ΠΑΝΕΠΙΣΤΗΜΙΟ
ΙΩΑΝΝΙΝΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Π.Μ.Σ. ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΔΙΚΤΥΩΝ

ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΚΑΤΑΝΟΜΗ ΦΟΡΤΟΥ ΕΡΓΑΣΙΑΣ ΣΕ ΠΕΡΙΒΑΛΛΟΝ FOG COMPUTING

Στέφανος Νίκου

Επιβλέπουσα: Σπυριδούλα Μαργαρίτη

Άρτα, Ιούλιος, 2025

WORKLOAD ALLOCATION IN A FOG COMPUTING ENVIRONMENT

Εγκρίθηκε από τριμελή εξεταστική επιτροπή

Άρτα, 30/9/2024

ΕΠΙΤΡΟΠΗ ΑΞΙΟΛΟΓΗΣΗΣ

1. Επιβλέπουσα

Σπυριδούλα Μαργαρίτη,

2. Μέλος επιτροπής

Ελευθέριος Στεργίου,

3. Μέλος επιτροπής

Χρήστος Γκόγκος

Δήλωση μη λογοκλοπής

Δηλώνω υπεύθυνα και γνωρίζοντας τις κυρώσεις του Ν. 2121/1993 περί Πνευματικής Ιδιοκτησίας, ότι η παρούσα μεταπτυχιακή εργασία είναι εξ ολοκλήρου αποτέλεσμα δικής μου ερευνητικής εργασίας, δεν αποτελεί προϊόν αντιγραφής ούτε προέρχεται από ανάθεση σε τρίτους. Όλες οι πηγές που χρησιμοποιήθηκαν (κάθε είδους, μορφής και προέλευσης) για τη συγγραφή της περιλαμβάνονται στη βιβλιογραφία.

Νίκου, Στέφανος

Υπογραφή

ΠΕΡΙΛΗΨΗ

Η παρούσα διπλωματική εργασία διερευνά το πρόβλημα της κατανομής φόρτου εργασίας σε περιβάλλοντα fog computing, προτείνοντας δύο συμπληρωματικές μεθόδους κατανομής: τη μέθοδο Neighbor-Aware και τη μέθοδο βασισμένη στον αλγόριθμο Particle Swarm Optimization (PSO). Αναπτύχθηκε το πρόγραμμα προσομοίωσης FogSim-NX για την αξιολόγηση των προτεινόμενων μεθόδων σε διαφορετικά σενάρια μεγέθους δικτύου και πλήθους εφαρμογών.

Η πειραματική αξιολόγηση έδειξε ότι η μέθοδος βασισμένη σε PSO επιτυγχάνει υψηλότερη ποιότητα κατανομής με 8-25% χαμηλότερη κατανάλωση ενέργειας, 12-31% μειωμένο λειτουργικό κόστος και σημαντικά βελτιωμένη απόδοση για εφαρμογές ευαίσθητες στην καθυστέρηση, με το κόστος όμως αυξημένης υπολογιστικής πολυπλοκότητας. Η μέθοδος Neighbor-Aware, αντίθετα, προσφέρει υπολογιστική αποδοτικότητα και ταχεία ανταπόκριση, καθιστώντας την καταλληλότερη για δυναμικά περιβάλλοντα με αυστηρούς χρονικούς περιορισμούς.

Παρουσιάζεται επίσης μια θεωρητική μελέτη περίπτωσης fog computing σε έξυπνη πόλη, που αναδεικνύει την πρακτική εφαρμογή των προτεινόμενων μεθόδων και προτείνει υβριδικές προσεγγίσεις που συνδυάζουν τα πλεονεκτήματά τους. Η εργασία καταλήγει ότι η επιλογή μεθόδου κατανομής πρέπει να καθοδηγείται από τις συγκεκριμένες απαιτήσεις του περιβάλλοντος ανάπτυξης, με τους κόμβους μεσαίου επιπέδου (district fog) να προσφέρουν την καλύτερη αναλογία απόδοσης-κόστους για πολλές εφαρμογές.

Οι προτεινόμενες μέθοδοι και το πλαίσιο FogSim-NX συμβάλλουν στην αποτελεσματικότερη διαχείριση πόρων σε κατανεμημένα συστήματα υπολογιστικού νέφους, εξισορροπώντας απόδοση, ενεργειακή αποδοτικότητα και λειτουργικό κόστος.

Λέξεις-κλειδιά: Fog Computing, Κατανομή Φόρτου Εργασίας , Αλγόριθμος Neighbor-Aware, Αλγόριθμος Particle Swarm Optimization, Διαχείριση Πόρων, Έξυπνες Πόλεις

ABSTRACT

This thesis explores the workload allocation problem in fog computing environments, proposing two complementary allocation methods: the Neighbor-Aware method and the Particle Swarm Optimization (PSO) based method. The FogSim-NX simulation framework was developed to evaluate the proposed methods across different network sizes and application loads.

Experimental evaluation demonstrated that the PSO-based method achieves higher allocation quality with 8-25% lower energy consumption, 12-31% reduced operational cost, and significantly improved performance for latency-sensitive applications, albeit at the cost of increased computational complexity. The Neighbor-Aware method, conversely, offers computational efficiency and rapid response, making it more suitable for dynamic environments with strict time constraints.

A theoretical case study of fog computing in a smart city environment is also presented, highlighting the practical application of the proposed methods and suggesting hybrid approaches that combine their advantages. The thesis concludes that the choice of allocation method should be guided by the specific requirements of the deployment environment, with mid-level nodes (district fog) offering the best performance-to-cost ratio for many applications.

The proposed methods and the FogSim-NX framework contribute to more effective resource management in distributed cloud computing systems, balancing performance, energy efficiency, and operational cost.

Keywords: Fog Computing, Workload Allocation, Neighbor-Aware Algorithm, Particle Swarm Optimization Algorithm, Resource Management, Smart Cities

ΑΠΟΔΟΣΗ ΟΡΩΝ / ΓΛΩΣΣΑΡΙΟ

Αγγλικός Όρος	Ελληνική Απόδοση
Fog Computing	Υπολογιστικό Νέφος Ομίχλης
Cloud Computing	Υπολογιστικό Νέφος
Workload Allocation	Κατανομή Φόρτου Εργασίας
Neighbor-Aware Allocation	Κατανομή με Επίγνωση Γειτνίασης
Particle Swarm Optimization (PSO)	Βελτιστοποίηση Σμήνους Σωματιδίων
Internet of Things (IoT)	Διαδίκτυο των Πραγμάτων
Edge Computing	Υπολογιστική Άκρου
Latency	Καθυστέρηση
Makespan	Συνολικός Χρόνος Εκτέλεσης
Quality of Service (QoS)	Ποιότητα Υπηρεσίας
Bandwidth	Εύρος Ζώνης
Smart City	Έξυπνη Πόλη
Gateway Node	Κόμβος Πύλης
Resource Allocation	Κατανομή Πόρων
Workload	Φόρτος Εργασίας
Multi-objective Optimization	Βελτιστοποίηση Πολλαπλών Στόχων
Energy Efficiency	Ενεργειακή Αποδοτικότητα
Operational Cost	Λειτουργικό Κόστος
Resource Utilization	Αξιοποίηση Πόρων
Simulation Framework	Πρόγραμμα προσομοίωσης

Πίνακας περιεχομένων

1 Εισαγωγή 1
1.1 Υπόβαθρο και Κίνητρο 1
1.2 Διατύπωση Προβλήματος 1
Σκοπός αυτής της εργασίας είναι να διερευνήσει τα παραπάνω πρόβλημα και να εξετάσει πιθανές λύσεις αποδοτικής κατανομής φόρτου εργασίας στους κόμβους fog. Συγκεκριμένα, επιλέγονται δύο βασικοί αλγόριθμοι όπως ο Neighbor-Aware και ο PSO. 2
1.3 Ερευνητικοί Στόχοι 2
2 Θεωρητικό Υπόβαθρο και Βιβλιογραφική Ανασκόπηση 4
2.1 Θεμελιώδεις Αρχές του Fog Computing 4
2.1.1 Ορισμός και Χαρακτηριστικά 4
2.1.2 Σύγκριση με Cloud Computing 5
2.1.3 Αρχιτεκτονική Fog Computing 5
2.2 Κατανομή Φόρτου Εργασίας σε περιβάλλον Fog Computing 6
2.2.1 Προκλήσεις Διαχείρισης Πόρων 6
2.2.2 Χαρακτηριστικά Φόρτου Εργασίας σε Περιβάλλοντα Fog 7
2.3 Τεχνικές Κατανομής Πόρων 8
2.3.1 Κεντρικές έναντι Κατανεμημένων Προσεγγίσεων 8
2.3.2 Ευρετικές Μέθοδοι 8
2.3.3 Μέθοδοι Βασισμένες σε Βελτιστοποίηση 9
2.4 Σχετικές Εργασίες 9
2.4.1 Στρατηγικές Κατανομής με Επίγνωση Γειτνίασης (Neighbor-Aware) 9
2.4.2 Τεχνικές Βελτιστοποίησης Εμπνευσμένες από τη Βιολογία 10
2.4.3 Particle Swarm Optimization στην Κατανομή Πόρων 11
3 Μέθοδοι Κατανομής Φόρτου Εργασίας 13
3.1 Μέθοδος Κατανομής Neighbor-Aware 13
3.1.1 Θεωρητικό Υπόβαθρο Μεθόδου Neighbor-Aware 13
3.1.2 Σχεδιασμός Αλγορίθμου 13
3.1.3 Λειτουργική Ροή 14
3.2 Κατανομή Particle Swarm Optimization (PSO) 16
3.2.1 Θεωρητικό Υπόβαθρο Αλγορίθμου PSO 16
3.2.2 Ρύθμιση Παραμέτρων 16
3.3 Συγκριτική Ανάλυση Μεθόδων Κατανομής 18

3.3.1 Αλγοριθμική Πολυπλοκότητα 18
3.3.2 Διαδικασία Λήψης Αποφάσεων 20
3.3.3 Διατύπωση Αντικειμενικής Συνάρτησης 21
3.3.4 Πλεονεκτήματα και Περιορισμοί 22
3.3.5 Σύγκριση με Άλλες Προσεγγίσεις Ευφυούς Κατανομής 23
4 Μοντελοποίηση του Προβλήματος Κατανομής Φόρτου Εργασίας 24
4.1 Εισαγωγή 24
4.2 Μοντελοποίηση του Περιβάλλοντος Fog Computing 25
4.2.1 Μεγέθη και συμβολισμοί του μοντέλου 25
4.2.2 Χαρακτηριστικά Κόμβων 27
4.2.3 Χαρακτηριστικά Συνδέσεων 27
4.3 Αναπαράσταση Τοπολογίας Δικτύου 27
4.4 Μοντελοποίηση Εφαρμογών 28
4.5 Αναλυτική Μοντελοποίηση του Προβλήματος Κατανομής 28
4.5.1 Μεταβλητές Απόφασης 28
4.5.2 Περιορισμοί 29
4.5.3 Υπολογισμοί Παραμέτρων 30
4.5.4 Πολυπλοκότητα του Προβλήματος 31
4.5.5 Σενάρια Προσομοίωσης 31
4.6 Μοντελοποίηση Μεθόδων Κατανομής 33
4.6.1 Μοντέλο Μεθόδου Neighbor-Aware 33
4.6.2 Μοντέλο Μεθόδου PSO 33
4.7 Συμπεράσματα Μοντελοποίησης 34
5 FogSim-NX: Πρόγραμμα Προσομοίωσης Fog Computing 35
5.1 Το Πλαίσιο Προσομοίωσης 35
5.2 Υλοποίηση FogSim-NX 35
5.3 Αρχιτεκτονική Συστήματος 36
5.3.1 Αρχές Σχεδιασμού 38
5.3.2 Τεχνολογίες Υλοποίησης 38
5.4 Συστατικά Στοιχεία Προσομοιωτή 38
5.4.1 Δημιουργία Τοπολογίας 38
5.4.2 Ταξινόμηση και Χαρακτηριστικά Κόμβων 40

5.4.3 Μοντελοποίηση Εφαρμογών 41
5.4.4 Μετρικές Απόδοσης 42
5.5 Λεπτομέρειες Υλοποίησης 43
5.5.1 Διαχείριση Τοπολογίας Βασισμένη στο NetworkX 43
5.5.2 Παράμετροι Διαμόρφωσης 43
5.5.3 Ροή Εργασίας Προσομοίωσης 44
5.6 Υλοποίηση αλγορίθμων 46
5.6.1 Υλοποίηση Neighbor-Aware 46
5.6.2 Υλοποίηση PSO 48
6 Πειραματική Αξιολόγηση 51
6.1 Περιβάλλον Προσομοίωσης 51
6.1.1 Περιγραφή Σεναρίων 51
6.1.2 Μετρικές Απόδοσης 52
6.1.3 Μεθοδολογία Αξιολόγησης 53
6.2 Παράμετροι Διαμόρφωσης Προσομοίωσης 54
6.3 Ανάλυση Αποτελεσμάτων 55
6.3.1 Κλιμάκωση με Μέγεθος Δικτύου 55
6.3.2 Κλιμάκωση φορτίου Εφαρμογών 57
6.3.3 Ανάλυση Αξιοποίησης Πόρων 60
6.3.4 Σύγκριση Latency και Makespan 62
6.3.5 Ανάλυση Κατανάλωσης Ενέργειας και Κόστους 64
6.4 Συζήτηση 65
6.4.1 Συμπεράσματα Απόδοσης 65
6.4.2 Αποδοτικότητα Αλγορίθμων 66
6.4.3 Ανάλυση Trade-offs 67
7 Περίπτωση Πραγματικού Κόσμου 68
7.1 Ανάπτυξη Fog Computing σε Έξυπνη Πόλη 68
7.1.1 Περιγραφή Υποθετικού Σεναρίου 68
7.1.2 Τυπικά Χαρακτηριστικά Workload 69
7.2 Εφαρμογή Προτεινόμενων Μεθόδων στο Θεωρητικό Σενάριο 70
7.2.1 Εφαρμογή Μεθόδου Neighbor-Aware 70
7.2.2 Εφαρμογή Μεθόδου PSO 71

7.3 Θεωρητική Ανάλυση Πλεονεκτημάτων και Μειονεκτημάτων 71
7.3.1 Συγκριτικά Πλεονεκτήματα 71
7.3.2 Θεωρητικά Σενάρια Εφαρμογής 72
7.3.3 Υβριδική Προσέγγιση 72
7.4 Θεωρητικές Προτάσεις Εφαρμογής 73
7.5 Συμπεράσματα Μελέτης Περίπτωσης 74
8 Μελλοντική Εργασία 75
8.1 Περίληψη Συνεισφορών 75
8.2 Βασικά Ευρήματα 75
8.2.1 Χαρακτηριστικά Απόδοσης 75
8.2.2 Αλγοριθμικές Γνώσεις 76
8.2.3 Πρακτικές Γνώσεις Εφαρμογής 76
8.3 Μειονεκτήματα 77
8.4 Κατευθύνσεις Μελλοντικής Έρευνας 78
8.4.1 Βελτιωμένοι Αλγόριθμοι Κατανομής 78
8.4.2 Βελτιώσεις Συστήματος 78
8.5 Παρατηρήσεις 79
9 Συμπεράσματα 80
10 Βιβλιογραφικές Αναφορές 81

1 Εισαγωγή

1.1 Υπόβαθρο και Κίνητρο

Η εκθετική αύξηση των συσκευών **Internet of Things (IoT)** έχει οδηγήσει σε μια πρωτοφανή αύξηση του όγκου δεδομένων στο edge του δικτύου. Σύμφωνα με πρόσφατες εκτιμήσεις, πάνω από 75 δισεκατομμύρια συσκευές IoT θα συνδεθούν στο διαδίκτυο μέχρι το 2025 παράγοντας τεράστιο όγκο δεδομένων που απαιτούν επεξεργασία και ανάλυση. Οι παραδοσιακές αρχιτεκτονικές cloud computing, παρά την ισχύ τους, αντιμετωπίζουν σημαντικές προκλήσεις στη διαχείριση αυτής της μαζικής εισροής δεδομένων λόγω περιορισμών του bandwidth και του latency αλλά και ζητημάτων απορρήτου.

