτιαίο μυελό για να προκαλέσουν από μυϊκές συσπάσεις μέχρι και να επηρεάσουν τις αδενικές εξόδους. Οι εσωτερικοί νευρώνες συνδέουν τους νευρώνες με άλλους νευρώνες εντός της ίδιας περιοχής του εγκεφάλου ή του νωτιαίου μυελού σε νευρικά δίκτυα.

Το μοντέλο Leaky Integrate-and-Fire

Ο ανθρώπινος εγκέφαλος αποτελείται από ένα σύνολο νευρώνων, καθένας εκ των οποίων συνδέεται με ένα σύνολο άλλων νευρώνων (τους γείτονες του). Κάθε νευρώνας δέχεται και παράγει ηλεκτρικά σήματα τα οποία μεταδίδονται προς τους γείτονες του μέσω των συνάψεων. Ταυτόχρονα όμως, η συμπεριφορά κάθε νευρώνα (δηλαδή το ηλεκτρικό σήμα που παράγει) επίσης εξαρτάται από το σύνολο των γειτόνων του.



Εικόνα 2

Η Εικόνα 2 παρουσιάζει την μεταβολή του ηλεκτρικού δυναμικού της μεμβράνης ενός νευρώνα με την πάροδο του χρόνου. Όπως φαίνεται και στην εικόνα, το δυναμικό αυξάνει (καθώς ο νευρώνας δέχεται σήματα από τους γείτονες του) και στην συνέχεια πέφτει απότομα (πυροδοτείται).

Το μοντέλο Leaky Integrate-and-Fire (LIF) προσομοιώνει την παραπάνω διαδικασία. Για έναν νευρώνα θεωρεί το δυναμικό της μεμβράνης u ως την βασική μεταβλητή, η οποία αυξάνει με ρυθμό μ καθώς δέχεται είσοδο από τους γείτονες του. Μόλις το δυναμικό ξεπεράσει ένα όριο (threshold) u_{th}, τότε ο νευρώνας πυροδοτείται και το δυναμικό του επανέρχεται στο μηδέν.

Θεωρούμε πως έχουμε ένα σύνολο n νευρώνων, με κάθε νευρώνα i να έχει δυναμικό $u_i(t)$ σε δεδομένη χρονική στιγμή t. Κάθε νευρώνας i είναι συνδεδεμένος με ένα υποσύνολο νευρώνων $\{N_i\}$, ή αλλιώς την "γειτονιά" (neighborhood). Συμβολίζουμε το μέγεθος αυτού του υποσυνόλου με N_c. Για να αναπαραστήσουμε την αλληλεπίδραση μεταξύ νευρώνων, χρησιμοποιούμε ένα δισδιάστατο μητρώο σ, κάθε στοιχείο σ_{ij} του οποίου περιγράφει πόσο επηρεάζει το δυναμικό του νευρώνα j το δυναμικό του νευρώνα i. Οι μαθηματικές σχέσεις του μοντέλου είναι οι εξής:

$$\frac{du_i(t)}{dt} = \mu - u_i(t) + \frac{1}{N_c}\sum_{j\in\{N_i\}}^{N_c} \sigma_{ij}\cdot\left[u_j(t) - u_i(t)\right]$$

$$\lim_{\epsilon\to 0}(t+\epsilon) \to 0, \text{ όταν } u_i \geq u_{th} \text{ με } u_{th} < \mu$$

Η πρώτη σχέση εκφράζει την επιρροή των δυναμικών των γειτόνων στο δυναμικό του νευρώνα i και η δεύτερη σχέση εκφράζει την πυροδότηση ενός νευρώνα και την επαναφορά του δυναμικού του στο μηδέν.

Στα πλαίσια μιας υλοποίησης μέσω προγράμματος θα πρέπει να γίνει διακριτοποίηση, δηλαδή δεν μπορούμε να θεωρήσουμε απείρως μικρές μεταβολές $du_i(t)$ και dt. Αντ' αυτού

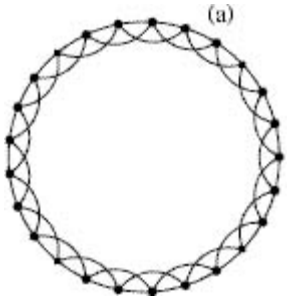
προσεγγίζουμε την παράγωγο χρησιμοποιώντας ένα μικρό (αλλά όχι απειροστά μικρό) Δt και η πρώτη σχέση μετατρέπεται σε:

$$\frac{u_i(t+\Delta t) - u_i(t)}{\Delta t} = \mu - u_i(t) + \frac{1}{N_c}\sum_{j\in\{N_i\}}^{N_c} \sigma_{ij}\cdot\left[u_j(t) - u_i(t)\right] \Rightarrow$$

$$u_i(t+\Delta t) = u_i(t) + \Delta t\cdot\left(\mu - u_i(t) + \frac{1}{N_c}\sum_{j\in\{N_i\}}^{N_c} \sigma_{ij}\cdot\left[u_j(t) - u_i(t)\right]\right)$$

