ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΕΠΙΚΟΙΝΩΝΙΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πτυχιακή εργασία

Τίτλος Πτυχιακής Εργασίας	Ταυτόχρονη προσομοίωση πολλαπλών δικτύων βιολογικών νευρώνων σε παράλληλα συστήματα Simultaneous simulation of multiple biological neural networks in parallel systems
Ονοματεπώνυμο Φοιτητή	Κωνσταντίνος Λοϊζίδης
Πατρώνυμο	Γιαννάκης
Αριθμός Μητρώου	Π20007
Επιβλέπων	Ιωάννης Βενέτης, Επίκουρος Καθηγητής

Ημερομηνία Παράδοσης: ΟΚΤΩΒΡΙΟΣ 2024

Copyright ©

Ευχαριστίες

Πρώτα απ' όλα, ευχαριστώ θερμά τον καθηγητή μου, Ιωάννη Βενέτη, για την ακαδημαϊκή του καθοδήγηση και τις πολύτιμες συμβουλές του καθ' όλη τη διάρκεια της εργασίας. Η υποστήριξή του και η κατεύθυνση που μου παρείχε στα ακαδημαϊκά και ερευνητικά ζητήματα ήταν καταλυτικής σημασίας για την πορεία και την εξέλιξη της εργασίας.

Ιδιαίτερες ευχαριστίες αξίζουν στην οικογένειά μου, που στάθηκε δίπλα μου σε κάθε βήμα και με στήριξε σε προσωπικό επίπεδο. Η υπομονή, η κατανόηση και η αγάπη τους αποτέλεσαν τη βάση για να αντιμετωπίσω τις προκλήσεις και να διατηρήσω τη συγκέντρωσή μου στις απαιτήσεις της εργασίας αυτής.

Επίσης, ευχαριστώ από καρδιάς τους φίλους μου, οι οποίοι με βοήθησαν να ξεφύγω από το άγχος και την πίεση, προσφέροντάς μου στιγμές χαλάρωσης και διασκέδασης. Η συντροφιά τους υπήρξε ανεκτίμητη, καθώς μου θύμιζαν την αξία της ισορροπίας μεταξύ προσωπικής και ακαδημαϊκής ζωής.

Περίληψη

Η παρούσα πτυχιακή εργασία εστιάζει στην υπολογιστική νευροεπιστήμη, με στόχο τη μοντελοποίηση και προσομοίωση της νευρωνικής δραστηριότητας μέσω δικτύων νευρώνων. Αναλύονται βιολογικά και υπολογιστικά μοντέλα νευρώνων, όπως το Leaky Integrate-and-Fire (LIF), τα οποία χρησιμοποιούνται για την απεικόνιση της πολύπλοκης δυναμικής των νευρωνικών δικτύων. Μέσω της προσομοίωσης αυτών των δικτύων, επιχειρείται η κατανόηση των μηχανισμών που διέπουν τη λειτουργία του εγκεφάλου και των υπολογιστικών διεργασιών που λαμβάνουν χώρα.

Η εργασία περιλαμβάνει συγκριτική μελέτη διαφόρων προτύπων και φαινομένων, όπως οι χιμαιρικές καταστάσεις, καθώς και την εφαρμογή του μοντέλου LIF για τη διερεύνηση της νευρωνικής συμπεριφοράς σε μεγάλης κλίμακας προσομοιώσεις. Για τη βελτίωση της αποδοτικότητας των προσομοιώσεων, χρησιμοποιείται παράλληλη επεξεργασία, αξιοποιώντας τις δυνατότητες των πολυπύρηνων συστημάτων, με στόχο τη μείωση του χρόνου εκτέλεσης και την εκτεταμένη ανάλυση παραμετρικών συνδυασμών.

Τα αποτελέσματα δείχνουν σημαντική βελτίωση στους χρόνους εκτέλεσης και στην επιτάχυνση, ιδιαίτερα σε χαμηλό αριθμό νημάτων, ενώ παρατηρείται μείωση της απόδοσης καθώς αυξάνονται τα νήματα λόγω συμφόρησης στη μνήμη. Η εργασία καταλήγει ότι η βελτιστοποίηση της χρήσης του εύρους ζώνης της μνήμης μπορεί να συμβάλλει στην αποτελεσματικότερη χρήση των πόρων, προσφέροντας πολύτιμα δεδομένα για την κατανόηση της νευρωνικής δραστηριότητας.

Επιστημονική περιοχή: Υπολογιστική Νευροεπιστήμη

Λέξεις-κλειδιά: Νευροεπιστήμη, Νευρώνες, Προσομοίωση, Παράλληλη Επεξεργασία, Leaky Integrate-and-Fire

Abstract

This thesis focuses on computational neuroscience, aiming to model and simulate neuronal activity through neural networks. Biological and computational neuron models, such as the Leaky Integrate-and-Fire (LIF), are analyzed to represent the complex dynamics of neural networks. Through the simulation of these networks, an attempt is made to understand the mechanisms governing brain function and the computational processes occurring within.

The study includes a comparative analysis of various models and phenomena, such as chimera states, and applies the LIF model to investigate neuronal behavior in large-scale simulations. To improve simulation efficiency, parallel processing is employed, leveraging the capabilities of multi-core systems, with the goal of reducing execution time and enabling an extensive analysis of parameter combinations.

The results demonstrate significant improvements in execution times and speedup, particularly with a low thread count, while a decrease in performance is observed as threads increase due to memory bottlenecks. The study concludes that optimizing memory bandwidth usage can enhance resource efficiency, providing valuable insights into neuronal activity.

Scientific Area: Computational Neuroscience

Keywords: Neuroscience, Neurons, Simulation, Parallel Processing, Leaky Integrate-and-Fire

Πίνακας Περιεχομένων

Κατάλογος Εικόνων

Κατάλογος Πινάκων

Κατάλογος Διαγραμμάτων

Εισαγωγή

Η παρούσα πτυχιακή εργασία επικεντρώνεται στην υπολογιστική νευροεπιστήμη, με κύριο στόχο τη μοντελοποίηση και προσομοίωση της νευρωνικής δραστηριότητας, αξιοποιώντας υπολογιστικά μοντέλα νευρώνων και παράλληλες τεχνικές επεξεργασίας. Η κατανόηση του τρόπου με τον οποίο οι νευρώνες διαχειρίζονται και επεξεργάζονται πληροφορίες αποτελεί αντικείμενο υψηλής επιστημονικής σημασίας, που συγκεντρώνει έντονο ερευνητικό ενδιαφέρον. Στην παγκόσμια βιβλιογραφία, σημαντικές ερευνητικές κατευθύνσεις περιλαμβάνουν τη μελέτη των βιολογικών νευρώνων, τη διασύνδεσή τους σε σύνθετα δίκτυα, και την ανάπτυξη υπολογιστικών μοντέλων που επιτρέπουν την εξερεύνηση και κατανόηση φαινομένων όπως οι χιμαιρικές καταστάσεις.

Η παρούσα εργασία στοχεύει στη διερεύνηση της συμπεριφοράς των νευρωνικών δικτύων μέσω της εφαρμογής του μοντέλου Leaky Integrate-and-Fire (LIF), το οποίο θεωρείται κατάλληλο για μεγάλες κλίμακες προσομοιώσεων. Η μεθοδολογική προσέγγιση περιλαμβάνει την αξιοποίηση αραιών δομών δεδομένων και την εφαρμογή παράλληλης επεξεργασίας με σκοπό την αύξηση της αποδοτικότητας και τη μείωση του χρόνου εκτέλεσης. Ειδικότερα, η χρήση της μορφής Compressed Sparse Row (CSR) και η ενσωμάτωση της βιβλιοθήκης oneAPI MKL διευκολύνουν την αποτελεσματική εκτέλεση των προσομοιώσεων, αξιοποιώντας τις δυνατότητες των πολυπύρηνων συστημάτων και των καρτών γραφικών για να καλύψουν τις αυξημένες υπολογιστικές απαιτήσεις.

Οι επιμέρους στόχοι της εργασίας περιλαμβάνουν μια συνοπτική αναφορά στα κύρια βιολογικά και υπολογιστικά μοντέλα νευρώνων, με έμφαση στο μοντέλο Leaky Integrate-and-Fire (LIF), το οποίο έχει επιλεγεί για τις προσομοιώσεις. Παράλληλα, η εργασία επικεντρώνεται στη βελτιστοποίηση της αποδοτικότητας του κώδικα μέσω παράλληλων τεχνικών επεξεργασίας. Επιπλέον, εξετάζεται ο χρόνος εκτέλεσης και η επίδραση των παραμέτρων προσομοίωσης στην επιτάχυνση, παρέχοντας μια ολοκληρωμένη ανάλυση της απόδοσης σε περιβάλλοντα με αυξημένες υπολογιστικές απαιτήσεις.

Η διάρθρωση της εργασίας ακολουθεί μια λογική ροή που ξεκινά με την εισαγωγή στη νευροεπιστήμη και τους βιολογικούς νευρώνες, παρέχοντας το απαραίτητο θεωρητικό υπόβαθρο. Στη συνέχεια, εξετάζονται τα διάφορα μοντέλα προσομοίωσης νευρώνων και αναλύεται η εφαρμογή του μοντέλου LIF στις προσομοιώσεις. Ακολουθεί η ενότητα που αναλύει τη βελτίωση της αποδοτικότητας των προσομοιώσεων μέσω τεχνικών παράλληλης επεξεργασίας και τη χρήση αραιών δομών δεδομένων, ενώ τέλος, παρουσιάζονται τα αποτελέσματα με ανάλυση των χρόνων εκτέλεσης και της επιτάχυνσης.

Η παρούσα εργασία φιλοδοξεί να συμβάλει στην κατανόηση της υπολογιστικής δυναμικής των νευρωνικών δικτύων και να προωθήσει τις τεχνικές βελτιστοποίησης προσομοιώσεων, διευκολύνοντας την περαιτέρω ανάπτυξη στον τομέα της υπολογιστικής νευροεπιστήμης.

1. Εισαγωγή στη Νευροεπιστήμη και τους Νευρώνες

1.1 Νευροεπιστήμη

Ο απώτερος στόχος της υπολογιστικής νευροεπιστήμης είναι να εξηγήσει πώς τα ηλεκτρικά και χημικά σήματα χρησιμοποιούνται στον εγκέφαλο για να αναπαραστήσουν και να επεξεργαστούν πληροφορίες(Goswami, 2004)(Sejnowski et al., 1988). Αυτός ο στόχος δεν είναι νέος, αλλά πολλά έχουν αλλάξει την τελευταία δεκαετία. Πλέον γνωρίζουμε περισσότερα για τον εγκέφαλο λόγω των προόδων στη νευροεπιστήμη, διαθέτουμε περισσότερη υπολογιστική ισχύ για την πραγματοποίηση ρεαλιστικών προσομοιώσεων νευρικών συστημάτων και έχουμε νέες γνώσεις από τη μελέτη απλουστευμένων μοντέλων μεγάλων δικτύων νευρώνων. Τα μοντέλα του εγκεφάλου χρησιμοποιούνται για να συνδέσουν το μικροσκοπικό επίπεδο, το οποίο είναι προσβάσιμο με μοριακές και κυτταρικές τεχνικές, με το επίπεδο των συστημάτων που είναι προσβάσιμο μέσω της μελέτης της συμπεριφοράς.Η κατανόηση του εγκεφάλου είναι μια πρόκληση που προσελκύει έναν συνεχώς αυξανόμενο αριθμό επιστημόνων από διάφορους κλάδους .

Παρά την πληθώρα ανακαλύψεων που έχουν γίνει τις τελευταίες δεκαετίες σχετικά με τη δομή του εγκεφάλου σε κυτταρικό και μοριακό επίπεδο, δεν κατανοούμε ακόμη πλήρως πώς το νευρικό σύστημα μας επιτρέπει να βλέπουμε και να ακούμε, να μαθαίνουμε δεξιότητες και να θυμόμαστε γεγονότα, να σχεδιάζουμε δράσεις και να λαμβάνουμε αποφάσεις. Απλά συστήματα αντανακλαστικών έχουν χρησιμεύσει ως χρήσιμα μοντέλα που προετοιμάζουν για τη μελέτη της δημιουργίας και τροποποίησης της συμπεριφοράς σε κυτταρικό επίπεδο. Ωστόσο, στα θηλαστικά, η σχέση μεταξύ της αντίληψης και της δραστηριότητας των μεμονωμένων νευρώνων είναι πιο δύσκολη να μελετηθεί, καθώς οι αισθητηριακές ικανότητες που αξιολογούνται με ψυχοφυσικές τεχνικές είναι αποτέλεσμα της δραστηριότητας πολλών νευρώνων από διαφορετικά μέρη του εγκεφάλου. Η εξήγηση των ανώτερων λειτουργιών είναι δύσκολη, εν μέρει επειδή τα νευρικά συστήματα έχουν πολλά επίπεδα οργάνωσης μεταξύ του μοριακού και του συστημικού επιπέδου, το καθένα με τις δικές του σημαντικές λειτουργίες .

Οι νευρώνες είναι οργανωμένοι σε τοπικά κυκλώματα, στήλες, στοιβάδες και τοπογραφικούς χάρτες, για σκοπούς που αρχίζουμε μόλις να κατανοούμε. Ιδιότητες που δεν βρίσκονται σε χαμηλότερα επίπεδα μπορούν να προκύψουν από την οργάνωση και την αλληλεπίδραση αυτών των επιπέδων σε ανώτερο επίπεδο. Για παράδειγμα, η ρυθμική παραγωγή μοτίβων σε ορισμένα νευρικά κυκλώματα είναι ιδιότητα του κυκλώματος και όχι μεμονωμένων νευρώνων που λειτουργούν ως «χρονομέτρες» (δηλαδή ένας νευρώνας που μεμονωμένα ελέγχει το χρονισμό των νευρικών σημάτων). Ανώτερες λειτουργίες του εγκεφάλου, όπως η αντίληψη και η προσοχή, μπορεί να εξαρτώνται από χρονικά συντονισμένες λειτουργικές μονάδες που διασπείρονται σε διάφορους χάρτες και πυρήνες . Οι πηγές αυτών των ιδιοτήτων των δικτύων δεν είναι προσβάσιμες μέσω των μεθόδων που είναι κατάλληλες για τη μελέτη μεμονωμένων νευρώνων.

Αν υποθέσουμε ότι υπάρχουν ιδιότητες που αναπτύσσονται στα δίκτυα νευρώνων καθώς αλληλεπιδρούν, είναι δύσκολο να φανταστούμε πώς θα προοδεύσουμε στην κατανόηση του εγκεφάλου χωρίς την ανάπτυξη πιο άμεσων και αποτελεσματικών τεχνικών μελέτης, χωρίς μια ταυτόχρονη ανάπτυξη άμεσων και αποτελεσματικών τεχνικών για τη διερεύνηση των μηχανισμών κατανεμημένης επεξεργασίας. Οι νέες πειραματικές τεχνικές που αναπτύσσονται περιλαμβάνουν μεθόδους για ταυτόχρονη καταγραφή από πολλαπλές μονάδες, οπτική καταγραφή της στήλης οργάνωσης στον φλοιό με βαφές ευαίσθητες σε τάση και ιόντα, καθώς και μεγάλες μετρήσεις της δομής και δραστηριότητας του εγκεφάλου με τη χρήση τομογραφίας εκπομπής ποζιτρονίων (PET), μαγνητοεγκεφαλογραφίας (MEG), 2-δεοξυγλυκόζης (2-DG) και μαγνητικής τομογραφίας (MRI).

1.1.1 Συνάψεις και Δίκτυα Νευρώνων

Οι συνάψεις μεταξύ των νευρώνων είναι τα σημεία όπου δημιουργούνται ηλεκτρικές και χημικές συνδέσεις, επιτρέποντας την επικοινωνία μεταξύ τους και τη δημιουργία δικτύων νευρώνων. Τα δίκτυα αυτά είναι υπεύθυνα για περίπλοκες λειτουργίες του εγκεφάλου.

1.2 Βιολογικοί Νευρώνες

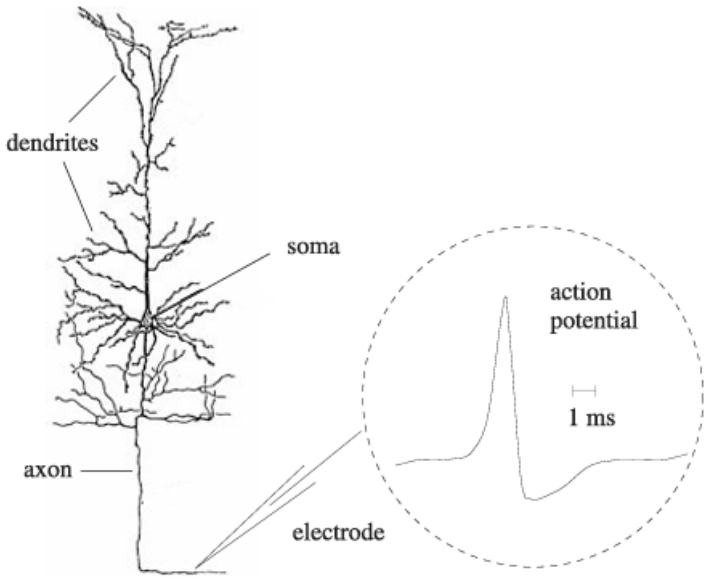
Τα τελευταία εκατό χρόνια, η βιολογική έρευνα έχει συσσωρεύσει τεράστιες ποσότητες λεπτομερών γνώσεων σχετικά με τη δομή και τη λειτουργία του εγκεφάλου (Gerstner, 1998). Οι βασικές μονάδες επεξεργασίας στον εγκέφαλο είναι οι νευρώνες, οι οποίοι συνδέονται μεταξύ τους με έναν περίπλοκο τρόπο. Ένα τμήμα ενός τέτοιου δικτύου νευρώνων στον φλοιό των θηλαστικών απεικονίζεται στο Σχήμα 1.1. Πρόκειται για αναπαραγωγή ενός διάσημου σκίτσου του Ramón y Cajal, ενός από τους πρωτοπόρους της νευροεπιστήμης στις αρχές του 20ου αιώνα. Μπορούμε να διακρίνουμε αρκετούς νευρώνες με τριγωνικά ή κυκλικά κυτταρικά σώματα και μακριές προεκτάσεις που μοιάζουν με σύρματα. Αυτό το σκίτσο δίνει μια εικόνα του δικτύου των νευρώνων στον φλοιό. Μόνο λίγοι από τους νευρώνες που υπάρχουν στο δείγμα έχουν καταστεί ορατοί μέσω της διαδικασίας χρώσης. Στην πραγματικότητα, οι νευρώνες και οι συνδέσεις τους σχηματίζουν ένα πυκνό δίκτυο με περισσότερα από 10^4 κυτταρικά σώματα και αρκετά χιλιόμετρα «συρμάτων» ανά κυβικό χιλιοστόμετρο. Σε άλλες περιοχές του εγκεφάλου, το μοτίβο σύνδεσης των νευρώνων είναι διαφορετικό. Σε όλες τις περιοχές, ωστόσο, νευρώνες διαφόρων μεγεθών και σχημάτων αποτελούν τα βασικά στοιχεία (Gerstner, 1998).

Σχήμα 1.1: Αυτή η αναπαράσταση ενός σκίτσου του Ramón y Cajal απεικονίζει έναν μικρό αριθμό νευρώνων στον φλοιό. Μόνο ένα μέρος των νευρώνων φαίνεται, ενώ στην πραγματικότητα η πυκνότητά τους είναι πολύ μεγαλύτερη. Το κύτταρο b αποτελεί ένα χαρακτηριστικό παράδειγμα πυραμιδικού κυττάρου με τριγωνικό κυτταρικό σώμα. Οι δενδρίτες, που εκτείνονται πλάγια και προς τα πάνω από το κύτταρο, ξεχωρίζουν λόγω της τραχείας τους επιφάνειας. Ο νευράξονας εκτείνεται προς τα κάτω, σχηματίζοντας μερικά παρακλάδια προς τα αριστερά και τα δεξιά (Gerstner, 1998).

Ένας τυπικός νευρώνας έχει τρία μέρη, που ονομάζονται δενδρίτες, σώμα και νευράξονας. Όπως φαίνεται και στο Σχήμα 1.2, τα σήματα από άλλους νευρώνες φτάνουν στη δενδριτική περιοχή και μεταφέρονται στο σώμα και τον νευράξονα. Η μεταβατική ζώνη ανάμεσα στο σώμα και τον νευράξονα είναι ιδιαίτερα σημαντική, καθώς εκεί λαμβάνει χώρα η βασική μη γραμμική επεξεργασία. Εάν η διέγερση που προκαλείται από την είσοδο είναι επαρκής, τότε παράγεται ένα σήμα εξόδου που προωθείται κατά μήκος του νευράξονα και των κλαδιών του προς άλλους νευρώνες. Η λειτουργία αυτή ονομάζεται «πυροδότηση».Η σύνδεση μεταξύ ενός κλάδου του νευράξονα και ενός δενδρίτη (ή του σώματος) ενός άλλου νευρώνα ονομάζεται σύναψη. Συνήθως αναφέρεται ως προσυναπτικός νευρώνας αυτός που στέλνει το σήμα και ως μετασυναπτικός νευρώνας αυτός που το λαμβάνει. Ένας νευρώνας στον φλοιό συνδέεται συχνά με περισσότερους από 10^4 μετασυναπτικούς νευρώνες. Πολλά από τα κλαδιά του νευράξονά του καταλήγουν στη γειτονιά του νευρώνα, ενώ ο νευράξονας μπορεί να εκτείνεται σε πολλά χιλιοστά και να συνδέεται με νευρώνες άλλων περιοχών του εγκεφάλου.

Μέχρι στιγμής έχουμε δηλώσει ότι οι νευρώνες μεταδίδουν σήματα μέσω του νευράξονα σε χιλιάδες άλλους νευρώνες – αλλά πώς μοιάζουν αυτά τα σήματα; Τα δυναμικά δράσης μπορούν να

παρατηρηθούν τοποθετώντας ένα λεπτό ηλεκτρόδιο κοντά στο σώμα ή τον νευράξονα ενός νευρώνα, δείτε το Σχήμα 1.2. Η καταγραφή της τάσης δείχνει μια ακολουθία μικρών παλμών, που ονομάζονται δυναμικά δράσης ή αιχμές. Μια τέτοια αλυσίδα παλμών που εκπέμπεται από έναν νευρώνα συνήθως αποκαλείται «ακολουθία αιχμών» – μια ακολουθία τυπικών γεγονότων που εμφανίζονται σε τακτά ή ακανόνιστα χρονικά διαστήματα. Η διάρκεια ενός δυναμικού ενέργειας είναι συνήθως της τάξης του 1-2 χιλιοστών του δευτερολέπτου. Αν και όλες οι αιχμές ενός συγκεκριμένου νευρώνα μοιάζουν ίδιες, η μορφή του δυναμικού ενέργειας δεν μεταφέρει καμία πληροφορία. Αυτό που έχει σημασία είναι ο αριθμός και η χρονική στιγμή των αιχμών.



Σχήμα 1.2: Ένας μεμονωμένος νευρώνας. Οι δενδρίτες, το σώμα και ο νευράξονας μπορούν να διακριθούν καθαρά. Η ένθετη εικόνα δείχνει ένα παράδειγμα ενός δυναμικού δράσης νευρώνα (σχηματικό). Το σκίτσο του νευρώνα είναι από τον Ramón y Cajal. Το δυναμικό ενέργειας είναι ένας σύντομος παλμός τάσης διάρκειας 1-2 χιλιοστών του δευτερολέπτου.

Το δυναμικό του νευρώνων αυξάνεται όταν αυτοί δεχθούν ερεθίσματα και όταν αυτό φτάσει μια συγκεκριμένη τιμή ο νευρώνας πυροδοτεί και το δυναμικό μειώνεται απότομα. Το σύνολο των χρόνων πυροδότησης ενός νευρώνα i περιγράφεται ως:

$$\mathcal{F}_i = \left\{ t_i^{(1)}, \ldots, t_i^{(n)} \right\}$$

όπου $t_i^{(n)}$ είναι ο πιο πρόσφατος χρόνος πυροδότησης του νευρώνα i.