Το **fog computing** έχει αναδειχθεί ως ένα ελκυστικό παράδειγμα για την αντιμετώπιση αυτών των περιορισμών, επεκτείνοντας τις δυνατότητες του **cloud** πιο κοντά στις πηγές δεδομένων. Κατανέμοντας τους πόρους υπολογισμού, αποθήκευσης και δικτύωσης μεταξύ του cloud και των τελικών συσκευών, το fog computing δημιουργεί ένα ενδιάμεσο στρώμα που μπορεί να επεξεργαστεί τα δεδομένα τοπικά, μειώνοντας τον όγκο των δεδομένων που μεταδίδονται στο cloud και επιτρέποντας την επεξεργασία σε πραγματικό χρόνο για εφαρμογές ευαίσθητες στο χρόνο [1].

Ωστόσο, η ετερογενής και κατανεμημένη φύση των περιβαλλόντων fog computing εισάγει νέες προκλήσεις στη διαχείριση πόρων και την κατανομή φόρτου εργασίας. Σε αντίθεση με τα κεντρικά data centers του cloud που έχουν ομοιογενείς πόρους, τα περιβάλλοντα fog αποτελούνται από πολυάριθμες συσκευές με διαφορετικές δυνατότητες, που είναι συνδεδεμένες μέσω διαφορετικών συνδέσεων δικτύου με διαφορετικό bandwidth και latency. Αυτή η **ετερογένεια**, σε συνδυασμό με τη δυναμική φύση των φόρτων εργασίας IoT, καθιστά την αποδοτική κατανομή φόρτου εργασίας ως ένα κρίσιμο και σύνθετο πρόβλημα [3].

1.2 Διατύπωση Προβλήματος

Καθώς οι συσκευές IoT και οι απαιτήσεις επεξεργασίας τους κατανέμονται δυναμικά στο χώρο και το χρόνο, η αξιοποίηση των πόρων σε fog computing περιβάλλοντα, συχνά δεν είναι ισορροπημένη. Ορισμένοι fog nodes μπορεί να υπερφορτωθούν, ενώ άλλοι λειτουργούν σε χαμηλή χωρητικότητα. Αυτή η ανισορροπία οδηγεί σε αρκετά κρίσιμα ζητήματα:

1. **Υποβάθμιση Απόδοσης** (Performance Degradation)**:** Οι υπερφορτωμένοι κόμβοι αντιμετωπίζουν συμφόρηση, αυξημένο latency και μειωμένο throughput, γεγονός που μπορεί να επηρεάσει σοβαρά τις time-sensitive εφαρμογές [6].

2. **Σπατάλη Πόρων:** Οι μη χρησιμοποιούμενοι κόμβοι αντιπροσωπεύουν σπατάλη πόρων, οδηγώντας σε αναποτελεσματικότητα και αυξημένο λειτουργικό και οικονομικό κόστος [5].

3. **Ενεργειακή ανεπάρκεια:** Η μη σωστή κατανομή φόρτου εργασίας μπορεί να οδηγήσει σε υψηλότερη κατανάλωση ενέργειας σε όλο το δίκτυο, αντίθετα δηλαδή με τα βασικά οφέλη του fog computing [2].

4. **Προκλήσεις σχετικά με την Ποιότητα Υπηρεσίας QoS (Quality of Service)**: Η μη βέλτιστη χρήση των διαθέσιμων πόρων σε περιβάλλοντα fog μπορεί να οδηγήσει σε σημαντικές παραβιάσεις των συμφωνημένων επιπέδων υπηρεσίας (SLAs), καθώς και σε αποτυχία ικανοποίησης των απαιτήσεων Quality of Service (QoS), όπως είναι η καθυστέρηση και η διαθεσιμότητα.

Σκοπός αυτής της εργασίας είναι να διερευνήσει τα παραπάνω πρόβλημα και να εξετάσει πιθανές λύσεις αποδοτικής κατανομής φόρτου εργασίας στους κόμβους fog. Συγκεκριμένα, επιλέγονται δύο βασικοί αλγόριθμοι όπως ο Neighbor-Aware και ο PSO.

1.3 Ερευνητικοί Στόχοι

Η διπλωματική εργασία στοχεύει στη διερεύνηση του προβλήματος της κατανομής του φόρτου εργασίας σε περιβάλλοντα fog computing (υπολογιστικά νέφη) και στην εξεύρεση αποδοτικών λύσεων. Για την επίτευξη αυτού του στόχου μοντελοποιούμε το περιβάλλον fog computing, τις τοπολογίες δικτύου, τα χαρακτηριστικά (attributes) κόμβων και τις απαιτήσεις (requirements) των εφαρμογών, και εφαρμόζουμε τους αλγορίθμους Neighbor-Aware και ο PSO.

Οι αλγόριθμοι αξιολογούνται πειραματικά, και για το σκοπό αυτό αναπτύχθηκε ένα ολοκληρωμένου πλαίσιο προσομοίωσης.

Συγκεκριμένα, η συνεισφορά μας συνοψίζεται στα ακόλουθα:

1. Σχεδιάζουμε και υλοποιούμε ένα γενικό πλαίσιο προσομοίωσης για την μοντελοποίηση και προσομοίωση του περιβάλλοντος fog computing και των αλγορίθμων κατανομής φορτίου:
    - Αλγόριθμος κατανομής Neighbor-Aware (επίγνωση γειτνίασης) που αξιοποιεί τους γειτονικούς κόμβους για την αποδοτική κατανομή των φόρτων εργασίας (workloads).
    - Αλγόριθμος κατανομής βασισμένος στη Βελτιστοποίηση Σμήνους Σωματιδίων (PSO) που χρησιμοποιεί βιο-εμπνευσμένη βελτιστοποίηση για την εύρεση των βέλτιστων λύσεων.
2. Αξιολογούμε και συγκρίνουμε πειραματικά την απόδοσης δυο μεθόδων κατανομής εργασιών για διάφορα σενάρια, με τη χρήση διάφορων μετρικών.
3. Δοκιμάζουμε και επαληθεύουμε την ορθή λειτουργία του συστήματος σε πραγματικές συνθήκες, όπως στη μελέτη περίπτωσης πραγματικού κόσμου (Real-World Case Study).

Η υπόλοιπη διπλωματική εργασία οργανώνεται ως εξής:

Κεφάλαιο 2: Θεωρητικό Υπόβαθρο και Βιβλιογραφική Ανασκόπηση. Παρέχει μια επισκόπηση των θεμελιωδών αρχών του fog computing, των προκλήσεων κατανομής φόρτου εργασίας και των υφιστάμενων προσεγγίσεων που βρίσκουμε στη βιβλιογραφία, με ιδιαίτερη έμφαση στις τεχνικές με Neighbor-Aware (επίγνωση γειτνίασης) και PSO (τεχνικές βελτιστοποίησης εμπνευσμένες από τη βιολογία).

Κεφάλαιο 3: FogSim-NX: Σχετικά με το Πρόγραμμα προσομοίωσης του Fog computing. Περιγράφει τον σχεδιασμό και την υλοποίηση του πλαισίου προσομοίωσης που αναπτύχθηκε για αυτή την έρευνα, συμπεριλαμβανομένης της αρχιτεκτονικής, των στοιχείων που χρησιμοποιήθηκαν και της λειτουργικότητάς του.

Κεφάλαιο 4: Μέθοδοι Κατανομής Φόρτου Εργασίας. Παρουσιάζει τον λεπτομερή σχεδιασμό και την υλοποίηση των δύο μεθόδων κατανομής που μελετώνται σε αυτή τη διπλωματική εργασία: τη μέθοδο κατανομής με επίγνωση γειτνίασης και τη μέθοδο κατανομής βασισμένη στον PSO αλγόριθμο.

Κεφάλαιο 5: Πειραματική Αξιολόγηση. Περιγράφει την πειραματική διάταξη, τα σενάρια δοκιμών και τις μετρικές απόδοσης που χρησιμοποιούνται για την αξιολόγηση των μεθόδων κατανομής. Ακολουθεί μια ολοκληρωμένη ανάλυση των αποτελεσμάτων.

Κεφάλαιο 6: Μελέτη Περίπτωσης (Real-World Case Study). Εφαρμόζει τις αναπτυγμένες μεθόδους σε ένα ρεαλιστικό σενάριο fog computing, επιδεικνύοντας την πρακτική χρησιμότητά τους και παρέχοντας πληροφορίες για αξιοποίηση στον πραγματικό κόσμο.

Κεφάλαιο 7: Συμπεράσματα και Μελλοντική Εργασία. Συνοψίζει τα βασικά ευρήματα και τις συνεισφορές αυτής της έρευνας, συζητά τους περιορισμούς και προτείνει κατευθύνσεις για μελλοντική εργασία.

Η διπλωματική εργασία ολοκληρώνεται με έναν κατάλογο αναφορών και παραρτήματα που περιέχουν πρόσθετες τεχνικές λεπτομέρειες και πειραματικά αποτελέσματα.

2 Θεωρητικό Υπόβαθρο και Βιβλιογραφική Ανασκόπηση

2.1 Θεμελιώδεις Αρχές του Fog Computing

2.1.1 Ορισμός και Χαρακτηριστικά

Ο όρος **fog computing**, που επινοήθηκε αρχικά από τη Cisco το 2012, αναφέρεται σε μια αρχιτεκτονική που επεκτείνει τις δυνατότητες του cloud computing στο άκρο (edge) του δικτύου [1].

Ενώ οι Bonomi et al. [1] παρείχαν την αρχική εννοιολογική προσέγγιση του fog computing, οι Vaquero και Rodero-Merino [10] διεύρυναν αυτόν τον ορισμό τονίζοντας το fog computing ως ένα σενάριο όπου ένας τεράστιος αριθμός ετερογενών και αποκεντρωμένων συσκευών επικοινωνούν και συνεργάζονται μεταξύ τους και με το δίκτυο για την εκτέλεση εργασιών αποθήκευσης και επεξεργασίας χωρίς την παρέμβαση τρίτων. Η εργασία τους ήταν καθοριστική στην καθιέρωση μιας πιο ολοκληρωμένης κατανόησης του fog computing που περιλαμβάνει όχι μόνο την επέκταση των δυνατοτήτων του cloud αλλά και τη φύση συνεργασίας των συσκευών fog, η οποία είναι ιδιαίτερα κοντά με τις neighbor-aware στρατηγικές κατανομής που διερευνώνται σε αυτή την εργασία. Αντιπροσωπεύει μια μεταφορά από το παραδοσιακό κεντρικοποιημένο μοντέλο cloud σε μια πιο κατανεμημένη προσέγγιση, όπου οι υπολογιστικοί πόροι τοποθετούνται πλησιέστερα στις πηγές δεδομένων και τους χρήστες.

Τα βασικά χαρακτηριστικά που διακρίνουν το fog computing περιλαμβάνουν:

1. **Τοποθεσία στα άκρα**: Σε αντίθεση με τα data centers του cloud, οι fog nodes αναπτύσσονται στο edge του δικτύου, κοντά στις τελικές συσκευές και τις πηγές δεδομένων [3].
2. **Γεωγραφική Κατανομή**: Η υποδομή fog αποτελείται από πολυάριθμους κατανεμημένους κόμβους που καλύπτουν πολλαπλές τοποθεσίες, σε αντίθεση με την κεντρικοποιημένη φύση του cloud computing [1].
3. **Ετερογένεια**: Τα περιβάλλοντα fog περιλαμβάνουν μια μεγάλη ποικιλία συσκευών με διαφορετικές υπολογιστικές δυνατότητες, από αισθητήρες με περιορισμένους πόρους έως ισχυρούς edge servers.
4. **Αλληλεπιδράσεις σε πραγματικό χρόνο**: Το fog computing υποστηρίζει εφαρμογές που απαιτούν χαμηλό latency και αποκρίσεις σε πραγματικό χρόνο, καθιστώντας το κατάλληλο για time-sensitive υπηρεσίες.
5. **Υποστήριξη Mobility**: Πολλές υλοποιήσεις fog computing υποστηρίζουν ρητά mobile συσκευές και υπηρεσίες, επιτρέποντας εφαρμογές με επίγνωση τοποθεσίας.
6. **Διαλειτουργικότητα**: Οι fog nodes συχνά χρειάζεται να συνεργάζονται και να αλληλεπιδρούν με διαφορετικές πλατφόρμες και υπηρεσίες, απαιτώντας τυποποιημένες διεπαφές και πρωτόκολλα.

2.1.2 Σύγκριση με Cloud Computing

Ενώ το fog computing επεκτείνει και συμπληρώνει το cloud computing, έχουν αρκετές βασικές διαφορές. Δείτε στον παρακάτω πίνακα.

Πίνακας 2-1: Σύγκριση Fog με Cloud computing

ΠΤΥΧΗ	FOG COMPUTING	CLOUD COMPUTING
ΤΟΠΟΘΕΣΙΑ	Άκρο δικτύου, κοντά σε πηγές δεδομένων	Κεντρικοποιημένα data centers
LATENCY	Χαμηλό (ms)	Υψηλότερο (δεκάδες έως εκατοντάδες ms)
ΓΕΩΓΡΑΦΙΚΗ ΚΑΤΑΝΟΜΗ	Υψηλά κατανεμημένο	Κεντρικοποιημένο ή περιφερειακά κατανεμημένο
ΑΠΟΣΤΑΣΗ CLIENT-SERVER	Μονό hop	Πολλαπλά hops
ΧΩΡΗΤΙΚΟΤΗΤΑ ΠΟΡΩΝ	Περιορισμένη ανά κόμβο	Πρακτικά απεριόριστη (elastic)
ΑΠΟΘΗΚΕΥΣΗ	Προσωρινή, περιορισμένη	Μόνιμη, κλιμακούμενη
ΔΙΑΘΕΣΙΜΟΤΗΤΑ	Μεταβλητή, πιθανώς λιγότερο αξιόπιστη	Υψηλή, με μηχανισμούς πλεονασμού
MOBILITY SUPPORT	Ισχυρή υποστήριξη για κινητές συσκευές	Περιορισμένη επίγνωση mobility
ΕΝΕΡΓΕΙΑΚΗ ΑΠΟΔΟΤΙΚΟΤΗΤΑ	Πιθανώς υψηλότερη λόγω τοπικής επεξεργασίας	Χαμηλότερη λόγω overhead μετάδοσης δεδομένων
ΠΡΟΚΛΗΣΕΙΣ ΑΣΦΑΛΕΙΑΣ	Κατανεμημένοι τομείς εμπιστοσύνης	Κεντρικοποιημένη διαχείριση ασφάλειας

Αντί να αντικαθιστά το cloud computing, το fog computing δημιουργεί μια συνέχεια πόρων από το edge έως το cloud, επιτρέποντας στις εφαρμογές να χρησιμοποιούν τους καταλληλότερους πόρους με βάση τις απαιτήσεις τους.

Οι Vaquero και Rodero-Merino [10] διαφοροποίησαν περαιτέρω το fog computing από το cloud computing τονίζοντας την υποστήριξή του για κινητικότητα, την έμφασή του στην ασύρματη πρόσβαση και την ισχυρότερη αποκέντρωση δεδομένων και υπολογισμών σε σύγκριση με τα παραδοσιακά μοντέλα cloud. Ανέφεραν ότι αυτά τα χαρακτηριστικά καθιστούν το fog computing κατάλληλο για σενάρια που περιλαμβάνουν κινητούς χρήστες και γεωγραφικά κατανεμημένες υπηρεσίες, όπως συνήθως βρίσκονται σε αναπτύξεις έξυπνων πόλεων.

2.1.3 Αρχιτεκτονική Fog Computing

Μια τυπική αρχιτεκτονική fog computing περιλαμβάνει τρία βασικά επίπεδα:

1. **IoT/End Devices Layer**: Αυτό το επίπεδο αποτελείται από συσκευές IoT, αισθητήρες, ενεργοποιητές και συσκευές χρηστών που παράγουν δεδομένα και αλληλεπιδρούν με τον φυσικό κόσμο. Αυτές οι συσκευές έχουν συνήθως περιορισμένες υπολογιστικές δυνατότητες και βασίζονται σε πόρους fog ή cloud για σύνθετη επεξεργασία.

2. **Fog Layer**: Το ενδιάμεσο επίπεδο που αποτελείται από fog nodes κατανεμημένους στο edge του δικτύου. Αυτοί οι κόμβοι μπορεί να είναι ειδικοί fog servers, εξοπλισμός δικτύου με υπολογιστικές δυνατότητες (π.χ., routers, gateways) ή ακόμα και συσκευές χρηστών με πλεονάζοντες πόρους. Αυτό το επίπεδο διαιρείται περαιτέρω σε:
    - **Edge Fog Nodes**: Βρίσκονται στο άκρο (edge), συνδεδεμένοι απευθείας με τελικές συσκευές
    - **Intermediate Fog Nodes**: Σημεία συγκέντρωσης που συντονίζουν πολλαπλούς edge nodes
    - **Gateway Fog Nodes**: Συνδέουν την υποδομή fog με το cloud
3. **Cloud Layer**: Παραδοσιακά data centers cloud που παρέχουν πρακτικά απεριόριστους υπολογιστικούς και αποθηκευτικούς πόρους, τυπικά χρησιμοποιούμενα για μακροπρόθεσμη αποθήκευση, σύνθετη ανάλυση και γενικό συντονισμό.