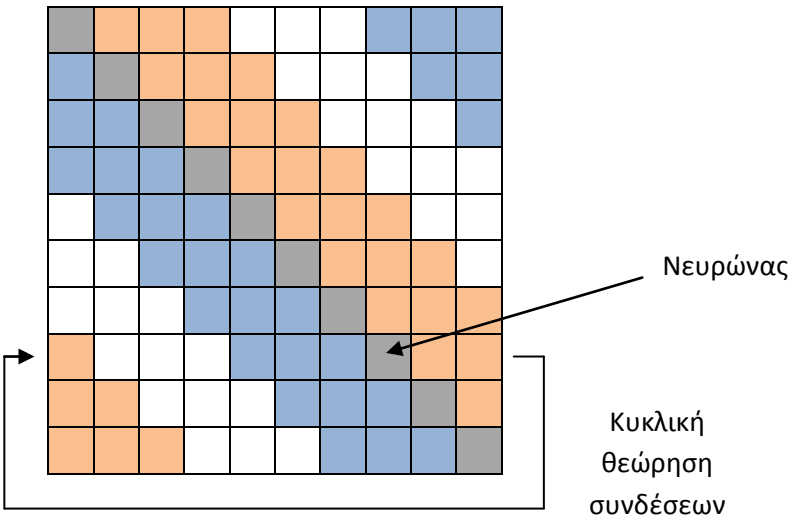
Η τελευταία σχέση δίνει το νέο δυναμικό του νευρώνα i την χρονική στιγμή $t+\Delta t$, βάσει του δυναμικού του νευρώνα και των δυναμικών όλων των γειτόνων την χρονική στιγμή t και είναι αυτή που θα χρησιμοποιηθεί στα πλαίσια της εργασίας. Κατά συνέπεια, αν γνωρίζουμε τις αρχικές τιμές των δυναμικών όλων των νευρώνων, τότε για κάθε νευρώνα μπορούμε να εφαρμόσουμε την παραπάνω σχέση και να δούμε πως εξελίσσεται το δυναμικό του στον χρόνο, με βάση τις συνδέσεις και την επιρροή που έχει από τους γείτονες του.

Η επιρροή που έχει κάθε νευρώνας στους υπόλοιπους νευρώνες, η οποία εκφράζεται μέσω του μητρώου σ, μπορεί να μοντελοποιηθεί με πολλούς τρόπους. Η πιο γενική περίπτωση είναι κάθε νευρώνας να συνδέεται και να επηρεάζει με διαφορετικό τρόπο κάθε άλλο νευρώνα, με λίγα λόγια όλο το μητρώο σ να έχει μη μηδενικές τιμές και αυτές να μην έχουν μεταξύ τους καμία συνάφεια. Η ακολουθιακή υλοποίηση που θα σας δωθεί υποθέτει την συγκεκριμένη προσέγγιση. Ωστόσο, για να διευκολύνουμε στα πλαίσια της εργασίας τον ορισμό του συνόλου των γειτόνων $\{N_i\}$ ενός νευρώνα i, θα χρησιμοποιήσουμε ένα μοντέλο σύνδεσης που συναντάται συχνά στην βιβλιογραφία. Συγκεκριμένα, θα θεωρήσουμε πως όλοι οι νευρώνες ανήκουν σε μια συνδεσμολογία δακτυλίου, και πως κάθε νευρώνας συνδέεται με τους r προηγούμενους και r επόμενους νευρώνες στον δακτύλιο, δηλαδή έχει συνολικά $N_c = 2\cdot r$ γείτονες, όπως φαίνεται και στην Εικόνα 3.



Εικόνα 3

Κατά συνέπεια, στο μητρώο σ το διαγώνιο στοιχείο στην γραμμή i θα αναπαριστά τον νευρώνα i και οι προηγούμενες και επόμενες r θέσεις της γραμμής θα περιέχουν μη μηδενικά στοιχεία, θεωρώντας μάλιστα πως η σύνδεση σε κάθε γραμμή είναι κυκλική, δηλαδή μετά το τελευταίο στοιχείο της γραμμής ακολουθεί το πρώτο. Αν θεωρήσουμε πως $n = 10$ και $r = 3$, τότε το μητρώο σ θα έχει την παρακάτω μορφή, με τα χρωματιστά στοιχεία να περιέχουν μη μηδενικές τιμές και τα λευκά στοιχεία να περιέχουν μηδέν. Τα γκρι στοιχεία δείχνουν τον νευρώνα αναφοράς, τα πορτοκαλί στοιχεία μιας γραμμής δείχνουν τους δεξιούς γείτονες του και τα μπλε στοιχεία μιας γραμμής τους αριστερούς γείτονες του.



<u>Προσοχή</u>: Μην βελτιστοποιήσετε την εφαρμογή θεωρώντας πως κάποια στοιχεία του μητρώου σ είναι μηδέν. Θέλουμε η εφαρμογή να υποστηρίζει κάθε πιθανή συνδεσμολογία νευρώνων, δηλαδή κάθε νευρώνας θα συνεχίσουμε να θεωρούμε πως συνδέεται και με τους $n-1$ υπόλοιπους νευρώνες. Ο μόνος λόγος για την παραπάνω παραδοχή είναι για να γνωρίζουμε το μέγεθος του συνόλου των γειτόνων, δηλαδή ότι $N_c = 2 \cdot r$.

Τέλος, παρατηρήστε πως η συνδεσμολογία αυτή είναι μονοδιάστατη, δηλαδή οι νευρώνες θεωρούμε πως είναι διατεταγμένοι σε μια ευθεία (με τον τελευταίο νευρώνα να συνδέεται βεβαίως με τον πρώτο). Για τον λόγο αυτό οι συνδέσεις κάθε νευρώνα με τους γείτονες του μπορούν να αναπαρασταθούν ως ένα διάνυσμα και συνολικά οι συνδέσεις για όλους τους νευρώνες ως ένα δισδιάστατο μητρώο. Φυσικά υπάρχουν και άλλες προσεγγίσεις, στις οποίες οι νευρώνες θεωρείται πως είναι διατεταγμένοι σε ένα δισδιάστατο επίπεδο. Έτσι η επιρροή ενός νευρώνα που βρίσκεται στις συντεταγμένες (k, l) πάνω σε έναν νευρώνα που βρίσκεται στις συντεταγμένες (i, j) περιγράφεται μέσω του στοιχείου σ_{ijkl} ενός τετραδιάστατου μητρώο σ. Αντίστοιχα, αν οι νευρώνες τοποθετηθούν σε έναν τρισδιάστατο χώρο, τότε το μητρώο σ θα έχει 6 διαστάσεις. Οι σχέσεις του μοντέλου LIF δεν αλλάζουν ουσιαστικά στις περιπτώσεις αυτές, με εξαίρεση τους δείκτες που χρησιμοποιούνται για να υποδείξουν τους νευρώνες και το αντίστοιχο στοιχείο του μητρώου σ. Για λόγους απλότητας λοιπόν θα μείνουμε στο μονοδιάστατο μοντέλο στα πλαίσια της εργασίας.