Σε πειραματικό περιβάλλον, οι χρόνοι πυροδότησης μετρώνται με κάποια ανάλυση Δt. Μια «ακολουθία αιχμών» μπορεί να περιγραφεί ως μια ακολουθία από 1 και 0, που αντιπροσωπεύουν την ύπαρξη ή την απουσία μιας αιχμής στους χρόνους Δt,2Δt,..., αντίστοιχα. Η επιλογή μεταξύ των 1 και 0 είναι αυθαίρετη. Μπορούμε, αντί για την τιμή 1, να χρησιμοποιήσουμε την τιμή 1/Δt για να δηλώσουμε την εμφάνιση μιας αιχμής. Με αυτόν τον ορισμό, η «ακολουθία αιχμών» ενός νευρώνα i αντιστοιχεί σε μια ακολουθία αριθμών $S_i(\Delta t), S_i(2\Delta t)$ ... με:

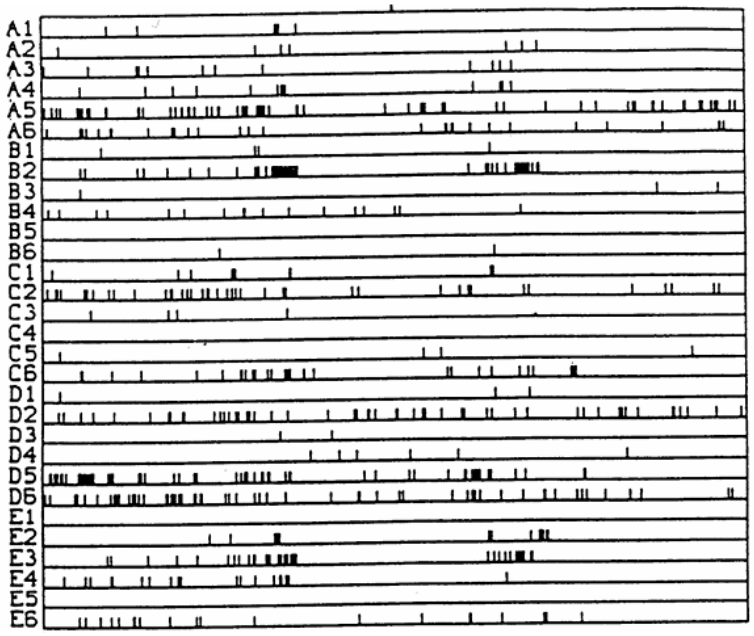
$$S_i(n\Delta t) = \begin{cases} 1/\Delta t & \text{αν} \quad n\Delta t \leq t_i^{(f)} < (n+1)\Delta t \\ 0 & \text{άλλως.} \end{cases}$$

Μπορούμε τυπικά να πάρουμε το όριο Δt→0 και να γράψουμε την ακολουθία αιχμών ως μια ακολουθία συναρτήσεων δέλτα (δ-functions).

$$S_i(t) = \sum_{t_i^{(f)} \in \mathcal{F}_i} \delta(t - t_i^{(f)})$$

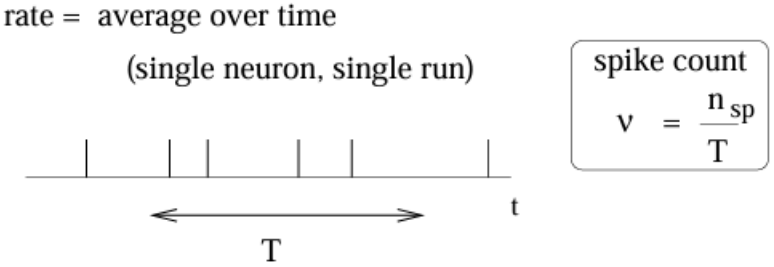
όπου $\delta(\cdot)$ είναι η συνάρτηση δέλτα του Dirac με $\delta(s) = 0$ για $s \neq 0$ και $\int_{-\infty}^{\infty} \delta(s)ds = 1$.

Εστιάσαμε μέχρι τώρα στην ακολουθία αιχμών ενός μεμονωμένου νευρώνα. Εφόσον υπάρχουν τόσοι πολλοί νευρώνες στον εγκέφαλο, χιλιάδες ακολουθίες αιχμών εκπέμπονται συνεχώς από διαφορετικούς νευρώνες, δείτε το Σχήμα 1.3. Ποια είναι η πληροφορία που περιέχεται σε ένα τέτοιο χωροχρονικό μοτίβο παλμών; Ποιος είναι ο κώδικας που χρησιμοποιούν οι νευρώνες για να μεταδώσουν αυτή την πληροφορία; Πώς μπορούν άλλοι νευρώνες να αποκωδικοποιήσουν το σήμα; Ως εξωτερικοί παρατηρητές, μπορούμε να διαβάσουμε αυτόν τον κώδικα και να κατανοήσουμε το μήνυμα του μοτίβου νευρωνικής δραστηριότητας;



Σχήμα 1.3: Χωροχρονικό μοτίβο παλμών. Οι αιχμές 30 νευρώνων (A1-E6, απεικονισμένοι κατά μήκος των κάθετων αξόνων) παρουσιάζονται ως συνάρτηση του χρόνου (οριζόντιος άξονας, συνολικός χρόνος 4.000 ms). Οι χρόνοι πυροδότησης σημειώνονται με μικρές κάθετες γραμμές. Από (Krüger and Aiple, 1988).

Προς το παρόν, δεν υπάρχει οριστική απάντηση σε αυτά τα ερωτήματα. Παραδοσιακά, έχει θεωρηθεί ότι η περισσότερη, αν όχι όλη, η σχετική πληροφορία περιέχεται στον μέσο ρυθμό πυροδότησης του νευρώνα. Ο ρυθμός πυροδότησης συνήθως ορίζεται από έναν χρονικό μέσο όρο, δείτε το Σχήμα 1.4.



Σχήμα 1.4: Ορισμός του μέσου ρυθμού πυροδότησης μέσω ενός χρονικού μέσου όρου.

Ο πειραματιστής ορίζει ένα χρονικό παράθυρο, ας πούμε T=100ms ή T=500ms, και μετρά τον αριθμό των αιχμών $n_{sp}(T)$ που συμβαίνουν σε αυτό το χρονικό παράθυρο. Η διαίρεση με το μήκος του χρονικού παραθύρου δίνει τον μέσο ρυθμό πυροδότησης:

$$\nu = \frac{n_{sp}(T)}{T}$$

συνήθως εκφρασμένο σε μονάδες sec^{-1} ή Hz.

Η έννοια του μέσου ρυθμού πυροδότησης έχει εφαρμοστεί επιτυχώς τα τελευταία 80 χρόνια. Προέρχεται από το πρωτοποριακό έργο του Adrian (Adrian, 1926, 1928), ο οποίος έδειξε ότι ο ρυθμός πυροδότησης των νευρώνων υποδοχέων διάτασης στους μύες σχετίζεται με την πίεση που ασκείται στον μυ. Τις επόμενες δεκαετίες, η μέτρηση των ρυθμών πυροδότησης έγινε ένα τυπικό εργαλείο για την περιγραφή των ιδιοτήτων όλων των ειδών αισθητήριων ή φλοιικών νευρώνων (Mountcastle, 1957; Hubel and Wiesel, 1959), εν μέρει λόγω της σχετικής ευκολίας στη μέτρηση αυτών των ρυθμών. Ωστόσο, είναι σαφές ότι μια προσέγγιση που βασίζεται σε έναν χρονικό μέσο όρο παραβλέπει όλες τις πληροφορίες που μπορεί να περιέχονται από τον ακριβή χρονισμό των αιχμών. Συνεπώς, δεν αποτελεί έκπληξη ότι η έννοια του ρυθμού πυροδότησης έχει επανειλημμένα δεχθεί κριτική και αποτελεί αντικείμενο συνεχιζόμενης συζήτησης (Abeles, 1994; Bialek et al., 1991; Hopfield, 1995; Shadlen and Newsome, 1994; Softky, 1995; Rieke et al., 1996). Τα τελευταία χρόνια, έχουν συσσωρευτεί περισσότερα πειραματικά στοιχεία που υποδεικνύουν ότι ο ρυθμός πυροδότησης βασισμένος σε χρονικό μέσο όρο μπορεί να είναι πολύ απλός για να περιγράψει τη δραστηριότητα του εγκεφάλου. Ένα από τα βασικά επιχειρήματα είναι ότι οι χρόνοι αντίδρασης σε πειράματα συμπεριφοράς είναι συχνά πολύ μικροί για να επιτρέψουν τον χρονικό μέσο όρο (Thorpe et al., 1996). Επιπλέον, σε πειράματα με έναν οπτικό νευρώνα σε μύγα, ήταν δυνατό να «διαβαστεί ο νευρωνικός κώδικας» και να ανακατασκευαστεί το χρονικά εξαρτώμενο ερέθισμα με βάση τους χρόνους πυροδότησης των νευρώνων (Bialek et al., 1991). Υπάρχουν στοιχεία για ακριβείς χρονικές συσχετίσεις μεταξύ παλμών διαφορετικών νευρώνων (Abeles, 1994; Lestienne, 1996) και χρονικά εξαρτώμενος συγχρονισμός της δραστηριότητας σε πληθυσμούς νευρώνων (Eckhorn et al., 1988; Gray and Singer, 1989; Gray et al., 1989; Engel et al., 1991; Singer, 1994). Τα περισσότερα από αυτά τα δεδομένα είναι ασύμβατα με την απλοϊκή έννοια της κωδικοποίησης μέσω μέσων ρυθμών πυροδότησης, όπου ο ακριβής χρόνος των αιχμών δεν πρέπει να παίζει κανένα ρόλο. Μια γρήγορη ματιά στη πειραματική βιβλιογραφία αποκαλύπτει ότι δεν υπάρχει μία μοναδική και σαφώς ορισμένη έννοια του «μέσου ρυθμού πυροδότησης». Στην πραγματικότητα, υπάρχουν τουλάχιστον τρεις διαφορετικές έννοιες του ρυθμού, οι οποίες συχνά συγχέονται και χρησιμοποιούνται ταυτόχρονα. Οι τρεις ορισμοί αναφέρονται σε τρεις διαφορετικές διαδικασίες μέσου όρου: είτε έναν μέσο όρο στο χρόνο, είτε έναν μέσο όρο σε αρκετές επαναλήψεις του πειράματος, είτε έναν μέσο όρο σε έναν πληθυσμό νευρώνων.

Ο πρώτος και πιο συχνά χρησιμοποιούμενος ορισμός του ρυθμού πυροδότησης αναφέρεται σε έναν χρονικό μέσο όρο. Όπως αναφέρθηκε στην προηγούμενη ενότητα, αυτό ουσιαστικά είναι ο αριθμός των αιχμών σε ένα διάστημα T διαιρεμένος με το T, δείτε το Σχήμα 1.4. Το μήκος του χρονικού παραθύρου καθορίζεται από τον πειραματιστή και εξαρτάται από τον τύπο του νευρώνα που καταγράφηκε και το ερέθισμα. Στην πράξη, για να έχουμε αξιόπιστους μέσους όρους, αρκετές αιχμές θα πρέπει να συμβούν μέσα στο χρονικό παράθυρο. Τυπικές τιμές για το T είναι 100 ms ή 500 ms, αλλά η διάρκεια μπορεί επίσης να είναι μεγαλύτερη ή μικρότερη.

Αυτός ο ορισμός του ρυθμού έχει εφαρμοστεί με επιτυχία σε πολλές πειραματικές διαδικασίες, ιδιαίτερα σε πειράματα σε αισθητήρια ή κινητικά συστήματα. Ένα κλασικό παράδειγμα είναι ο υποδοχέας διάτασης σε έναν μυ (Adrian, 1926). Ο αριθμός των αιχμών που εκπέμπεται από τον υποδοχέα αυξάνεται με τη δύναμη που εφαρμόζεται στον μυ.

Αυτά τα κλασικά αποτελέσματα δείχνουν ότι ο πειραματιστής, ως εξωτερικός παρατηρητής, μπορεί να αξιολογήσει και να ταξινομήσει τη νευρωνική πυροδότηση μετρώντας τον αριθμό των αιχμών – αλλά είναι αυτός ο κώδικας που χρησιμοποιούν οι νευρώνες στον εγκέφαλο; Με άλλα λόγια, ένας νευρώνας που λαμβάνει σήματα από έναν αισθητηριακό νευρώνα αντιδρά απλώς και μόνο κοιτώντας και αντιδρώντας στον αριθμό των αιχμών που λαμβάνει σε ένα χρονικό παράθυρο.

Από τα πειράματα συμπεριφοράς είναι γνωστό ότι οι χρόνοι αντίδρασης είναι συχνά αρκετά σύντομοι. Μια μύγα μπορεί να αντιδράσει σε ένα νέο ερέθισμα και να αλλάξει κατεύθυνση πτήσης

μέσα σε 30-40 ms, δείτε (Rieke et al., 1996). Αυτό δεν είναι αρκετό για την καταμέτρηση αιχμών και τον υπολογισμό μέσων όρων. Συνεπώς, οι μύγες πρέπει να αντιδρούν σε μεμονωμένες αιχμές. Οι άνθρωποι μπορούν να αναγνωρίσουν οπτικά ερεθίσματα σε μόλις λίγες εκατοντάδες χιλιοστά του δευτερολέπτου (Thorpe et al., 1996), παρόλο που η αναγνώριση θεωρείται ότι περιλαμβάνει διάφορα επίπεδα επεξεργασίας. Και πάλι, αυτό δεν αφήνει αρκετό χρόνο για να υπολογιστούν οι χρονικοί μέσοι όροι σε κάθε επίπεδο.

Ο χρονικός μέσος όρος μπορεί να λειτουργήσει καλά όταν το ερέθισμα είναι σταθερό ή κινείται αργά και δεν απαιτεί ταχεία αντίδραση από τον οργανισμό – και αυτός είναι ο λόγος που χρησιμοποιείται συχνά σε ελεγχόμενα πειραματικά πρωτόκολλα. Ωστόσο, στον πραγματικό κόσμο, τα ερεθίσματα σπάνια παραμένουν σταθερά, αλλά αλλάζουν γρήγορα. Για παράδειγμα, ακόμη και όταν κοιτάζουμε μια στατική εικόνα, κάνουμε σακκαδικές κινήσεις, ραγδαίες αλλαγές της κατεύθυνσης του βλέμματος. Οι φωτοϋποδοχείς στον αμφιβληστροειδή λαμβάνουν νέα εικόνα κάθε μερικές εκατοντάδες χιλιοστά του δευτερολέπτου.

Παρά τις αδυναμίες της, η έννοια του ρυθμού πυροδότησης χρησιμοποιείται ευρέως τόσο σε πειράματα όσο και σε μοντέλα δικτύων νευρώνων. Υποτίθεται ότι ένας νευρώνας μετατρέπει το ερέθισμα που λαμβάνει σε μια συνεχή έξοδο, η οποία εκφράζεται ως ρυθμός πυροδότησης. Αυτός ο ρυθμός αυξάνεται ανάλογα με την ένταση του ερεθίσματος. Με αυτόν τον τρόπο, οι αιχμές θεωρούνται ως μέσο για τη μετάδοση της συνεχούς αναλογικής πληροφορίας που προκύπτει από την απόκριση του νευρώνα.

1.3 Υπολογιστική Προσομοίωση Νευρώνων και Μελέτη Φαινομένων

Ενώ η έρευνα για τον εγκέφαλο σε μοριακό επίπεδο αποτελεί πρόκληση για τους χημικούς και βιολόγους, το επίπεδο επεξεργασίας πληροφοριών και οι ανταλλαγές μεταξύ των νευρώνων είναι ιδιαίτερα ενδιαφέροντα στοιχεία για τους υπολογιστικούς επιστήμονες. Ένα στοιχείο της πολυπλοκότητας που σχετίζεται με τη μετάδοση πληροφοριών αφορά το μέγεθος του εγκεφάλου. Ο συνολικός αριθμός των νευρώνων στον εγκέφαλο ενός ενήλικα ανθρώπου εκτιμάται περίπου σε 10^{10} και κάθε νευρώνας συνδέεται περίπου με 7000 άλλους νευρώνες. Η πληροφορία μεταδίδεται μέσα σε αυτό το τεράστιο δίκτυο με τη μορφή ηλεκτρικών και χημικών παλμών που ανταλλάσσονται μεταξύ των νευρώνων, οι οποίοι αποτελούν τους κόμβους του δικτύου.

Η δραστηριότητα ενός μεμονωμένου νευρώνα μοντελοποιείται μαθηματικά, και κατ’ επέκταση και υπολογιστικά, ως ένας μη γραμμικός ταλαντωτής που παράγει αιχμές (spiking activity). Τα προτεινόμενα μοντέλα αντλούν δεδομένα από πειράματα in vitro καθώς και από πειράματα in vivo της ηλεκτροεγκεφαλογραφίας (EEG)(Mormann, 2000). Αυτά τα μοντέλα επεκτείνονται για να λαμβάνουν υπόψη τις αλληλεπιδράσεις μεταξύ συνδεδεμένων νευρώνων, δημιουργώντας δίκτυα νευρώνων. Οι νευρώνες σε αυτά τα δίκτυα επικοινωνούν μέσω αιχμών, οι οποίες μεταφέρουν πληροφορίες και ρυθμίζουν τη συνολική δραστηριότητα του δικτύου. Τα κοινώς χρησιμοποιούμενα μοντέλα ταλαντωτών νευρώνων που εκπέμπουν σήματα περιλαμβάνουν τα Hodgkin-Huxley (Hodgkin, 1952), Hindmarsh-Rose(Hindmarsh, 1982)(Hindmarsh, 1984), FitzHugh-Nagumo(FitzHugh, 1961)(Nagumo, 1962), Lattice Limit Cycle(Hizanidis, 2015), τον ταλαντωτή Van der Pol (Omelchenko, 2015)(Omelchenko, 2016)και το Leaky Integrate-and-Fire (LIF)(Abrams, 2004).

Ένα από τα πιο ενδιαφέροντα και ανεξήγητα φαινόμενα που παρατηρούνται κατά την αλληλεπίδραση νευρώνων σε μεγάλα δίκτυα είναι η λεγόμενη "χιμαιρική κατάσταση". Σε συστήματα που αποτελούνται από πολλούς νευρώνες που εκπέμπουν σήματα, συχνά παρατηρούνται τοπικά σύνολα νευρώνων που παράγουν μη συνεκτικά σήματα, συνυπάρχοντας με άλλα σύνολα που παράγουν συνεκτικά σήματα. Το πιο ιδιόμορφο χαρακτηριστικό των χιμαιρικών καταστάσεων είναι η παρουσία τους σε συστήματα που αποτελούνται από πανομοιότυπα στοιχεία, τα οποία είναι συνδεδεμένα μεταξύ τους με πανομοιότυπο τρόπο. Σε τέτοια συμμετρικά συστήματα, δεν υπάρχει προφανής λόγος για "σπάσιμο της συμμετρίας" και σχηματισμό συνεκτικών και μη συνεκτικών συνόλων που είναι χρονικά σταθερά. Αυτός είναι ο λόγος που αυτό το φαινόμενο έχει πρόσφατα προκαλέσει έντονο ερευνητικό ενδιαφέρον.

Ένας άλλος λόγος για τη μελέτη των χιμερικών καταστάσεων είναι ότι έχουν συσχετιστεί με βιολογικά φαινόμενα, όπως ύπνος με μόνο ένα ημισφαίριο. Αυτός ο τύπος ύπνου έχει παρατηρηθεί σε πτηνά και θηλαστικά, όπου το ένα μισό του εγκεφάλου τους βρίσκεται σε βαθύ ύπνο και το μάτι

που αντιστοιχεί σε αυτό το μισό είναι κλειστό, ενώ το άλλο μάτι παραμένει ανοιχτό. Με άλλα λόγια, ένα μέρος του εγκεφάλου τους βρίσκεται σε διαφορετική κατάσταση από το υπόλοιπο. Αυτός ο ύπνος θεωρείται ως ένας τύπος χιμαιρικής κατάστασης και επιτρέπει στον εγκέφαλο των θηλαστικών να ξεκουράζεται σε περιοχές υψηλής θήρευσης και στον εγκέφαλο των πτηνών να ξεκουράζεται κατά τη διάρκεια μακρών μεταναστευτικών πτήσεων. Οι χιμαιρικές καταστάσεις έχουν επίσης συσχετιστεί με νευρολογικές διαταραχές όπως η επιληψία ή η κοιλιακή μαρμαρυγή. Παρά το γεγονός ότι αρχικά αυτές οι απρόσμενες καταστάσεις συσχετίστηκαν με τη δυναμική του εγκεφάλου, σε πιο πρόσφατες μελέτες έχουν επίσης παρατηρηθεί πειραματικά σε χημικά συστήματα, σε συνδεδεμένα λέιζερ και σε δίκτυα συνδεδεμένων μηχανικών ταλαντωτών.

Οι χιμερικές καταστάσεις παρατηρήθηκαν για πρώτη φορά από τους Kuramoto και Battogtokh το 2002, ενώ ο όρος προτάθηκε δύο χρόνια αργότερα από τους Abrams και Strogatz. Έλαβαν την ονομασία τους από το τέρας Χίμαιρα της ελληνικής μυθολογίας, ένα υβριδικό ζώο με κεφάλι λιονταριού, σώμα κατσίκας και ουρά φιδιού. Το όνομα θεωρήθηκε κατάλληλο για αυτή την υβριδική κατάσταση. Πολλές αριθμητικές μελέτες έχουν αποκαλύψει τη δυνατότητα σχηματισμού χιμερικών καταστάσεων σε πολλά από τα προαναφερθέντα μοντέλα ταλαντωτών. Παρά τη μελέτη της ανάπτυξης και της μορφής των χιμερικών καταστάσεων, ο μηχανισμός πίσω από το σπάσιμο της χωρικής συμμετρίας που οδηγεί στη δημιουργία των χιμερικών καταστάσεων δεν έχει ακόμη αποσαφηνιστεί πλήρως.

Το μέγεθος του συστήματος (ο αριθμός των ταλαντωτών) παίζει κρίσιμο ρόλο στην εμφάνιση των χιμερικών καταστάσεων. Συνήθως, απαιτείται μεγάλος αριθμός συνδεδεμένων στοιχείων για να ανιχνευθούν. Πρόσφατες προσομοιώσεις έχουν δείξει ότι οι χιμερικές καταστάσεις είναι δυνατές σε "μη τοπική" συνδεσιμότητα, όπου κάθε ταλαντωτής συνδέεται με μερικούς γείτονες. Για καθαρά τοπική συνδεσιμότητα, η συνοχή είναι δύσκολο να επιτευχθεί, ενώ για συνολική συνδεσιμότητα (όλοι προς όλους) παρατηρείται πλήρης συνοχή. Οι χιμερικές καταστάσεις εμφανίζονται μόνο στο ενδιάμεσο καθεστώς.