Οι ροές δεδομένων σε αυτή την αρχιτεκτονική μπορεί να είναι αμφίδρομες:

- Ανοδική ροή: από τελικές συσκευές μέσω fog nodes στο cloud
- Καθοδική ροή: από το cloud μέσω fog nodes στις τελικές συσκευές

Αυτή η ιεραρχική δομή επιτρέπει μια διαβαθμισμένη προσέγγιση στην επεξεργασία δεδομένων, όπου οι time-sensitive υπολογισμοί πραγματοποιούνται στο edge, ενώ οι εργασίες που απαιτούν πόρους ή γενικό συντονισμό μεταφέρονται στο cloud.

2.2 Κατανομή Φόρτου Εργασίας σε περιβάλλον Fog Computing

Η κατανομή φόρτου εργασίας (workload allocation) αναφέρεται στη διαδικασία διάθεσης υπολογιστικών εργασιών και εφαρμογών στους διαθέσιμους πόρους ενός κατανεμημένου συστήματος με τρόπο που βελτιστοποιεί συγκεκριμένες μετρικές απόδοσης όπως ο χρόνος απόκρισης, η καθυστέρηση, η χρήση πόρων και η κατανάλωση ενέργειας. Σε περιβάλλοντα fog computing, αυτή η διαδικασία περιλαμβάνει την απόφαση για το ποιες εφαρμογές θα εκτελεστούν σε κόμβους fog, ποιες θα μεταφερθούν στο cloud, καθώς και τη βέλτιστη αντιστοίχιση εφαρμογών σε συγκεκριμένους κόμβους με βάση τα χαρακτηριστικά τους και τις απαιτήσεις των εφαρμογών.

2.2.1 Προκλήσεις Διαχείρισης Πόρων

Η διαχείριση πόρων σε περιβάλλοντα fog computing αντιμετωπίζει αρκετές δυσκολίες λόγω της κατανεμημένης και ετερογενούς φύσης τους:

1. **Διαφορετικότητα (Ετερογένεια) Πόρων**: Οι fog nodes έχουν διαφορετικές υπολογιστικές δυνατότητες, χωρητικότητες μνήμης, χώρους αποθήκευσης και bandwidth δικτύου, καθιστώντας τις ομοιόμορφες στρατηγικές κατανομής πόρων αναποτελεσματικές.
2. **Δυναμικότητα Δικτύου**: Η τοπολογία και οι συνθήκες δικτύου σε περιβάλλοντα fog είναι συχνά δυναμικές, με μεταβαλλόμενα bandwidth, latency, ακόμη και διαθεσιμότητα κόμβων λόγω mobility ή ενεργειακών περιορισμών.
3. **Ποικιλομορφία Φορτίου Εργασίας (Workload Diversity)**: Οι εφαρμογές σε περιβάλλοντα fog έχουν ποικίλες απαιτήσεις όσον αφορά την υπολογιστική δύναμη, τη χρήση μνήμης, τις ανάγκες αποθήκευσης και την ευαισθησία στην απόκριση (latency).

4. **Multi-objective Optimization**: Η κατανομή πόρων πρέπει να βελτιστοποιεί ταυτόχρονα πολλαπλούς, συχνά αντικρουόμενους στόχους, όπως την ελαχιστοποίηση του latency, τη μεγιστοποίηση της χρήσης πόρων, τη μείωση της κατανάλωσης ενέργειας και τον έλεγχο του λειτουργικού κόστους.

5. **Ζητήματα Επεκτασιμότητας (Scalability Issues)**: Καθώς ο αριθμός των fog nodes και των εφαρμογών αυξάνεται, οι κεντρικοποιημένες προσεγγίσεις κατανομής γίνονται όλο και πιο πρακτικές λόγω overhead επικοινωνίας και υπολογιστικής πολυπλοκότητας.

6. **Αξιοπιστία και Ανοχή Σφάλματος**: Οι fog nodes μπορεί να αποτύχουν ή να αποσυνδεθούν, απαιτώντας στρατηγικές κατανομής που διατηρούν τη συνέχεια εξυπηρέτησης παρά τη μη διαθεσιμότητα των πόρων.

2.2.2 Χαρακτηριστικά Φόρτου Εργασίας σε Περιβάλλοντα Fog

Τα Workload (Φόρτοι Εργασίας) σε περιβάλλοντα fog computing παρουσιάζουν αρκετά διακριτικά χαρακτηριστικά που επηρεάζουν τις στρατηγικές κατανομής:

1. **Locality Sensitivity**: Πολλές εφαρμογές fog επωφελούνται σημαντικά από την εκτέλεση κοντά σε πηγές δεδομένων ή χρηστών, απαιτώντας στρατηγικές κατανομής που λαμβάνουν υπόψη τη γεωγραφική απόσταση.

2. **Latency Requirements**: Εφαρμογές κρίσιμες για το χρόνο, όπως επαυξημένη πραγματικότητα, αυτόνομα οχήματα ή συστήματα βιομηχανικού ελέγχου, έχουν αυστηρούς περιορισμούς latency που πρέπει να πληρούνται μέσω κατάλληλων αποφάσεων κατανομής.

3. **Resource Intensity Variation**: Τα Workload fog κυμαίνονται από ελαφρύ φιλτράρισμα δεδομένων αισθητήρων έως υπολογιστικά εντατική συμπερασματολογία μηχανικής μάθησης, απαιτώντας στρατηγικές κατανομής που ταιριάζουν τις ανάγκες εφαρμογών με κατάλληλους πόρους.

4. **Data Volume Considerations**: Ο όγκος των δεδομένων που παράγονται και επεξεργάζονται από εφαρμογές επηρεάζει τη χρήση δικτύου και μπορεί να επηρεάσει τις αποφάσεις κατανομής για ελαχιστοποίηση του κόστους της μεταφοράς των δεδομένων.

5. **Temporal Patterns**: Πολλοί φόρτοι εργασίας fog παρουσιάζουν χρονικά μοτίβα, με προβλέψιμες ή απρόβλεπτες διακυμάνσεις στις απαιτήσεις πόρων που οι στρατηγικές κατανομής πρέπει να λαμβάνουν υπόψη.

6. **Data Dependencies**: Οι σύνθετες εφαρμογές συχνά περιλαμβάνουν πολλαπλές αλληλεξαρτώμενες εργασίες με απαιτήσεις ροής δεδομένων που περιορίζουν τις δυνατότητες κατανομής.

7. **Mobility Patterns**: Σε σενάρια που περιλαμβάνουν χρήστες που μετακινούνται ή συσκευές, οι στρατηγικές κατανομής πρέπει να λαμβάνουν υπόψη τα μοτίβα mobility για τη διατήρηση της ποιότητας υπηρεσίας παρά τις μεταβαλλόμενες συνθήκες δικτύου.

2.3 Τεχνικές Κατανομής Πόρων

2.3.1 Κεντρικές έναντι Κατανεμημένων Προσεγγίσεων

Οι προσεγγίσεις κατανομής πόρων σε περιβάλλοντα fog computing χωρίζονται σε δύο κατηγορίες:

Κεντρικοποιημένες Προσεγγίσεις:

- Χρησιμοποιούν έναν κεντρικό controller ή orchestrator που έχει καθολική γνώση της κατάστασης του συστήματος
- Λαμβάνουν αποφάσεις κατανομής βάσει ολοκληρωμένων πληροφοριών για όλους τους πόρους και φόρτους εργασίας
- Πλεονεκτήματα: ενδεχομένως βέλτιστες λύσεις, συνεπής επιβολή πολιτικής, απλούστερη υλοποίηση
- Μειονεκτήματα: μοναδικό σημείο αποτυχίας, περιορισμοί κλιμάκωσης, υψηλό overhead επικοινωνίας [6].

Κατανεμημένες Προσεγγίσεις:

- Κατανέμουν τις αποφάσεις κατανομής σε πολλαπλές οντότητες στο σύστημα
- Βασίζονται σε τοπικές πληροφορίες ή περιορισμένο συντονισμό μεταξύ γειτονικών κόμβων
- Πλεονεκτήματα: υψηλότερη κλιμάκωση, ανοχή σφαλμάτων, μειωμένο overhead επικοινωνίας
- Μειονεκτήματα: μη βέλτιστες λύσεις, προβλήματα συντονισμού, πολύπλοκη υλοποίηση

Υβριδικές προσεγγίσεις που συνδυάζουν στοιχεία τόσο των κεντρικοποιημένων όσο και των κατανεμημένων κατανομών έχουν επίσης προταθεί για την εξισορρόπηση των αντίστοιχων πλεονεκτημάτων και περιορισμών τους.

2.3.2 Ευρετικές Μέθοδοι

Οι ευρετικές μέθοδοι παρέχουν πρακτικές λύσεις στο πρόβλημα κατανομής φόρτου εργασίας σε περιβάλλοντα fog μέσω κανόνων, κατευθυντήριων γραμμών ή διαδικασιών που δίνουν λογικές λύσεις χωρίς να εγγυώνται βέλτιστη λύση. Κοινές ευρετικές προσεγγίσεις περιλαμβάνουν:

1. **Greedy Algorithms**: Κάνουν τοπικά βέλτιστες επιλογές σε κάθε βήμα, όπως η ανάθεση φόρτων εργασίας στον λιγότερο φορτωμένο κόμβο ή στον κόμβο με τον ελάχιστο αναμενόμενο χρόνο ολοκλήρωσης.
2. **First-Fit and Best-Fit Strategies**: Κατανέμουν φόρτους εργασίας στον πρώτο κατάλληλο κόμβο ή στον καταλληλότερο κόμβο με βάση προκαθορισμένα κριτήρια.
3. **Neighborhood-Based Methods**: Λαμβάνουν υπόψη τοπικές πληροφορίες και καταστάσεις γειτονικών κόμβων για τη λήψη αποφάσεων κατανομής, επιτρέποντας τοπική εξισορρόπηση φορτίου και κοινή χρήση πόρων.
4. **List Scheduling**: Θέτουν προτεραιότητες στους φόρτους εργασίας με βάση συγκεκριμένα χαρακτηριστικά (π.χ., προθεσμία, απαιτήσεις πόρων) και τους κατανέμουν διαδοχικά σύμφωνα με αυτή την προτεραιότητα.

Αυτές οι ευρετικές μέθοδοι συχνά προτιμώνται σε πρακτικές αναπτύξεις fog computing λόγω της απλότητάς τους, της υπολογιστικής αποδοτικότητας και της προσαρμοστικότητάς τους σε μεταβαλλόμενες συνθήκες.

2.3.3 Μέθοδοι Βασισμένες σε Βελτιστοποίηση

Οι μέθοδοι βασισμένες σε βελτιστοποίηση διατυπώνουν την κατανομή φόρτου εργασίας ως προβλήματα μαθηματικής βελτιστοποίησης και χρησιμοποιούν διάφορες τεχνικές για την εύρεση βέλτιστων ή σχεδόν βέλτιστων λύσεων. Αυτές οι προσεγγίσεις περιλαμβάνουν:

1. **Linear and Integer Programming**: Διατυπώνουν την κατανομή ως πρόβλημα γραμμικού ή ακέραιου προγραμματισμού με περιορισμούς που αντιπροσωπεύουν περιορισμούς πόρων και στόχους όπως η ελαχιστοποίηση της κατανάλωσης ενέργειας ή του latency.
2. **Constraint Satisfaction**: Ορίζουν την κατανομή ως πρόβλημα ικανοποίησης περιορισμών, διασφαλίζοντας ότι πληρούνται όλοι οι περιορισμοί πόρων και οι απαιτήσεις εφαρμογών.
3. **Meta-heuristic Approaches**:
    - **Genetic Algorithms**: Εξελικτική προσέγγιση που παράγει λύσεις μέσω μηχανισμών εμπνευσμένων από τη φυσική επιλογή.
    - **Particle Swarm Optimization (PSO)**: Στοχαστική τεχνική βελτιστοποίησης βασισμένη σε πληθυσμό εμπνευσμένη από την κοινωνική συμπεριφορά του σμήνους πτηνών ή κοπαδιού ψαριών.
    - **Ant Colony Optimization**: Πιθανολογική τεχνική εμπνευσμένη από τη συμπεριφορά αναζήτησης τροφής των μυρμηγκιών.
4. **Machine Learning Methods**: Χρησιμοποιούν reinforcement learning, neural networks ή άλλες τεχνικές ML για την εκμάθηση αποτελεσματικών πολιτικών κατανομής από την εμπειρία.

Αυτές οι μέθοδοι βασισμένες σε βελτιστοποίηση μπορούν ενδεχομένως να βρουν ανώτερες κατανομές σε σύγκριση με απλές ευρετικές, ιδιαίτερα για σύνθετα σενάρια με πολλαπλούς στόχους, αλλά συχνά με κόστος υψηλότερης υπολογιστικής πολυπλοκότητας.

2.4 Σχετικές Εργασίες

2.4.1 Στρατηγικές Κατανομής με Επίγνωση Γειτνίασης (Neighbor-Aware)

Οι στρατηγικές κατανομής με επίγνωση γειτνίασης αξιοποιούν την τοπική γνώση και την αλληλεπίδραση μεταξύ γειτονικών κόμβων για τη λήψη αποτελεσματικών αποφάσεων κατανομής χωρίς να απαιτείται καθολική γνώση του συστήματος. Αρκετές αξιοσημείωτες εργασίες έχουν διερευνήσει αυτή την προσέγγιση:

Οι Gupta et al. [3] πρότειναν μια αρχιτεκτονική fog computing όπου οι γειτονικοί fog nodes συνεργάζονται για την εξισορρόπηση φορτίου και τη μείωση του latency για εφαρμογές IoT. Η προσέγγισή τους επιτρέπει στους fog nodes να μεταφέρουν εργασίες σε γειτονικούς κόμβους όταν οι τοπικοί πόροι είναι ανεπαρκείς, βελτιώνοντας τη χρήση πόρων και μειώνοντας την εξάρτηση από το cloud.

Οι Taneja et al. [11] ανέπτυξαν έναν αλγόριθμο κατανομής πόρων με επίγνωση εγγύτητας που λαμβάνει υπόψη τη γεωγραφική εγγύτητα των fog nodes προς τις συσκευές IoT, επιτρέποντας αποδοτική τοποθέτηση υπηρεσιών με παράλληλη ελαχιστοποίηση του overhead επικοινωνίας.

Οι Skarlat et al. [6] εισήγαγαν την έννοια των "fog colonies" όπου οι γειτονικοί fog nodes σχηματίζουν ομάδες συνεργασίας για κοινή χρήση πόρων και συνεργατική εκτέλεση υπηρεσιών. Η προσέγγισή τους χρησιμοποιεί ένα ιεραρχικό σύστημα διαχείρισης πόρων που δίνει προτεραιότητα στην τοπική εκτέλεση εντός των fog colonies πριν εξετάσει το offloading στο cloud. Επίσης πρότειναν μια στρατηγική δυναμικής μετεγκατάστασης υπηρεσιών με επίγνωση γειτνίασης που επιτρέπει στις υπηρεσίες να μεταναστεύουν μεταξύ κοντινών fog nodes με βάση τα μοτίβα mobility των χρηστών, ελαχιστοποιώντας τη διακοπή υπηρεσιών και διατηρώντας χαμηλό latency.

Αυτές οι προσεγγίσεις με επίγνωση γειτνίασης αποδεικνύουν την αποτελεσματικότητα της τοπικής λήψης αποφάσεων σε περιβάλλοντα fog, ιδιαίτερα για σενάρια με γεωγραφικά κατανεμημένους φόρτους εργασίας και περιορισμένο bandwidth επικοινωνίας.

2.4.2 Τεχνικές Βελτιστοποίησης Εμπνευσμένες από τη Βιολογία

Οι τεχνικές βελτιστοποίησης εμπνευσμένες από τη βιολογία έχουν αποκτήσει δημοτικότητα στην κατανομή πόρων για fog computing λόγω της ικανότητάς τους να βρίσκουν σχεδόν βέλτιστες λύσεις για σύνθετα προβλήματα με πολλαπλούς στόχους. Αυτές οι τεχνικές αντλούν έμπνευση από φυσικές διαδικασίες και συμπεριφορές:

Ο αλγόριθμος Particle Swarm Optimization (PSO), που εισήχθη από τους Kennedy και Eberhart [4], αποτελεί χαρακτηριστικό παράδειγμα βιο-εμπνευσμένης τεχνικής βελτιστοποίησης που μιμείται τη συμπεριφορά σμήνους πτηνών ή κοπαδιού ψαριών. Η προσέγγιση αυτή χαρακτηρίζεται από την παράλληλη εξερεύνηση του χώρου λύσεων μέσω πολλαπλών "σωματιδίων" που μοιράζονται πληροφορίες, επιτρέποντας γρήγορη σύγκλιση προς βέλτιστες λύσεις για πολύπλοκα προβλήματα βελτιστοποίησης.