Ζητούμενα της άσκησης

Στα πλαίσια της άσκησης θα σας δωθεί μια ακολουθιακή υλοποίηση του μοντέλου LIF. Η υλοποίηση αυτή είναι μια απευθείας απεικόνιση των παραπάνω μαθηματικών σχέσεων σε πρόγραμμα, όπως θα το έγραφε κάποιος νευροεπιστήμονας που έχει μικρή σχέση με τον προγραμματισμό. Αυτή είναι μια περίπτωση που είναι πολύ πιθανόν να συναντήσετε στο μέλλον, να πρέπει δηλαδή να δουλέψετε με βάση κώδικα που δεν έχει γραφεί από καλούς γνώστες προγραμματισμού και εσείς να πρέπει να τον βελτιστοποιήσετε/επιταχύνετε. Σκοπός σας λοιπόν θα είναι:

1) Να βελτιστοποιήσετε τον κώδικα και να τον επιταχύνετε **πριν ξεκινήσετε την παραλληλοποίηση του**.
    a. Παρατηρήστε πως πρέπει να διατηρούμε δύο διανύσματα, τα οποία περιέχουν τις τρέχουσες (u) και τις επόμενες ($uplus$) τιμές του δυναμικού κάθε νευρώνα. Στην υλοποίηση που σας δίνεται, κάθε νέα τιμή αντιγράφεται τελικά στην προηγούμενη στο τέλος κάθε επανάληψης. Επομένως για n νευρώνες πρέπει να γίνουν n αντιγραφές, το οποίο κοστίζει σημαντικά σε χρόνο όταν το n είναι μεγάλο. Σκεφτείτε μια πιο αποδοτική προσέγγιση.
    b. Επιπροσθέτως στο παραπάνω, μπορείτε να αναδιοργανώσετε τους υπολογισμούς που πρέπει να γίνουν σε κάθε νευρώνα ώστε σε κάθε επανάληψη να χρειάζονται λιγότερες πράξεις; Π.χ. μπορείτε να υπολογίσετε μόνο μια φορά κάποιο τμήμα των υπολογισμών για κάθε νευρώνα, να το αποθηκεύσετε και μετά απλά να το χρησιμοποιείτε σε κάθε επανάληψη αντί να το υπολογίζετε εκ νέου;
    c. Αν καταφέρατε το παραπάνω, τότε αυτομάτως σας δίνετε η δυνατότητα να κάνετε άλλη μια βελτιστοποίηση. Σε πολλές εφαρμογές πρέπει να υπολογισθούν βασικές αλγεβρικές πράξεις, όπως γινόμενο μητρώου με μητρώο, γινόμενο μητρώου με διάνυσμα, κλπ. Υπάρχουν πολλοί έξυπνοι αλγόριθμοι οι οποίοι μειώνουν το συνολικό πλήθος το πράξεων που απαιτούνται για την πραγματοποίηση των υπολογισμών αυτών και/ή εκμεταλλεύονται καλύτερα στοιχεία της αρχιτεκτονικής ενός υπολογιστή, π.χ. την κρυφή μνήμη (cache). Ωστόσο, ο προγραμματισμός τους είναι συνήθως πολύπλοκος. Το πρότυπο συναρτήσεων Basic Linear Algebra Subroutines (BLAS) ορίζει ένα σύνολο συναρτήσεων για την πραγματοποίηση βασικών αλγεβρικών πράξεων. Η υλοποίηση των συναρτήσεων αυτών μπορεί να βασίζεται σε εξαιρετικά αποδοτικούς αλγόριθμους, αλλά κρύβει την πολυπλοκότητα υλοποίησης τους από τον χρήστη. Εγκαταστήστε μια έτοιμη υλοποίηση των BLAS στον υπολογιστή σας και αντικαταστήστε τον βασικό υπολογισμό (ποιος είναι μετά τις αλλαγές του προηγούμενου βήματος;) με την αντίστοιχη κλήση στην σωστή συνάρτηση BLAS.
2) Αφού αξιολογήσετε την βελτίωση στην επίδοση που πήρατε από τα παραπάνω βήματα (δείτε για λεπτομέρειες το Παράρτημα Β), είστε έτοιμοι να δημιουργήσετε παράλληλες εκδόσεις του προγράμματος. Συγκεκριμένα:
    a. Θα ξεκινήσετε από την έκδοση του προγράμματος που φτιάξατε στο Βήμα 1.b (πριν δηλαδή χρησιμοποιήσετε την βιβλιοθήκη BLAS, την οποία θα επαναφέρουμε στην συνέχεια και στην παράλληλη έκδοση). Παρατηρήστε πως το κυρίως τμήμα του υπολογισμού έχει 3 εμφωλευμένους βρόχους (loop), οι οποίοι χρησιμοποιούν τις μεταβλητές it, i και j. Χρησιμοποιήστε την