2. Μοντέλα Εξομοίωσης Νευρώνων

2.1 Εισαγωγή στα Μοντέλα Προσομοίωσης Νευρώνων

Κατά την τελευταία δεκαετία, η έρευνα στα δίκτυα τεχνητών νευρώνων έχει μετατοπιστεί προς τα δίκτυα νευρώνων με χρονικό υπολογισμό αιχμών (spiking neural networks). Παρακινούμενες από βιολογικές ανακαλύψεις, πολλές μελέτες εστιάζουν στα pulse-coupled δίκτυα νευρώνων με αιχμές ως κύριο συστατικό για την επεξεργασία πληροφοριών από τον εγκέφαλο. Στη μελέτη της δυναμικής των δικτύων υπάρχουν δύο κρίσιμα ζητήματα: (1) Ποιο μοντέλο περιγράφει τη δυναμική της πυροδότησης για κάθε νευρώνα και (2) Πώς συνδέονται οι νευρώνες. Λανθασμένη επιλογή του μοντέλου ή της συνδεσιμότητας μπορεί να οδηγήσει σε αποτελέσματα που δεν σχετίζονται με την επεξεργασία πληροφοριών από τον εγκέφαλο.Αυτή η προσέγγιση μας βοηθά να συγκρίνουμε διάφορα μοντέλα των νευρώνων με χρονικές αιχμές, όπως αναφέρεται στο άρθρο του (Izhikevich, 2004) . Τα διάφορα πρότυπα πυροδοτήσεων νευρώνων που έχουν αναγνωριστεί συνοψίζονται στην παρακάτω λίστα:

A. Tonic Spiking: Οι περισσότεροι νευρώνες ενεργοποιούνται μόνο όταν τους δοθεί κάποιο ερέθισμα, και εκπέμπουν μια σειρά από δυναμικά δράσης όταν ερεθιστούν συνεχώς. Αυτός ο τύπος συμπεριφοράς είναι γνωστός ως "συνεχές (tonic) spiking", και παρατηρείται σε διάφορους τύπους νευρώνων του εγκεφαλικού φλοιού.

B. Phasic Spiking: Μερικοί νευρώνες πυροδοτούν ένα μόνο δυναμικό ενέργειας στην αρχή του ερεθίσματος και παραμένουν ανενεργοί στη συνέχεια. Αυτή η συμπεριφορά είναι χρήσιμη για την ανίχνευση της αρχής ενός ερεθίσματος.

C. Tonic Bursting: Ορισμένοι νευρώνες, όπως αυτοί που παρατηρούνται σε γάτες, πυροδοτούν μια σειρά από εκρήξεις με υψηλή συχνότητα κατά τη διάρκεια της διέγερσης. Αυτή η συμπεριφορά συνδέεται με την ανίχνευση συνεχών εισροών και τη μείωση του θορύβου στις συνάψεις.

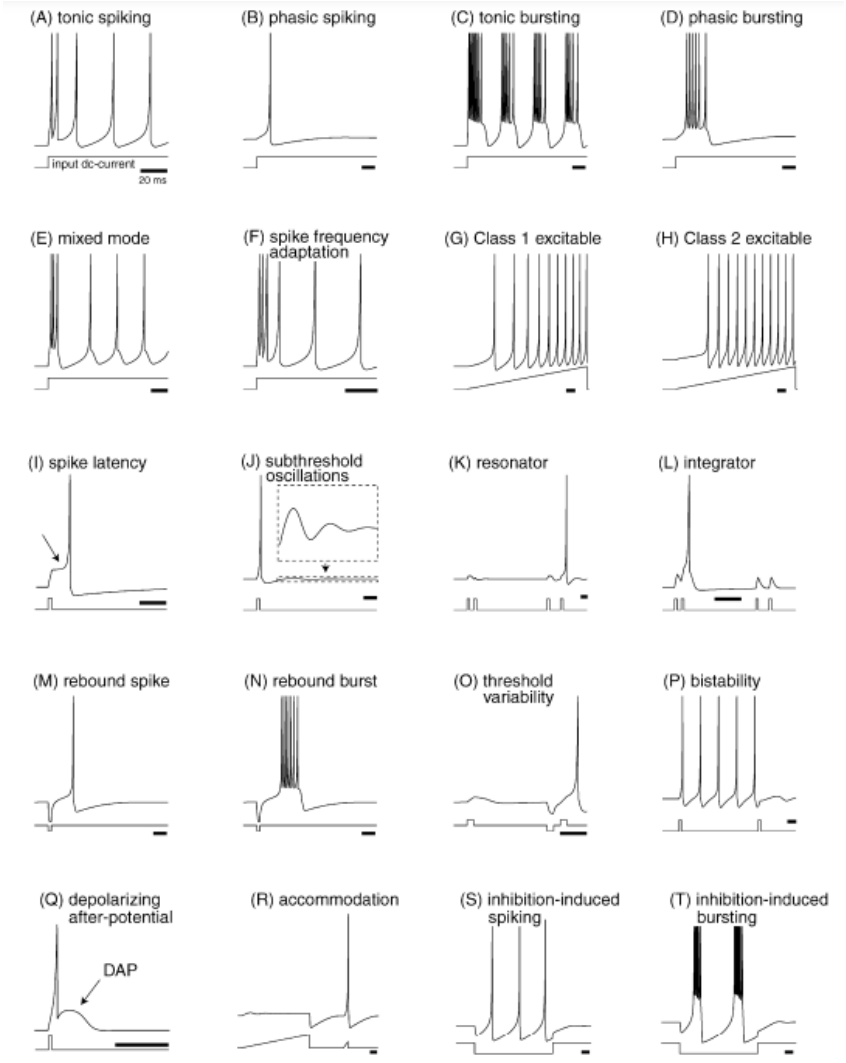
D. Phasic Bursting: Παρόμοιο με τα phasic spiking, αλλά σε αυτή την περίπτωση οι νευρώνες παράγουν διαδοχικές εκρήξεις (bursts) στην αρχή του ερεθίσματος και παραμένουν ανενεργοί αργότερα.

E. Mixed Model (Bursting Then Spiking): Οι νευρώνες αυτού του τύπου, που βρίσκονται στο θηλαστικό νεοφλοιό, εμφανίζουν μικτές μορφές ενεργοποίησης, με πρώτα μια έκρηξη και στη συνέχεια συνεχόμενη εκπομπή σήματος (Izhikevich, 2004).

F. Προσαρμογή Συχνότητας Αιχμής: Ο πιο κοινός τύπος διεγερτικών νευρώνων στον θηλαστικό νεοφλοιό, όπως τακτικά πυροδοτούμενα κύτταρα (regular spiking cells - RS), πυροδοτούν αιχμές με μειωμένη συχνότητα, όπως φαίνεται στο Σχημα 2.1(f). Η συχνότητα είναι σχετικά υψηλή στην αρχή του ερεθίσματος και μετά μειώνεται. Οι νευρώνες ανασταλτικών αιχμών χαμηλού κατωφλίου (LTS) έχουν επίσης αυτή την ιδιότητα. Η συχνότητα των διακοπών μεταξύ των αιχμών αυτών των κυττάρων μπορεί να κωδικοποιήσει το χρόνο που έχει περάσει από την έναρξη του ερεθίσματος.

G. Επαγωγή Κλάσης 1 (Class 1 Excitability): Η συχνότητα της τακτικής πυροδότησης των διεγερτικών νευρώνων του νεοφλοιού (RS) εξαρτάται από την ένταση του ερεθίσματος και μπορεί να εκτείνεται σε εύρος από 2 Hz έως 200 Hz ή ακόμη και περισσότερο. Η ικανότητα να πυροδοτούν χαμηλής συχνότητας αιχμές όταν η είσοδος είναι αδύναμη (αλλά πάνω από το κατώφλι) ονομάζεται Επαγωγή Κλάσης 1. Οι νευρώνες Κλάσης 1 μπορούν να κωδικοποιήσουν την ένταση του εισερχόμενου σήματος μέσω της συχνότητας πυροδότησης τους, όπως φαίνεται στο Σχημα 2.1(g).

H. Επαγωγή Κλάσης 2 (Class 2 Excitability): Μερικοί νευρώνες δεν μπορούν να παράγουν χαμηλής συχνότητας ακολουθίες αιχμών. Αυτοί οι νευρώνες είναι είτε ανενεργοί είτε πυροδοτούν μια ακολουθία αιχμών με σχετικά υψηλή συχνότητα, π.χ., 40 Hz, όπως φαίνεται στο Σχημα 2.1(h). Αυτοί οι νευρώνες ονομάζονται Νευρώνες Κλάσης 2. Η συχνότητα πυροδότησής τους δεν είναι καλός δείκτης της έντασης του ερεθίσματος (Izhikevich, 2004).



Σχήμα 2.1: Οι προσομοιώσεις που παρουσιάζονται είναι του ίδιου μοντέλου με διαφορετικές παραμέτρους, με κάθε οριζόντια μπάρα να αντιστοιχεί σε ένα χρονικό διάστημα 20 ms (Izhikevich, 2004).

I. Καθυστέρηση Ακίδας (Spike Latency): Οι νευρώνες μπορεί να παράγουν ακίδες με καθυστέρηση, η οποία εξαρτάται από την ένταση του εισερχόμενου σήματος. Για παράδειγμα, η καθυστέρηση ακίδας μπορεί να είναι μεγάλη σε νευρώνες φλοιού που ανταποκρίνονται σε σήματα αδύναμης έντασης.

J. Υποκατώφλιες Ταλαντώσεις (Subthreshold Oscillations): Οι νευρώνες μπορεί να εμφανίζουν ταλαντώσεις που δεν παράγουν ακίδες αλλά παίζουν σημαντικό ρόλο στη ρύθμιση του σήματος και της συχνότητας πυροδότησης.

K. Προτίμηση Συχνότητας και Συντονισμός (Frequency Preference and Resonance): Οι νευρώνες με ταλαντώσεις μπορεί να συντονίζονται σε συγκεκριμένες συχνότητες εισερχόμενου σήματος. Αυτή η συμπεριφορά μπορεί να εξηγήσει πώς οι νευρώνες μεταφέρουν πληροφορίες με πολλαπλασιασμό συχνοτήτων.

L. Συνδυασμός και Ενσωμάτωση (Integration and Coincidence Detection): Ορισμένοι νευρώνες λειτουργούν ως ενσωματωτές, όπου η συχνότητα της πυροδότησης αυξάνεται με την ένταση του σήματος, ενώ άλλοι λειτουργούν ως ανιχνευτές σύμπτωσης για τον εντοπισμό ταυτόχρονων σημάτων.

M. Ακίδα Επαναφοράς (Rebound Spike): Όταν ένας νευρώνας απελευθερώνεται από ανασταλτικό σήμα, μπορεί να παράγει μια ακίδα, κάτι που είναι συνήθως γνωστό ως φαινόμενο ανασταλτικής επανα

N. Εκρήξεις Επαναφοράς (Rebound Burst): Ορισμένοι νευρώνες, όπως οι θάλαμο-φλοιώδεις, μπορεί να παράγουν εκρήξεις επαναφοράς μετά από ανασταλτικά σήματα. Οι εκρήξεις αυτές πιστεύεται ότι συμβάλλουν στις διαδικασίες ύπνου.

O. Μεταβλητότητα Κατωφλίου (Threshold Variability): Μια κοινή παρανόηση στη νευρωνική έρευνα είναι ότι οι νευρώνες έχουν σταθερό κατώφλιο διέγερσης. Στην πραγματικότητα, το κατώφλι τους μπορεί να είναι μεταβλητό, ανάλογα με την προηγούμενη δραστηριότητά τους.

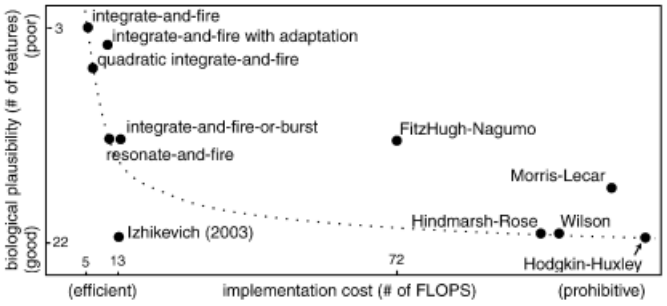
P. Διπλή Σταθερότητα των Καταστάσεων Ανάπαυσης και Πυροδότησης: Ορισμένοι νευρώνες μπορούν να εμφανίζουν δύο σταθερές καταστάσεις λειτουργίας: ανάπαυση και τονική πυροδότηση (ή ακόμα και εκρήξεις). Ένα διεγερτικό ή ανασταλτικό παλμό μπορεί να εναλλάσσει μεταξύ αυτών των καταστάσεων, δημιουργώντας έτσι μια ενδιαφέρουσα δυνατότητα για διπλή σταθερότητα και βραχυπρόθεσμη μνήμη. Για να αλλάξει από την τονική πυροδότηση στην κατάσταση ανάπαυσης, η είσοδος πρέπει να φτάσει σε κατάλληλη φάση ταλάντωσης, τονίζοντας έτσι τη σημασία του χρονισμού της αιχμής σε τέτοιου είδους επεξεργασία πληροφοριών.

Q. Μεταπολαρικά Μεταδυναμικά: Μετά από την πυροδότηση μιας αιχμής, το δυναμικό μεμβράνης ενός νευρώνα μπορεί να εμφανίσει παρατεταμένη υπερπόλωση (AHP) ή παρατεταμένο αποπολωτικό μεταδυναμικό (DAP). Τα DAPs μπορούν να προκύψουν λόγω της επίδρασης των δενδριτικών εισροών, της ενεργοποίησης υψηλού ορίου ρεύματος κατά τη διάρκεια της αιχμής ή λόγω αλληλεπίδρασης μεταξύ των υποουδικών ρευμάτων που ρυθμίζονται από τάση. Σε κάθε περίπτωση, ένας τέτοιος νευρώνας έχει μειωμένη περίοδο αντοχής και γίνεται υπερευαίσθητος.

R. Προσαρμογή: Οι νευρώνες είναι εξαιρετικά ευαίσθητοι σε σύντομες συμπτωματικές εισροές, αλλά μπορεί να μην πυροδοτούν ως απόκριση σε μια ισχυρή αλλά αργά αυξανόμενη είσοδο. Όταν το ρεύμα αυξάνεται αργά, οι εσωτερικές εισροές έχουν αρκετό χρόνο να απενεργοποιηθούν και οι εξωτερικές ροές να ενεργοποιηθούν, με αποτέλεσμα ο νευρώνας να προσαρμόζεται και να μην μπορεί να παράγει αιχμή.

S. Αναστολή-Επαγόμενη Πυροδότηση: Ένα παράξενο χαρακτηριστικό πολλών θαλαμοφλοιωδών νευρώνων είναι ότι παραμένουν ήσυχοι όταν δεν υπάρχει είσοδος, αλλά πυροδοτούν όταν υπερπολώνονται από ένα ανασταλτικό ερέθισμα ή ένα ενέσιμο ρεύμα. Αυτό συμβαίνει επειδή το εισερχόμενο ρεύμα ενεργοποιεί το h-ρεύμα και απενεργοποιεί το ρεύμα ασβεστίου T, οδηγώντας σε τονική πυροδότηση.

T. Αναστολή-Επαγόμενη Έκρηξη: Αντί για απλή πυροδότηση, ένας θαλαμοφλοιώδης νευρώνας μπορεί να παράγει τονικές εκρήξεις αιχμών σε απόκριση σε παρατεταμένη υπερπόλωση. Πιστεύεται ότι οι εκρήξεις αυτές λαμβάνουν χώρα κατά τη διάρκεια ταλαντώσεων στο σύστημα θαλαμο-φλοιού και παίζουν σημαντικό ρόλο στους ρυθμούς ύπνου (Izhikevich, 2004).



Models	biophysically meaningful	tonic spiking	phasic spiking	tonic bursting	phasic bursting	mixed mode	spike frequency adaptation	class 1 excitable	class 2 excitable	spike latency	subthreshold oscillations	resonator	integrator	rebound spike	rebound burst	threshold variability	bistability	DAP	accommodation	inhibition-induced spiking	inhibition-induced bursting	chaos	# of FLOPS
integrate-and-fire	-	+	-	-	-	-	-	+	-	-	-	-	+	-	-	-	-	-	-	-	-	-	5
integrate-and-fire with adapt.	-	+	-	-	-	-	+	+	-	-	-	-	+	-	-	-	-	+	-	-	-	-	10
integrate-and-fire-or-burst	-	+	+		+	-	+	+	-	-	-	-	+	+	+	-	+	+	-	-	-		13
resonate-and-fire	-	+	+	-	-	-	-	+	+	-	+	+	+	+	-	-	+	+	+	-	-	+	10
quadratic integrate-and-fire	-	+	-	-	-	-	-	+	-	+	-	-	+	-	-	+	+	-	-	-	-	-	7
Izhikevich (2003)	-	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	13
FitzHugh-Nagumo	-	+	+	-		-	-	+	-	+	+	+	-	+	-	+	+	-	+	+	-	-	72
Hindmarsh-Rose	-	+	+	+			+	+	+	+	+	+	+	+	+	+	+	+	+	+		+	120
Morris-Lecar	+	+	+	-		-	-	+	+	+	+	+	+	+		+	+	-	+	+	-	-	600
Wilson	-	+	+	+			+	+	+	+	+	+	+	+	+		+	+					180
Hodgkin-Huxley	+	+	+	+			+	+	+	+	+	+	+	+	+	+	+	+	+	+		+	1200

Σχήμα 2.2: Παρουσιάζεται μια σύγκριση των νευροϋπολογιστικών ιδιοτήτων των μοντέλων αιχμής και πυροδότησης. Η ένδειξη "# των FLOPS" αναφέρεται στον κατά προσέγγιση αριθμό των λειτουργιών κινητής υποδιαστολής (π.χ. προσθέσεις, πολλαπλασιασμοί) που απαιτούνται για την προσομοίωση του μοντέλου σε μια χρονική περίοδο 1 ms. Κάθε κενό τετράγωνο υποδεικνύει την ιδιότητα που το μοντέλο θα έπρεπε θεωρητικά να εμφανίζει, εφόσον οι παράμετροι επιλέγονταν κατάλληλα, αλλά ο συγγραφέας δεν κατάφερε να βρει τις κατάλληλους παραμέτρους εντός εύλογου χρονικού διαστήματος (Izhikevich, 2004).

Παρακάτω κάνουμε μια συνοπτική παρουσίαση μερικών από τα πιο ευρέως χρησιμοποιούμενα μοντέλα νευρώνων που "πυροδοτούν" και "εκρήγνυνται", τα οποία μπορούν να εκφραστούν με τη μορφή διαφορικών εξισώσεων (ODE) (συνεπώς, εξαιρούμε το μοντέλο απόκρισης αιχμής). Εκτός από τα 20 νευρο-υπολογιστικά χαρακτηριστικά που αναφέρθηκαν παραπάνω, εξετάζουμε επίσης αν τα μοντέλα έχουν βιοφυσικά σημαντικές και μετρήσιμες παραμέτρους και αν μπορούν να εμφανίσουν αυτόνομη χαοτική δραστηριότητα. Ξεκινάμε με τα πιο απλά μοντέλα. Η σύνοψη της σύγκρισής μας φαίνεται στο Σχήμα 2.2.

Σε όλη αυτή την ενότητα, το v αντιπροσωπεύει το δυναμικό της μεμβράνης και το v' αντιπροσωπεύει την παράγωγο του δυναμικού ως προς τον χρόνο. Όλες οι παράμετροι στα μοντέλα επιλέγονται ώστε το v να έχει τάξη μεγέθους mV (μιλιβόλτ) και ο χρόνος να έχει τάξη μεγέθους ms (χιλιοστά του δευτερολέπτου). Για να συγκρίνουμε το υπολογιστικό κόστος, υποθέτουμε ότι κάθε μοντέλο, γραμμένο ως δυναμικό σύστημα $\dot{x}=f(x)$, υλοποιείται χρησιμοποιώντας τη μέθοδο Euler πρώτης τάξης με σταθερό βήμα $x(t+\tau)=x(t)+\tau f(x(t))$, όπου το βήμα ολοκλήρωσης τ επιλέγεται ώστε να επιτευχθεί μια ικανοποιητική αριθμητική ακρίβεια.

2.1.1 Μοντέλο Integrate-and-Fire

Ένα από τα πιο ευρέως χρησιμοποιούμενα μοντέλα στην υπολογιστική νευροεπιστήμη είναι ο "διαρρέων" νευρώνας που πυροδοτεί σήμα (Leaky Integrate-and-Fire, I&F):

$$v' = I + a - bv,\ \text{αν}\ v \geq v_{thresh},\ \text{τότε}\ v \leftarrow c$$

όπου το v είναι το δυναμικό της μεμβράνης, I είναι το ρεύμα εισόδου, και τα a, b, c, και v_{thresh} είναι οι παράμετροι. Όταν το δυναμικό της μεμβράνης v φτάσει την τιμή κατωφλίου v_{thresh}, ο νευρώνας πυροδοτεί ένα σήμα και το δυναμικό v επαναφέρεται στην τιμή c.

Ο νευρώνας I&F ανήκει στην κλάση των διεγέρσιμων νευρώνων Class 1. Μπορεί να εκπέμπει σήματα με σταθερή συχνότητα και λειτουργεί ως ολοκληρωτής. Είναι το πιο απλό μοντέλο.

Το μοντέλο είναι εύκολο να υλοποιηθεί όταν το βήμα ολοκλήρωσης τ είναι 1 ms. Πράγματι, η επανάληψη v(t+1)=v(t)+(I+a−bv(t)) απαιτεί μόνο τέσσερις πράξεις κινητής υποδιαστολής (προσθέσεις, πολλαπλασιασμούς, κ.λπ.), συν μία σύγκριση με το κατώφλιο v_{thresh}. Επειδή το I&F έχει μόνο μία μεταβλητή, δεν μπορεί να έχει φασική εκπομπή, εκρήξεις οποιουδήποτε είδους, αποκρίσεις επαναφοράς, μεταβλητότητα κατωφλίου, διπλή σταθερότητα των ελκυστών ή αυτόνομη χαοτική δυναμική. Λόγω του σταθερού κατωφλίου, οι εκπομπές δεν έχουν καθυστερήσεις. Συνοπτικά, παρόλο που το I&F μοντέλο μπορεί να φαίνεται απλό, εξακολουθεί να προσφέρει σημαντικά πλεονεκτήματα σε ορισμένες περιπτώσεις. Η απλότητά του το καθιστά ιδανικό για θεωρητικές αναλύσεις και γρήγορες προσομοιώσεις, ενώ παραμένει ένα από τα βασικά εργαλεία για την κατανόηση των βασικών μηχανισμών των νευρωνικών πυροδοτήσεων. Αν και υπάρχουν πιο προηγμένα μοντέλα, το I&F συνεχίζει να αποτελεί ένα πολύτιμο εργαλείο για τους ερευνητές που επιδιώκουν να εξετάσουν και να αναπτύξουν αναλυτικά αποτελέσματα.

2.1.2 Μοντέλο Integrate-and-Fire με Προσαρμογή

Το μοντέλο I&F είναι μονοδιάστατο (1-D), συνεπώς δεν μπορεί να εκρήγνυται ή να έχει άλλες ιδιότητες των φλοιικών νευρώνων. Κάποιος θα μπορούσε να σκεφτεί ότι η προσθήκη μιας δεύτερης γραμμικής εξίσωσης

$$v' = I + a - bv + g(d - v)$$

$$g' = \frac{(e\delta(t) - g)}{\tau}$$

που περιγράφει τη δυναμική ενεργοποίησης ενός ρεύματος καλίου (K) υψηλού κατωφλίου θα μπορούσε να επιφέρει βελτίωση, π.χ., να ενσωματώσει στο μοντέλο την προσαρμογή της συχνότητας εκπομπής σημάτων. Πράγματι, κάθε πυροδότηση αυξάνει την πύλη ενεργοποίησης του K με την συνάρτηση δέλτα του Dirac και παράγει ένα ρεύμα εξόδου που μειώνει τη συχνότητα εκπομπής σημάτων. Οι προσομοιώσεις αυτού του μοντέλου απαιτούν 10 πράξεις κινητής υποδιαστολής ανά χρονικό βήμα 1 ms, ωστόσο το μοντέλο εξακολουθεί να υστερεί σε πολλές σημαντικές ιδιότητες των φλοιικών νευρώνων που εκπέμπουν σήματα.

2.1.3 Μοντέλο Integrate-and-Fire-or-Burst

Ο Smith και οι συνεργάτες του πρότειναν μια βελτίωση — το μοντέλο "integrate-and-fire-or-burst" (I&FB) (Smith, 2000):

$$v' = I + a - bv + gH(v - v_h)h(v_T - v)$$

Εάν $v = v_{thresh}$, τότε $v \leftarrow c$

$$h' = \begin{cases} -\frac{h}{\tau^-}, & \text{αν}\ v > v_h \\ \frac{(1-h)}{\tau^+}, & \text{αν}\ v < v_h \end{cases}$$

Αυτό το μοντέλο προορίζεται για τη μοντελοποίηση των θαλαμο-φλοιικών νευρώνων. Εδώ το h περιγράφει την απενεργοποίηση του T-ρεύματος ασβεστίου, ενώ οι παράμετροι g, v_h, v_T, τ^+ και τ^- περιγράφουν τη δυναμική του T-ρεύματος. Η συνάρτηση H είναι η συνάρτηση βήματος Heaviside.