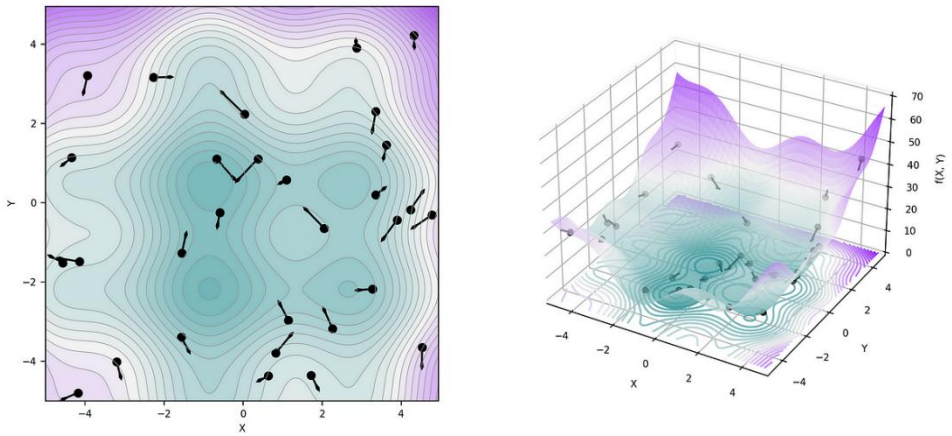
Στο πλαίσιο του fog computing, οι βιο-εμπνευσμένες τεχνικές εφαρμόζονται αποτελεσματικά για τη βελτιστοποίηση κατανομής φορτίου εργασίας με στόχο την εξισορρόπηση αντικρουόμενων παραμέτρων. Οι Deng et al. [2] διερεύνησαν την εφαρμογή τεχνικών βελτιστοποίησης για την επίτευξη ισορροπίας μεταξύ καθυστέρησης και κατανάλωσης ενέργειας στο fog computing, αναδεικνύοντας τη σημασία της βέλτιστης κατανομής φορτίου εργασίας μεταξύ fog και cloud.

Οι συνεργατικές προσεγγίσεις βελτιστοποίησης, όπως αυτές που προτείνουν οι Zhou et al. [7], επεκτείνουν αυτές τις τεχνικές συνδυάζοντας πολλαπλά σμήνη που συνεργάζονται για τη βελτιστοποίηση διαφορετικών πτυχών της κατανομής πόρων. Αυτή η προσέγγιση είναι ιδιαίτερα αποτελεσματική σε περιβάλλοντα fog-cloud, όπου η βελτιστοποίηση πρέπει να λαμβάνει υπόψη ετερογενείς πόρους, μεταβαλλόμενες συνθήκες δικτύου και ποικίλες απαιτήσεις εφαρμογών.

Η ολοκληρωμένη επισκόπηση τεχνικών διαχείρισης εφαρμογών σε περιβάλλοντα fog computing από τους Mahmud et al. [5] αναδεικνύει τη σημασία των βιο-εμπνευσμένων προσεγγίσεων για την αντιμετώπιση πολύπλοκων προβλημάτων βελτιστοποίησης πολλαπλών στόχων. Οι συγγραφείς επισημαίνουν πώς αυτές οι τεχνικές μπορούν να εξισορροπήσουν αποτελεσματικά το κέρδος του παρόχου υπηρεσιών και την εμπειρία χρήστη, επιδεικνύοντας την ικανότητα εύρεσης ισορροπημένων λύσεων για ανταγωνιστικούς στόχους.

Αυτές οι προσεγγίσεις που είναι εμπνευσμένες από τη βιολογία επιδεικνύουν αξιοσημείωτη προσαρμοστικότητα στη δυναμική φύση των περιβαλλόντων fog και ικανότητα χειρισμού της πολυπλοκότητας και της βελτιστοποίησης πολλαπλών στόχων στην κατανομή φόρτου εργασίας.

2.4.3 Particle Swarm Optimization στην Κατανομή Πόρων



Εικόνα 2.1: Τυχαία αρχικοποίηση με N = 30 αντικείμενα με ταχύτητα (velocity)

Το Particle Swarm Optimization (PSO), συγκεκριμένα, έχει αναδειχθεί ως μια υποσχόμενη προσέγγιση για την κατανομή πόρων σε fog computing λόγω της υπολογιστικής του αποδοτικότητας, της δυνατότητας παραλληλοποίησης και της αποτελεσματικότητας σε πολυδιάστατους χώρους αναζήτησης [2].

Οι Deng et al. [2] εφάρμοσαν PSO για τη βελτιστοποίηση του προγραμματισμού εργασιών σε περιβάλλοντα cloud-fog, εστιάζοντας στην ελαχιστοποίηση του χρόνου εκτέλεσης και της κατανάλωσης ενέργειας. Η προσέγγισή τους επέδειξε ταχύτερη σύγκλιση και καλύτερες λύσεις σε σύγκριση με γενετικούς αλγόριθμους για τα ίδια σενάρια .

Οι Mahmud et al. [5] εισήγαγαν μια πολιτική τοποθέτησης εφαρμογών με επίγνωση κέρδους χρησιμοποιώντας PSO που εξισορροπεί το κέρδος του παρόχου υπηρεσιών και την εμπειρία χρήστη, επιδεικνύοντας την ικανότητα του αλγορίθμου να βρίσκει ισορροπημένες λύσεις για ανταγωνιστικούς στόχους.

Οι Zhou et al. [7] πρότειναν μια συνεργατική προσέγγιση PSO όπου πολλαπλά σμήνη συνεργάζονται για τη βελτιστοποίηση διαφορετικών πτυχών της κατανομής πόρων σε περιβάλλοντα fog-cloud, δείχνοντας βελτιωμένη ταχύτητα σύγκλισης και ποιότητα λύσης σε σύγκριση με παραδοσιακές υλοποιήσεις PSO.

Η δημοτικότητα του PSO για την κατανομή πόρων σε fog computing πηγάζει από αρκετά βασικά πλεονεκτήματα:

- Η φύση του που βασίζεται σε πληθυσμό επιτρέπει την εξερεύνηση πολλαπλών πιθανών λύσεων ταυτόχρονα

- Η απλότητα του αλγορίθμου και το χαμηλό υπολογιστικό overhead τον καθιστούν κατάλληλο για δυναμικά περιβάλλοντα.

- Ο εγγενής παραλληλισμός του PSO ευθυγραμμίζεται καλά με την κατανεμημένη φύση του fog computing.

- Η ευελιξία του αλγορίθμου επιτρέπει εύκολη προσαρμογή σε προβλήματα βελτιστοποίησης πολλαπλών στόχων.

Αυτά τα χαρακτηριστικά καθιστούν το PSO ιδιαίτερα κατάλληλο για την αντιμετώπιση των σύνθετων προκλήσεων κατανομής φόρτου εργασίας σε ετερογενή περιβάλλοντα fog computing.

3 Μέθοδοι Κατανομής Φόρτου Εργασίας

3.1 Μέθοδος Κατανομής Neighbor-Aware

Ο αλγόριθμος Neighbor-Aware σε αυτή την εργασία αποτελεί **πρωτότυπη** συνεισφορά που συνδυάζει και επεκτείνει έννοιες από υπάρχουσες προσεγγίσεις. Εμπνευσμένος από την έννοια των "fog colonies" των Skarlat et al. [6] και τη στρατηγική γειτονικής συνεργασίας των Gupta et al. [3]. Ο συγκεκριμένος αλγόριθμος εισάγει μια δομημένη διαδικασία τριών επιπέδων κατανομής με ειδικούς μηχανισμούς load balancing και makespan βελτιστοποίησης που δεν έχουν παρουσιαστεί στη βιβλιογραφία με αυτόν τον συνδυασμό.

3.1.1 Θεωρητικό Υπόβαθρο Μεθόδου Neighbor-Aware

Η μέθοδος κατανομής Neighbor-Aware βασίζεται στις αρχές της κατανεμημένης λήψης αποφάσεων και της τοπικής βελτιστοποίησης, αξιοποιώντας την έννοια της γειτνίασης στα δίκτυα για την αποδοτική κατανομή φόρτου εργασίας. Η θεωρητική της βάση αντλεί από τη θεωρία γράφων και τις ευρετικές προσεγγίσεις που εστιάζουν στην εκμετάλλευση της τοπολογίας του δικτύου.

Βασικές Θεωρητικές Αρχές:

Αρχή Εγγύτητας (Proximity Principle): Οι εφαρμογές κατανέμονται κατά προτίμηση σε κόμβους που είναι τοπολογικά κοντά στα σημεία εισόδου τους, μειώνοντας το overhead επικοινωνίας και την καθυστέρηση μετάδοσης.
Greedy Optimization: Κάθε απόφαση κατανομής λαμβάνεται με στόχο την τοπική βελτιστοποίηση (ελαχιστοποίηση αύξησης makespan), χωρίς να λαμβάνεται υπόψη η επίδραση στη συνολική κατανομή.
Ιεραρχική Αναζήτηση: Η μέθοδος ακολουθεί μια δομημένη διαδικασία επέκτασης του χώρου αναζήτησης από τοπικούς κόμβους σε ευρύτερες περιοχές δικτύου, εφαρμόζοντας την αρχή της σταδιακής κλιμάκωσης.
Load-Aware Decision Making: Οι αποφάσεις κατανομής λαμβάνουν υπόψη το τρέχον φορτίο των κόμβων, προωθώντας την εξισορρόπηση φορτίου μέσω της επιλογής λιγότερο φορτωμένων gateways.

Αυτή η θεωρητική προσέγγιση παρέχει υπολογιστική αποδοτικότητα και ταχεία απόκριση, καθιστώντας τη μέθοδο κατάλληλη για δυναμικά περιβάλλοντα με χρονικούς περιορισμούς.

3.1.2 Σχεδιασμός Αλγορίθμου

Η μέθοδος κατανομής Neighbor-Aware είναι μια κατανεμημένη, ευρετική προσέγγιση σχεδιασμένη για την αποδοτική κατανομή εφαρμογών σε περιβάλλοντα fog computing αξιοποιώντας την τοπική γνώση γειτονικών κόμβων. Αυτή η μέθοδος ακολουθεί μια ιεραρχική στρατηγική κατανομής που ξεκινά με τους gateway nodes και επεκτείνεται στους γειτονικούς fog nodes πριν καταφύγει σε πόρους cloud όταν είναι απαραίτητο.

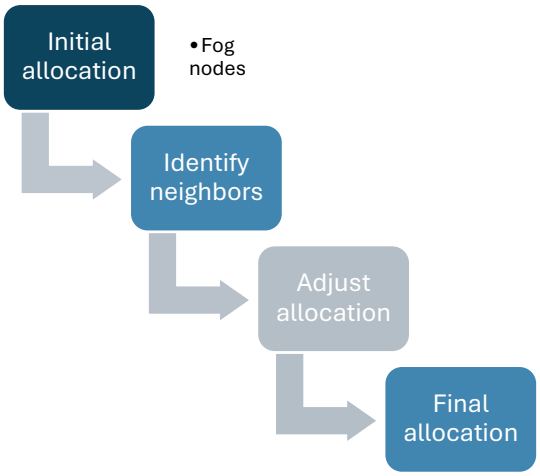
Οι βασικές αρχές της μεθόδου κατανομής Neighbor-Aware περιλαμβάνουν:

1. **Gateway-Centric Distribution:** Οι gateway nodes λειτουργούν ως σημεία εισόδου για εφαρμογές και συντονίζουν τη διαδικασία κατανομής με βάση τη γνώση τους για τους τοπικούς πόρους, παρόμοια με την προσέγγιση fog colony που προτάθηκε από τους Skarlat et al. [6].

2. **Proximity-Based Allocation:** Οι εφαρμογές κατανέμονται κατά προτίμηση σε κόμβους που είναι τοπολογικά κοντά στα σημεία εισόδου τους, μειώνοντας το overhead επικοινωνίας και το latency.
3. **Load-Aware Forwarding:** Οι gateway nodes διανέμουν εφαρμογές σε fog nodes με βάση το τρέχον φορτίο και τη χωρητικότητά τους για την ελαχιστοποίηση των αυξήσεων makespan.
4. **Hierarchical Fallback Mechanism:** Εάν η τοπική κατανομή αποτύχει, ο αλγόριθμος σταδιακά επεκτείνει την αναζήτησή του σε γειτονικούς κόμβους και τελικά στο cloud ως έσχατη λύση, ακολουθώντας αρχές παρόμοιες με αυτές που περιγράφηκαν από τους Bonomi et al. [1] για αρχιτεκτονικές fog πολλαπλών επιπέδων.

Η διαδικασία λήψης αποφάσεων κατανομής καθοδηγείται κυρίως από την ελαχιστοποίηση του makespan, που αντιπροσωπεύει το χρόνο που απαιτείται για την ολοκλήρωση όλων των εφαρμογών που έχουν ανατεθεί σε έναν κόμβο. Ελαχιστοποιώντας την αύξηση του makespan για κάθε κατανομή, ο αλγόριθμος επιχειρεί να εξισορροπήσει το φορτίο σε όλο το δίκτυο σεβόμενος τους περιορισμούς πόρων.

Η Εικόνα 3.1 απεικονίζει την ιεραρχική ροή κατανομής της μεθόδου Neighbor-Aware.



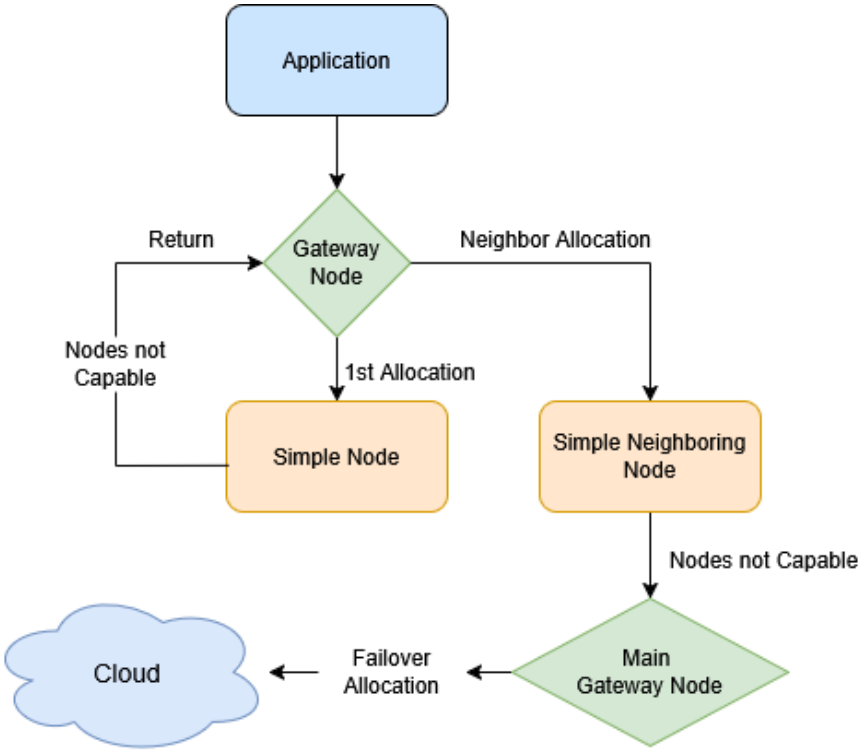
Εικόνα 3.1: Neighbor-Aware που δείχνει την ιεραρχική διαδικασία κατανομής

3.1.3 Λειτουργική Ροή

Η μέθοδος κατανομής Neighbor-Aware ακολουθεί μια ιεραρχική στρατηγική λήψης αποφάσεων που επεξεργάζεται τις εφαρμογές μία προς μία, εφαρμόζοντας μια σταδιακή διαδικασία τριών βημάτων για κάθε εφαρμογή. Η διαδικασία ξεκινά με την προσπάθεια τοπικής κατανομής σε γειτονικούς κόμβους και επεκτείνεται προοδευτικά σε ευρύτερες περιοχές του δικτύου εάν η αρχική προσπάθεια αποτύχει.