οδηγία “pragma omp parallel” του προτύπου OpenMP σε κάθε έναν από τους 3 βρόχους (κάθε φορά σε έναν μόνο από τους 3 βρόχους). Χρησιμοποιείστε επιπλέον όποιες άλλες οδηγίες χρειαστείτε για διαμοιρασμό εργασίας μεταξύ νημάτων (#pragma omp for), συγχρονισμό #pragma omp barrier, #pragma omp critical, #pragma omp atomic, …), κλπ για να παραλληλοποιήσετε **σωστά** το πρόγραμμα. Σε ποια περίπτωση αναμένετε την καλύτερη βελτίωση της απόδοσης και γιατί; Είναι όντως η περίπτωση που προβλέψατε η καλύτερη;

b. Κρατήστε την καλύτερη υλοποίηση από τις παραπάνω 3 που δοκιμάσατε και επαναφέρετε σε αυτήν την χρήση της βιβλοθήκης BLAS. Μπορεί αυτό να γίνει με χρήση της αυτόματης οδηγίας διαμοιρασμού εργασίας “#pragma omp for” (που πιθανότατα χρησιμοποιήσατε); Αν όχι, τι πρέπει να κάνετε;

3) Μέχρι στιγμής ο κώδικας αποθήκευε αποτελέσματα σε αρχεία ανά τακτά χρονικά διαστήματα της εξομοίωσης (π.χ. κάθε 10000 επαναλήψεις του βρόχου “it”). Αυτό έγινε για να μειώσουμε τον χρόνο εκτέλεσης της εφαρμογής και τον απαιτούμενο αποθηκευτικό χώρο, δίνοντας όμως ταυτόχρονα την δυνατότητα να ελέγχετε και την ορθότητα των προσεγγίσεων σας. Στην πραγματικότητα όμως χρειαζόμαστε **όλα** τα αποτελέσματα, για κάθε βήμα της εξομοίωσης.

   Μεταγλωττίστε τώρα το πρόγραμμα δίνοντας επιπλέον κατά τον χρόνο μεταγλώττισης την παράμετρο –DALL_RESULTS και εκτελέστε ξανά το παράλληλο πρόγραμμα που δημιουργήσατε στο Βήμα 2.b. Τι παρατηρείτε ως προς τον χρόνο που απαιτείται για τους υπολογισμούς και για την αποθήκευση των αποτελεσμάτων;

   Και αυτό το σενάριο είναι πολύ συχνό σε παράλληλες εφαρμογές. Αναζητείστε και εφαρμόστε τεχνικές για την μείωση του χρόνου που απαιτείται για την αποθήκευση αποτελεσμάτων σε ένα παράλληλο πρόγραμμα (buffering, ασύγχρονη εγγραφή σε αρχείο, διαφορετική αναπαράσταση των αποτελεσμάτων, …).

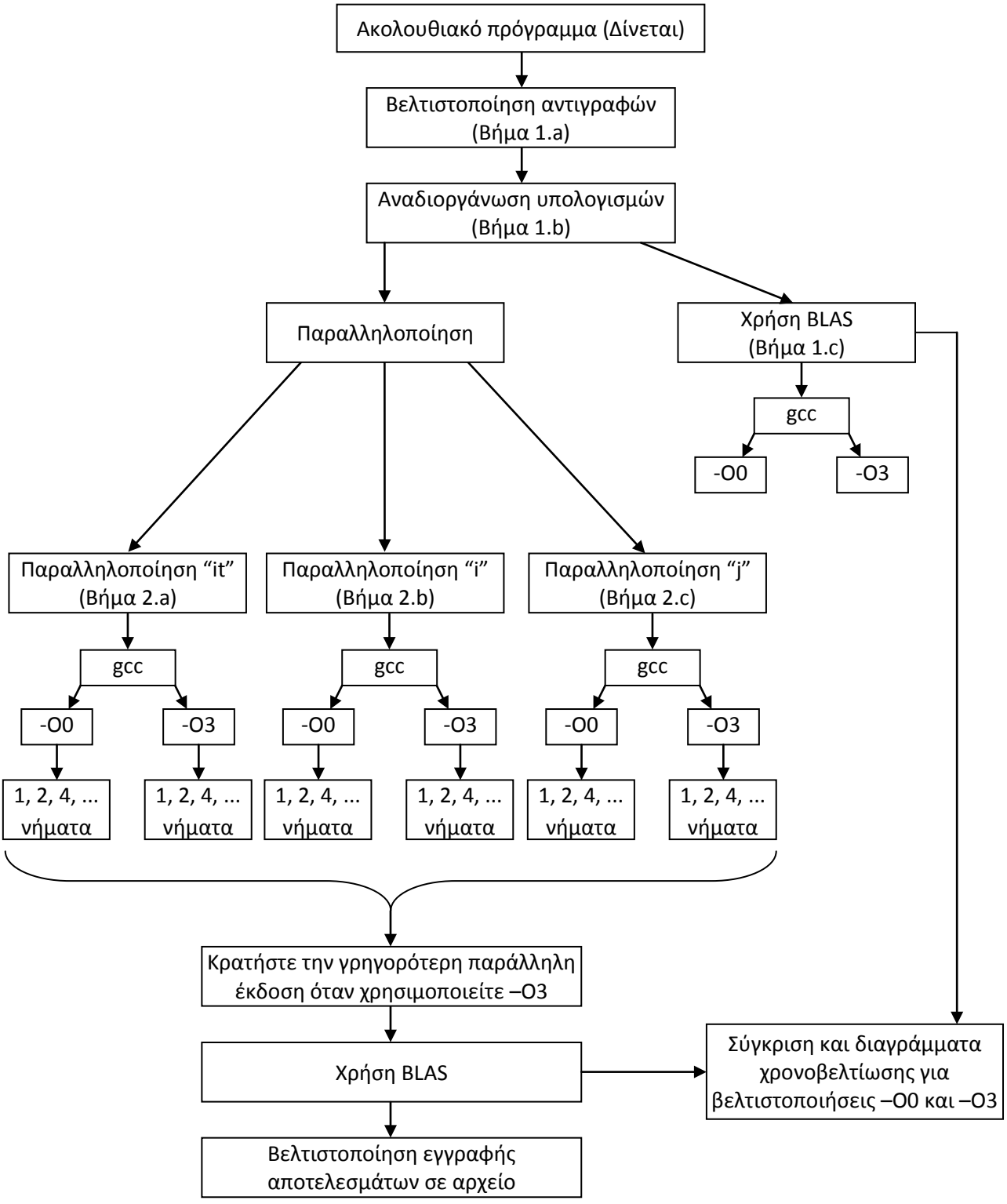
Εκτός όλων των παραπάνω είσαστε προφανώς ελεύθεροι να βελτιστοποιήσετε και να παραλληλοποιήσετε τον κώδικα με όποιον άλλο τρόπο θέλετε. Αν θεωρήσετε πως μια οποιαδήποτε αναδιοργάνωση του κώδικα μπορεί να οδηγήσει σε καλύτερη παραλληλοποίηση και απόδοση είστε ελεύθεροι να το κάνετε. Ωστόσο, θεωρείται αυτονόητο πως η υλοποίηση θα πρέπει να είναι σωστή (να δίνει **πάντα** τα ίδια αποτελέσματα με την ακολουθιακή έκδοση του προγράμματος) και πως **στην αναφορά που θα παραδώσετε θα πρέπει να αιτιολογήσετε γιατί παραλληλοποιήσατε με τον συγκεκριμένο τρόπο την εφαρμογή.**