Η προσθήκη αυτής της δεύτερης μεταβλητής επιτρέπει τη δυνατότητα για εκρήξεις και άλλα ενδιαφέροντα φαινόμενα, όπως συνοψίζονται στο Σχήμα 2.2. Ωστόσο, αυτό έχει ένα κόστος: απαιτούνται 9 έως 13 πράξεις (ανάλογα με την τιμή του v) για την προσομοίωση 1 ms του μοντέλου.

2.1.4 Μοντέλο Resonate-and-Fire

Ο νευρώνας "resonate-and-fire" είναι ένα δισδιάστατο (2-D) ανάλογο του νευρώνα I&F ('Resonate-and-fire neurons,' 2001):

$$z' = I + (b + \omega)z$$

Εάν Im $z=a_{thresh}$, τότε $z \leftarrow z_0(z)$

όπου το πραγματικό μέρος της μιγαδικής μεταβλητής z είναι το δυναμικό της μεμβράνης. Εδώ τα b, ω, και a_{thresh} είναι παράμετροι, και το $z_0(z)$ είναι μια αυθαίρετη συνάρτηση που περιγράφει την επαναφορά μετά την εκπομπή σήματος, η οποία εξαρτάται από την προηγούμενη δραστηριότητα.

Το μοντέλο "resonate-and-fire" είναι απλό και αποδοτικό — απαιτεί 10 πράξεις για την προσομοίωση 1 ms. Όταν η συχνότητα ταλάντωσης ω=0, το μοντέλο μετατρέπεται σε ολοκληρωτή. Οι νευρο-υπολογιστικές ιδιότητές του συνοψίζονται στο Σχήμα 2.2.

2.1.5 Μοντέλο Quadratic I&F

Μια εναλλακτική στο διαρρέον μοντέλο I&F είναι ο τετραγωνικός νευρώνας I&F, γνωστός επίσης ως ο "theta-neuron" (Ermentrout,1996), (Ermentrout & Kopell,1986) ή το κανονικό μοντέλο Ermentrout-Kopell (όταν γράφεται σε τριγωνομετρική μορφή) (Hoppensteadt,1997). Το παρουσιάζουμε εδώ σύμφωνα με (Latham, 2000):

$$v' = I + a(v - v_{rest})(v - v_{thresh})$$

Εάν $v=v_{peak}$, τότε $v \leftarrow v_{reset}$

όπου v_{reset} και v_{thresh} είναι οι τιμές ηρεμίας και κατωφλίου του δυναμικού της μεμβράνης. Αυτό το μοντέλο θεωρείται κανονικό, με την έννοια ότι κάθε νευρώνας Κλάσης 1 που περιγράφεται από ομαλές διαφορικές εξισώσεις μπορεί να μετατραπεί σε αυτό μέσω μιας συνεχούς αλλαγής μεταβλητών ('Class 1 neural excitability', 1999). Απαιτεί μόνο επτά πράξεις για την προσομοίωση 1 ms του μοντέλου, και θα πρέπει να προτιμάται όταν προσομοιώνουμε μεγάλα δίκτυα ολοκληρωτών. Σε αντίθεση με το γραμμικό ανάλογό του, ο τετραγωνικός νευρώνας I&F έχει λανθάνουσες εκπυρσοκροτήσεις, ένα κατώφλι που εξαρτάται από τη δραστηριότητα (το οποίο είναι το v_{thresh} μόνο όταν I=0) και διπλή σταθερότητα ανάμεσα στις καταστάσεις ηρεμίας και εκπυρσοκρότησης σημάτων.

2.1.6 Μοντέλο Νευρώνα με Εκπομπή Σημάτων από τον Izhikevich (2003)

Όλα τα αποτελέσματα που παρουσιάζονται στο Σχήμα 2.1 ελήφθησαν χρησιμοποιώντας ένα απλό μοντέλο νευρώνων που εκπέμπουν σήμα, το οποίο προτάθηκε πρόσφατα από τον Izhikevich (Izhikevich, 2003):

$$v' = 0.04v^2 + 5v + 140 - u + I$$

$$u' = a(bv - u)$$

με την βοηθητική επαναφορά μετά την πυροδότηση:

$$\text{if } v \geq +30\ \text{mV},\ \text{τότε } \begin{cases} v \leftarrow c \\ u \leftarrow u + d \end{cases}$$

Εδώ η μεταβλητή v αντιπροσωπεύει το δυναμικό της μεμβράνης του νευρώνα, ενώ η u αντιπροσωπεύει μια μεταβλητή αποκατάστασης της μεμβράνης, που λαμβάνει υπόψη την ενεργοποίηση των ιοντικών ρευμάτων K+ και την απενεργοποίηση των ιοντικών ρευμάτων Na+, παρέχοντας αρνητική ανάδραση στο v. Αφού η πυροδότηση φτάσει την κορυφή της (+30 mV), το δυναμικό της μεμβράνης και η μεταβλητή αποκατάστασης επαναφέρονται. Εάν το v υπερβεί το 30, τότε πρώτα επαναφέρεται στο 30, και στη συνέχεια στο c, ώστε όλες οι εκπομπές να έχουν ίσες μεγάλες τιμές.

Το τμήμα 0.04v2+5v+140 επιλέγεται ώστε το v να έχει τάξη μεγέθους mV και ο χρόνος να έχει τάξη μεγέθους ms. Η γεωμετρική ανάλυση του μοντέλου βασίζεται στις γρήγορες και αργές nullclines, όπως περιγράφει ο Izhikevich στο βιβλίο του Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting..

Το μοντέλο μπορεί να παρουσιάσει πρότυπα πυροδότησης όλων των γνωστών τύπων φλοιικών νευρώνων, με την επιλογή των παραμέτρων a, b, c και d δεδομένων στο (Izhikevich, 2003). Απαιτεί μόνο 13 πράξεις κινητής υποδιαστολής για να προσομοιώσει 1 ms, οπότε είναι αρκετά αποδοτικό για μεγάλες προσομοιώσεις δικτύων φλοιικών νευρώνων.

Όταν οι παράμετροι είναι (a,b,c,d)=(0.2,2,−56,−16) και I=−99, το μοντέλο παρουσιάζει χαοτική πυροδότηση σημάτων. Ωστόσο, το βήμα ολοκλήρωσης τ πρέπει να είναι μικρό για να επιτευχθεί επαρκής αριθμητική ακρίβεια.

Τονίζεται ότι τα +30 mV δεν είναι το κατώφλι, αλλά η κορυφή της πυροδότησης. Η κατωφλιακή τιμή του νευρώνα του μοντέλου είναι μεταξύ -70 και -50, και είναι δυναμική, όπως στους βιολογικούς νευρώνες.

2.1.7 Μοντέλο FitzHugh–Nagumo

Οι παράμετροι στο μοντέλο FitzHugh–Nagumo (Izhikevich, 2004):

$$v' = a + bv + cv^2 + dv^3 - u$$

$$u' = \varepsilon(ev - u)$$

μπορούν να ρυθμιστούν έτσι ώστε το μοντέλο να περιγράφει τη δυναμική εκπομπής σημάτων πολλών αντηχούντων νευρώνων. Οι νευρο-υπολογιστικές ιδιότητες του μοντέλου συνοψίζονται στο Σχήμα 2.2. Δεδομένου ότι είναι απαραίτητη η προσομοίωση του σχήματος κάθε σήματος, το βήμα χρόνου στο μοντέλο πρέπει να είναι σχετικά μικρό, π.χ., τ=0.25 ms. Απαιτούνται 18 πράξεις κινητής υποδιαστολής ανά 0.25 ms, δηλαδή 72 πράξεις ανά 1 ms προσομοίωσης. Εφόσον το μοντέλο είναι ένα δισδιάστατο σύστημα διαφορικών εξισώσεων (ODEs) χωρίς επαναφορά, δεν μπορεί να παρουσιάσει αυτόνομη χαοτική δυναμική ή εκρήξεις. Η προσθήκη θορύβου σε αυτό ή σε άλλα 2-D μοντέλα επιτρέπει την στοχαστική εκπομπή σημάτων.

2.1.8 Μοντέλο Hindmarsh–Rose

Το μοντέλο Hindmarsh–Rose για τον θαλαμικό νευρώνα μπορεί να γραφτεί ως εξής (Rose,1989):

$$v' = u - F(v) + I - w$$

$$u' = G(v) - u$$

$$w' = \frac{(H(v) - w)}{\tau}$$

όπου οι F, G, και H είναι κάποιες συναρτήσεις. Ανάλογα με την επιλογή τους, το μοντέλο μπορεί, κατ' αρχήν, να παρουσιάσει όλες τις νευρο-υπολογιστικές ιδιότητες που φαίνονται στο Σχήμα 2.1. Το πρόβλημα είναι, φυσικά, πώς να βρεθούν οι συναρτήσεις για το μοντέλο, π.χ., για νευρώνες RS ή LTS. Ας υποθέσουμε ότι αυτό το πρόβλημα λύθηκε και ότι οι συναρτήσεις είναι πολυώνυμα τρίτου βαθμού (στην καλύτερη περίπτωση). Δεδομένου ότι πρέπει να προσομοιώσουμε το σχήμα του δυναμικού δράσης, το μέγιστο βήμα χρόνου είναι 0.25 ms. Δεδομένου ότι απαιτούνται 30 πράξεις κινητής υποδιαστολής ανά 0.25 ms προσομοίωσης, θα χρειάζονταν 120 πράξεις για να προσομοιωθεί 1 ms του μοντέλου. Και πάλι, αυτή είναι μια αισιόδοξη εκτίμηση που μπορεί να μην επιτευχθεί.

2.1.9 Μοντέλο Hodgkin–Huxley

Το μοντέλο Hodgkin–Huxley (Izhikevich, 2004) είναι ένα από τα πιο σημαντικά μοντέλα στην υπολογιστική νευροεπιστήμη. Αποτελείται από τέσσερις εξισώσεις και δεκάδες παραμέτρους, που δεν παρέχονται εδώ, και περιγράφουν το δυναμικό της μεμβράνης, την ενεργοποίηση των ρευμάτων νατρίου (Na) και καλίου (K), και την απενεργοποίηση του ρεύματος νατρίου. Παρόλο που το μοντέλο έχει αρκετά περιορισμένη συμπεριφορά με τις αρχικές τιμές των παραμέτρων, μπορεί στην πραγματικότητα να παρουσιάσει όλες τις ιδιότητες στο Σχήμα 2.1, αν οι παράμετροι ρυθμιστούν σωστά.

Γενικά, οι επιστήμονες αναφέρονται σε όλα τα μοντέλα που βασίζονται στη διαγωγιμότητα ως τύπου Hodgkin–Huxley. Αυτά τα μοντέλα είναι σημαντικά όχι μόνο επειδή οι παράμετροί τους είναι βιοφυσικά σημαντικοί και μετρήσιμοι, αλλά και επειδή μας επιτρέπουν να εξετάσουμε ερωτήματα σχετικά με την ολοκλήρωση των συνάψεων, τη δενδριτική φιλτράρισμα καλωδίων, τα αποτελέσματα της μορφολογίας των δενδριτών, την αλληλεπίδραση των ιοντικών ρευμάτων και άλλα ζητήματα που σχετίζονται με τη δυναμική των κυττάρων.

Το μοντέλο είναι εξαιρετικά ακριβό σε ό,τι αφορά την υπολογιστική υλοποίηση. Απαιτεί 120 πράξεις κινητής υποδιαστολής για την αξιολόγηση 0.1 ms χρόνου προσομοίωσης (υποθέτοντας ότι κάθε εκθέτης απαιτεί μόνο δέκα πράξεις), δηλαδή 1200 πράξεις ανά 1 ms. Συνεπώς, κάποιος μπορεί να χρησιμοποιήσει τη μέθοδο Hodgkin–Huxley μόνο για να προσομοιώσει έναν μικρό αριθμό νευρώνων ή όταν ο χρόνος προσομοίωσης δεν αποτελεί ζήτημα.

2.2 Επιλογή του Κατάλληλου Μοντέλου

Όπως μπορεί να δει ο αναγνώστης στο Σχήμα 2.2, έχουν προταθεί πολλά μοντέλα νευρώνων που εκπέμπουν σήματα. Πως γίνεται η επιλογή του πλέον κατάλληλου για την προσομοίωση που θέλουμε να πραγματοποιήσουμε;Η απάντηση εξαρτάται από το είδος του προβλήματος. Αν ο στόχος είναι να μελετηθεί πώς η συμπεριφορά των νευρώνων εξαρτάται από μετρήσιμες φυσιολογικές παραμέτρους, όπως οι μέγιστες αγωγιμότητες, οι συναρτήσεις (απ)ενεργοποίησης σε σταθερή κατάσταση, και οι χρονικές σταθερές, τότε το μοντέλο τύπου Hodgkin–Huxley είναι το καλύτερο. Φυσικά, θα μπορούσαμε να προσομοιώσουμε μόνο δεκάδες νευρώνες που εκπέμπουν σήματα σε πραγματικό χρόνο.

Αντίθετα, αν θέλουμε να προσομοιώσουμε χιλιάδες νευρώνες που εκπέμπουν σήματα σε πραγματικό χρόνο με ανάλυση 1 ms, τότε υπάρχουν πολλά μοντέλα να διαλέξουμε. Το πιο αποδοτικό είναι το μοντέλο I&F. Παρόλο που το μοντέλο μπορεί να μην αποδίδει όλες τις σύνθετες ιδιότητες των φλοιικών νευρώνων, παραμένει ένα χρήσιμο εργαλείο για ορισμένες εφαρμογές. Με περαιτέρω βελτιώσεις και εξερεύνηση, μπορεί να προσφέρει σημαντική συνεισφορά στην κατανόηση της νευρωνικής δραστηριότητας σε συγκεκριμένα πλαίσια. Ένα από τα κύρια πλεονεκτήματα του μοντέλου I&F είναι η γραμμική του φύση, η οποία το καθιστά ιδανικό για μαθηματική ανάλυση. Παρόλο που μπορεί να μην είναι το καταλληλότερο μοντέλο για όλες τις προσομοιώσεις, είναι εξαιρετικά χρήσιμο σε περιπτώσεις όπου η εξαγωγή αναλυτικών αποτελεσμάτων είναι επιθυμητή, και μπορεί να παρέχει πολύτιμα στοιχεία σε συγκεκριμένες εφαρμογές.

Το τετραγωνικό μοντέλο I&F είναι πρακτικά εξίσου αποδοτικό με το γραμμικό μοντέλο και παρουσιάζει πολλές σημαντικές ιδιότητες των πραγματικών νευρώνων, όπως εκπομπές σημάτων με καθυστερήσεις και διπλή σταθερότητα μεταξύ καταστάσεων ηρεμίας και τόνου εκπομπών. Ωστόσο, είναι μονοδιάστατο (1-D), και επομένως δεν μπορεί να παρουσιάσει εκρήξεις και δεν μπορεί να παρουσιάσει προσαρμογή της συχνότητας εκπομπής σημάτων. Επομένως, μπορεί να

χρησιμοποιηθεί σε προσομοιώσεις φλοιικών δικτύων νευρώνων μόνο όταν η βιολογική αληθοφάνεια δεν είναι σημαντική.

Αν ο στόχος είναι να κατανοηθεί η λεπτομερής χρονική δομή των εκπομπών φλοιικών σημάτων και να χρησιμοποιηθεί ο χρόνος εκπομπής ως πρόσθετη μεταβλητή για να κατανοηθεί πώς ο θηλαστικός νεοφλοιός επεξεργάζεται τις πληροφορίες, απαιτείται ένα μοντέλο εκπομπών που μπορεί να παρουσιάσει όλες ή τις περισσότερες από τις 20 νευρο-υπολογιστικές ιδιότητες των βιολογικών νευρώνων που συνοψίζονται στο Σχήμα 2.1. Το μοντέλο που προτάθηκε πρόσφατα από τον Izhikevich (Izhikevich, 2003) αναπτύχθηκε ακριβώς για αυτούς τους σκοπούς. Είναι το απλούστερο δυνατό μοντέλο που μπορεί να παρουσιάσει όλα τα πρότυπα εκπομπών στο Σχήμα 2.1.

Συμπερασματικά, η ύπαρξη ενός δικτύου φλοιικών νευρώνων που εκπέμπουν σήματα, το οποίο είναι υπολογιστικά αποδοτικό και βιολογικά αληθοφανές, και που συνδέεται σύμφωνα με τις αρχές της γνωστής ανατομίας του νεοφλοιού, θα πρέπει να είναι ο στόχος κάθε επιστήμονα που εξερευνά την επεξεργασία πληροφοριών στον εγκέφαλο των θηλαστικών.

2.3 Εφαρμογή Μοντέλου Leaky Integrate-and-Fire (LIF)

Στα πλαίσια της Πτυχιακής Εργασίας, χρησιμοποιούμε το μοντέλο Leaky Integrate-and-Fire (LIF), το οποίο αποτελεί μία από τις απλούστερες και πιο διαδεδομένες προσεγγίσεις για την προσομοίωση της συμπεριφοράς των νευρώνων. Το συγκεκριμένο μοντέλο είναι κατάλληλο για μεγάλης κλίμακας προσομοιώσεις, καθώς έχει χαμηλότερες απαιτήσεις σε υπολογιστικούς πόρους συγκριτικά με πιο περίπλοκα βιολογικά μοντέλα, όπως το Hodgkin-Huxley.

Η συνεχώς αυξανόμενη επιθυμία των επιστημόνων για προσομοίωση ολοένα και μεγαλύτερων νευρωνικών δικτύων αποσκοπεί στην καλύτερη κατανόηση της λειτουργίας του εγκεφάλου και των πολύπλοκων υπολογιστικών διεργασιών που πραγματοποιούνται σε αυτά τα δίκτυα. Όμως, η προσομοίωση αυτών των δικτύων συχνά απαιτεί μεγάλο χρόνο εκτέλεσης, καθώς η πολυπλοκότητα αυξάνεται εκθετικά με το μέγεθος του δικτύου και τον αριθμό των παραμέτρων που πρέπει να συνδυαστούν.

Για να βελτιωθεί η αποδοτικότητα και να μειωθεί ο χρόνος εκτέλεσης αυτών των προσομοιώσεων, γίνεται όλο και πιο διαδεδομένη η χρήση παράλληλης επεξεργασίας. Μεταξύ άλλων, οι κάρτες γραφικών (GPUs) αποτελούν μία καλή λύση σε αυτό το πρόβλημα, επειδή διαθέτουν μεγάλο αριθμό πυρήνων που επιτρέπουν την ταυτόχρονη επεξεργασία δεδομένων. Έτσι, μπορούμε να επιταχύνουμε σημαντικά την εκτέλεση των προσομοιώσεων και να εκτελέσουμε μεγαλύτερο αριθμό από αυτές, επιτυγχάνοντας μια εκτεταμένη ανάλυση παραμετρικών συνδυασμών και πιο ολοκληρωμένη έρευνα.

Με την υιοθέτηση παράλληλης επεξεργασίας, μπορούμε να πραγματοποιήσουμε εκτενείς σαρώσεις παραμέτρων, προσεγγίζοντας πληρέστερα την πολυπλοκότητα των νευρωνικών δικτύων και βελτιστοποιώντας τα αποτελέσματα των προσομοιώσεων. Αυτό όχι μόνο συμβάλλει στην αποδοτική χρήση υπολογιστικών πόρων αλλά και στην ανάπτυξη πιο ακριβών και ρεαλιστικών μοντέλων.

Γνωρίζουμε ιδιότητες του μοντέλου LIF (Leaky Integrate-and-Fire) και την επίδρασή τους στην απόδοση (Venetis & Provata, 2022). Το μοντέλο LIF έχει αρκετά χαρακτηριστικά που καθιστούν δύσκολη την επίτευξη υψηλής απόδοσης σε παράλληλα συστήματα, συμπεριλαμβανομένων των συστημάτων κοινής μνήμης και των GPUs. Ειδικά, γνωρίζουμε ότι το LIF είναι περιορισμένο από τη μνήμη (memory bound), κάτι που δημιουργεί δυσκολίες στην υλοποίηση αποτελεσματικών παράλληλων λύσεων, είτε πρόκειται για πολυπύρηνα συστήματα κοινής μνήμης είτε για GPUs.

Είναι σημαντικό να σημειωθεί ότι οι βελτιστοποιήσεις στην πρόσβαση στη μνήμη παίζουν καίριο ρόλο στη βελτίωση της απόδοσης αυτού του μοντέλου, κυρίως όταν ο αριθμός των νευρώνων είναι μικρός. Ωστόσο, τα όρια του εύρους ζώνης της μνήμης καλύπτονται γρήγορα, περιορίζοντας περαιτέρω την αύξηση της απόδοσης. Παρά τις προκλήσεις αυτές, σημαντικά κέρδη απόδοσης έχουν επιτευχθεί στις νεότερες GPUs, οι οποίες διαθέτουν βελτιωμένα υποσυστήματα μνήμης με τη χρήση μνήμης 3D, βελτιώνοντας σημαντικά το εύρος ζώνης της μνήμης.

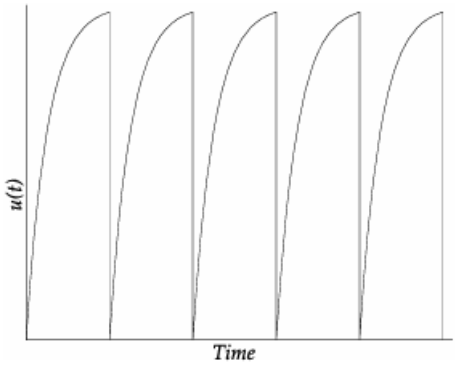
Αξίζει να σημειωθεί πως τα κόστη συγχρονισμού είναι γενικά αμελητέα, ωστόσο μπορεί να επιτευχθεί μια επιπλέον αύξηση στην απόδοση της τάξης του 5%-10% με τη χρήση των εσωτερικών βελτιστοποιήσεων που προσφέρουν τα CUDA Graphs. Στα πλαίσια της Πτυχιακής Εργασίας δεν έχει αξιοποιηθεί ωστόσο η δυνατότητα αυτή.

Το μοντέλο LIF περιγράφει την εκπομπή σήματος ενός μεμονωμένου νευρώνα και αποτελείται από τρία στάδια: α) ένα στάδιο εκθετικής ολοκλήρωσης, β) μια απότομη επαναφορά του δυναμικού και γ) μια περίοδο ανάρρωσης.

$$\frac{du(t)}{dt} = \mu - u(t) \tag{1a}$$

$$\lim_{\epsilon \to 0} u(t_q + \epsilon) \to u_{rest}, \quad \text{όταν } u(t_q) \geq u_{th}, \quad q = 1, 2, \ldots \tag{1b}$$

$$u(t) = u_{rest}, \quad \text{όταν } t_q \leq t < t_q + T_r \tag{1c}$$



Σχήμα 2.3. Δυναμικό της μεμβράνης (άξονας y) ενός απομονωμένου (μοναδικού) νευρώνα με την πάροδο του χρόνου (άξονας x), όπως προσομοιώνεται με το μοντέλο LIF. Μόλις το δυναμικό φτάσει στο κατώφλι u_{th}, το δυναμικό επαναφέρεται. Σε αυτό το σχήμα, η περίοδος ανάρρωσης Tr=0, δηλαδή, το δυναμικό αρχίζει αμέσως να αυξάνεται μετά από κάθε επαναφορά.