Χαρακτηριστικά της λειτουργικής ροής:

- Διαδοχική επεξεργασία: Κάθε εφαρμογή επεξεργάζεται ξεχωριστά με βάση την τρέχουσα κατάσταση των πόρων
- Δυναμική προσαρμογή: Οι αποφάσεις κατανομής λαμβάνουν υπόψη τις αλλαγές που προκαλούνται από προηγούμενες κατανομές
- Ιεραρχική κλιμάκωση: Από τοπική αναζήτηση σε προοδευτικά ευρύτερες περιοχές του δικτύου



Εικόνα 3.2: Απεικόνιση της λειτουργικής ροής της μεθόδου κατανομής ***Neighbor-Aware***

Οι βασικές λειτουργικές πτυχές περιλαμβάνουν:

1. **Gateway Load Balancing**: Οι εφαρμογές κατανέμονται αρχικά μεταξύ των περιφερειακών gateways με βάση το τρέχον φορτίο τους, διασφαλίζοντας ότι κανένα μεμονωμένο gateway δεν γίνεται σημείο συμφόρησης.
2. **Επιλογή κόμβου με γνώμονα το Makespan**: Για κάθε υποψήφιο κόμβο, ο αλγόριθμος υπολογίζει την πιθανή αύξηση του makespan και επιλέγει τον κόμβο που ελαχιστοποιεί αυτή την αύξηση, εξισορροπώντας αποτελεσματικά το χρόνο εκτέλεσης σε όλο το δίκτυο.
3. **Προγραμματισμός Συγχρονισμού**: Ο αλγόριθμος λαμβάνει υπόψη τις δυνατότητες παράλληλης επεξεργασίας των κόμβων υπολογίζοντας θέσεις ταυτόχρονης εκτέλεσης με βάση τη χωρητικότητα CPU, επιτρέποντας πιο αποδοτική χρήση πόρων.
4. **Προοδευτική επέκταση του χώρου αναζήτησης**: Η διαδικασία κατανομής τριών βημάτων επεκτείνει συστηματικά το χώρο αναζήτησης από άμεσους γείτονες σε ευρύτερες περιοχές δικτύου και τελικά στο cloud, βελτιστοποιώντας τις τοπικές κατανομές όταν είναι δυνατόν ενώ διασφαλίζει ότι όλες οι εφαρμογές τελικά εξυπηρετούνται.
5. **Ολοκληρωμένη παρακολούθηση μετρικών**: Καθ' όλη τη διαδικασία κατανομής, η μέθοδος παρακολουθεί πολλαπλές μετρικές απόδοσης, παρέχοντας πληροφορίες για τη χρήση πόρων, το latency, την κατανάλωση ενέργειας και το κόστος.

Αυτή η ροή εργασίας αντιπροσωπεύει μια πρακτική προσέγγιση στην κατανομή φόρτου εργασίας σε περιβάλλοντα fog, εξισορροπώντας τους στόχους βελτιστοποίησης με θέματα υπολογιστικής αποδοτικότητας και κλιμάκωσης.

3.2 Κατανομή Particle Swarm Optimization (PSO)

3.2.1 Θεωρητικό Υπόβαθρο Αλγορίθμου PSO

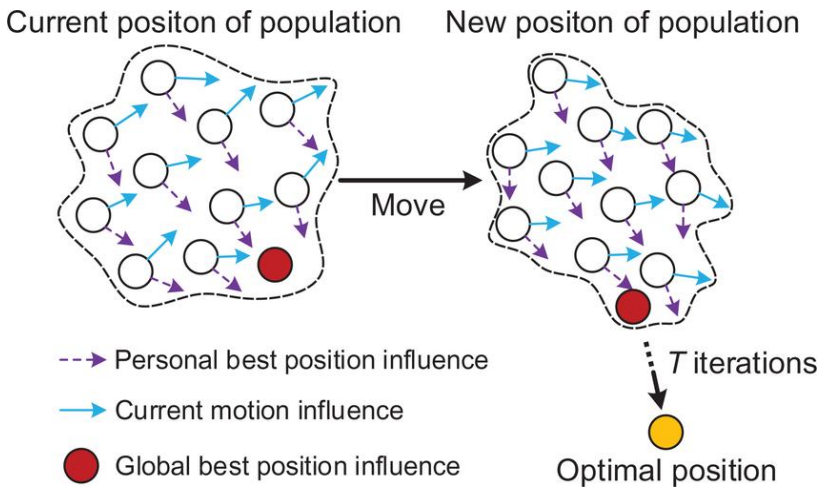
Το Particle Swarm Optimization (PSO) είναι μια στοχαστική τεχνική βελτιστοποίησης βασισμένη σε πληθυσμό εμπνευσμένη από την κοινωνική συμπεριφορά σμήνους πτηνών ή κοπαδιού ψαριών. Αναπτύχθηκε από τους Kennedy και Eberhart [4] και έχει γίνει ευρέως χρησιμοποιούμενη για την επίλυση σύνθετων προβλημάτων βελτιστοποίησης, ιδιαίτερα αυτών που περιλαμβάνουν πολλαπλούς στόχους και μεγάλους χώρους αναζήτησης.

Η θεμελιώδης έννοια πίσω από το PSO είναι η προσομοίωση ενός "σμήνους" σωματιδίων που εξερευνούν το χώρο λύσεων και μοιράζονται πληροφορίες για υποσχόμενες περιοχές. Κάθε σωματίδιο αντιπροσωπεύει μια υποψήφια λύση και κινείται μέσα στο χώρο αναζήτησης σύμφωνα με τη δική του εμπειρία (cognitive component) και την εμπειρία ολόκληρου του σμήνους (social component).

Τα βασικά συστατικά του αλγορίθμου PSO περιλαμβάνουν:

1. **Particles**: Μεμονωμένες υποψήφιες λύσεις που εξερευνούν τον χώρο αναζήτησης.
2. **Position**: Η θέση κάθε σωματιδίου στον χώρο αναζήτησης, που αντιπροσωπεύει μια συγκεκριμένη λύση στο πρόβλημα.
3. **Velocity**: Η κατεύθυνση και το μέγεθος της κίνησης ενός σωματιδίου μέσα στον χώρο αναζήτησης.
4. **Personal Best (pbest)**: Η καλύτερη λύση που βρέθηκε από κάθε μεμονωμένο σωματίδιο.
5. **Global Best (gbest)**: Η καλύτερη λύση που βρέθηκε από οποιοδήποτε σωματίδιο στο σμήνος.

Η Εικόνα 3.3 απεικονίζει τη βασική έννοια του PSO με σωματίδια που κινούνται προς βέλτιστες λύσεις.



Εικόνα 3.3: Οπτικοποίηση της κίνησης σωματιδίων PSO προς βέλτιστες λύσεις

3.2.2 Ρύθμιση Παραμέτρων

Η αποτελεσματικότητα της κατανομής βασισμένης σε PSO εξαρτάται σε μεγάλο βαθμό από την επιλογή κατάλληλων παραμέτρων αλγορίθμου. Ο Πίνακας 3.1 συνοψίζει τις βασικές παραμέτρους που χρησιμοποιούνται στην υλοποίηση και την επιρροή τους στη συμπεριφορά του αλγορίθμου.

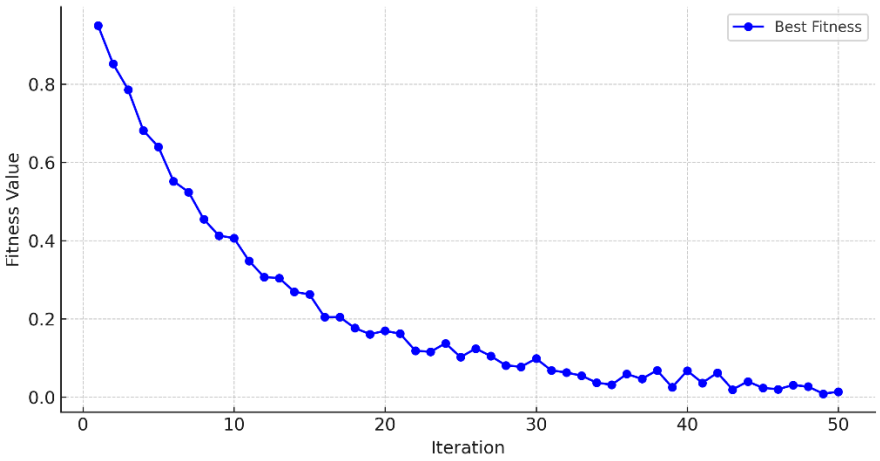
Πίνακας 3-1: Παράμετροι PSO και η επιρροή τους στη συμπεριφορά κατανομής

Παράμετρος	Επιρροή στη Συμπεριφορά Κατανομής
Αριθμός σωματιδίων	Εξισορροπεί το εύρος εξερεύνησης και την υπολογιστική αποδοτικότητα· κλιμακώνεται με το μέγεθος του προβλήματος
Μέγιστες επαναλήψεις	Περιορίζει το χρόνο σύγκλισης επιτρέποντας παράλληλα επαρκή βελτίωση
Inertia weight (w)	Μέτρια τιμή εξισορροπεί την εκμετάλλευση και την εξερεύνηση
Cognitive coefficient (c1)	Ενθαρρύνει τα σωματίδια να επιστρέφουν στις προσωπικές τους βέλτιστες θέσεις
Social coefficient (c2)	Ισχυρή έλξη προς το global best ενθαρρύνει τη σύγκλιση
Fitness function weights	Δίνει προτεραιότητα στην επιτυχία κατανομής (10000) και τη μείωση του makespan (5000) έναντι του κόστους (0,5) και της ενέργειας (0,5)

Αυτές οι παράμετροι επιλέχθηκαν με βάση προκαταρκτικούς πειραματισμούς και ανασκόπηση της βιβλιογραφίας εφαρμογών PSO σε προβλήματα κατανομής πόρων. Ο σχετικά υψηλός social coefficient (c2 = 3.0) σε σύγκριση με τον cognitive coefficient (c1 = 1.0) δίνει έμφαση στην ικανότητα καθολικής αναζήτησης του αλγορίθμου, βοηθώντας τον να συγκλίνει γρηγορότερα προς υποσχόμενες περιοχές του χώρου λύσεων.

Τα βάρη της συνάρτησης καταλληλότητας αντιπροσωπεύουν μια βασική πτυχή της ρύθμισης παραμέτρων, καθώς επηρεάζουν άμεσα τη συμπεριφορά βελτιστοποίησης πολλαπλών στόχων. Η υλοποίηση χρησιμοποιεί μια προσέγγιση σταθμισμένου αθροίσματος με σημαντικά υψηλότερα βάρη για την επιτυχία κατανομής (unallocated_penalty = 10000 ανά εφαρμογή) και το makespan (5000), μεσαίο βάρος για τη χρήση cloud (cloud_penalty = 1000) και χαμηλότερα βάρη για latency (10), κόστος (0,5) και κατανάλωση ενέργειας (0,5). Αυτό το σχήμα στάθμισης αντικατοπτρίζει την προτεραιότητα που δίνεται στην επιτυχία κατανομής και την απόδοση έναντι των οικονομικών παραμέτρων.

Η Εικόνα 3.4 δείχνει τη συμπεριφορά σύγκλισης του αλγορίθμου PSO σε διάφορες επαναλήψεις για ένα τυπικό σενάριο κατανομής.



Εικόνα 3.4: Τυπική συμπεριφορά σύγκλισης του αλγορίθμου κατανομής PSO που δείχνει τη βελτίωση του fitness σε επαναλήψεις

3.3 Συγκριτική Ανάλυση Μεθόδων Κατανομής

3.3.1 Αλγοριθμική Πολυπλοκότητα

Οι μέθοδοι κατανομής Neighbor-Aware και PSO διαφέρουν σημαντικά στην αλγοριθμική τους πολυπλοκότητα, η οποία επηρεάζει την κλιμάκωσή τους και το υπολογιστικό τους overhead:

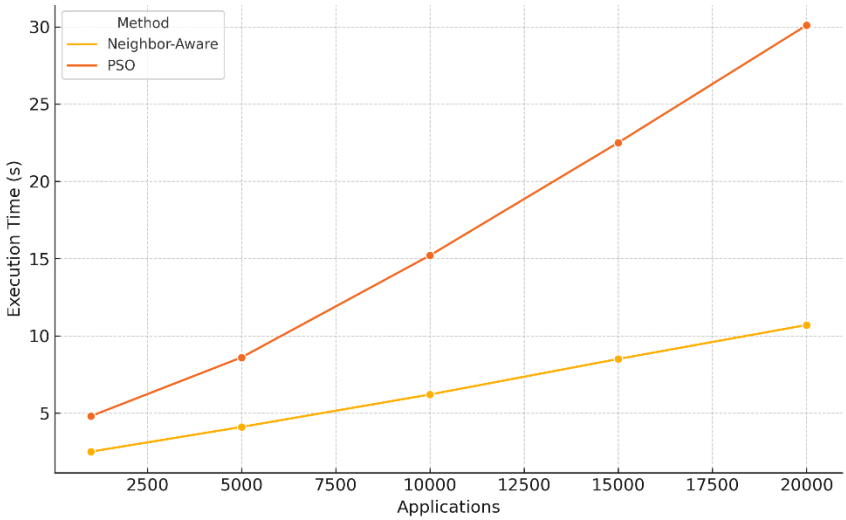
Μέθοδος Neighbor-Aware:

- **Χρονική Πολυπλοκότητα**: O(n×m), όπου n είναι ο αριθμός των εφαρμογών και m είναι ο αριθμός των κόμβων. Κάθε εφαρμογή απαιτεί την αξιολόγηση έως και m πιθανών κόμβων.
- **Χωρική Πολυπλοκότητα**: O(m), κυρίως για την παρακολούθηση των πόρων κόμβων, makespan και πλήθους εφαρμογών.
- **Συμπεριφορά Κλιμάκωσης**: Γραμμική αύξηση του χρόνου εκτέλεσης με τον αριθμό των εφαρμογών.

Μέθοδος Βασισμένη σε PSO:

- **Χρονική Πολυπλοκότητα**: O(p×i×n×m), όπου p είναι ο αριθμός των σωματιδίων, i είναι ο αριθμός των επαναλήψεων, n είναι ο αριθμός των εφαρμογών και m είναι ο αριθμός των κόμβων. Κάθε επανάληψη απαιτεί την αξιολόγηση όλων των σωματιδίων, με κάθε αξιολόγηση σωματιδίου να έχει πολυπλοκότητα O(n×m).
- **Χωρική Πολυπλοκότητα**: O(p×n + m), για την αποθήκευση θέσεων σωματιδίων, ταχυτήτων και πόρων κόμβων.
- **Συμπεριφορά Κλιμάκωσης**: Υπερ-γραμμική (Hyperlinear) αύξηση του χρόνου εκτέλεσης με τον αριθμό των εφαρμογών, ενδεχομένως περιορίζοντας την εφαρμογή της για σενάρια πολύ μεγάλης κλίμακας.

Η Εικόνα 3.5 απεικονίζει τις διαφορές στην κλιμάκωση του χρόνου εκτέλεσης μεταξύ των δύο μεθόδων.



Εικόνα 3.5: Κλιμάκωση χρόνου εκτέλεσης των μεθόδων κατανομής Neighbor-Aware και PSO με αυξανόμενο αριθμό εφαρμογών

Η υψηλότερη υπολογιστική πολυπλοκότητα της μεθόδου PSO αντιπροσωπεύει ένα trade-off (ανταλλαγή) μεταξύ ποιότητας λύσης και υπολογιστικού overhead.

Ενώ η μέθοδος Neighbor-Aware μπορεί να λάβει αποφάσεις κατανομής γρήγορα, ακόμη και για μεγάλους αριθμούς εφαρμογών, η μέθοδος PSO ενδεχομένως βρίσκει καλύτερες λύσεις κατανομής με κόστος αυξημένου χρόνου υπολογισμού.

Πίνακας 3-2: Συγκριτική αξιολόγηση των μεθόδων κατανομής PSO και Neighbor-Aware σε πέντε βασικά κριτήρια απόδοσης.