Οδηγίες για την μεταγλώττιση και εκτέλεση του ακολουθιακού κώδικα που σας δίνεται και του παράλληλου κώδικα που θα φτιάξετε δίνονται στο Παράρτημα Β της εκφώνησης. Εκεί επίσης αναφέρεται για ποιες παραμέτρους θα πρέπει να τρέξετε την κάθε έκδοση του προγράμματος και ποια αποτελέσματα θα συμπεριλάβετε στην αναφορά σας.

Συγκεντρωτικά, θα πρέπει στην αναφορά σας να παραθέσετε μετρήσεις και να σχολιάσετε τις παρακάτω περιπτώσεις:

1) Βελτιστοποίηση ακολουθιακού προγράμματος
    a. Ακολουθιακό πρόγραμμα που δίνεται.
    b. Ακολουθιακό πρόγραμμα με βελτιστοποίηση αντιγραφών (Βήμα 1.a).
    c. Ακολουθιακό πρόγραμμα με βελτιστοποίηση αντιγραφών + αναδιοργάνωση υπολογισμών (Βήμα 1.a + Βήμα 1.b).
    d. Ακολουθιακό πρόγραμμα με βελτιστοποίηση αντιγραφών + αναδιοργάνωση υπολογισμών + χρήση βιβλιοθήκης BLAS (Βήμα 1.a + Βήμα 1.b + Βήμα 1.c).
    e. Μεταγλώττιση και εκτέλεση κάθε παραπάνω προγράμματος χωρίς βελτιστοποιήσεις (-O0) και με μέγιστες βελτιστοποιήσεις (-O3) του μεταφραστή.
    f. Δώστε ιδιαίτερη βάση στις μετρήσεις με και χωρίς βελτιστοποιήσεις του μεταφραστή. Τι επίπτωση έχει κάθε βελτιστοποίηση που κάνετε εσείς (Βήματα 1.a, 1.b, 1.c) στον χρόνο εκτέλεσης του προγράμματος όταν χρησιμοποιούνται οι βελτιστοποιήσεις του μεταφραστή και όταν δεν χρησιμοποιούνται;
2) Παραλληλοποιημένος κώδικας (βασιζόμενοι στο ακολουθιακό πρόγραμμα που προέκυψε από το Βήμα 1.b)
    a. Παραλληλοποίηση μόνο του βρόχου “it”
    b. Παραλληλοποίηση μόνο του βρόχου “i”
    c. Παραλληλοποίηση μόνο του βρόχου “j”
    d. Κρατήστε το καλύτερο από τις παραπάνω 3 περιπτώσεις (όταν χρησιμοποιείτε μέγιστες βελτιστοποιήσεις –O3)
        i. Προσθέστε χρήση βιβλιοθήκης BLAS
    e. Μεταγλώττιση και εκτέλεση κάθε παραπάνω προγράμματος χωρίς βελτιστοποιήσεις (-O0) και με μέγιστες βελτιστοποιήσεις (-O3) του μεταφραστή.
    f. **Δώστε ιδιαίτερη βάση στις μετρήσεις με και χωρίς βελτιστοποιήσεις του μεταφραστή. Τι επίπτωση έχει κάθε βελτιστοποίηση που κάνετε εσείς (Βήματα 2.a, 2.b, 2.c, 2.d) στον χρόνο εκτέλεσης του προγράμματος όταν χρησιμοποιούνται οι βελτιστοποιήσεις του μεταφραστή και όταν δεν χρησιμοποιούνται;**
    g. Εκτέλεση κάθε περίπτωσης με 1, 2 και 4 νήματα τουλάχιστον. Αν το σύστημα σας διαθέτει περισσότερους πυρήνες, ακόμα καλύτερα!
3) Στο καλύτερο παράλληλο πρόγραμμα που προέκυψε
    a. Βελτιστοποίηση εγγραφής αποτελεσμάτων σε αρχείο.