Συγκεκριμένα:

a. Η Εξίσωση (1a) περιγράφει τη φάση ολοκλήρωσης του δυναμικού. Η παράμετρος μ εκφράζει το μέγιστο δυναμικό που επιτυγχάνεται κατά την ολοκλήρωση.

b. Η Εξίσωση (1b) περιγράφει την απότομη, στιγμιαία επαναφορά του δυναμικού όταν αυτό υπερβαίνει μια συγκεκριμένη τιμή κατωφλίου, που δηλώνεται με u_{th}. Το δυναμικό στη συνέχεια επαναφέρεται στην κατάσταση ηρεμίας, που δηλώνεται με urest, και το q=1,2,...q είναι ένας μετρητής των επαναφορών, με $u_{rest}<u_{th}<\mu$.

c. Η Εξίσωση (1c) περιγράφει το στάδιο ανάρρωσης· μετά την επαναφορά, το δυναμικό του νευρώνα παραμένει στην κατάσταση ηρεμίας για ένα χρονικό διάστημα T_r, πριν εισέλθει ξανά στη φάση ολοκλήρωσης.

Το Σχήμα 2.3 παρουσιάζει την εξέλιξη του δυναμικού της μεμβράνης ενός μοναδικού νευρώνα σύμφωνα με το μοντέλο LIF. Το μοντέλο LIF έχει χρησιμοποιηθεί εκτενώς στη δυναμική των δικτύων νευρώνων λόγω της απλότητάς του, η οποία επιτρέπει την προσομοίωση ενός μεγάλου αριθμού νευρώνων, ενώ διατηρεί όλα τα βασικά στοιχεία της δραστηριότητας των βιολογικών νευρώνων.

Στην πτυχιακή εργασία εστιάζουμε στην εξέλιξη του δυναμικού της μεμβράνης ενός μεγάλου αριθμού αλληλεπιδρώντων νευρώνων σε ένα γραμμικό δίκτυο. Ο συνολικός αριθμός των νευρώνων δηλώνεται με N και το δυναμικό του νευρώνα i σε χρόνο t δηλώνεται με $u_i(t)$. Για να αποφευχθούν τα φαινόμενα πλευρικών επιδράσεων, οι γραμμικές διατάξεις σχηματίζουν ένα κυκλικό δίκτυο, όπου το στοιχείο N γίνεται γείτονας του στοιχείου 1 (βλ. Σχήμα 2.5). Κάθε νευρώνας συνδέεται με έναν αριθμό άλλων νευρώνων, όπως ορίζεται από τη μήτρα συνδεσιμότητας ή γειτνίασης σ. Το στοιχείο σ_{ij} δηλώνει τη δύναμη της σύζευξης μεταξύ των νευρώνων i και j. Γενικά, η μήτρα σσ δεν είναι συμμετρική, δηλαδή $\sigma_{ij} \neq \sigma_{ji}$. Τα στοιχεία της μήτρας συνδεσιμότητας μπορεί να είναι θετικά (διεγερτική σύζευξη) ή αρνητικά (ανασταλτική σύζευξη).

Λόγω αυτών των συνδέσεων, το δυναμικό του νευρώνα i επηρεάζεται από το δυναμικό των γειτόνων του, με την αλλαγή του δυναμικού να εκφράζεται ως εξής:

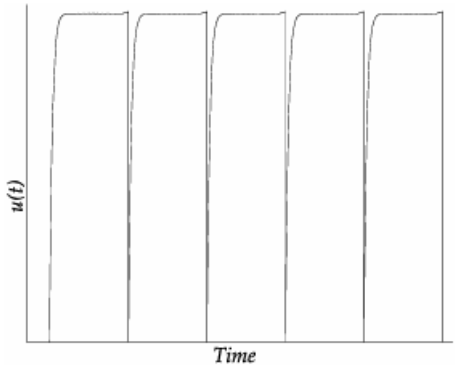
$$\frac{du_i(t)}{dt} = \mu - u_i(t) + \frac{1}{N_i}\sum_{j=1}^{N}\sigma_{ij}[u_j(t) - u_i(t)] \tag{2a}$$

$$\lim_{\epsilon\to 0} u_i(t_q + \epsilon) \to u_{rest}, \quad \text{όταν } u_i(t_q) \geq u_{th}, \quad q = 1, 2, \ldots \tag{2b}$$

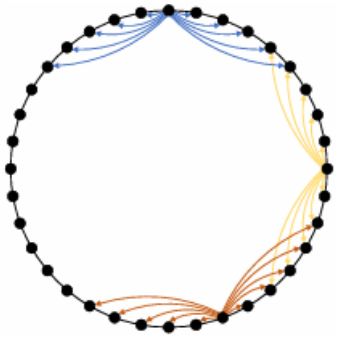
$$u_i(t) = u_{rest}, \quad \text{όταν } t_q \leq t < t_q + T_r \tag{2c}$$

Το Σχήμα 2.4 παρουσιάζει ένα παράδειγμα για το πώς το δυναμικό της μεμβράνης ενός τυχαίου νευρώνα σε ένα δίκτυο νευρώνων αλλάζει. Παρατηρήστε ότι το δυναμικό είναι διαφορετικό σε σύγκριση με αυτό που παρουσιάζεται στο Σχήμα 2.3, λόγω των συνδέσεων με τους άλλους νευρώνες.

Όπως φαίνεται από τις εξισώσεις (2), όλοι οι νευρώνες μοιράζονται τις ίδιες παραμέτρους μ, T_r, u_{th} και u_{rest}, ενώ σε γενικές γραμμές ενδέχεται να μην ισχύει το ίδιο για τις υπόλοιπες παραμέτρους.

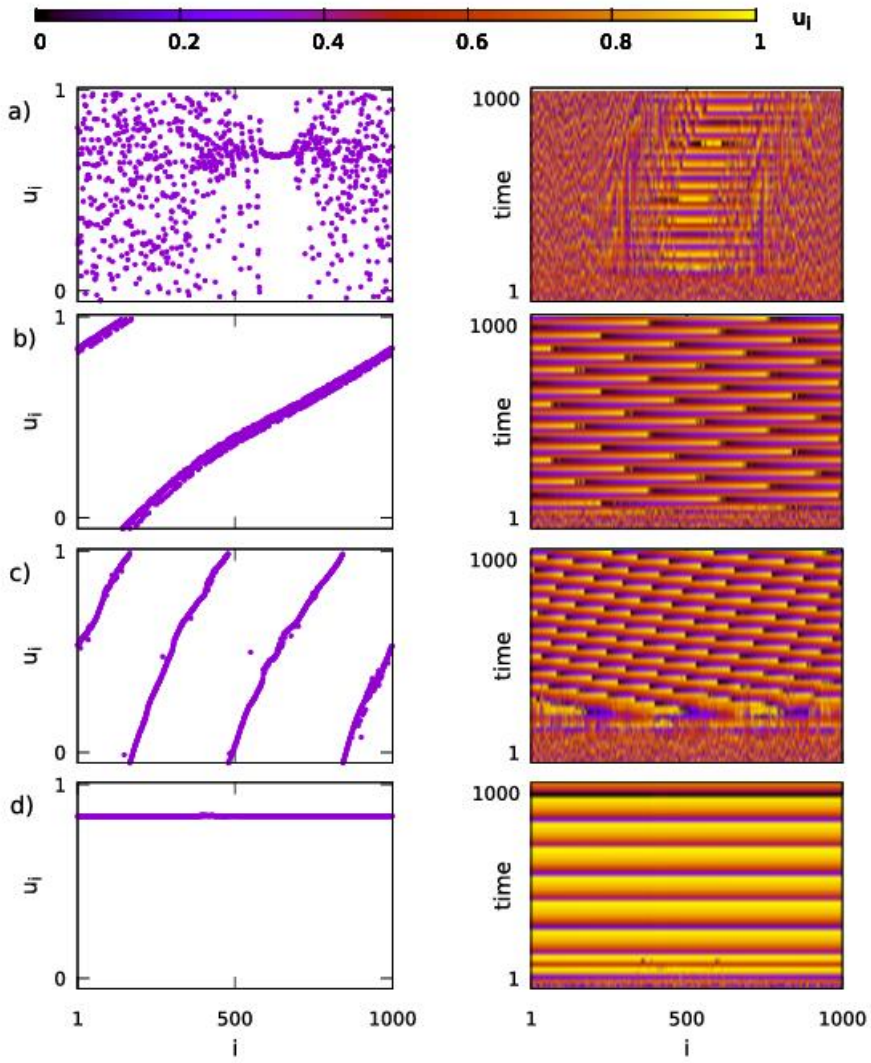


Σχήμα 2.4. Δυναμικό της μεμβράνης (άξονας y) ενός συνδεδεμένου νευρώνα με την πάροδο του χρόνου (άξονας x), όπως προσομοιώνεται με το μοντέλο LIF. Σε αυτό το παράδειγμα, ο νευρώνας που εξετάζεται είναι μέρος ενός δικτύου N=1000 νευρώνων και χρησιμοποιείται το σχήμα μη τοπικής συμμετρικής συνδεσιμότητας με R=300 γείτονες. Η περίοδος ανάρρωσης T_r=0.



Σχήμα 2.5. Ένα παράδειγμα της τοπολογίας δακτυλίου υπό το σχήμα μη τοπικής συμμετρικής συνδεσιμότητας. Το δίκτυο αποτελείται από N=36 νευρώνες και ο καθένας έχει R=5 γείτονες σε κάθε πλευρά (σύνολο 2R). Παρουσιάζονται λεπτομερείς συνδέσεις για τρεις νευρώνες. Διαφορετικά χρώματα χρησιμοποιούνται για να αναπαραστήσουν τις συνδέσεις του καθενός από αυτούς.

Κάθε νευρώνας μπορεί να έχει διαφορετικό αριθμό συνδέσεων, $N_i \leq N$, με i=1,2,...,N. Στην πιο γενική περίπτωση, κάθε νευρώνας μπορεί να έχει διαφορετικές τιμές παραμέτρων. Στη συνέχεια, μελετάμε τις παραλλαγές στην πολυπλοκότητα της μήτρας συνδεσιμότητας σ_{ij}. Λόγω της κυκλικής διάταξης του δικτύου δακτυλίου, όλοι οι δείκτες και τα αθροίσματά τους υπολογίζονται με βάση το modulo N.



Σχήμα 2.6. Το μοντέλο LIF υπό διαφορετικά σχήματα συνδεσιμότητας και τιμές παραμέτρων. Τυπικά προφίλ εμφανίζονται στα αριστερά, ενώ χρονικά διαγραμματα εμφανίζονται στα δεξιά. a) Μη τοπική συνδεσιμότητα, σ=−0.7, b) Ασύμμετρη συνδεσιμότητα, σ=−0.7, S=0, c) Ασύμμετρη συνδεσιμότητα, σ=−0.7, S=80, d) Μη τοπική συνδεσιμότητα, σ=−0.3. Οι υπόλοιπες παράμετροι είναι R=280, μ=1, και T_r=0.

Χωρίς απώλεια γενικότητας, σε αυτή τη μελέτη χρησιμοποιούμε το ακόλουθο σύνολο παραμέτρων: μ=1, T_r=0, u_{th}=0.98, u_{rest}=0 , και $\Delta t=10^{-4}$, ενώ οι παράμετροι R, σ_{ij}, και N_i ποικίλλουν, όπως υποδεικνύεται στο κείμενο. Πρέπει επίσης να επισημανθεί ότι δεν έγιναν ειδικές βελτιστοποιήσεις για την εκμετάλλευση της δομής των μητρώων συνδεσιμότητας που σχηματίζονται από τα διαφορετικά σχήματα σύζευξης, καθώς σε πραγματικές καταστάσεις, η συνδεσιμότητα των νευρώνων είναι πολύπλοκη και οι μήτρες συνδεσιμότητας δεν έχουν ειδική δομή.

Στο Σχήμα 2.6 παρουσιάζονται τυπικά προφίλ χιμαιρικών και συγχρονισμένων καταστάσεων σε δίκτυα νευρώνων και συγχρονισμένες καταστάσεις, που επιτυγχάνονται για διαφορετικά σχήματα συνδεσιμότητας και τιμές παραμέτρων. Τα τυπικά προφίλ απεικονίζονται στα αριστερά, ενώ στα δεξιά απεικονίζονται χρονικά διαγράμματα.

Το άθροισμα στην εξίσωση (2a),

$$\sum_{j=1}^{N} \sigma_{ij} \left[u_j(t) - u_i(t) \right],$$

αποτελεί το πιο υπολογιστικά απαιτητικό κομμάτι της εξίσωσης. Στην ουσία, αυτή η έκφραση περιγράφει την αλληλεπίδραση μεταξύ των δυναμικών κάθε νευρώνα και των γειτονικών του

νευρώνων μέσω του μητρώου συνδεσιμότητας, σ_{ij}, το οποίο δείχνει τις συνδέσεις ανάμεσα στους νευρώνες. Αυτό το κομμάτι της εξίσωσης απαιτεί υπολογισμό για κάθε συνδεδεμένο ζεύγος νευρώνων, κάτι που αυξάνει σημαντικά τον υπολογιστικό φόρτο, ειδικά για μεγάλα δίκτυα με μεγάλο αριθμό νευρώνων και συνδέσεων.

Για να μειωθεί η υπολογιστική πολυπλοκότητα, το άθροισμα μπορεί να αναδιαμορφωθεί στην εξίσωση (3a)ως:

$$\sum_{j=1}^{N} \sigma_{ij} u_j(t) - u_i(t) \cdot \sum_{j=1}^{N} \sigma_{ij} \tag{3a}$$

Ο τελευταίος όρος ορίζεται ως sum_of_σ και μπορεί να υπολογιστεί μία φορά για κάθε νευρώνα i, καθώς παραμένει σταθερός καθ' όλη τη διάρκεια της προσομοίωσης. Αντικαθιστώντας αυτόν τον όρο, καταλήγουμε στην εξίσωση (3b):

$$\sum_{j=1}^{N} \sigma_{ij} u_j(t) - u_i(t) \cdot \text{sum\_of\_}\sigma \tag{3b}$$

Αυτός ο αναδιαμορφωμένος υπολογισμός μειώνει τον αριθμό των πράξεων που απαιτούνται για το άθροισμα και καθιστά τη διαδικασία πιο αποδοτική. Συγκεκριμένα, το πρώτο άθροισμα, είναι ένα γινόμενο μητρώου-διανύσματος, όπου το μητρώο είναι το μητρώο συνδεσιμότητας σ και το διάνυσμα είναι το διάνυσμα των δυναμικών u(t). Αυτή η πράξη αποτελεί τη βασική υπολογιστική πράξη κατά την ενημέρωση των δυναμικών των νευρώνων σε κάθε χρονικό βήμα.

3. Αύξηση της Αποδοτικότητας της Προσομοίωσης

3.1 Αξιοποίηση Αραιών Δομών Δεδομένων

Για την προσομοίωση του δικτύου νευρώνων, θέλουμε να εξετάσουμε ταυτόχρονα πολλαπλές αρχικές συνθήκες για το δυναμικό των νευρώνων. Αυτό σημαίνει ότι, αντί να προσομοιώνουμε μία μόνο κατάσταση του δυναμικού, θα διαχειριστούμε πολλές αρχικές συνθήκες ταυτόχρονα, επιτρέποντας παράλληλες προσομοιώσεις στο ίδιο δίκτυο. Αυτή η προσέγγιση οδηγεί στη μετατροπή του αρχικού πολλαπλασιασμού αραιού μητρώου επί διάνυσμα σε αραιό μητρώο επί πυκνό μητρώο.

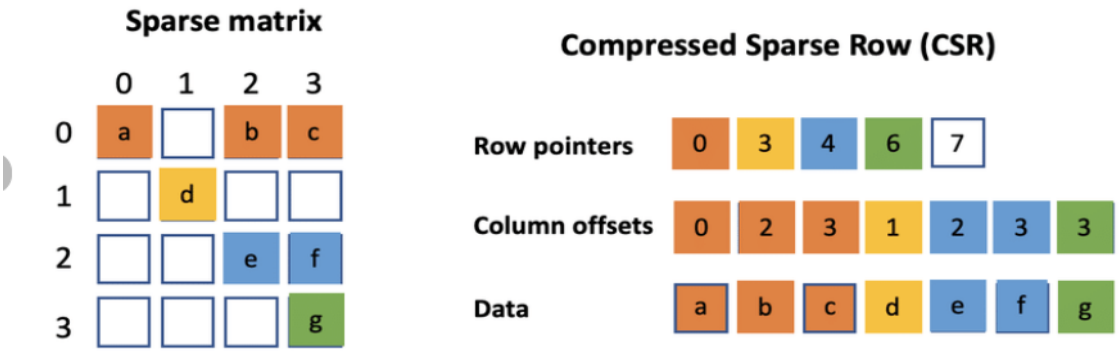
Η μετατροπή αυτή προσφέρει πλεονεκτήματα στη διαχείριση των δεδομένων και μειώνει την εξάρτηση του αλγορίθμου από τη μνήμη, αξιοποιώντας τη δομή και την κατανομή των μη μηδενικών στοιχείων του μητρώου για αποδοτικότερη χρήση των διαθέσιμων πόρων. Έτσι, επιτυγχάνεται επιτάχυνση της συνολικής διαδικασίας εκτέλεσης και πιο αποτελεσματική προσομοίωση του δικτύου για πολλαπλές αρχικές συνθήκες ταυτόχρονα.

Η χρήση αραιών δομών δεδομένων, όπως το Compressed Sparse Row (CSR) format, είναι κρίσιμη για τη βελτιστοποίηση των προσομοιώσεων σε μεγάλα δίκτυα νευρώνων, καθώς προσφέρει σημαντικά πλεονεκτήματα στην αποδοτικότητα της μνήμης και την ταχύτητα επεξεργασίας. Σε ένα μεγάλο δίκτυο, οι μήτρες συνδεσιμότητας είναι γεμάτες με μηδενικά στοιχεία, τα οποία, αν αποθηκευτούν με παραδοσιακές μεθόδους, καταλαμβάνουν σημαντικό χώρο μνήμης και επιβαρύνουν τους υπολογισμούς. Το CSR format επιλύει αυτό το ζήτημα αποθηκεύοντας μόνο τα μη μηδενικά στοιχεία μαζί με τις θέσεις τους.

Αυτή η βελτίωση έχει διπλό όφελος: Πρώτον, μειώνει δραστικά τις απαιτήσεις σε μνήμη, επιτρέποντας την αποθήκευση μεγαλύτερων μητρώων χωρίς να υπερβαίνεται η διαθέσιμη χωρητικότητα. Δεύτερον, επιταχύνει τις πράξεις που εκτελούνται με τις μήτρες αυτές, καθώς οι υπολογισμοί μπορούν να αγνοήσουν τα μηδενικά στοιχεία, εστιάζοντας μόνο στα σημαντικά δεδομένα. Αυτό σημαίνει ότι η επεξεργασία των δεδομένων γίνεται πιο αποδοτική, ειδικά σε μεγάλα δίκτυα νευρώνων όπου οι συνδέσεις είναι αραιές.

Επιπλέον, η επιλογή του CSR format διευκολύνει την καλύτερη εκμετάλλευση των σύγχρονων υπολογιστικών πόρων, όπως τα πολυπύρηνα συστήματα και οι GPUs, που είναι βελτιστοποιημένα για να επεξεργάζονται αραιές μήτρες. Οι πράξεις πολλαπλασιασμού μεταξύ αραιών και πυκνών μητρώων, όπως αυτές που γίνονται για τον υπολογισμό των δυναμικών των νευρώνων, μπορούν να εκτελούνται ταχύτερα, αξιοποιώντας στο μέγιστο τις δυνατότητες του υλικού. Αυτό καθιστά την προσομοίωση μεγάλων δικτύων πιο εφικτή και αποδοτική, επιτρέποντας στους ερευνητές να πειραματίζονται με μεγαλύτερες κλίμακες δεδομένων χωρίς να περιορίζονται από τις απαιτήσεις μνήμης και επεξεργασίας.

3.1.1 Compressed Sparse Row (CSR) Format



Σχήμα 3.1. Η εικόνα παρουσιάζει τον τρόπο αποθήκευσης ενός αραιού πίνακα χρησιμοποιώντας την αναπαράσταση Compressed Sparse Row (CSR).

Στην αριστερή πλευρά απεικονίζεται ένας αραιός πίνακας 4x4, όπου μόνο ορισμένα στοιχεία έχουν μη μηδενικές τιμές (a, b, c, d, e, f, g), ενώ στην δεξιά πλευρά φαίνεται η αναπαράσταση του πίνακα σε μορφή CSR. Η CSR αποθήκευση αποτελείται από τρία τμήματα: Row pointers που δείχνει τα σημεία έναρξης κάθε σειράς στον πίνακα δεδομένων, Column offsets που υποδεικνύει τη στήλη κάθε μη μηδενικού στοιχείου, και Data που περιέχει τα μη μηδενικά στοιχεία του πίνακα. Αυτή η μέθοδος αποθήκευσης μειώνει σημαντικά την απαιτούμενη μνήμη, καθώς αποθηκεύει μόνο τα απαραίτητα δεδομένα και τις θέσεις τους, αποφεύγοντας τις μηδενικές τιμές.

Το CSR format είναι μία από τις πιο κοινές αραιές αναπαραστάσεις πινάκων, ειδικά χρήσιμη σε εφαρμογές όπως τα δίκτυα νευρώνων. Αποθηκεύει μόνο τα μη μηδενικά στοιχεία της μήτρας, μαζί με πληροφορίες για τη θέση τους. Έτσι, μειώνεται σημαντικά ο χώρος που απαιτείται για την αποθήκευση, ενώ επιταχύνεται η εκτέλεση αλγορίθμων που χρησιμοποιούν αυτές τις μήτρες.

3.1.2 Εφαρμογή στον Κώδικα

Στον κώδικα, χρησιμοποιήσαμε τη βιβλιοθήκη Intel MKL, η οποία παρέχει βελτιστοποιημένες ρουτίνες για πράξεις με αραιές μήτρες. Τα αρχεία εισόδου του προγράμματος περιγράφουν τις μήτρες συνδεσιμότητας σε μορφή Market Matrix, η οποία είναι παρόμοια με τη μορφή COO (Coordinate List). Μετατρέψαμε τη μήτρα συνδεσιμότητας στη μορφή CSR (Compressed Sparse Row), καθώς διαπιστώσαμε ότι η χρήση του συγκεκριμένου format βελτίωσε σημαντικά την απόδοση σε σχέση με το COO. Η Intel MKL υποστηρίζει την αποδοτική εκτέλεση πράξεων σε αυτή τη μορφή.

3.2 Παράλληλη Επεξεργασία και Πράξεις Μητρώων

Η αντικατάσταση της βασικής πράξης πολλαπλασιασμού από αραιό μητρώο επί διάνυσμα σε αραιό μητρώο επί πυκνό μητρώο προσφέρει σημαντικά πλεονεκτήματα όσον αφορά την αποδοτικότητα και την κλιμάκωση. Στην παραδοσιακή μέθοδο αραιό μητρώο επί διάνυσμα, κάθε πολλαπλασιασμός μεταξύ μιας αραιής μήτρας συνδεσιμότητας και ενός διανύσματος δυναμικών γίνεται ξεχωριστά, πράγμα που περιορίζει τις δυνατότητες βελτιστοποίησης. Με την αλλαγή σε αραιό μητρώο επί πυκνό

μητρώο, πολλαπλές προσομοιώσεις μπορούν να συνδυαστούν και να υπολογιστούν ταυτόχρονα, εκμεταλλευόμενες πλήρως την παράλληλη επεξεργασία που προσφέρουν οι σύγχρονες GPUs και τα πολυπύρηνα συστήματα.

Η παράλληλη επεξεργασία είναι ιδιαίτερα χρήσιμη για την ταχεία προσομοίωση μεγάλων δικτύων νευρώνων, όπου απαιτείται η ταυτόχρονη επεξεργασία μεγάλου αριθμού δεδομένων. Με τον πολλαπλασιασμό αραιού μητρώου επί πυκνό μητρώο, μπορούμε να εκμεταλλευτούμε καλύτερα τη διαθέσιμη υπολογιστική ισχύ, καθώς οι σύγχρονες αρχιτεκτονικές επιτρέπουν τον υπολογισμό πολλαπλών γραμμών του μητρώου συνδεσιμότητας ταυτόχρονα. Αυτό μειώνει τις καθυστερήσεις που προκύπτουν από τη συνεχή πρόσβαση στη μνήμη και αυξάνει την αποδοτικότητα.