Κριτήριο	**PSO Allocation**	**Neighbor-Aware Allocation**	**Trade-off (Αντιστάθμισμα)**
Ποιότητα Κατανομής	Πολύ καλή – Βελτιστοποιημένη καθολικά	Μέτρια – Τοπικά βέλτιστη	PSO αποδίδει καλύτερα σε καθυστέρηση και κόστος
Υπολογιστικός Χρόνος	Υψηλός – Πολλαπλές επαναλήψεις & πληθυσμοί	Πολύ χαμηλός – Άμεσες αποφάσεις με γείτονες	PSO είναι ~10–12x πιο αργός σε μεγάλα σενάρια
Υπολογιστικό Overhead	Βαρύς – Χρήζει περισσότερης μνήμης και CPU	Ελαφρύς – Κατάλληλος για real-time edge συστήματα	PSO απαιτεί περισσότερους πόρους για εκτέλεση
Εκμετάλλευση Πόρων	Υψηλή – Οδηγεί σε βέλτιστη χρήση fog κόμβων	Χαμηλότερη – Περισσότερα tasks στέλνονται στο cloud	PSO έχει καλύτερη κατανομή σε διαθέσιμους κόμβους
Προσαρμοστικότητα	Καλή – Μπορεί να ρυθμιστεί για πολλαπλά κριτήρια	Περιορισμένη – Στατική λογική βασισμένη σε γειτονία	PSO προσφέρει περισσότερη ευελιξία

Κλίμακα Αξιολόγησης:

- Πολύ Υψηλή/Πολύ Καλή: >80% βελτίωση ή άριστη απόδοση
- Υψηλή/Καλή: 60-80% βελτίωση ή καλή απόδοση
- Μέτρια: 40-60% ή μέση απόδοση
- Χαμηλή: 20-40% ή υποδεέστερη απόδοση
- Πολύ Χαμηλή: <20% ή ανεπαρκής απόδοση

Οι αξιολογήσεις βασίζονται σε πειραματικά δεδομένα και θεωρητική ανάλυση από τα Κεφάλαια 5 και 6.

3.3.2 Διαδικασία Λήψης Αποφάσεων

Η θεμελιώδης διαφορά μεταξύ των δύο μεθόδων κατανομής βρίσκεται στην προσέγγισή τους στη λήψη αποφάσεων:

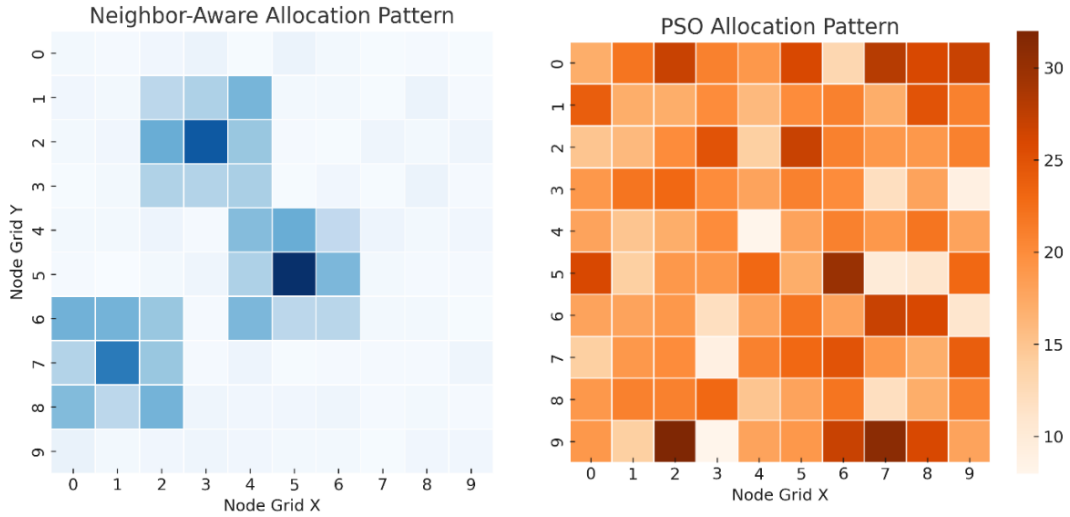
Μέθοδος Neighbor-Aware:

- **Τοπική Βελτιστοποίηση**: Λαμβάνει greedy αποφάσεις για κάθε εφαρμογή μεμονωμένα, βελτιστοποιώντας την τοπική αύξηση του makespan.
- **Διαδοχική Επεξεργασία**: Επεξεργάζεται τις εφαρμογές μία προς μία, με τις προηγούμενες κατανομές να επηρεάζουν τους διαθέσιμους πόρους για μεταγενέστερες εφαρμογές.
- **Προσδιοριστική Συμπεριφορά**: Παράγει συνεπή αποτελέσματα για πανομοιότυπες εισόδους, ακολουθώντας μια σταθερή διαδρομή απόφασης.
- **Περιορισμένη Προβλεπτικότητα**: Δεν μπορεί να προβλέψει τον αντίκτυπο των τρέχουσων αποφάσεων σε μελλοντικές κατανομές.

Μέθοδος Βασισμένη σε PSO:

- **Καθολική Βελτιστοποίηση**: Αξιολογεί πλήρεις λύσεις κατανομής (όλες τις εφαρμογές ταυτόχρονα), επιτρέποντας τη βελτιστοποίηση συστημικών στόχων.
- **Παράλληλη Εξερεύνηση**: Εξερευνά ταυτόχρονα πολλαπλές δυνατότητες κατανομής μέσω της συμπεριφοράς του σμήνους σωματιδίων.
- **Στοχαστική Συμπεριφορά**: Ενσωματώνει τυχαιότητα στη διαδικασία αναζήτησης, ενδεχομένως ανακαλύπτοντας διαφορετικά μοτίβα λύσεων.
- **Βελτίωση Λύσης**: Βελτιώνει επαναληπτικά τις λύσεις κατανομής με βάση την αξιολόγηση καταλληλότητας και τη νοημοσύνη σμήνους.

Αυτές οι διαφορές στην προσέγγιση λήψης αποφάσεων οδηγούν σε διαφορετικά μοτίβα κατανομής:



Εικόνα 3.6: Σύγκριση μοτίβων κατανομής από τις μεθόδους Neighbor-Aware και PSO

***Σκουρότερο** = περισσότερες εφαρμογές κατανεμημένες, **Ανοιχτότερο** = λιγότερες εφαρμογές κατανεμημένες.*

Neighbor-Aware → γρήγορο, τοπικό, αλλά μη ισορροπημένο

PSO → πιο αργό, αλλά πολύ καλύτερο στην κατανομή του φορτίου

3.3.3 Διατύπωση Αντικειμενικής Συνάρτησης

Μια άλλη σημαντική διαφορά μεταξύ των μεθόδων είναι η προσέγγισή τους στο χειρισμό πολλαπλών στόχων βελτιστοποίησης:

Μέθοδος Neighbor-Aware:

- **Πρωτεύων Στόχος**: Ελαχιστοποιεί την αύξηση του makespan για κάθε απόφαση κατανομής.
- **Έμμεσοι Στόχοι**: Έμμεσα λαμβάνει υπόψη τη χρήση πόρων μέσω ελέγχων ικανοποίησης και υπολογισμού του makespan.
- **Ιεραρχική Δομή Απόφασης**: Χρησιμοποιεί μια διαδικασία τριών βημάτων που έμμεσα δίνει προτεραιότητα στην κατανομή σε κόμβους fog έναντι της κατανομής σε cloud.

Μέθοδος Βασισμένη σε PSO:

- **Πολλαπλοί Στόχοι**: Συνδυάζει άμεσα πολλαπλούς στόχους στη συνάρτηση καταλληλότητας με συγκεκριμένα βάρη.
- **Ρυθμιζόμενες Προτεραιότητες**: Επιτρέπει την προσαρμογή των βαρών των στόχων λαμβάνοντας υπόψιν διαφορετικές πτυχές της κατανομής.
- **Ολοκληρωμένη Αξιολόγηση**: Λαμβάνει υπόψη όλους τους στόχους ταυτόχρονα κατά τη σύγκριση λύσεων κατανομής.

Η συνάρτηση καταλληλότητας στη μέθοδο PSO παρέχει ρητό έλεγχο των προτεραιοτήτων βελτιστοποίησης μέσω των ακόλουθων σταθμισμένων συστατικών:

$$fitness = unallocated\_penalty + makespan \cdot 5000 + total\_cost \cdot 0.5 + total\_latency \cdot 10 + total\_energy \cdot 0.5 + cloud\_penalty \cdot 1000$$

Αυτή η διατύπωση επιτρέπει ευέλικτη ιεράρχηση των στόχων, με την τρέχουσα υλοποίηση να δίνει βάρος στην επιτυχία της κατανομής και τη μείωση του makespan έναντι άλλων παραμέτρων.

Τα βάρη της συνάρτησης fitness (w_1=10000, w_2=5000, w_3=0.5, w_4=10, w_5=0.5, w_6=1000) προσδιορίστηκαν μέσω συνδυασμού εμπειρικής ρύθμισης και ανάλυσης κλίμακας τιμών.

Η επιλογή ακολούθησε την ιεραρχία προτεραιοτήτων:

- Εγγύηση επιτυχούς κατανομής,
- Βελτιστοποίηση απόδοσης συστήματος,
- Προτίμηση fog έναντι cloud επεξεργασίας, και
- Ελαχιστοποίηση λειτουργικών παραμέτρων.

Μεγαλύτερο βάρος = πιο σημαντικό

Παράδειγμα: Επιτυχία κατανομής (10000) > Απόδοση (5000) > Κόστος (0.5)

Δοκιμάζουμε διάφορους συνδυασμούς και κρατάμε αυτόν που δίνει καλύτερα αποτελέσματα.

Τα πειράματα επιβεβαίωσαν ότι αυτή η σταθμισμένη προσέγγιση παράγει ισορροπημένες λύσεις που ικανοποιούν τους στόχους βελτιστοποίησης.

3.3.4 Πλεονεκτήματα και Περιορισμοί

Πλεονεκτήματα της Μεθόδου Neighbor-Aware:

1. **Υπολογιστική Αποδοτικότητα**: Χαμηλότερο υπολογιστικό overhead την καθιστά κατάλληλη για αποφάσεις κατανομής σε πραγματικό χρόνο σε δυναμικά περιβάλλοντα.
2. **Κλιμάκωση**: Γραμμική χρονική πολυπλοκότητα επιτρέπει εφαρμογή σε σενάρια μεγάλης κλίμακας.
3. **Προσδιορισμός**: Παράγει συνεπή, προβλέψιμα μοτίβα κατανομής, διευκολύνοντας το σχεδιασμό και την ανάλυση συστημάτων.
4. **Απλότητα Υλοποίησης**: Απλούστερη δομή αλγορίθμου απλοποιεί την υλοποίηση και την αποσφαλμάτωση.

Περιορισμοί της Μεθόδου Neighbor-Aware:

1. **Τοπικά Βέλτιστα**: Η greedy λήψη αποφάσεων μπορεί να οδηγήσει σε μη βέλτιστες λύσεις.
2. **Περιορισμένο Εύρος Βελτιστοποίησης**: Εστιάζει κυρίως στην ελαχιστοποίηση του makespan, μειώνοντας την ενέργεια και το κόστος.
3. **Εξάρτηση**: Οι αρχικές αποφάσεις κατανομής μπορεί να αδικήσουν την κατανομή άλλων εφαρμογών.
4. **Σταθερή Δομή Απόφασης**: Η σταθερή στρατηγική κατανομής μπορεί να μην προσαρμόζεται καλά σε διαφορετικές απαιτήσεις άλλων σεναρίων.

Πλεονεκτήματα της Μεθόδου PSO:

1. **Καθολική Βελτιστοποίηση**: Εξετάζει ολόκληρα μοτίβα κατανομής, βρίσκοντας καλύτερες συστημικές λύσεις.
2. **Εξισορρόπηση Πολλαπλών Στόχων**: Εξισορροπεί πολλαπλούς στόχους μέσω σταθμισμένης συνάρτησης καταλληλότητας.
3. **Ποικιλομορφία Λύσης**: Η στοχαστική φύση επιτρέπει την εξερεύνηση διαφορετικών μοτίβων κατανομής.
4. **Προσαρμοστικότητα**: Η ρύθμιση παραμέτρων επιτρέπει την προσαρμογή σε διαφορετικά σενάρια.

Περιορισμοί της Μεθόδου Βασισμένης σε PSO:

1. **Υπολογιστικό Overhead (υπολογιστική επιβάρυνση)**: Υψηλότερη υπολογιστική πολυπλοκότητα περιορίζει την εφαρμογή σε σενάρια πραγματικού χρόνου ή πολύ μεγάλης κλίμακας.
2. **Ευαισθησία Παραμέτρων**: Η απόδοση εξαρτάται από την κατάλληλη ρύθμιση των παραμέτρων του αλγορίθμου.
3. **Ζητήματα Σύγκλισης**: Μπορεί να μην συγκλίνει πλήρως εντός του ορίου επαναλήψεων για σύνθετα σενάρια.
4. **Πολυπλοκότητα Υλοποίησης**: Πιο σύνθετη δομή αλγορίθμου αυξάνει την πολυπλοκότητα υλοποίησης και αποσφαλμάτωσης.

Πίνακας 3-3: Ο Πίνακας συνοψίζει τις βασικές διαφορές μεταξύ των δύο μεθόδων κατανομής. Σύγκριση των μεθόδων κατανομής Neighbor-Aware και PSO

Πτυχή	Μέθοδος Neighbor-Aware	Μέθοδος Βασισμένη σε PSO
Προσέγγιση Απόφασης	Greedy, διαδοχική	Καθολική, παράλληλη
Εύρος Βελτιστοποίησης	Τοπικό (ανά εφαρμογή)	Καθολικό (ολόκληρη κατανομή)
Χρονική Πολυπλοκότητα	O(n×m)	O(p×i×n×m)
Πρωτεύουσα Βελτιστοποίηση	Ελαχιστοποίηση makespan	Πολλαπλοί στόχοι (σταθμισμένοι)
Προσδιορισμός	Προσδιοριστική	Στοχαστική
Ικανότητα Προσαρμογής	Περιορισμένη	Παραμετροποιήσιμη
Πολυπλοκότητα Υλοποίησης	Χαμηλότερη	Υψηλότερη
Κατάλληλα Σενάρια	Πραγματικού χρόνου, δυναμικά	Σχεδιασμού, εστιασμένα στη βελτιστοποίηση

Αυτά τα πλεονεκτήματα και περιορισμοί υποδηλώνουν ότι η επιλογή μεταξύ μεθόδων κατανομής θα πρέπει να εξαρτάται από συγκεκριμένες απαιτήσεις ανάπτυξης, με τη μέθοδο Neighbor-Aware να προσφέρει αποδοτικότητα και απλότητα για σενάρια πραγματικού χρόνου και τη μέθοδο PSO να παρέχει ενδεχομένως καλύτερη ποιότητα κατανομής για εφαρμογές εστιασμένες στο σχεδιασμό και τη βελτιστοποίηση.

3.3.5 Σύγκριση με Άλλες Προσεγγίσεις Ευφυούς Κατανομής

Οι μέθοδοι κατανομής που αναπτύχθηκαν σε αυτή την εργασία μπορούν να τοποθετηθούν στο ευρύτερο τοπίο των τεχνικών ευφυούς κατανομής φόρτου εργασίας για περιβάλλοντα fog. Ιδιαίτερα, οι Abbasi et al. [8] πρότειναν ένα πλαίσιο ευφυούς κατανομής φόρτου εργασίας για αρχιτεκτονικές IoT-Fog-Cloud που χρησιμοποιεί μια προσέγγιση λήψης αποφάσεων πολλαπλών κριτηρίων βασισμένη στη μέθοδο TOPSIS. Η προσέγγισή τους λαμβάνει υπόψη πολλαπλά χαρακτηριστικά, συμπεριλαμβανομένων της CPU, της μνήμης, του εύρους ζώνης και της κατανάλωσης ενέργειας κατά τη λήψη αποφάσεων κατανομής.