Για κάθε έναν από τους παραπάνω συνδυασμούς συμπεριλάβετε στην αναφορά σας πίνακες με τους χρόνους εκτέλεσης του βασικού αλγόριθμου (υπάρχει έτοιμο στον κώδικα που σας δίνεται) και **διαγράμματα της χρονοβελτίωσης**. Συγκεντρωτικά, όλες οι περιπτώσεις που θα πρέπει να συμπεριλάβετε φαίνονται στο παρακάτω σχήμα:


Ακολουθιακό πρόγραμμα (Δίνεται)
Βελτιστοποίηση αντιγραφών (Βήμα 1.a)
Αναδιοργάνωση υπολογισμών (Βήμα 1.b)
Παραλληλοποίηση
Χρήση BLAS (Βήμα 1.c)
gcc
-O0
-O3
Παραλληλοποίηση “it” (Βήμα 2.a)
Παραλληλοποίηση “i” (Βήμα 2.b)
Παραλληλοποίηση “j” (Βήμα 2.c)
gcc
gcc
gcc
-O0
-O3
-O0
-O3
-O0
-O3
1, 2, 4, ... νήματα
1, 2, 4, ... νήματα
1, 2, 4, ... νήματα
1, 2, 4, ... νήματα
1, 2, 4, ... νήματα
1, 2, 4, ... νήματα
Κρατήστε την γρηγορότερη παράλληλη έκδοση όταν χρησιμοποιείτε –O3
Χρήση BLAS
Σύγκριση και διαγράμματα χρονοβελτίωσης για βελτιστοποιήσεις –O0 και –O3
Βελτιστοποίηση εγγραφής αποτελεσμάτων σε αρχείο

Διαδικαστικά

Η εργασία θα πρέπει να γίνει σε ομάδες των 3 ή 4 ατόμων **ακριβώς**. Δεν θα γίνει δεκτή ομάδα με λιγότερα ή περισσότερα άτομα **για κανέναν λόγο**. Αν δεν συμπληρώσετε ομάδα 3 ή 4 ατόμων, τότε θα προστεθούν άτομα στην ομάδα σας από εμάς.

Η δήλωση των ομάδων θα γίνει στην ηλεκτρονική πλατφόρμα "Open eClass" του Πανεπιστημίου Πατρών (http://eclass.upatras.gr). **Για τον σκοπό αυτό θα πρέπει όλοι οι φοιτητές που επιθυμούν να παραδώσουν εργασία να εγγραφούν πρώτα στην παραπάνω πλατφόρμα.** Στην συνέχεια, ένα άτομο από κάθε ομάδα θα αναλάβει να δηλώσει την ομάδα του μέχρι την **Παρασκευή, 06/04/2018 και ώρα 23:59:59**. Το άτομο αυτό θα είναι επίσης υπεύθυνο για όλη την επικοινωνία της ομάδας μαζί μας, καθ' όλη την διάρκεια του εξαμήνου και μέχρι την παράδοση της άσκησης. Η ομάδα θα δηλωθεί μέσω e-mail στην διεύθυνση venetis@ceid.upatras.gr. Για την ευκολότερη ταξινόμηση από την μεριά μας και την δυνατότητα αυτόματης προώθησης, το e-mail θα πρέπει να έχει τον εξής τίτλο:

`[ParPro17-18] Δήλωση ομάδας`

Το περιεχόμενο του e-mail θα πρέπει να είναι ο Α.Μ. και το ονοματεπώνυμο του φοιτητή που κάνει την δήλωση της ομάδας, καθώς επίσης το πλήθος των ατόμων της ομάδας. Στην συνέχεια θα αναλάβουμε να φτιάξουμε μια ομάδα στο "Open eClass" και θα σας ενημερώσουμε για τον αριθμό της ομάδας σας. **Η δημιουργία των ομάδων θα γίνει συγκεντρωτικά για όλους και μετά την λήξη της προθεσμίας** (δεν θα υπάρξει δηλαδή άμεση απάντηση από την μεριά μας για το θέμα).

Σε περίπτωση που χρειαστεί επιπλέον επικοινωνία μαζί μας μέσω e-mail, αυτή θα πρέπει να γίνει με τον κ. Ιωάννη Βενέτη (venetis@ceid.upatras.gr). **Για την ευκολότερη ταξινόμηση από την μεριά μας και την δυνατότητα αυτόματης προώθησης, ο τίτλος κάθε e-mail θα πρέπει να ξεκινάει με [ParPro17-18].**

Παραδοτέα

Τα παραδοτέα για την εργασία σας είναι μια γραπτή αναφορά και ο κώδικας της άσκησης που θα αναπτύξετε. **Η προθεσμία παράδοσης της εργασίας ορίζεται η Κυριακή 20/05/2018 και ώρα 23:59:59. Η εργασία θα πρέπει να παραδωθεί αποκλειστικά μέσω της ηλεκτρονικής πλατφόρμας "Open eClass" (εργασίες που θα αποσταλούν μέσω e-mail δεν θα βαθμολογηθούν) και η πλατφόρμα θα κλειδώσει αυτόματα την δυνατότητα υποβολής εργασιών μετά την λήξη της προθεσμίας. Οργανώστε λοιπόν σωστά τον χρόνο σας.** Μετά την είσοδο σας στο σύστημα θα πρέπει να μεταβείτε στο μάθημα "Παράλληλη Επεξεργασία" και στο μενού αριστερά να μεταβείτε στο "Εργασίες". **Κάθε ομάδα θα παραδώσει μια φορά μόνο την εργασία (όχι κάθε φοιτητής ξεχωριστά).**

Στην αναφορά **δεν** θα πρέπει να περιλαμβάνεται επεξήγηση του ακολουθιακού αλγόριθμου που σας δώθηκε. Επικεντρωθείτε στην επεξήγηση της παραλληλοποίησης που κάνατε, στις μετρήσεις σας και στα διαγράμματα που θα προσθέσετε. Σχολιάστε τις διαφορές από την χρήση βελτιστοποιήσεων στον χρόνο εκτέλεσης της εφαρμογής και (κυρίως) στην χρονοβελτίωση. Γενικότερα, δώστε ιδιαίτερο βάρος στην αναφορά σε αυτά που κάνατε εσείς.