Επιπλέον, αυτή η προσέγγιση επιτρέπει τη μείωση της εξάρτησης από την ταχύτητα της μνήμης, καθώς οι πράξεις μεταξύ αραιών και πυκνών πινάκων εκτελούνται πιο αποδοτικά. Με τη χρήση CSR format για τις αραιές μήτρες, μπορούμε να εκτελούμε πράξεις πολλαπλασιασμού πιο γρήγορα, ελαχιστοποιώντας την ανάγκη για συνεχείς προσπελάσεις της μνήμης και επιτυγχάνοντας υψηλότερη απόδοση στις προσομοιώσεις δικτύων νευρώνων.

3.2.1 Πολλαπλασιασμός Μήτρας με Μήτρα

Ο πολλαπλασιασμός του μητρώου συνδεσιμότητας με το διάνυσμα δυναμικών είναι η κύρια πράξη για την εξομοίωση δικτύων νευρώνων. Κάθε χρονική στιγμή, αυτή η πράξη παράγει τα νέα δυναμικά των νευρώνων για την επόμενη χρονική μονάδα. Παρά τη σημαντική συμβολή του, ο υπολογισμός αυτός είναι memory-bound, καθώς η πρόσβαση στη μνήμη καθυστερεί την επεξεργασία, ειδικά όταν το μητρώο συνδεσιμότητας είναι αραιό.

Μια λύση σε αυτό το πρόβλημα είναι η ταυτόχρονη επεξεργασία πολλών προσομοιώσεων. Καθώς τα διαφορετικά δυναμικά των νευρώνων για κάθε προσομοίωση εξαρτώνται από τις αρχικές συνθήκες, μπορούμε να συνδυάσουμε αυτές τις αρχικές συνθήκες σε ένα δισδιάστατο μητρώο. Έτσι, οι πολλαπλές προσομοιώσεις μετατρέπονται σε μια μόνο προσομοίωση πολλαπλών δυναμικών.

Ο πολλαπλασιασμός μεταξύ αραιού μητρώου συνδεσιμότητας και πυκνού μητρώου δυναμικών νευρώνων επιτρέπει την καλύτερη διαχείριση της ταχύτητας μνήμης. Ειδικότερα, η χρήση του πολλαπλασιασμού αραιού μητρώου επί πυκνό μητρώο βελτιώνει την αποδοτικότητα σε σχέση με τη χρήση πολλαπλασιασμού αραιού μητρώου επί διάνυσμα, διότι εκμεταλλεύεται την παράλληλη επεξεργασία σε GPUs και πολυπύρηνα συστήματα, ενώ παράλληλα μειώνει το κόστος μεταφοράς δεδομένων από και προς την μνήμη.

Επιπλέον, οι βελτιώσεις στη διαχείριση της μνήμης, όπως η μετατροπή του format από COO σε CSR, βελτιώνουν την απόδοση ακόμα περισσότερο, ειδικά όταν έχουμε μεγάλα μητρώα με πολλές αρχικές συνθήκες, κάνοντας τη συνολική προσομοίωση πιο αποδοτική και γρήγορη.

3.2.2 Υλοποίηση στον Κώδικα

Οι απαραίτητες μεταβλητές επεκτάθηκαν από διανύσματα σε δισδιάστατα μητρώα ώστε να υποστηρίζουν τις πολλαπλές αρχικές συνθήκες. Αυτό επιτεύχθηκε με την τροποποίηση των διαστάσεών τους σε n*m και την προσαρμογή των βρόχων επανάληψης ώστε να λαμβάνουν υπόψη τη νέα διάσταση, όπου m είναι το πλήθος των αρχικών συνθηκών.

3.3 oneAPI MKL

Η συνάρτηση mkl_sparse_d_mm από την Intel oneMKL χρησιμοποιείται για τον υπολογισμό του γινομένου ενός αραιού μητρώου και ενός πυκνού μητρώου, με το αποτέλεσμα να αποθηκεύεται ως πυκνό μητρώο (Intel, 2024). Η συνάρτηση έχει τη μορφή:

```
sparse_status_t mkl_sparse_d_mm(
  const sparse_operation_t operation,
  const double alpha,
  const sparse_matrix_t A,
  const struct matrix_descr descr,
```

```
  const sparse_layout_t layout,
  const double *B,
  const MKL_INT columns,
  const MKL_INT ldb,
  const double beta,
  double *C,
  const MKL_INT ldc
);
```

Αρχικά, το operation καθορίζει τον τρόπο με τον οποίο θα γίνει ο πολλαπλασιασμός (χωρίς μετασχηματισμό ή με μετασχηματισμό). Στη συνάρτηση πολλαπλασιασμού μήτρας με μήτρα, η παράμετρος operation καθορίζει αν η μήτρα A θα πολλαπλασιαστεί κανονικά, ανάστροφα ή συζυγία. Οι διαθέσιμοι μετασχηματισμοί είναι ο κανονικός πολλαπλασιασμός op(A)=A, ο ανάστροφος op(A)=A^T, και ο συζυγής op(A) = A^H. Ανάλογα με την επιλογή του μετασχηματισμού, οι μήτρες B και C προσαρμόζονται και επεξεργάζονται κατάλληλα για να υπολογιστεί το τελικό αποτέλεσμα C.Η παράμετρος alpha είναι ένας συντελεστής που πολλαπλασιάζει το μητρώο A. Το μητρώο A είναι αραιός και αντιπροσωπεύεται από το sparse_matrix_t A, ενώ η παράμετρος descr παρέχει πρόσθετες ιδιότητες για το μητρώο (όπως αν είναι συμμετρικό ή τριγωνικό). Το layout καθορίζει αν το πυκνό μητρώο B αποθηκεύεται σε διάταξη σειρών ή στηλών.

Η συνάρτηση εκτελεί την πράξη C:=α×A×B+β×C, όπου οι παράμετροι B και C είναι τα μητρώα εισόδου και εξόδου αντίστοιχα, ενώ οι παράμετροι columns, ldb, και ldc καθορίζουν τις διαστάσεις και τη διάταξη των πυκνών πινάκων. Η συνάρτηση επιστρέφει μια τιμή κατάστασης που υποδεικνύει την επιτυχία ή αποτυχία της εκτέλεσης.Η συνάρτηση αυτή επιτρέπει την παράλληλη εκτέλεση του πολλαπλασιασμού σε πολυπύρηνα συστήματα, κάτι που εκμεταλλευτήκαμε για την περαιτέρω αξιολόγηση της υλοποίησής μας. Η δυνατότητα αυτή μας δίνει τη δυνατότητα να επιταχύνουμε σημαντικά τον υπολογισμό, ειδικά σε περιπτώσεις όπου τα δεδομένα είναι μεγάλα ή η πράξη πολλαπλασιασμού είναι υπολογιστικά εντατική. Χάρη στην παράλληλη επεξεργασία, ο φόρτος κατανέμεται σε πολλαπλούς πυρήνες, βελτιώνοντας την απόδοση και μειώνοντας τον συνολικό χρόνο εκτέλεσης. Αυτή η προσέγγιση μας επιτρέπει να αξιοποιούμε πλήρως τις δυνατότητες του υλικού, διασφαλίζοντας ότι η υλοποίηση μπορεί να ανταποκριθεί σε απαιτητικές εφαρμογές.

4. Παράμετροι Προσομοιώσεων

Σε αυτό το κεφάλαιο παρουσιάζονται τα αποτελέσματα της εκτέλεσης του τελικού κώδικα, που υλοποιήθηκε με βάση τις βελτιστοποιήσεις που αναλύθηκαν στα προηγούμενα κεφάλαια.Η CPU του συστήματος είναι η Intel Core i7 8750H με έξι πυρήνες, η μνήμη του συστήματος είναι 8 GB. Οι προσομοιώσεις εκτελέστηκαν με διαφορετικά πλήθη αρχικών συνθηκών (1, 2, 4, 8 και 16), διαφορετικό πλήθος νημάτων (1, 2, 4 και 8), και διαφορετικά μητρώα συνδεσιμότητας.Στα ακόλουθα διαγράμματα, αναλύεται ο χρόνος εκτέλεσης και η επιτάχυνση (speedup) της προσομοίωσης.

Τα μητρώα συνδεσιμότητας που χρησιμοποιήθηκαν προήλθαν από το SuiteSparse Matrix Collection, μια δημόσια βάση δεδομένων αραιών μητρώων που χρησιμοποιείται για δοκιμές και εφαρμογές σε αριθμητικές μεθόδους και προσομοιώσεις.

Χρησιμοποιήθηκαν οι: tuma2, ww_vref_6405, ex19, ca-HepPh, lhr11, kmnist_norm_10NN, vsp_c-30_data_data, bloweybq, G66, και 3elt_dual. Τα μητρώα προέρχονται από διάφορους τομείς της Επιστήμης, όπως ανάλυση γραφημάτων, πρόβληματα 2D/3D, πρόβληματα υπολογιστικής ρευστοδυναμικής, πρόβληματα υλικών και έχουν διαφορετική δομή, κατανομή και πλήθος μη μηδενικών στοιχείων. Στα πλαίσια της Πτυχιακής Εργασίας αξιοποιήθηκαν για την προσομοίωση διαφορετικών ειδών δικτύων νευρώνων.

Τα μητρώα που επιλέχθηκαν είναι όλα τετραγωνικά. Είναι απαραίτητο αυτά τα μητρώα να είναι τετραγωνικά. Η τετραγωνική δομή αντιπροσωπεύει τις συνδέσεις κάθε στοιχείου με όλα τα υπόλοιπα του συστήματος, καθώς και με τον εαυτό του, αν αυτό είναι απαραίτητο. Κάθε γραμμή και κάθε στήλη αντιστοιχεί σε ένα στοιχείο του συστήματος, και η θέση (i, j) του πίνακα δείχνει αν και πώς το στοιχείο i συνδέεται με το στοιχείο j. Εφόσον ο αριθμός των στοιχείων παραμένει σταθερός, το μητρώο πρέπει να είναι τετραγωνικό ώστε να διατηρεί την πλήρη χαρτογράφηση των συνδέσεων αυτών.

Ο παρακάτω πίνακας συνοψίζει τα βασικά χαρακτηριστικά των μητρώων που χρησιμοποιήθηκαν:

Πίνακας 1. Χαρακτητιστικά μητρώων

Όνομα	Μη Μηδενικά Στοιχεία	Διάσταση	Είδος
3elt_dual	26,556	9000x9000	2D/3D Problem
G66	36,000	9000x9000	Undirected Weighted Random Graph
kmnist_norm_10NN	156,932	10000x10000	Undirected Weighted Graph
lhr11	231,806	10964x10964	Chemical Process Simulation Problem
bloweybq	49,999	10001x10001	Materials Problem
vsp_c-30_data_data	124,368	11023x11023	Random Unweighted Graph
ca-HepPh	237,010	12008x12008	Undirected Graph
ex19	259,577	12005x12005	Computational Fluid Dynamics Problem
tuma2	49,365	12992x12992	2D/3D Problem
ww_vref_6405	48,737	13251x13251	Eigenvalue/Model Reduction Problem

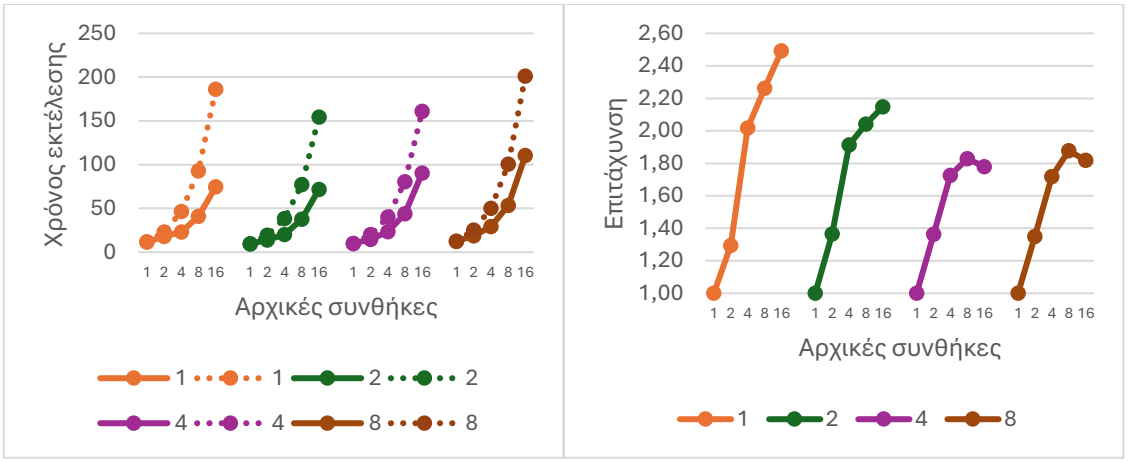
4.1 Χρόνος Εκτέλεσης και Επιτάχυνση

Εκτελέσαμε το αρχικό πρόγραμμα που υλοποιεί την προσομοίωση με μια μόνο αρχική συνθήκη για τα δυναμικά των νευρώνων για κάθε μητρώο. Οι διακεκομμένες γραμμές στο αριστερό σκέλος κάθε διαγράμματος αντιστοιχούν στον χρόνο εκτέλεσης του προγράμματος αυτού για συνολικά 1, 2, 4, 8, και 16 διαφορετικές αρχικές συνθήκες. Οι εκτελέσεις αυτές πραγματοποιήθηκαν ξεχωριστά και αθροίστηκαν οι αντίστοιχοι χρόνοι εκτέλεσης. Διαφορετικά χρώματα αντιστοιχούν στην χρήση ενός η περισσότερων (1, 2, 4 και 8) νημάτων για την εκτέλεση του προγράμματος.

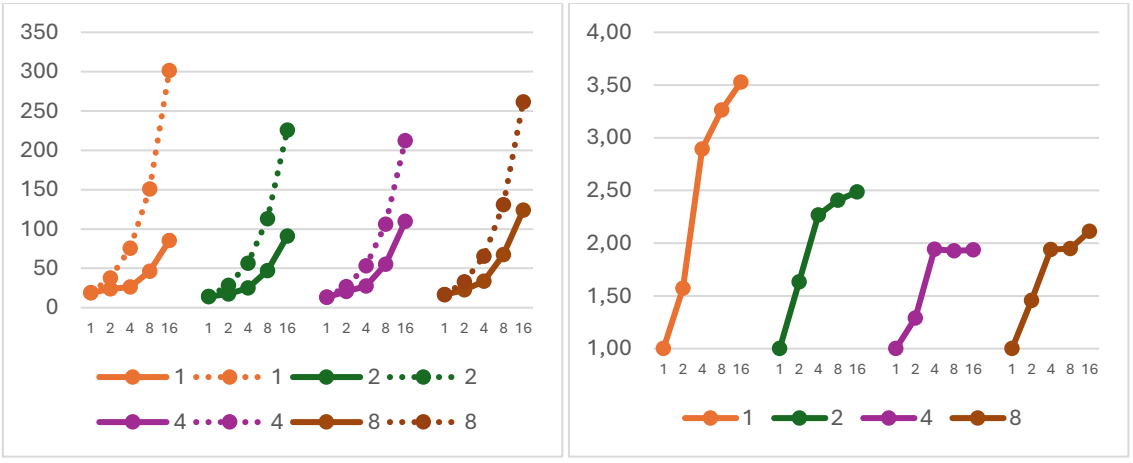
Αντίστοιχα, εκτελέσαμε το νέο πρόγραμμα που υποστηρίζει την ταυτόχρονη προσομοίωση πολλαπλών αρχικών συνθηκών με ένα ή περισσότερα (1, 2, 4 και 8) νήματα. Οι συνεχείς γραμμές στο αριστερό σκέλος κάθε διαγράμματος αντιστοιχούν στον χρόνο εκτέλεσης για το νέο πρόγραμμα.

Τα ίδια αποτελέσματα συνοψίζονται στο δεξί σκέλος κάθε διαγράμματος, όπου παρουσιάζεται η επιτάχυνση που επιτυγχάνεται με την ταυτόχρονη χρήση περισσότερων αρχικών συνθηκών. Ακολουθούν τα διαγράμματα για όλα τα μητρώα που χρησιμοποιήθηκαν.

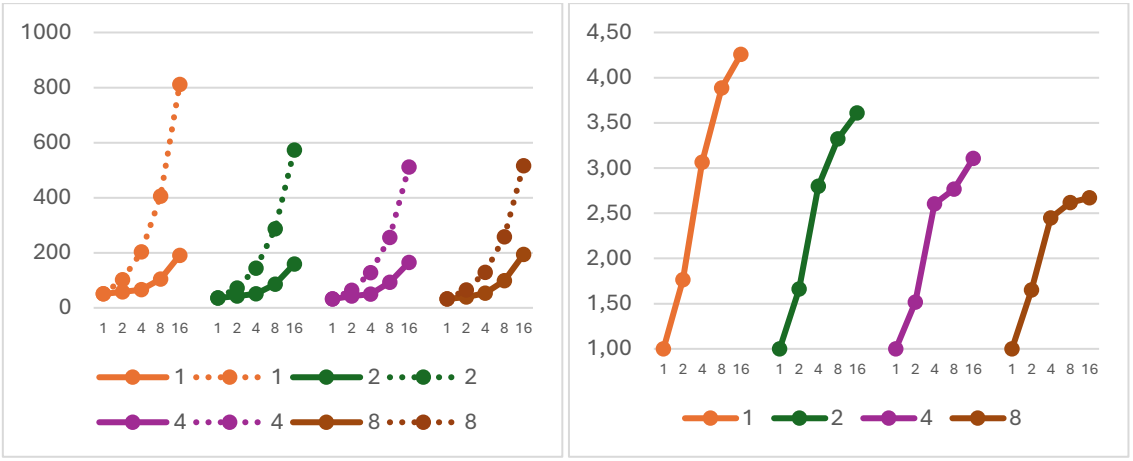
4.1.1 Διάγραμματα Χρόνου Εκτέλεσης και Επιτάχυνσης



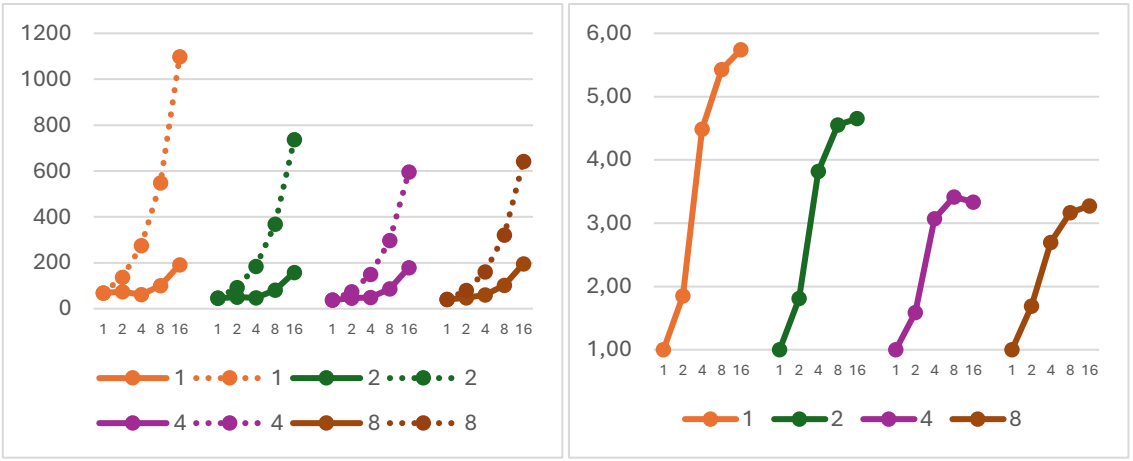
Διάγραμμα 1: Χρόνοι εκτέλεσης αρχικού προγράμματος (μία αρχική συνθήκη) και νέου προγράμματος (πολλαπλές αρχικές συνθήκες) για διαφορετικό πλήθος νημάτων για το μητρώο 3elt_dual (αριστερά). Διάγραμμα επιτάχυνσης για διαφορετικό πλήθος νημάτων των ίδιων αποτελεσμάτων (δεξιά).



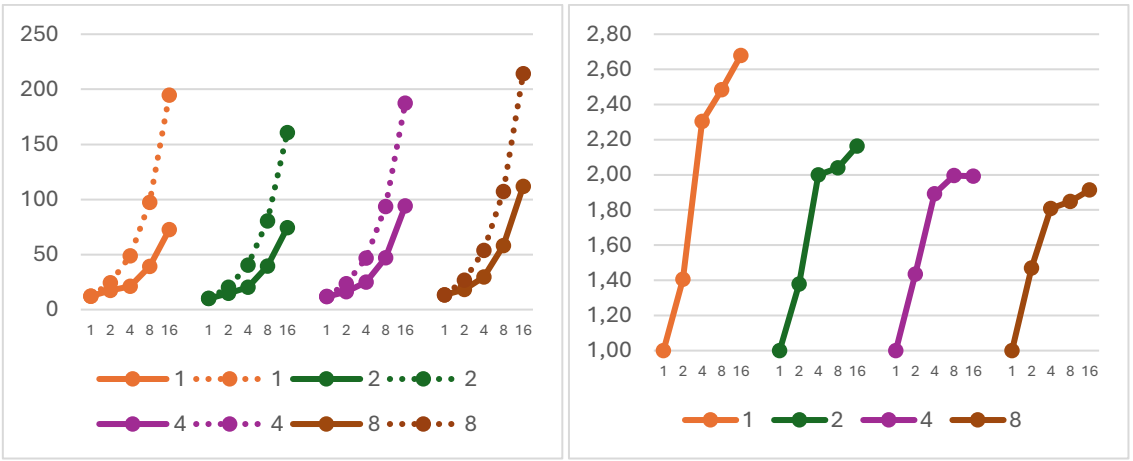
Διάγραμμα 2: Χρόνοι εκτέλεσης αρχικού προγράμματος (μία αρχική συνθήκη) και νέου προγράμματος (πολλαπλές αρχικές συνθήκες) για διαφορετικό πλήθος νημάτων για το μητρώο bloweybq (αριστερά). Διάγραμμα επιτάχυνσης για διαφορετικό πλήθος νημάτων των ίδιων αποτελεσμάτων (δεξιά).



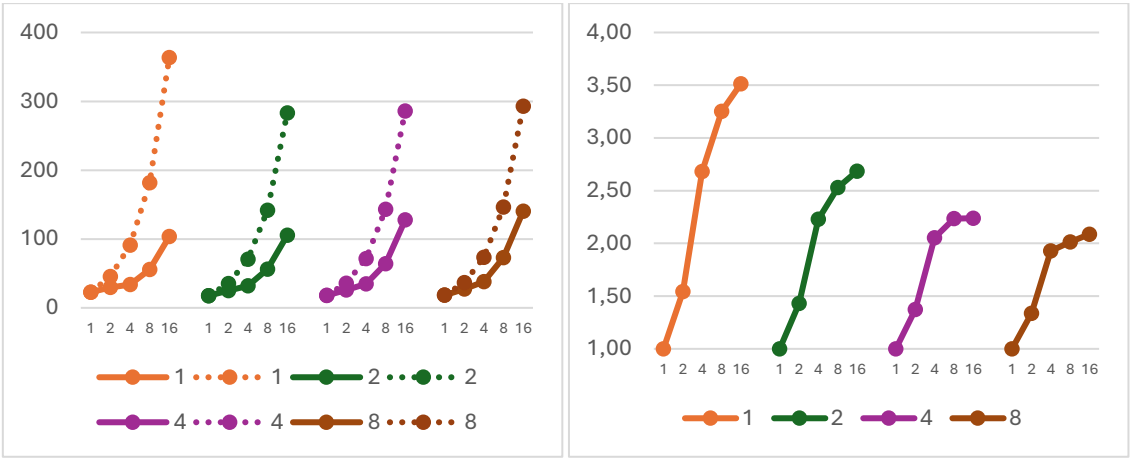
Διάγραμμα 3: Χρόνοι εκτέλεσης αρχικού προγράμματος (μία αρχική συνθήκη) και νέου προγράμματος (πολλαπλές αρχικές συνθήκες) για διαφορετικό πλήθος νημάτων για το μητρώο ca-HepPh (αριστερά). Διάγραμμα επιτάχυνσης για διαφορετικό πλήθος νημάτων των ίδιων αποτελεσμάτων (δεξιά).