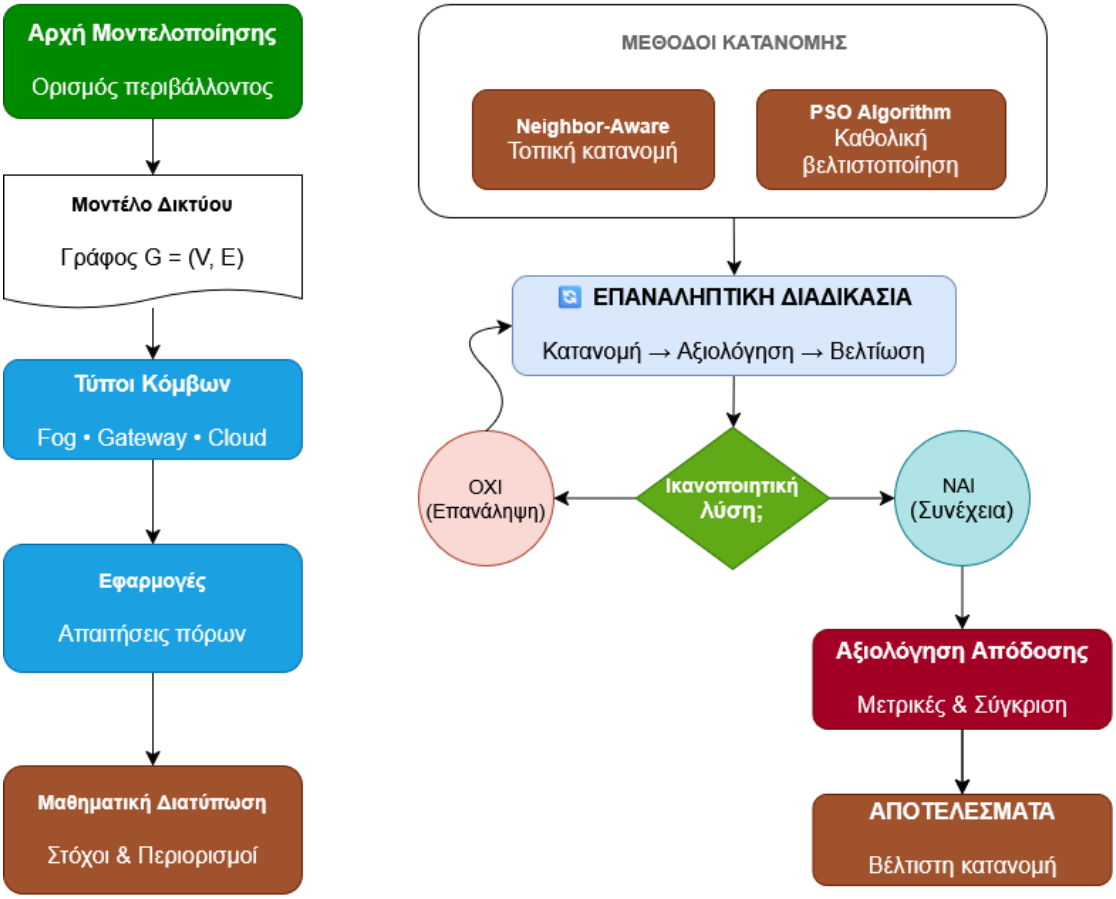
Σε αντίθεση με τη μέθοδο PSO που διερευνά ταυτόχρονα πολλαπλές υποψήφιες λύσεις, η προσέγγιση που βασίζεται στο TOPSIS των Abbasi et al. [8] αξιολογεί εναλλακτικές λύσεις έναντι μιας ιδανικής λύσης. Το πλαίσιό τους πέτυχε σημαντικές βελτιώσεις στο χρόνο απόκρισης (20,1%) και στην κατανάλωση ενέργειας (29,3%) σε σύγκριση με τις βασικές προσεγγίσεις. Αυτά τα αποτελέσματα είναι συγκρίσιμα με τα οφέλη απόδοσης που παρατηρήθηκαν με τη μέθοδο PSO, αν και η προσέγγισή μας επιδεικνύει πρόσθετα πλεονεκτήματα στο ποσοστό επιτυχίας κατανομής και στη βελτιστοποίηση της χρήσης του cloud.

Η εργασία των Abbasi et al. [8] ενισχύει την αξία της ευφυούς βελτιστοποίησης πολλαπλών στόχων για την κατανομή φόρτου εργασίας σε ετερογενή περιβάλλοντα fog, ένα συμπέρασμα που συμπίπτει με τα ευρήματά μας σχετικά με την υπεροχή της μεθόδου PSO έναντι απλούστερων ευρετικών προσεγγίσεων .

4 Μοντελοποίηση του Προβλήματος Κατανομής Φόρτου Εργασίας

4.1 Εισαγωγή

Η αποτελεσματική κατανομή φόρτου εργασίας σε περιβάλλοντα fog computing απαιτεί προσεκτική μοντελοποίηση, λαμβάνοντας υπόψη την διαφορετικότητα των πόρων, τους περιορισμούς του δικτύου και τις διαφορετικές απαιτήσεις των εφαρμογών. Αυτό το κεφάλαιο παρουσιάζει μια ολοκληρωμένη προσέγγιση για τη δομική και μαθηματική μοντελοποίηση του προβλήματος.



*(*Εικόνα *4.1: Συνολική ροή μοντελοποίησης του προβλήματος κατανομής φόρτου εργασίας σε περιβάλλοντα υπολογιστικής ομίχλης)*

4.2 Μοντελοποίηση του Περιβάλλοντος Fog Computing

Η αρχιτεκτονική του συστήματος fog computing (Εικόνα 4.1) περιλαμβάνει τρία επίπεδα: το επίπεδο των συσκευών (IoT), το επίπεδο fog και το επίπεδο cloud. Για τη μοντελοποιησή του θα αναπτύξουμε ένα θεωρητικό μοντέλο που αποτυπώνει το δίκτυο και τη συμπεριφορά των οντοτήτων του για διάφορα σενάρια χρήσης.

4.2.1 Μεγέθη και συμβολισμοί του μοντέλου

Για διευκόλυνση του αναγνώστη, στον παρακάτω πίνακα συνοψίζονται τα μεγέθη και οι αντίστοιχοι συμβολισμοί που θα χρησιμοποιηθούν στη συνέχεια για την περιγραφή του μοντέλου.

Πίνακας 4-1: Μεγέθη και οι αντίστοιχοι συμβολισμοί

Σύμβολο	**Περιγραφή**	**Σύμβολο**	**Περιγραφή**
$G = (V, E)$	Γράφος δικτύου fog computing	V	Σύνολο κόμβων δικτύου
E	Σύνολο ακμών (συνδέσεων) δικτύου	$n = \|V\|$	Αριθμός κόμβων στο δίκτυο
$m = \|E\|$	Αριθμός ακμών στο δίκτυο	v, u	Μεμονωμένοι κόμβοι
$e(u,v)$	Ακμή που συνδέει κόμβους u και v	$Type_v$	Τύπος κόμβου v (FN, GW, MGW, CN)
p	Πιθανότητα δημιουργίας ακμής (Erdős-Rényi)	$p_{gateway}$	Ποσοστό κόμβων που ορίζονται ως gateways
CPU_v	Υπολογιστική ισχύς κόμβου v	RAM_v	Διαθέσιμη μνήμη κόμβου v
$Storage_v$	Διαθέσιμος αποθηκευτικός χώρος κόμβου v	CPU_C, RAM_C, $STORAGE_C$	Πρακτικά απεριόριστοι πόροι cloud
$propagation\_delay_C$	Καθυστέρηση διάδοσης προς cloud	$Bandwidth_c$	Εύρος ζώνης cloud
$Bandwidth_g$	Εύρος ζώνης main gateway	$Runtime_g$	Χρόνος απόκρισης gateway
$Coords_v$	Γεωγραφικές συντεταγμένες κόμβου v	$Bandwidth_e$	Εύρος ζώνης ακμής e
$Latency_e$	Καθυστέρηση διάδοσης ακμής e	$Reliability_e$	Αξιοπιστία σύνδεσης ακμής e
A	Σύνολο εφαρμογών	A	Μεμονωμένη εφαρμογή
$k = \|A\|$	Αριθμός εφαρμογών	CPU_a	Απαιτούμενη υπολογιστική ισχύς εφαρμογής a
RAM_a	Απαιτούμενη μνήμη εφαρμογής a	$Storage_a$	Απαιτούμενος αποθηκευτικός χώρος εφαρμογής a
$Runtime_a$	Χρόνος εκτέλεσης εφαρμογής a	$MsgSize_a$	Μέγεθος μηνυμάτων επικοινωνίας εφαρμογής a
$x(a,v)$	Δυαδική μεταβλητή κατανομής	$Apps_v$	Σύνολο εφαρμογών στον κόμβο v
$N_v = \|Apps_v\|$	Πλήθος εφαρμογών σε κόμβο v	S_v	Αριθμός concurrent slots κόμβου v
L	Παράμετρος υπολογισμού concurrent slots	$BW(v)$	Ελάχιστο bandwidth διαδρομής από κόμβο v
$AppMakespan(a,v)$	Χρόνος εκτέλεσης εφαρμογής a σε κόμβο v	M_v	Makespan κόμβου v
$f_{makespan}(x)$	Συνολικός makespan συστήματος	$P^{idle}v$	Κατανάλωση ενέργειας αδρανούς κατάστασης κόμβου v
$P_{processing}$	Κατανάλωση ενέργειας επεξεργασίας	$P^{transmission}$	Κατανάλωση ενέργειας μετάδοσης
$f_{energy}(x)$	Συνολική κατανάλωση ενέργειας	UC_{energy}	Κόστος μονάδας ενέργειας
UC_{cpu}	Κόστος μονάδας CPU	UC_{ram}	Κόστος μονάδας RAM
$UC_{storage}$	Κόστος μονάδας αποθήκευσης	$f_{cost}(x)$	Συνολικό λειτουργικό κόστος

f(x)	Αντικειμενική συνάρτηση (fitness function)	$w_1, w_2, \ldots, w_6$	Βάρη της αντικειμενικής συνάρτησης
$f_{latency}(x)$	Μέση καθυστέρηση επικοινωνίας	$f_{unallocated}(x)$	Ποινή για μη κατανεμημένες εφαρμογές
$f_{cloud\_penalty}(x)$	Ποινή για χρήση cloud	$p_i(t)$	Θέση σωματιδίου i στην επανάληψη t
$v_i(t)$	Ταχύτητα σωματιδίου i στην επανάληψη t	$pbest_i$	Καλύτερη προσωπική θέση σωματιδίου i
gbest	Καλύτερη καθολική θέση σμήνους	W	Βάρος αδράνειας
c_1	Γνωστικός συντελεστής	c_2	Κοινωνικός συντελεστής
r_1, r_2	Τυχαίοι αριθμοί στο [0,1]	BC(v)	Betweenness centrality κόμβου v
σ_{st}	Αριθμός συντομότερων διαδρομών από s σε t	$\sigma_{st}(v)$	Συντομότερες διαδρομές από s σε t μέσω v
$MaxLatency_a$	Μέγιστη επιτρεπόμενη καθυστέρηση για εφαρμογή a	$A_{critical}$	Υποσύνολο εφαρμογών με αυστηρές απαιτήσεις

Η υποδομή του μοντελοποιείται ως ένας μη κατευθυνόμενος γράφος G = (V, E), όπου:

- $\boldsymbol{V} = \{v_1, v_2, \ldots, v_n\}$: Το σύνολο των κόμβων, που αντιπροσωπεύουν τις συσκευές (fog nodes, gateways, cloud).Το μέγεθος του συστήματος συμβολίζεται ως $n = |V|$.
- $\boldsymbol{E} = \{e_1, e_2, \ldots, e_m\}$: Το σύνολο των ακμών, που αντιπροσωπεύουν τις απευθείας συνδέσεις μεταξύ των κόμβων του δικτύου. Μία ακμή $e(u, v) \in E$ συνδέει απευθείας τους κόμβους u και v.

Κατηγορίες Κόμβων

Οι κόμβοι ταξινομούνται σε τέσσερις κατηγορίες:

Fog Nodes (FN): Κόμβοι με μέτριους υπολογιστικούς πόρους για τοπική επεξεργασία, **σημαντικά χαμηλότερους από τους πόρους cloud** (περίπου 0.04 - 0.2% της υπολογιστικής ισχύος cloud) αλλά επαρκείς για τοπική επεξεργασία.

Gateway Nodes (GW): Κόμβοι που λειτουργούν ως διαμεσολαβητές μεταξύ fog nodes **με παρόμοιες υπολογιστικές δυνατότητες με τους Fog Nodes** αλλά εξειδικευμένο ρόλο συντονισμού και προώθησης (δεν κάνουν επεξεργασία).

Main Gateway Node (MGW): **Κεντρικός κόμβος υψηλότερης δυνατότητας** που συνδέεται απευθείας με το Cloud, με ενισχυμένες δικτυακές δυνατότητες:

- Bandwidth: 1,000 Mbps (έναντι 100-1,000 Mbps των fog nodes)
- Runtime: 1,000 sec εγγυημένου χρόνου λειτουργίας

Cloud Node (CN): Απομακρυσμένος κόμβος με **πρακτικά απεριόριστους πόρους** (>500x υψηλότερους από fog nodes), **υψηλή καθυστέρηση** (200ms έναντι 1-10ms των fog nodes) και **υψηλό κόστος επικοινωνίας**.

4.2.2 Χαρακτηριστικά Κόμβων

Κάθε κόμβος v στο σύνολο V περιγράφεται από:

- $\boldsymbol{CPU_v}$: Υπολογιστική ισχύς (MIPS)
- $\boldsymbol{RAM_v}$: Διαθέσιμη μνήμη (MB)
- $\boldsymbol{Storage_v}$: Διαθέσιμος αποθηκευτικός χώρος (MB)
- $\boldsymbol{Type_v}$: Τύπος κόμβου (FN, GW, MGW, CN)
- $\boldsymbol{Coords_v}$: Γεωγραφικές συντεταγμένες (όπου εφαρμόζεται)

4.2.3 Χαρακτηριστικά Συνδέσεων

Κάθε ακμή e(u,v) στο E που συνδέει κόμβους u και v περιγράφεται από:

- $\boldsymbol{Bandwidth_e}$: Εύρος ζώνης (Mbps)
- $\boldsymbol{Latency_e}$: Καθυστέρηση διάδοσης (δευτερόλεπτα)
- $\boldsymbol{Reliability_e}$: Αξιοπιστία σύνδεσης (ποσοστό)

4.3 Αναπαράσταση Τοπολογίας Δικτύου

Για την αναπαραγωγή της τοπολογίας του δικτύου, δηλαδή του τρόπου με τον οποίο οι κόμβοι του δικτύου συνδέονται μεταξύ τους, θεωρούμε το μοντέλο Erdős-Rényi (ER) [12], το οποίο παράγει τυχαίους γράφους G(n,E). Σε έναν τυχαίο γράφο μεγέθους n, κάθε ακμή που περιλαμβάνεται στο γράφημα επιλέγεται με πιθανότητα p και ανεξάρτητα από όλες τις άλλες ακμές, από τις συνολικά $\binom{n}{2}$ διαθέσιμες ακμές. Η τιμή της παραμέτρου p καθορίζει την πυκνότητα (σε ακμές) και τη συμπεριφορά του δικτύου.

Σε αυτό το μοντέλο:

- **n**: Αριθμός κόμβων στο δίκτυο
- **p**: Πιθανότητα δημιουργίας ακμής μεταξύ οποιωνδήποτε δύο κόμβων

Μαθηματικές Ιδιότητες:

Αναμενόμενος βαθμός κόμβου: ⟨k⟩ = p(n-1)

Κατώφλι συνδεσιμότητας: p > ln(n)/n για συνδεδεμένο γράφο

Επιλογή Gateway Nodes:

Χρησιμοποιείται η μετρική **betweenness centrality** για τον εντοπισμό στρατηγικά σημαντικών κόμβων:

$$BC(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

όπου σ_{st} είναι ο αριθμός συντομότερων διαδρομών από τον κόμβο s στον κόμβο t, και $\sigma_{st}(v)$ ο αριθμός των μονοπατιών που διέρχονται από τον κόμβο v.