Ο βαθμός της εργασίας αποτελεί το 30% της τελικής βαθμολογίας. Το υπόλοιπο 70% προκύπτει από την τελική εξέταση. Για να περάσει κάποιος φοιτητής το μάθημα δεν είναι

απαραίτητη η παράδοση της εργασίας. **Στην περίπτωση αυτή ωστόσο, θεωρείται πως η εργασία έχει πάρει βαθμό 0 (μηδέν). Ο τελικός βαθμός τότε προκύπτει μόνο από το 70% της τελικής εξέτασης και θα πρέπει να είναι προβιβάσιμος (≥ 5).**

Ο βαθμός της εργασίας διατηρείται μέχρι και την άτυπη εξεταστική Φεβρουαρίου 2019. Αν κάποιος φοιτητής δεν περάσει το μάθημα μέχρι τότε θα πρέπει να παρακολουθήσει εξ αρχής το μάθημα και να ανταποκριθεί στις υποχρεώσεις του μαθήματος για το ακαδημαϊκό έτος που θα το παρακολουθήσει ξανά.

Σημειώνεται επίσης πως δεν είναι δυνατή η παράδοση της εργασίας σε άλλη ημερομηνία, π.χ. κατά την εξεταστική του Σεπτεμβρίου ή την άτυπη εξεταστική Φεβρουαρίου.

Παράρτημα Α

Για την εγκατάσταση των προγραμμάτων που θα χρειαστείτε στα πλαίσια της άσκησης, θα πρέπει να ακολουθήσετε τις παρακάτω οδηγίες.

1) Εγκατάσταση Linux

   Στα πλαίσια της εργασίας συστήνεται να εγκαταστήσετε την έκδοση 16.04.4 LTS της διανομής Ubuntu (http://www.ubuntu.com). Οι οδηγίες που ακολουθούν υποθέτουν την εγκατάσταση της συγκεκριμένης διανομής. Φυσικά μπορείτε να εγκαταστήσετε όποια άλλη διανομή θέλετε, όπως για παράδειγμα Fedora (http://fedoraproject.org) ή openSUSE (http://www.opensuse.org). Σε κάθε περίπτωση επιλέξτε την τελευταία έκδοση που είναι διαθέσιμη για κάθε διανομή. Αν εγκαταστήσετε άλλη διανομή εκτός της Ubuntu 16.04.4 LTS θα πρέπει να προσαρμόσετε κατάλληλα της οδηγίες για την εγκατάσταση των επιπλέον πακέτων που απαιτούνται.

   Οι διανομές συνήθως είναι διαθέσιμες ως αρχεία τύπου ISO. Αν εγκαταστήσετε την διανομή απευθείας στον σκληρό δίσκο του υπολογιστή σας (βλέπε παρακάτω) θα πρέπει να γράψετε το αρχείο ISO σε ένα CD, DVD ή USB stick. Αν εγκαταστήσετε την διανομή σε ένα Virtual Machine (βλέπε παρακάτω) μπορείτε να χρησιμοποιήσετε απευθείας το αρχείο ISO. Η εγκατάσταση της διανομής μπορεί να γίνει με δύο τρόπους:

   a. Απευθείας στον σκληρό δίσκο του μηχανήματος σας

      Αν ακολουθήσετε την μέθοδο αυτή θα πρέπει να φτιάξετε ένα ξεχωριστό partition στον σκληρό σας δίσκο. Όταν ξεκινήσετε την εγκατάσταση της διανομής θα πρέπει να επιλέξετε το συγκεκριμένο partition. Ακολουθήστε αυτή τη μέθοδο αν θέλετε να κρατήσετε την διανομή που θα εγκαταστήσετε και για αργότερα.

   b. Μέσω Virtual Machine (VM)

      "Virtual Machine" είναι στην πραγματικότητα οποιοδήποτε πρόγραμμα που μπορούμε να εγκαταστήσουμε σε ένα λειτουργικό σύστημα που διαθέτουμε και προσομοιώνει έναν υπολογιστή. Αν για παράδειγμα διαθέτουμε Windows, μπορούμε να εγκαταστήσουμε ένα Virtual Machine και στην συνέχεια να εγκαταστήσουμε στον προσομοιούμενο υπολογιστή ένα άλλο λειτουργικό σύστημα, όπως για παράδειγμα Linux.

      Αν ακολουθήσετε αυτή τη μέθοδο θα πρέπει κατ' αρχάς να κατεβάσετε και να εγκαταστήσετε στο λειτουργικό σύστημα που διαθέτετε ένα Virtual Machine. Υπάρχουν αρκετά ελεύθερα προγράμματα διαθέσιμα για αυτό, όπως το Virtual Box (https://www.virtualbox.org), **το οποίο και συστήνουμε για την εργασία**, το VMware Player (http://www.vmware.com/products/player/overview.html) και το QEMU (http://wiki.qemu.org). Επειδή θα χρειαστεί να τρέξουμε παράλληλα προγράμματα, διαβάστε τις οδηγίες εγκατάστασης και δημιουργίας Virtual Machine, ώστε η Virtual Machine στην οποία θα εγκαταστήσετε την διανομή Linux που θα επιλέξετε να υποστηρίζει πολλαπλούς πυρήνες. Επίσης, δηλώστε έναν αρκετά μεγάλο σκληρό δίσκο (τουλάχιστον 10GB). Τέλος, εγκαταστήστε την διανομή Linux που επιλέξατε στην Virtual Machine που φτιάξατε.