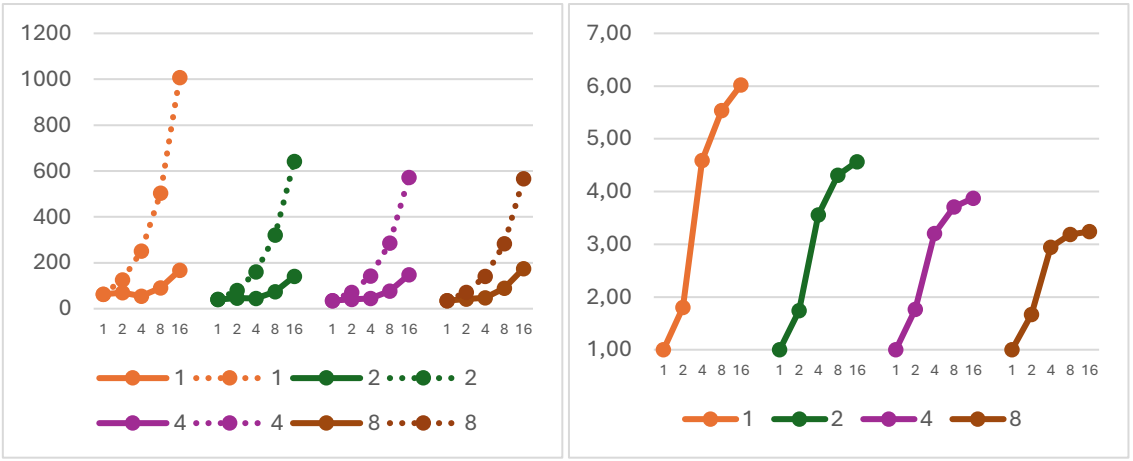
Διάγραμμα 4: Χρόνοι εκτέλεσης αρχικού προγράμματος (μία αρχική συνθήκη) και νέου προγράμματος (πολλαπλές αρχικές συνθήκες) για διαφορετικό πλήθος νημάτων για το μητρώο ex19 (αριστερά). Διάγραμμα επιτάχυνσης για διαφορετικό πλήθος νημάτων των ίδιων αποτελεσμάτων (δεξιά).



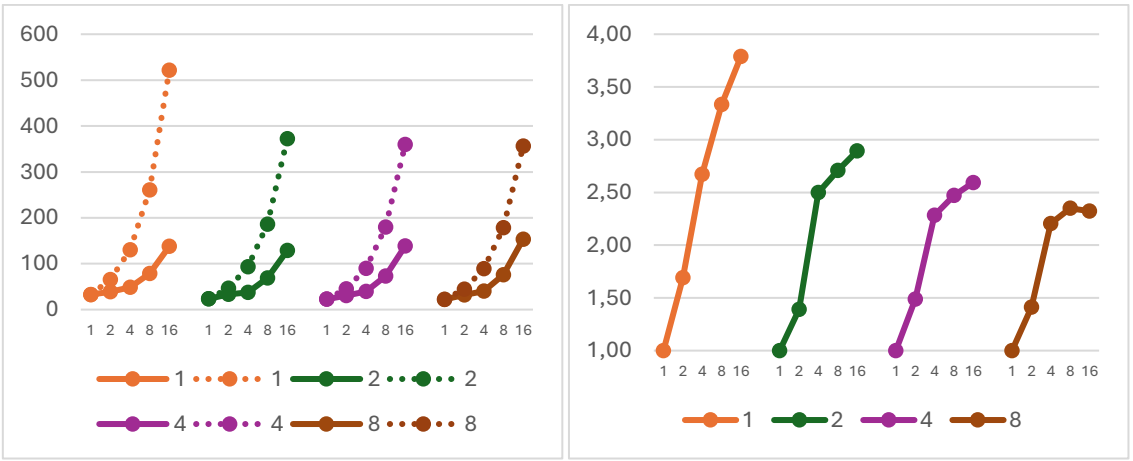
Διάγραμμα 5: Χρόνοι εκτέλεσης αρχικού προγράμματος (μία αρχική συνθήκη) και νέου προγράμματος (πολλαπλές αρχικές συνθήκες) για διαφορετικό πλήθος νημάτων για το μητρώο G66 (αριστερά). Διάγραμμα επιτάχυνσης για διαφορετικό πλήθος νημάτων των ίδιων αποτελεσμάτων (δεξιά).



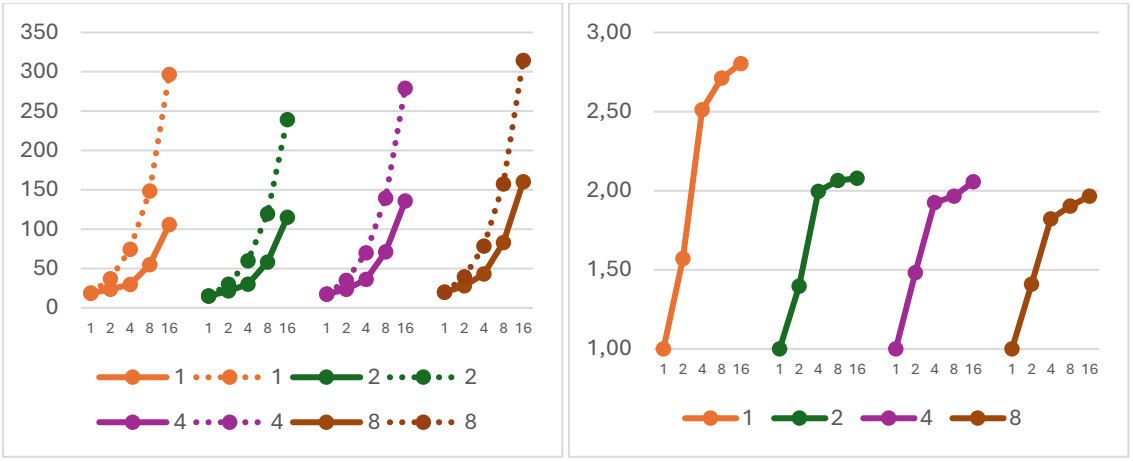
Διάγραμμα 6: Χρόνοι εκτέλεσης αρχικού προγράμματος (μία αρχική συνθήκη) και νέου προγράμματος (πολλαπλές αρχικές συνθήκες) για διαφορετικό πλήθος νημάτων για το μητρώο vsp_c-30_data_data (αριστερά). Διάγραμμα επιτάχυνσης για διαφορετικό πλήθος νημάτων των ίδιων αποτελεσμάτων (δεξιά).



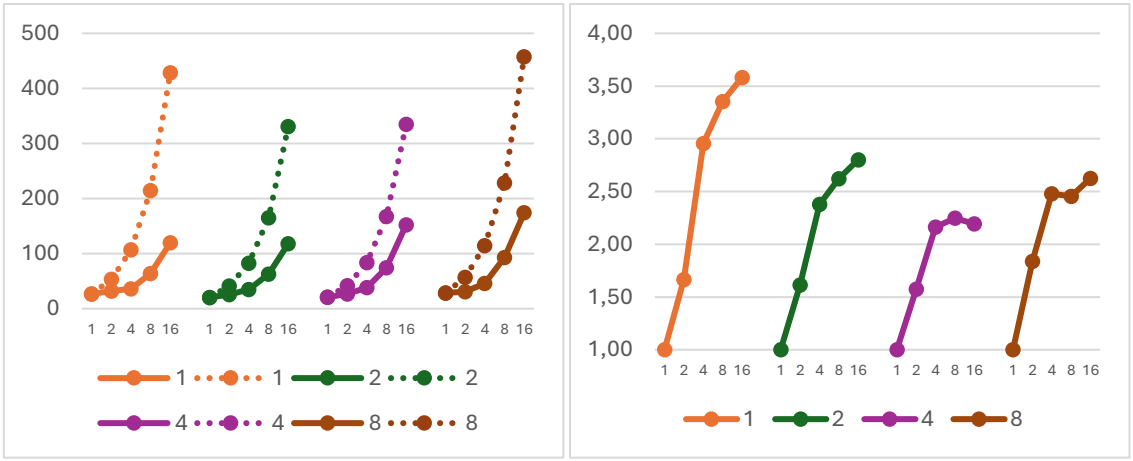
Διάγραμμα 7: Χρόνοι εκτέλεσης αρχικού προγράμματος (μία αρχική συνθήκη) και νέου προγράμματος (πολλαπλές αρχικές συνθήκες) για διαφορετικό πλήθος νημάτων για το μητρώο lhr11 (αριστερά). Διάγραμμα επιτάχυνσης για διαφορετικό πλήθος νημάτων των ίδιων αποτελεσμάτων (δεξιά).



Διάγραμμα 8: Χρόνοι εκτέλεσης αρχικού προγράμματος (μία αρχική συνθήκη) και νέου προγράμματος (πολλαπλές αρχικές συνθήκες) για διαφορετικό πλήθος νημάτων για το μητρώο kmnist_norm_10NN (αριστερά). Διάγραμμα επιτάχυνσης για διαφορετικό πλήθος νημάτων των ίδιων αποτελεσμάτων (δεξιά).



Διάγραμμα 9: Χρόνοι εκτέλεσης αρχικού προγράμματος (μία αρχική συνθήκη) και νέου προγράμματος (πολλαπλές αρχικές συνθήκες) για διαφορετικό πλήθος νημάτων για το μητρώο tuma2 (αριστερά). Διάγραμμα επιτάχυνσης για διαφορετικό πλήθος νημάτων των ίδιων αποτελεσμάτων (δεξιά).



Διάγραμμα 10: Χρόνοι εκτέλεσης αρχικού προγράμματος (μία αρχική συνθήκη) και νέου προγράμματος (πολλαπλές αρχικές συνθήκες) για διαφορετικό πλήθος νημάτων για το μητρώο ww_vref_6405 (αριστερά). Διάγραμμα επιτάχυνσης για διαφορετικό πλήθος νημάτων των ίδιων αποτελεσμάτων (δεξιά).

Το πρώτο διάγραμμα παρουσιάζει τον χρόνο εκτέλεσης (σε δευτερόλεπτα) για κάθε σενάριο με διαφορετικά πλήθη αρχικών συνθηκών και διαφορετικό πλήθος νημάτων. Παρατηρούμε ότι ο χρόνος εκτέλεσης αυξάνεται καθώς αυξάνεται ο αριθμός των αρχικών συνθηκών, κάτι που είναι αναμενόμενο λόγω της αυξημένης πολυπλοκότητας των υπολογισμών.

Αξιοσημείωτο είναι ότι, παρά την αύξηση του χρόνου εκτέλεσης, η παράλληλη επεξεργασία με περισσότερα νήματα μειώνει σημαντικά τον συνολικό χρόνο εκτέλεσης σε σύγκριση με τη μονονηματική εκτέλεση. Για παράδειγμα, βλέπουμε ότι με 16 αρχικές συνθήκες και 8 threads, ο χρόνος εκτέλεσης είναι αισθητά μικρότερος σε σχέση με την εκτέλεση με 1 ή 2 threads.

Στο δεύτερο διάγραμμα παρουσιάζεται η επιτάχυνση (speedup) που επιτυγχάνεται με την ταυτόχρονη χρήση περισσότερων αρχικών συνθηκών. Η επιτάχυνση υπολογίζεται ως ο λόγος του χρόνου εκτέλεσης με 1 αρχική συνθήκη προς τον χρόνο εκτέλεσης με περισσότερες αρχικές συνθήκες. Παρατηρούμε ότι η επιτάχυνση αυξάνεται γραμμικά μέχρι ένα σημείο, αλλά αρχίζει να σταθεροποιείται όταν το πλήθος των αρχικών συνθηκών αυξάνει.

Συμπεράσματα

Από τα διαγράμματα παρατηρούμε ότι το νέο πρόγραμμα παρουσιάζει βελτίωση στους χρόνους εκτέλεσης για όλες τις περιπτώσεις, ανεξαρτήτως του αριθμού των νημάτων και των αρχικών συνθηκών. Η βελτίωση στους χρόνους εκτέλεσης αποτυπώνεται στα διαγράμματα επιτάχυνσης (δεξιά), όπου η μικρότερη επιτάχυνση είναι περίπου 1.20x και φτάνει έως και 6.00x στις βέλτιστες περιπτώσεις.

Αυτή η βελτίωση οφείλεται στην καλύτερη αξιοποίηση του εύρους ζώνης της μνήμης στο νέο πρόγραμμα. Με τη βελτιστοποίηση του κώδικα, επιτυγχάνεται αποδοτικότερη χρήση των διαθέσιμων πόρων, μειώνοντας τον χρόνο που δαπανάται στην ανάκτηση και αποθήκευση δεδομένων από τη μνήμη. Το αποτέλεσμα είναι η ταχύτερη εκτέλεση των διαδικασιών, όπως αναμενόταν λόγω των τροποποιήσεων στον τρόπο διαχείρισης της μνήμης.

Παρατηρούμε ότι η βελτίωση στην επιτάχυνση μειώνεται καθώς αυξάνεται ο αριθμός των νημάτων. Στα διαγράμματα επιτάχυνσης (δεξιά), η αύξηση από 1 σε 2 νήματα αποδίδει υψηλή βελτίωση, αλλά η επίδοση σταδιακά μειώνεται με τη χρήση 4 ή 8 νημάτων. Αυτό οφείλεται στην αυξημένη ζήτηση για πρόσβαση στη μνήμη, η οποία απαιτεί μεγαλύτερο εύρος ζώνης καθώς τα νήματα πολλαπλασιάζονται.

Όσο περισσότερα νήματα χρησιμοποιούνται, τόσο περισσότερες αιτήσεις για ανάγνωση και εγγραφή δεδομένων πρέπει να εξυπηρετηθούν ταυτόχρονα από το σύστημα μνήμης, το οποίο οδηγεί σε συμφόρηση και μείωση της αποδοτικότητας. Η προσπέλαση της μνήμης γίνεται σημείο συμφόρησης, καθώς η υποδομή της μνήμης δεν μπορεί να καλύψει τις αυξημένες ανάγκες εύρους ζώνης που προκύπτουν από τις αιτήσεις πολλαπλών νήματα, με αποτέλεσμα η επιτάχυνση να φτάνει σε σημείο κορεσμού ή ακόμα και να μειώνεται.

Παρατηρείται ένα μοτίβο που συνδέεται με το μέγεθος του μητρώου και το πλήθος των μη μηδενικών στοιχείων του. Σε μεγαλύτερα μητρώα ή σε μητρώα με περισσότερα μη μηδενικά στοιχεία, η απόδοση επιδεινώνεται πιο γρήγορα με την αύξηση των νημάτων, λόγω των υψηλότερων απαιτήσεων για εύρος ζώνης μνήμης. Η αυξημένη ποσότητα δεδομένων που πρέπει να διακινηθεί καθιστά την παράλληλη επεξεργασία αναποτελεσματική, καθώς η μνήμη δεν μπορεί να ανταποκριθεί στις αυξημένες απαιτήσεις. Από την άλλη πλευρά, σε μικρότερα μητρώα ή σε μητρώα με λιγότερα μη μηδενικά στοιχεία, παρατηρείται βελτίωση στους χρόνους εκτέλεσης με την αύξηση των νημάτων, αλλά η απόδοση φτάνει σύντομα σε σημείο κορεσμού, καθώς ο απαιτούμενος συγχρονισμός μεταξύ νημάτων επιφέρει επιβαρύνσεις στον χρόνο εκτέλεσης..

Συμπερασματικά, η αποδοτικότητα της παράλληλης επεξεργασίας επηρεάζεται από το μέγεθος και την κατανομή των μη μηδενικών στοιχείων στα μητρώα. Όσο μεγαλύτερο και πιο πυκνά δομημένο είναι το μητρώο, τόσο περισσότερο περιορίζεται η δυνατότητα βελτίωσης των χρόνων εκτέλεσης καθώς αυξάνεται ο αριθμός των νημάτων.

Το μητρώο lhr11 αποτελεί μια ιδιαίτερη περίπτωση, καθώς επιτυγχάνει τη μεγαλύτερη επιτάχυνση με ένα νήμα σε σύγκριση με τα υπόλοιπα μητρώα. Αυτό πιθανόν αποδίδεται στη μοναδική δομή και διάταξη των μη μηδενικών στοιχείων του, που εξασφαλίζουν υψηλή αποδοτικότητα όταν η επεξεργασία πραγματοποιείται σειριακά. Με τη χρήση μόνο ενός νήματος, το σύστημα εκτελεί τις απαραίτητες προσπελάσεις στη μνήμη χωρίς την ανάγκη συγχρονισμού ή ανταγωνισμού με άλλα νήματα, γεγονός που επιτρέπει την πιο αποδοτική διαχείριση των δεδομένων.

Η συγκεκριμένη βελτίωση με ένα νήμα υποδηλώνει ότι το μητρώο lhr11 διαθέτει πιθανώς διάταξη των μη μηδενικών στοιχείων που είναι ιδανική για σειριακή επεξεργασία. Αυτό σημαίνει ότι η προσπέλαση και η εκτέλεση των υπολογισμών πραγματοποιούνται πιο αποδοτικά χωρίς την ανάγκη παράλληλης διαχείρισης, η οποία θα μπορούσε να αυξήσει τις απαιτήσεις μνήμης. Ως αποτέλεσμα, επιτυγχάνεται η μέγιστη επιτάχυνση, καθώς οι απαιτήσεις σε εύρος ζώνης μνήμης παραμένουν χαμηλές και αποφεύγονται οι συμφόρησεις που συνήθως εμφανίζονται κατά τη χρήση πολλαπλών νημάτων.

Το μητρώο 3elt_dual παρουσιάζει τη μικρότερη επιτάχυνση για ένα νήμα, πιθανόν λόγω της ιδιαίτερης δομής και κατανομής των μη μηδενικών στοιχείων του, η οποία δεν ευνοεί τη σειριακή επεξεργασία. Αυτό μπορεί να σημαίνει ότι τα δεδομένα του μητρώου είναι αραιά ή διάσπαρτα, απαιτώντας πολλαπλές προσπελάσεις μνήμης που μειώνουν την απόδοση, ειδικά όταν δεν υπάρχει δυνατότητα παράλληλης επεξεργασίας για να επιταχυνθεί ο χειρισμός των δεδομένων.

Σε αυτή την περίπτωση, το ένα νήμα αναγκάζεται να εκτελεί όλες τις προσπελάσεις μνήμης και υπολογισμούς διαδοχικά, χωρίς την υποστήριξη παράλληλων νημάτων που θα μπορούσαν να κατανείμουν το φορτίο. Αυτό δημιουργεί καθυστερήσεις, καθώς οι απαιτήσεις μνήμης πιθανόν είναι υψηλότερες λόγω των διάσπαρτων προσπελάσεων, μειώνοντας έτσι την αποδοτικότητα της εκτέλεσης με ένα μόνο νήμα και οδηγώντας σε χαμηλότερη επιτάχυνση.

Συνοπτικά, η μικρή επιτάχυνση για το μητρώο 3elt_dual με ένα νήμα οφείλεται στην αραιή ή μη ευνοϊκή διάταξη των δεδομένων, η οποία επιβαρύνει τη σειριακή εκτέλεση λόγω αυξημένων προσπελάσεων μνήμης και υπολογιστικής καθυστέρησης.

Αναλύοντας τα αποτελέσματα, παρατηρούνται ορισμένα επιπλέον μοτίβα. Κατ' αρχήν, για τα περισσότερα μητρώα, η επιτάχυνση αυξάνεται όσο αυξάνεται ο αριθμός των νημάτων, αλλά αυτή η αύξηση φτάνει σε ένα σημείο κορεσμού. Από έναν συγκεκριμένο αριθμό νημάτων και μετά (συνήθως από τέσσερα νήματα και πάνω), η επιπλέον βελτίωση μειώνεται ή ακόμη και αναστρέφεται, οδηγώντας σε σταθεροποίηση ή επιδείνωση των χρόνων εκτέλεσης. Αυτό το φαινόμενο παρατηρείται ιδιαίτερα στα μεγάλα και πυκνά μητρώα, όπου η συμφόρηση μνήμης, λόγω του αυξημένου αριθμού προσπελάσεων, μειώνει την αποδοτικότητα της παράλληλης εκτέλεσης.

Επιπλέον, τα αραιά μητρώα, δηλαδή αυτά με λιγότερα μη μηδενικά στοιχεία, τείνουν να επιτυγχάνουν καλύτερη επιτάχυνση με μικρότερα πλήθη νημάτων. Η δομή τους επιτρέπει αποδοτικότερη χρήση της μνήμης και των υπολογιστικών πόρων, καθώς απαιτούν λιγότερη επεξεργασία και μικρότερο εύρος ζώνης μνήμης. Το μοτίβο αυτό υποδηλώνει ότι η παράλληλη εκτέλεση είναι πιο αποτελεσματική σε αραιά δεδομένα, όπου τα νήματα μπορούν να εργαστούν πιο ανεξάρτητα χωρίς να δημιουργούν συμφόρηση στη μνήμη.

Σε ορισμένα μητρώα, όπως το 3elt_dual και το G66, παρατηρείται όχι μόνο σταθεροποίηση αλλά και μείωση της επιτάχυνσης καθώς αυξάνονται τα νήματα πέρα από έναν αριθμό. Αυτό πιθανόν οφείλεται στη μη ευνοϊκή διάταξη των δεδομένων, η οποία καθιστά τις παράλληλες προσπελάσεις μνήμης αναποτελεσματικές, δημιουργώντας συμφόρηση. Τα μητρώα αυτά ενδέχεται να περιέχουν δεδομένα διάσπαρτα, αυξάνοντας τις καθυστερήσεις όταν τα νήματα χρειάζεται να συγχρονιστούν.

Επιπλέον, παρατηρείται διαφορετική απόδοση μεταξύ μικρών και μεγάλων μητρώων. Τα μικρά μητρώα τείνουν να επιτυγχάνουν μεγαλύτερες βελτιώσεις με λιγότερα νήματα, ενώ τα μεγάλα μητρώα αποδίδουν καλύτερα με μεγαλύτερο πλήθος νημάτων, μέχρι να φτάσουν στο σημείο κορεσμού. Αυτό συμβαίνει επειδή τα μεγάλα μητρώα έχουν περισσότερα δεδομένα που μπορούν να διαχωριστούν και να επεξεργαστούν παράλληλα, έως ότου η μνήμη αρχίσει να περιορίζει την αποδοτικότητα.

Συνολικά, η αποδοτικότητα της παράλληλης επεξεργασίας εξαρτάται από το μέγεθος, την κατανομή των μη μηδενικών στοιχείων και τη διάταξη των δεδομένων κάθε μητρώου, καθώς και από την ικανότητα της μνήμης να υποστηρίζει παράλληλες προσπελάσεις. Ο βέλτιστος αριθμός νημάτων επηρεάζεται από αυτά τα χαρακτηριστικά, με τα αραιά μητρώα να παρουσιάζουν καλύτερη κλιμάκωση με μικρότερο πλήθος νημάτων, ενώ τα μεγάλα και πιο πυκνά μητρώα να φτάνουν γρηγορότερα σε σημείο κορεσμού καθώς αυξάνεται ο αριθμός των νημάτων.