2) Εγκατάσταση Guest Additions

Αν έχετε εγκαταστήσει VirtualBox τότε μια σημαντική προσθήκη που μπορείτε να κάνετε είναι τα “Guest Additions”. Τα εργαλεία αυτά προσφέρουν επιπλέον δυνατότητες στον εικονικό υπολογιστή, με κυριότερες την καλύτερη ανάλυση οθόνης και το Clipboard για την αντιγραφή δεδομένων και αρχείων μεταξύ του πραγματικού και του εικονικού υπολογιστή. **Η εγκατάσταση των “Guest Additions” είναι εντελώς προαιρετική. Μπορείτε να ολοκληρώσετε την εργασία και χωρίς αυτά.** Αν θέλετε να τα εγκαταστήσετε ακολουθήστε τα παρακάτω βήματα:

a. Πριν εκκινήσετε την VM εγκαταστήστε στο VirtualBox το Extension Pack: https://www.virtualbox.org/manual/ch01.html#intro-installing
b. Εκκινήστε την VM και εγκαταστήστε τα προαπαιτούμενα πακέτα στο λειτουργικό σύστημα του εικονικού υπολογιστή. Από την γραμμή εντολών εκτελέστε την εντολή:

   ```
   sudo apt-get install dkms
   ```

c. Κάντε επανεκκίνηση του υπολογιστή σας.
d. Στο μενού του Virtual Box επιλέξτε “Devices → Insert Guest Additions CD image…”.
e. Επιτρέψτε την αυτόματη εκτέλεση του προγράμματος εγκατάστασης.
f. Κάντε επανεκκίνηση του υπολογιστή σας.
g. Ανοίξτε το πρόγραμμα Διαχείρισης Αρχείων (File Manager).
h. Κάντε unmount το “Guest Additions CD” κάνοντας κλικ στο σύμβολο “⏏”.

Παράρτημα Β

1) Για την εκτέλεση του ακολουθιακού προγράμματος που σας δίνεται και των παράλληλων προγραμμάτων που θα φτιάξετε απαιτείται το πέρασμα παραμέτρων από την γραμμή εντολής. Κατά την διάρκεια ανάπτυξης και αποσφαλμάτωσης των προγραμμάτων σας μπορείτε να χρησιμοποιείται τις παρακάτω παραμέτρους, οι οποίες οδηγούν σε σχετικά μικρούς χρόνους εκτέλεσης:

   ```
   --n 200 --r 60
   --n 400 --r 120
   --n 600 --r 200
   --n 800 --r 250
   --n 1000 --r 350
   ```

   Με αυτές μπορείτε να ελέγχετε γρήγορα αν οι αλλαγές που κάνετε για την βελτιστοποίηση ή/και παραλληλοποίηση είναι σωστές. Για τις μετρήσεις που θα περιλαμβάνονται στην αναφορά σας θα χρησιμοποιήσετε τις παρακάτω παραμέτρους, που απαιτούν περισσότερο χρόνο:

   ```
   --n 1000 --r 350
   --n 2000 --r 700
   --n 3000 --r 1000
   --n 4000 --r 1300
   --n 5000 --r 1600
   ```

2) Για την μεταγλώττιση του ακολουθιακού προγράμματος που σας δίνεται μπορείτε να χρησιμοποιήσετε την παρακάτω εντολή:

   ```
   gcc -O3 (ή –O0) -Wall -Wextra -o lif1d lif1d.c
   ```

   Αυτή θα δημιουργήσει ένα εκτελέσιμο αρχείο με το όνομα “lif1d”, το οποίο μπορείτε να εκτελέσετε με την εντολή:

   ```
   ./lif1d <Παράμετροι γραμμής εντολής>
   ```

   Οι παράμετροι γραμμής εντολής είναι κάποιο από τα σύνολα που δώθηκαν παραπάνω. **Συνιστάται να τρέξετε την ακολουθιακή εφαρμογή με όλα τα παραπάνω σύνολα παραμέτρων, ώστε να έχετε τα αποτελέσματα σαν αναφορά για τις αλλαγές που θα κάνετε κατά την παραλληλοποίηση. Μην ξεχνάτε πως θα πρέπει να παίρνετε τα ίδια αποτελέσματα από το ακολουθιακό και το παράλληλο πρόγραμμα! Μετά την εκτέλεση του ακολουθιακού προγράμματος για κάθε σύνολο παραμέτρων μην ξεχάσετε να μετονομάσετε κατάλληλα τα αρχεία που παράγονται, καθώς σε αυτά θα εγγραφούν τα αποτελέσματα της επόμενης εκτέλεσης του προγράμματος.**

3) Για την μεταγλώττιση του παράλληλου προγράμματος με OpenMP που θα φτιάξετε μπορείτε να χρησιμοποιήσετε την παρακάτω εντολή (θεωρώντας πως το όνομα του αρχείου που φτιάξατε είναι “lif1d_omp.c”):

   ```
   gcc -O3 (ή –O0) –fopenmp -Wall -Wextra -o lifcd_omp lifcd_omp.cpp
   ```

Αυτή θα δημιουργήσει ένα εκτελέσιμο αρχείο με το όνομα “lif1d_omp”. Μπορείτε να ορίσετε το πλήθος των νημάτων που θα δημιουργούνται σε κάθε παράλληλη περιοχή και στην συνέχεια να το εκτελέσετε με τις εντολές:

```
export OMP_NUM_THREADS=<Πλήθος νημάτων ανά παράλληλη περιοχή>
./lif1d_omp <Παράμετροι γραμμής εντολής>
```

4) Για να κάνετε εύκολα την σύγκριση αποτελεσμάτων που έχετε αποθηκεύσει σε αρχεία μπορείτε να χρησιμοποιήσετε το πρόγραμμα numdiff. Η ακρίβεια των αποτελεσμάτων πρέπει να είναι τουλάχιστον μέχρι το 12 δεκαδικό ψηφίο:

```
numdiff -a 1e-12 spacetime1.txt spacetime2.txt
```