Βιβλιογραφία

- Abeles, M. (1994). Firing rates and well-timed events. In Models of Neural Networks 2, E. Domany, K. Schulten, and J. L. van Hemmen (Eds.), Springer, New York, Chapter 3, 121–140.
- Ioannis E. Venetis, Astero Provata, Analysis of the Leaky Integrate-and-Fire neuron model for GPU implementation, Journal of Parallel and Distributed Computing, Vol. 163, pp. 1-19, 2022. DOI: https://doi.org/10.1016/j.jpdc.2022.01.021
- Adrian, E. D. (1926). The impulses produced by sensory nerve endings. Journal of Physiology (London), 61:49–72.
- Adrian, E. D. (1928). The Basis of Sensation. W. W. Norton, New York.
- Bialek, W., Rieke, F., de Ruyter van Stevenick, R. R., and Warland, D. (1991). Reading a neural code. Science, 252:1854–1857.
- Eckhorn, R., Bauer, R., Jordan, W., Brosch, M., Kruse, W., Munk, M., & Reitboeck, H. J. (1988). Coherent oscillations: A mechanism of feature linking in the visual cortex? Biol. Cybern., 60:121-130.
- Engel, A. K., König, P., and Singer, W. (1991a). Direct physiological evidence for scene segmentation by temporal coding. Proc. Natl. Acad. Sci. USA, 88:9136–9140.
- Engel, A. K., König, P., Kreiter, A. K., and Singer, W. (1991b). Interhemispheric synchronization of oscillatory neural responses in cat visual cortex. Science, 252:1177–1179.
- Ermentrout, G. B. (1996). Type I membranes, phase resetting curves, and synchrony. Neural Comput., vol. 8, pp. 979–1001.
- Ermentrout, G. B., & Kopell, N. (1986). Parabolic bursting in an excitable system coupled with a slow oscillation. SIAM J. Appl. Math., vol. 46, pp. 233–253.
- FitzHugh, R. (1961). Impulses and physiological states in theoretical models of nerve membrane. Biophys. J. 1 (1961) 445.
- FitzHugh, R. (1961). Impulses and physiological states in models of nerve membrane. Biophys. J., vol. 1, pp. 445–466.
- Gray, C. M., and Singer, W. (1989). Stimulus-specific neuronal oscillations in orientation columns of cat visual cortex. Proc. Natl. Acad. Sci. USA, 86:1698–1702.
- Gray, C. M., König, P., Engel, A. K., and Singer, W. (1989). Oscillatory responses in cat visual cortex exhibit intercolumnar synchronization which reflects global stimulus properties. Nature, 338:334–337.
- Hindmarsh, J. L., & Rose, R. M. (1982). A model of the nerve impulse using two first-order differential equations. Nature, 296:162–164.
- Hindmarsh, J. L., & Rose, R. M. (1984). A model of neuronal bursting using three coupled first-order differential equations. Proc. R. Soc. B 221 (1984) 87–102.
- Hodgkin, A. L., & Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in the nerve. J. Physiol., 177:500–544.
- Hopfield, J. J. (1995). Pattern recognition computation using action potential timing for stimulus representation. Nature, 376:33–36.
- Hoppensteadt, F. C., & Izhikevich, E. M. (1997). Weakly Connected Neural Networks. New York: Springer-Verlag.
- Hubel, D. H., and Wiesel, T. N. (1959). Receptive fields of single neurons in the cat's striate cortex. J. Physiol., 148:574–591.
- Izhikevich, E. M. (2003). Simple model of spiking neurons. IEEE Trans. Neural Networks, vol. 14, pp. 1569–1572.
- Izhikevich, E. M. (To be published). Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting.

- Latham, P. E., Richmond, B. J., Nelson, P. G., & Nirenberg, S. (2000). Intrinsic dynamics in neuronal networks. I. Theory. J. Neurophysiol., vol. 83, pp. 808–827.
- Lestienne, R. (1996). Determination of the precision of spike timing in the visual cortex of anaesthetised cats. Biological Cybernetics, 74:55–61.
- Mormann, F., Lehnertz, K., David, P., & Elger, C. E. (2000). Mean phase coherence as a measure for phase synchronization and its application to the EEG of epilepsy patients. Physica D, 144:358.
- Mountcastle, V. B. (1957). Modality and topographic properties of single neurons of cat's somatosensory cortex. Journal of Neurophysiology, 20, 408–434.
- Nagumo, J., Arimoto, A., & Yoshizawa, S. (1962). An active pulse transmission line simulating nerve axon. Proc. Inst. Radio Eng., 50:2061.
- Omelchenko, I., Omel'chenko, O., Zakharova, A., & Schöll, E. (2015). Nonlinearity of local dynamics promotes multi-chimeras. Chaos, 25:083104.
- Omelchenko, I., Zakharova, A., Hövel, P., Siebert, J., & Schöll, E. (2015). Tweezers for chimeras in small networks. Phys. Rev. Lett., 116:114101.
- Rieke, F., Warland, D., de Ruyter van Steveninck, R., & Bialek, W. (1996). Spikes - Exploring the Neural Code. MIT Press, Cambridge, MA.
- Sejnowski, T. J., Koch, C., & Churchland, P. S. (1988). Computational neuroscience. Science, 241(4871), 1299-1306.
- Shadlen, M. N., & Newsome, W. T. (1994). Noise, neural codes and cortical organization. Current Opinion in Neurobiology, 4:569–579.
- Singer, W. (1994). Models of Neural Networks 2. Springer, Berlin Heidelberg New York, pp. 141–173.
- Smith, G. D., Cox, C. L., Sherman, S. M., & Rinzel, J. (2000). Fourier analysis of sinusoidally driven thalamocortical relay neurons and a minimal integrate-and-fire-or-burst model. J. Neurophysiol., 83:588–610.
- Softky, W. R. (1995). Simple codes versus efficient codes. Current Opinion in Neurobiology, 5:239–247.
- Thorpe, S., Fize, D., & Marlot, C. (1996). Speed of processing in the human visual system. Nature, 381:520–522.
- Hunter, Kevin & Spracklen, Lawrence & Ahmad, Subutai. (2021). Two Sparsities Are Better Than One: Unlocking the Performance Benefits of Sparse-Sparse Networks. 10.48550/arXiv.2112.13896.

Παραρτήματα

```
#include <stdio.h>
#include <math.h>
#include <limits.h>
#include <getopt.h>
#include <stdlib.h>
#include <sys/time.h>

#include "../../spm.h"

#define DEF_DT        (1.0e-04)
#define DEF_MU        (1.0)
#define DEF_U_TH      (0.98)
#define DEF_U_REST    (0.0)
#define DEF_S_MIN     (0.7)
#define DEF_S_MAX     (0.7)
#define DEF_SAMPLING      (100L)
#define DEF_SIM_TIME      (20.0)
#define DEF_T_TRANSIENT   (-1.0)
#define DEF_T_REFRACTORY  (0.0)
#define DEF_SPFORMAT      (CSR)
#define DEF_NUM_INIT_CONDITIONS  (1)  // Default value of numOfInitConditions

static struct option long_options[] =
{
 {"dt",       required_argument, 0, 'a'},
 {"mu",       required_argument, 0, 'b'},
 {"uth",      required_argument, 0, 'c'},
 {"urest",    required_argument, 0, 'd'},
 {"time",     required_argument, 0, 'e'},
 {"transient",  required_argument, 0, 'f'},
 {"refractory", required_argument, 0, 'g'},
 {"s_min",    required_argument, 0, 'h'},
 {"s_max",    required_argument, 0, 'i'},
 {"sampling",  required_argument, 0, 'j'},
 {"spfile",   required_argument, 0, 'k'},
 {"spformat",  required_argument, 0, 'l'},
 {"numOfInitConditions", required_argument, 0, 'm'},  // New option for numOfInitConditions
 {0, 0, 0, 0}
};

int main(int argc, char *argv[])
{
   FILE        *output1, *output2;
```

```
INDEX_TYPE     n;
long        numOfInitConditions =  DEF_NUM_INIT_CONDITIONS;  // Set default value
long        i;
long        it;
long        sampling;
double       dt;
double       sim_time;
double       t_transient;
double       t_refractory;
long        total_time_steps;
long        spformat;
long        *num_of_neighs, *omega1, *t_refractory_until_it;
long        t_refractory_time_steps;
double       u_th;
double       u_rest;
double       mu;
double       s_min;
double       s_max;
double       *u, *u_next, *omega, *sum_s, *temp;
struct timeval     global_start, global_end, IO_start, IO_end;
double       global_usec, IO_usec = 0.0;
int       c, option_index;
char         *end_ptr;
char         *filename0 = NULL, filename1[NAME_MAX + 1], filename2[NAME_MAX + 1];
struct COOMatrix   *s_coo;
sparse_matrix_t    s;
sparse_status_t    status;
struct matrix_descr descr;

dt        = DEF_DT;
mu        = DEF_MU;
u_th      = DEF_U_TH;
u_rest    = DEF_U_REST;
s_min     = DEF_S_MIN;
s_max     = DEF_S_MAX;
sampling   = DEF_SAMPLING;
sim_time   = DEF_SIM_TIME;
t_transient  = DEF_T_TRANSIENT;
t_refractory = DEF_T_REFRACTORY;
spformat   = DEF_SPFORMAT;

while (1) {
   c = getopt_long (argc, argv, "", long_options, &option_index);
```

```
    if (c == -1) {
      break;
    }

    switch (c) {
      case 'a':
        dt = strtod(optarg, &end_ptr);
        if (*end_ptr != '\0') {
          printf("Option \"%s\": Invalid argument \"%s\".\n", long_options[option_index].name, optarg);
          exit(1);
        }
        if (dt <= 0.0) {
          printf("Option \"%s\": \"dt\" must be larger than zero.\n", long_options[option_index].name);
          exit(1);
        }
        break;
      case 'b':
        mu = strtod(optarg, &end_ptr);
        if (*end_ptr != '\0') {
          printf("Option \"%s\": Invalid argument \"%s\".\n", long_options[option_index].name, optarg);
          exit(1);
        }
        if (mu <= 0.0) {
          printf("Option \"%s\": \"mu\" must be larger than zero.\n", long_options[option_index].name);
          exit(1);
        }
        break;
      case 'c':
        u_th = strtod(optarg, &end_ptr);
        if (*end_ptr != '\0') {
          printf("Option \"%s\": Invalid argument \"%s\".\n", long_options[option_index].name, optarg);
          exit(1);
        }
        if (u_th <= 0.0) {
          printf("Option \"%s\": \"uth\" must be larger than zero.\n", long_options[option_index].name);
          exit(1);
        }
        break;
      case 'd':
        u_rest = strtod(optarg, &end_ptr);
        if (*end_ptr != '\0') {
          printf("Option \"%s\": Invalid argument \"%s\".\n", long_options[option_index].name, optarg);
          exit(1);
        }
```

```
            break;
        case 'e':
            sim_time = strtod(optarg, &end_ptr);
            if (*end_ptr != '\0') {
                printf("Option \"%s\": Invalid argument \"%s\".\n", long_options[option_index].name, optarg);
                exit(1);
            }
            if (sim_time <= 0.0) {
                printf("Option \"%s\": Total simulation time must be larger than zero.\n",
long_options[option_index].name);
                exit(1);
            }
            break;
        case 'f':
            t_transient = strtod(optarg, &end_ptr);
            if (*end_ptr != '\0') {
                printf("Option \"%s\": Invalid argument \"%s\".\n", long_options[option_index].name, optarg);
                exit(1);
            }
            if (t_transient < 0.0) {
                printf("Option \"%s\": \"t_transient\" must be larger or equal than zero.\n",
long_options[option_index].name);
                exit(1);
            }
            break;
        case 'g':
            t_refractory = strtod(optarg, &end_ptr);
            if (*end_ptr != '\0') {
                printf("Option \"%s\": Invalid argument \"%s\".\n", long_options[option_index].name, optarg);
                exit(1);
            }
            if (t_refractory < 0.0) {
                printf("Option \"%s\": \"t_refractory\" must be larger or equal than zero.\n",
long_options[option_index].name);
                exit(1);
            }
            break;
        case 'h':
            s_min = strtod(optarg, &end_ptr);
            if (*end_ptr != '\0') {
                printf("Option \"%s\": Invalid argument \"%s\".\n", long_options[option_index].name, optarg);
                exit(1);
            }
            break;
        case 'i':
```

```
            s_max = strtod(optarg, &end_ptr);
            if (*end_ptr != '\0') {
              printf("Option \"%s\": Invalid argument \"%s\".\n", long_options[option_index].name, optarg);
              exit(1);
            }
            break;
          case 'j':
            sampling = strtol(optarg, &end_ptr, 10);
            if (*end_ptr != '\0') {
              printf("Option \"%s\": Invalid argument \"%s\".\n", long_options[option_index].name, optarg);
              exit(1);
            }
            if (sampling < 1) {
              printf("Option \"%s\": \"sampling\" must be larger or equal than one.\n",
long_options[option_index].name);
              exit(1);
            }
            break;
          case 'k':
            filename0 = optarg;
            break;
          case 'l':
            if (strcmp(optarg, "COO") == 0) {
              spformat = COO;
            } else if (strcmp(optarg, "CSR") == 0) {
              spformat = CSR;
            } else if (strcmp(optarg, "BSR") == 0) {
              spformat = BSR;
            } else {
              printf("Option \"%s\": Acceptable values are ", long_options[option_index].name);
              for (i = 0; i < MAX_SPFORMAT; i++) {
                printf("%s, ", spformat_name[i]);
              }
              printf("%s.\n", spformat_name[MAX_SPFORMAT]);
              exit(1);
            }
            break;
          case 'm':  // New case for numOfInitConditions
            numOfInitConditions = strtol(optarg, NULL, 10);
            if (numOfInitConditions <= 0) {
              printf("Option \"%s\": numOfInitConditions must be larger than zero.\n",
long_options[option_index].name);
              exit(1);
            }
```

```
            break;
        case '?':
        default:
            exit(1);
            break;
    }
}

if (optind != argc) {
    printf("Unknown option \"%s\".\n", argv[optind]);
    exit(1);
}

if (filename0 == NULL) {
    printf("Please provide an input file for the connectivity matrix.\n");
    exit(1);
}

if (s_min > s_max) {
    printf("s_min (%17.15f) must be smaller or equal than s_max (%17.15f).\n", s_min, s_max);
    exit(1);
}

if (t_transient == DEF_T_TRANSIENT) {
    t_transient = sim_time / 2.0;
}
t_refractory_time_steps = t_refractory / dt;
total_time_steps = sim_time / dt;

s_coo = readCOOMatrix(filename0, true, s_min, s_max);

if (s_coo->rows != s_coo->cols) {
    printf("Connectivity matrix must be square.\n");
    exit(1);
}
n = s_coo->rows;

sum_s = (double *)calloc(n, sizeof(double));
if (sum_s == NULL) {
    printf("Could not allocate memory for \"sum_s\".\n");
    exit(1);
}

num_of_neighs = (long *)calloc(n, sizeof(long));
```

```
  if (num_of_neighs == NULL) {
    printf("Could not allocate memory for \"num_of_neighs\".\n");
    exit(1);
  }

  for (i = 0; i < s_coo->nnz; i++) {
    sum_s[s_coo->rowindx[i]] += s_coo->values[i];
    /*
     * If there is a connection between neuron i and another neuron
     * update the number of neighbours for neuron i.
     */
    if ((s_coo->rowindx[i] != s_coo->colindx[i]) && (s_coo->values[i] != 0.0)) {
      num_of_neighs[s_coo->rowindx[i]]++;
    }
  }

  for (i = 0; i < n; i++) {
    if (num_of_neighs[i] == 0) {
      num_of_neighs[i]++;
    }
//    printf("%10.6f  %ld\n", sum_s[i], num_of_neighs[i]);
  }

  status = mkl_sparse_d_create_coo(&s, SPARSE_INDEX_BASE_ZERO, s_coo->rows, s_coo->cols, s_coo-
>nnz, s_coo->rowindx, s_coo->colindx, s_coo->values);
  if (status != SPARSE_STATUS_SUCCESS) {
    printf("Creation of sparse matrix in COO format failed.\n");
    exit(1);
  }

  if (spformat == CSR) {
    status = mkl_sparse_convert_csr(s, SPARSE_OPERATION_NON_TRANSPOSE, &s);
    if (status != SPARSE_STATUS_SUCCESS) {
      printf("Conversion of sparse matrix from COO to CSR format failed.\n");
      exit(1);
    }
  } else if (spformat == BSR) {
    /*
     * The size of the blocks used seems to be a good compromise for most cases,
     * but further fine-tuning can be tested.
     */
    status = mkl_sparse_convert_bsr(s, n / 100, SPARSE_LAYOUT_ROW_MAJOR,
SPARSE_OPERATION_NON_TRANSPOSE, &s);
    if (status != SPARSE_STATUS_SUCCESS) {
```

```
        printf("Conversion of sparse matrix from COO to BSR format failed.\n");
        exit(1);
      }
    }

    descr.type = SPARSE_MATRIX_TYPE_GENERAL;
    descr.diag = SPARSE_DIAG_NON_UNIT;
    mkl_sparse_set_mv_hint (s, SPARSE_OPERATION_NON_TRANSPOSE, descr, total_time_steps);
    mkl_sparse_set_memory_hint(s, SPARSE_MEMORY_AGGRESSIVE);
    mkl_sparse_optimize(s);

    printf("Running simulation with following parameters:\n");
    printf("  Number of neurons    : %d\n", n);
    printf("  Connectivity matrix  : %s\n", filename0);
    printf("  Sparse matrix format : %s\n", spformat_name[spformat]);
    printf("  Simulation time      : %08.4f seconds (%ld time steps)\n", sim_time, total_time_steps);
    printf("  Transient time       : %08.4f seconds\n", t_transient);
    printf("  Refractory time      : %08.4f seconds\n", t_refractory);
    printf("  Sampling rate        : %ld time steps\n", sampling);
    printf("  dt                   : %.1e seconds \n", dt);
    printf("  mu                   : %17.15f\n", mu);
    printf("  u_th                 : %17.15f\n", u_th);
    printf("  u_rest               : %17.15f\n", u_rest);
    printf("  s_min                : %17.15f\n", s_min);
    printf("  s_max                : %17.15f\n", s_max);

    i = snprintf(filename1, NAME_MAX + 1,
"spacetime_%07d_%+6.4f_%+6.4f_%8.6f_%08.4f_%08.4f_%8.6f_%8.6f_%8.6f_%07ld.out",
                 n, s_min, s_max, dt, sim_time, t_transient, mu, u_th, u_rest, sampling);
    if (i >= NAME_MAX + 1) {
      printf("Filename to store space-time information is too long.\n");
      exit(1);
    }

    i = snprintf(filename2, NAME_MAX + 1,
"omega_%07d_%+6.4f_%+6.4f_%8.6f_%08.4f_%08.4f_%8.6f_%8.6f_%8.6f_%07ld.out",
                 n, s_min, s_max, dt, sim_time, t_transient, mu, u_th, u_rest, sampling);
    if (i >= NAME_MAX + 1) {
      printf("Filename to store omega information is too long.\n");
      exit(1);
    }

    output1 = fopen(filename1, "w");
    if (output1 == NULL) {
```

```
    printf("Could not open file \"%s\".\n", filename1);
    exit(1);
  }

  output2 = fopen(filename2, "w");
  if (output2 == NULL) {
    printf("Could not open file \"%s\".\n", filename2);
    exit(1);
  }

  u = (double *)calloc(n*numOfInitConditions, sizeof(double));
  if (u == NULL) {
    printf("Could not allocate memory for \"u\".\n");
    exit(1);
  }

  u_next = (double *)calloc(n*numOfInitConditions, sizeof(double));
  if (u_next == NULL) {
    printf("Could not allocate memory for \"u_next\".\n");
    exit(1);
  }

  omega = (double *)calloc(n*numOfInitConditions, sizeof(double));
  if (omega == NULL) {
    printf("Could not allocate memory for \"omega\".\n");
    exit(1);
  }

  omega1 = (long *)calloc(n*numOfInitConditions, sizeof(long));
  if (omega1 == NULL) {
    printf("Could not allocate memory for \"omega1\".\n");
    exit(1);
  }

  t_refractory_until_it = (long *)calloc(n*numOfInitConditions, sizeof(long));
  if (t_refractory_until_it == NULL) {
    printf("Could not allocate memory for \"t_refractory_until_it\".\n");
    exit(1);
  }

  /*
   * Initialize elements of array u[i] with random numbers in the range [0, u_th).
  */
```

```
  for (long j = 0; j < numOfInitConditions; j++) {
    srand48(j);
    for (i = 0; i < n; i++) {
      u[i * numOfInitConditions + j] = u_th * drand48();
    }
  }

  /*
   * Print initial values of potential to output file.
   */
  //fprintf(output1, "0\t");
  for (i = 0; i < n; i++) {
    for (long j = 0; j < numOfInitConditions; j++ ){
    //fprintf(output1, "%19.15f", u[i * numOfInitConditions + j]);
    }
  }
  //fprintf(output1, "\n");

  /*
   * Temporal iteration.
   */
  gettimeofday(&global_start, NULL);
  for (it = 0; it < total_time_steps; it++) {

    mkl_sparse_d_mm(SPARSE_OPERATION_NON_TRANSPOSE, 1.0, s, descr,
SPARSE_LAYOUT_ROW_MAJOR, u, numOfInitConditions, numOfInitConditions, 0.0, u_next,
numOfInitConditions);

    /*
     * Iteration over neurons.
     */
    for (i = 0; i < n; i++) {
      for (long j = 0; j < numOfInitConditions; j++ ){
      /*
       * Keep at u_rest, if still in refractory period.
       */
      if (it < t_refractory_until_it[i * numOfInitConditions + j]) {
        u_next[i * numOfInitConditions + j] = u_rest;
      } else {
        /*
         * Update network elements and set u[i] = u_rest if u[i] > u_th
         */
        u_next[i * numOfInitConditions + j] = u[i * numOfInitConditions + j] + dt * (mu - u[i *
numOfInitConditions + j]) + dt * (u_next[i * numOfInitConditions + j] - sum_s[i] * u[i * numOfInitConditions + j])
/ num_of_neighs[i];
```

```
          if (u_next[i * numOfInitConditions + j] > u_th) {
            t_refractory_until_it[i * numOfInitConditions + j] = it + t_refractory_time_steps;
            u_next[i * numOfInitConditions + j] = u_rest;
            /*
             * Calculate omega's.
             */
            if (it * dt >= t_transient) {
              omega1[i * numOfInitConditions + j]++;
            }
          }
        }
      }
    }

    /*
     * Exchange u and u_next
     */
    temp = u;
    u = u_next;
    u_next = temp;

    /*
     * Print out of results.
     */
    if ((it + 1) % sampling == 0) {
      //printf("Time is %ld\n", it + 1);

      gettimeofday(&IO_start, NULL);
      //fprintf(output1, "%ld\t", it + 1);
      for (i = 0; i < n; i++) {
        for (long j = 0; j < numOfInitConditions; j++){
        //fprintf(output1, "%19.15f", u[i * numOfInitConditions + j]);
        }
      }
      //fprintf(output1, "\n");

      if (it * dt > t_transient) {
        //fprintf(output2, "%ld\t", it + 1);
        for (i = 0; i < n; i++) {
          for (long j = 0; j < numOfInitConditions; j++){
          omega[i * numOfInitConditions + j] = 2.0 * M_PI * omega1[i * numOfInitConditions + j] / (it * dt -
t_transient);
          //fprintf(output2, "%19.15f", omega[i * numOfInitConditions + j]);
```

```
            }
          }
          //fprintf(output2, "\n");
        }
        gettimeofday(&IO_end, NULL);
        IO_usec += ((IO_end.tv_sec - IO_start.tv_sec) * 1000000.0 + (IO_end.tv_usec - IO_start.tv_usec));
      }
    }
    gettimeofday(&global_end, NULL);
    global_usec = ((global_end.tv_sec - global_start.tv_sec) * 1000000.0 + (global_end.tv_usec -
global_start.tv_usec));

    printf("Time for calculations = %13.6f sec\n", (global_usec - IO_usec) / 1000000.0);
    printf("Time for I/O          = %13.6f sec\n", IO_usec / 1000000.0);
    printf("Total execution time  = %13.6f sec\n", global_usec / 1000000.0);

    fclose(output1);
    fclose(output2);

    return 0;
}
```