4.4 Μοντελοποίηση Εφαρμογών

Οι εφαρμογές αναπαρίστανται ως σύνολο $A = \{a_1, a_2, \dots, a_k\}$. Κάθε εφαρμογή a στο A περιγράφεται από:

- $\boldsymbol{CPU_a}$: Απαιτούμενη υπολογιστική ισχύς (MIPS)
- $\boldsymbol{RAM_a}$: Απαιτούμενη μνήμη (MB)
- $\boldsymbol{Storage_a}$: Απαιτούμενος αποθηκευτικός χώρος (MB)
- $\boldsymbol{Runtime_a}$: Χρόνος εκτέλεσης (δευτερόλεπτα)
- $\boldsymbol{MsgSize_a}$: Μέγεθος μηνυμάτων επικοινωνίας (MB)

4.5 Αναλυτική Μοντελοποίηση του Προβλήματος Κατανομής

Έστω η αρχιτεκτονική του συστήματος fog computing που περιγράφεται στην Ενότητα 3.2 και αποτελείται από ένα σύνολο κόμβων υπολογισμού $\boldsymbol{V} = \{v_1, v_2, \dots, v_n\}$. Θεωρούμε τα δύο υψηλότερα επίπεδα ενός συστήματος, και συγκεκριμένα το επίπεδο fog και το επίπεδο cloud, στο οποίο φθάνουν αιτήματα για επεξεργασία από ένα σύνολο εφαρμογών $A = \{a_1, a_2, \dots, a_k\}$. που εκτελούνται στις τελικές συσκευές. Τα αιτήματα επεξεργασίας λαμβάνονται από τους κόμβους τύπου Gateway. Το πρόβλημα ανάγεται στην εύρεση τρόπου τοποθέτησης των εφαρμογών προς επεξεργασία, στους διαθέσιμους πόρους του συστήματος έτσι ώστε να:

- μεγιστοποιείται η αξιοποίηση των κόμβων fog ενώ ικανοποιούνται οι απαιτήσεις των εφαρμογών,
- μοιράζονται εξίσου τους πόρους του συστήματος,
- ελαχιστοποιείται το κόστος,
- ελαχιστοποιείται η κατανάλωση ενέργειας

4.5.1 Μεταβλητές Απόφασης

Το πρόβλημα κατανομής φόρτου εργασίας μοντελοποιείται ως ακολούθως: Έστω ένα σύστημα με n κόμβους και m εφαρμογές προς εκτέλεση. Να προσδιοριστεί το σύνολο των κόμβων που θα αναλάβει την εκτέλεση των εφαρμογών έτσι ώστε η κατανομή των m εφαρμογών στα διαθέσιμα fog nodes να μεγιστοποιεί την εξίσωση (1).

$$\max\left(\sum_{i=1}^{m} a_i \, x(a, v)\right) \qquad (1)$$

όπου ο όρος x(a,v) αποτελεί την κεντρική μεταβλητή απόφασης του προβλήματος. Για κάθε ζεύγος εφαρμογής a και κόμβου v, η μεταβλητή παίρνει την τιμή 1 αν η εφαρμογή κατανέμεται στον συγκεκριμένο κόμβο, και 0 σε αντίθετη περίπτωση, δηλαδή

$$x(a, v) = \begin{cases} 1, \text{η εφαρμογή a εκχωρήθηκε στον κόμβο v} \\ 0, \text{αλλιώς} \end{cases} \quad \forall a \in A, \forall v \in V$$

Για κάθε ζεύγος εφαρμογής a και κόμβου v, η μεταβλητή παίρνει την τιμή 1 αν η εφαρμογή κατανέμεται στον συγκεκριμένο κόμβο, και 0 σε αντίθετη περίπτωση.

4.5.2 Περιορισμοί

Το πρόβλημα κατανομής υπόκειται στους ακόλουθους περιορισμούς:

Μοναδική Ανάθεση Εφαρμογής: Κάθε εφαρμογή a κατανέμεται σε έναν και μόνο έναν κόμβο του δικτύου.

$$\sum_{v \in V} x(a, v) = 1, \forall a \in A$$

Μπορεί δύο ή περισσότερες εφαρμογές στο ίδιο node.

Αυτός ο περιορισμός διασφαλίζει ότι κάθε εφαρμογή εκχωρείται ακριβώς σε έναν κόμβο. Το άθροισμα των μεταβλητών απόφασης για κάθε εφαρμογή σε όλους τους κόμβους πρέπει να ισούται με 1, αποκλείοντας τη δυνατότητα μη κατανομής ή πολλαπλής κατανομής της ίδιας εφαρμογής.

Περιορισμοί Χωρητικότητας Πόρων: Οι συνολικές απαιτήσεις των εφαρμογών που κατανέμονται σε κάθε κόμβο δεν μπορούν να υπερβαίνουν τη διαθέσιμη χωρητικότητά του.

- CPU: $\sum_{a \in A} x(a, v) \cdot CPU_a \leq CPU_v, \forall v \epsilon V$
- RAM: $\sum_{a \in A} x(a, v) \cdot RAM_a \leq RAM_v, \forall v \epsilon V$
- Storage: $\sum_{a \in A} x(a, v) \cdot Storage_a \leq Storage_v, \forall v \epsilon V$

Για κάθε τύπο πόρου (CPU, RAM, Storage) και κάθε κόμβο, το άθροισμα των απαιτήσεων όλων των κατανεμημένων εφαρμογών δεν πρέπει να ξεπερνάει τη διαθέσιμη χωρητικότητα του κόμβου. Αυτό αποτρέπει την υπερδέσμευση πόρων.

Καθυστέρηση (για κρίσιμες εφαρμογές):

$$x(a, v) * \ Latency(a, v) \leq \ MaxLatency_a, \forall \ a \epsilon A_{critical} \forall \ v \epsilon \ V.$$

όπου A_critical το υποσύνολο των εφαρμογών με αυστηρές απαιτήσεις καθυστέρησης.

Αυτός ο περιορισμός εφαρμόζεται μόνο για εφαρμογές κρίσιμες στον χρόνο που ανήκουν στο σύνολο A_critical. Διασφαλίζει ότι αν μια κρίσιμη εφαρμογή κατανεμηθεί σε έναν κόμβο, η καθυστέρηση επικοινωνίας δεν θα υπερβεί το μέγιστο επιτρεπόμενο όριο για αυτή την εφαρμογή.

Σημείωση: Οι κόμβοι μπορούν να εξυπηρετήσουν πολλές εφαρμογές ταυτόχρονα, υπό την προϋπόθεση ότι οι συνολικές απαιτήσεις πόρων δεν υπερβαίνουν τη διαθέσιμη χωρητικότητά τους. Ο παραλληλισμός εκτέλεσης μοντελοποιείται μέσω των concurrent slots που υπολογίζονται βάσει της διαθέσιμης CPU ισχύος κάθε κόμβου.

4.5.3 Υπολογισμοί Παραμέτρων

Χρόνος Εκτέλεσης Εφαρμογής:

$$AppMakespan(a, v) = Runtime_a + (Msg\_Size_a \times 8) / BW(v)$$

Ο χρόνος εκτέλεσης μιας εφαρμογής αποτελείται από δύο συνιστώσες: τον καθαρό χρόνο επεξεργασίας (Runtime_a) και τον χρόνο μετάδοσης δεδομένων. Ο χρόνος μετάδοσης υπολογίζεται διαιρώντας το μέγεθος των δεδομένων (σε bits, γι' αυτό πολλαπλασιάζουμε με 8) με το διαθέσιμο εύρος ζώνης.

Makespan Κόμβου:

$$M\_v = \begin{cases} max\{a \in Apps_v\}\, A[\text{CPUTime(a, v)} + \text{TransmissionTime(a, v)}] & \alpha v\ N_v \leq S_v\ (parallel\ execution) \\ \sum\{a \in Apps_v\}\, [\text{CPUTime(a, v)} + \text{TransmissionTime(a, v)}] & \alpha v\ N_v > S_v\ (sequential\ execution) \end{cases}$$

Το makespan ενός κόμβου εξαρτάται από τον αριθμό των εφαρμογών που του έχουν ανατεθεί σε σχέση με τις δυνατότητες παράλληλης εκτέλεσης (concurrent slots). Αν ο αριθμός εφαρμογών είναι μικρότερος ή ίσος με τα διαθέσιμα slots, οι εφαρμογές εκτελούνται παράλληλα και ο makespan είναι ο μέγιστος χρόνος εκτέλεσης. Διαφορετικά, οι εφαρμογές εκτελούνται διαδοχικά και ο makespan είναι το άθροισμα όλων των χρόνων.

Τελικός Τύπος Makespan:

$$f_{makespan}(x) = max\{v \in V\}\, [M_v]$$

Το συνολικό makespan του συστήματος είναι το μέγιστο makespan μεταξύ όλων των κόμβων, καθώς το σύστημα ολοκληρώνει την επεξεργασία όταν ολοκληρωθεί και ο τελευταίος κόμβος.

Κατανάλωση Ενέργειας (Energy Consumption):

Υπολογίζεται η συνολική ενέργεια που καταναλώνει το σύστημα fog computing όταν κατανέμονται εφαρμογές στους κόμβους.

$$f_{energy}(x) = \sum v \in V \sum a \in A\, x(a, v) \cdot \left[P_{idle} \cdot Runtime_a + P_{processing} \cdot \left(\frac{CPU_a}{CPU_{capacity\_v}}\right) \cdot Runtime_a + P_{transmission} \cdot \left(\frac{MsgSize\_a}{Bandwidth\_path}\right)\right]$$

Για κάθε εφαρμογή που κατανέμεται σε έναν κόμβο (x_a,v = 1), υπολογίζει τρεις τύπους ενέργειας:
Idle Energy: Ενέργεια "αναμονής" του κόμβου κατά τη διάρκεια εκτέλεσης
Processing Energy: Ενέργεια επεξεργασίας (εξαρτάται από το CPU load)
Transmission Energy: Ενέργεια μετάδοσης δεδομένων (εξαρτάται από bandwidth)

Bandwidth Calculation:

$$BW(v) = min\{e \in shortest\_path(v, cloud)\}\, bandwidth_e$$

Το διαθέσιμο εύρος ζώνης για έναν κόμβο καθορίζεται από τη μικρότερη διαδρομή προς το cloud, σύμφωνα με την αρχή ότι η ταχύτητα μετάδοσης περιορίζεται από το πιο αργό τμήμα της διαδρομής.

Καθυστέρηση (Latency):

$$f_{latency}(x) = \Sigma(\forall a \in A)\, \Sigma(\forall v\, V)\, x(a,v) * Latency(a,v)$$

Η συνολική καθυστέρηση υπολογίζεται ως το άθροισμα των καθυστερήσεων όλων των κατανεμημένων εφαρμογών. Για κάθε εφαρμογή που κατανέμεται σε έναν κόμβο (x(a,v) = 1), προστίθεται η αντίστοιχη καθυστέρηση επικοινωνίας.

Κόστος (Cost):

Συνιστώσες Κόστους για κάθε Εφαρμογή:

$$Cost(a,v) = C_{energy}(a,v) + C_{CPU}(a,v) + C_{RAM}(a,v) + C_{Storage}(a,v)$$

Γενικός Τύπος Κόστους:

$$f_{cost}(x) = \sum(a \in A) \sum(v \in V)\, x(a,v) \times [Energy(a,v) \times UC_{energy} + CPU_a \times UC_{cpu} \times Runtime_a + RAM_a \times UC_{ram} \times Runtime_a + Storage_a \times UC_{storage} \times Runtime_a]$$

Το συνολικό κόστος αποτελείται από τέσσερις συνιστώσες: κόστος ενέργειας, κόστος χρήσης CPU, κόστος χρήσης RAM και κόστος χρήσης αποθηκευτικού χώρου. Κάθε συνιστώσα υπολογίζεται πολλαπλασιάζοντας τη χρήση του πόρου με το αντίστοιχο κόστος μονάδας (UC - Unit Cost). Για τους υπολογιστικούς πόρους, το κόστος εξαρτάται και από τον χρόνο εκτέλεσης.

4.5.4 Πολυπλοκότητα του Προβλήματος

Το πρόβλημα κατανομής φόρτου εργασίας σε περιβάλλοντα fog computing ανήκει στην κλάση των NP-hard προβλημάτων. Αυτό σημαίνει ότι δεν υπάρχει γνωστός αλγόριθμος πολυωνυμικού χρόνου που να εγγυάται τη βέλτιστη λύση για όλες τις περιπτώσεις.

4.5.5 Σενάρια Προσομοίωσης

Για την ολοκληρωμένη αξιολόγηση των μεθόδων κατανομής, ορίζονται δύο κύρια πειραματικά σενάρια:

Σενάριο Α: Κλιμάκωση Δικτύου

Σταθερός αριθμός εφαρμογών: 2000

Μεταβλητός αριθμός κόμβων: {50, 100, 200, 300, 400}

Στόχος: Αξιολόγηση συμπεριφοράς κλιμάκωσης με το μέγεθος δικτύου

Σενάριο Β: Κλιμάκωση Φόρτου Εργασίας

Σταθερός αριθμός κόμβων: 100

Μεταβλητός αριθμός εφαρμογών: {500, 1000, 5000, 15000, 20000}

Στόχος: Αξιολόγηση συμπεριφοράς με αυξανόμενο φόρτο

Αυτά τα δύο σενάρια σχεδιάστηκαν για να αξιολογήσουν διαφορετικές πτυχές της κλιμάκωσης:

- το πρώτο εξετάζει πώς οι μέθοδοι αντιδρούν σε αυξανόμενους πόρους (περισσότεροι κόμβοι),
- ενώ το δεύτερο εξετάζει την αντίδραση σε αυξανόμενες απαιτήσεις (περισσότερες εφαρμογές). Αυτή η προσέγγιση παρέχει μια πλήρη εικόνα της συμπεριφοράς των αλγορίθμων υπό διαφορετικές συνθήκες φόρτωσης.

Παράμετροι Συστήματος:

Edge probability (p): 0.3

Gateway percentage: 20%

Επαναλήψεις PSO: 20

Αριθμός σωματιδίων PSO: max(300, app_count/10)

4.6 Μοντελοποίηση Μεθόδων Κατανομής

4.6.1 Μοντέλο Μεθόδου Neighbor-Aware

Η μέθοδος Neighbor-Aware είναι μια ευρετική προσέγγιση σταδιακής κατανομής με τοπική βελτιστοποίηση. Η επιλογή κόμβου δίνεται από:

$$v^* = argmin\ \forall\ v\ \in \text{Neighbors(g)} * \Delta_{\text{Makespan}}(\text{v, a})$$

υπό τους περιορισμούς διαθεσιμότητας πόρων, όπου g είναι το επιλεγμένο gateway.

4.6.2 Μοντέλο Μεθόδου PSO

Η μέθοδος PSO (Particle Swarm Optimization) μοντελοποιεί το πρόβλημα ως καθολική βελτιστοποίηση. Για κάθε σωματίδιο i ισχύει $potition[i] = j$ που σημαίνει ότι η εφαρμογή a_i τοποθετήθηκε στον κόμβο v_j. Η ενημέρωση θέσης σωματιδίου δίνεται από [4]:

- $v_i(t+1) = w \cdot v_i(t) + c_1 \cdot r_1 \cdot (pbest_i - p_i(t)) + c_2 \cdot r_2 \cdot (gbest - p_i(t))$
- $p_i(t+1) = p_i(t) + v_i(t+1)$

Όπου:

- $\boldsymbol{v_i}$ **(t)**: Ταχύτητα σωματιδίου i στην επανάληψη t.
- $\boldsymbol{p_i}$ **(t)**: Θέση σωματιδίου i στην επανάληψη t.
- **w**: Βάρος αδράνειας.
- $\mathbf{c_1}$**,** $\mathbf{c_2}$: Γνωστικός και κοινωνικός συντελεστής.
- $\mathbf{r_1}$**,** $\mathbf{r_2}$: Τυχαίοι αριθμοί στο [0,1].

4.6.2.1 Αντικειμενική Συνάρτηση

Στόχος είναι η ελαχιστοποίηση μιας πολυκριτηριακής συνάρτησης (fitness function του PSO):

$$f(x) = w_1 \cdot f_{\text{makespan}}(x) + w_2 \cdot f_{energy}(x) + w_3 \cdot f_{latency}(x) + w_4 \cdot f_{cost}(x) + w_5 \cdot f_{\text{unallocated}}(x) + w_6 \cdot f_{cloud\_penalty}(x)$$

Συνδυάζει έξι διαφορετικούς στόχους βελτιστοποίησης μέσω σταθμισμένου αθροίσματος. Κάθε βάρος wi καθορίζει τη σχετική σημασία του αντίστοιχου στόχου. Τα βάρη w_5 και w_6 εισάγουν ποινές για μη κατανεμημένες εφαρμογές και υπερβολική χρήση cloud αντίστοιχα, καθοδηγώντας τη βελτιστοποίηση προς επιθυμητές λύσεις.

Όπου:

- $\boldsymbol{f}_{\text{makespan}}(x)$: Συνολικός χρόνος ολοκλήρωσης όλων των εφαρμογών.
- $\boldsymbol{f}_{\boldsymbol{energy}}(x)$: Συνολική κατανάλωση ενέργειας.
- $\boldsymbol{f}_{\boldsymbol{latency}}(x)$: Μέση καθυστέρηση επικοινωνίας.
- $\boldsymbol{f}_{\boldsymbol{cost}}(x)$: Συνολικό λειτουργικό κόστος.

Βάρη (weights) που καθορίζουν τη σημασία κάθε στόχου:

- w_1 (για makespan)
- w_2 (για energy)
- w_3 (για latency)
- w_4 (για cost)
- w_5 (για unallocated penalty)
- w_6 (για cloud penalty)

4.7 Συμπεράσματα Μοντελοποίησης

Η προτεινόμενη μοντελοποίηση παρέχει ένα ολοκληρωμένο πλαίσιο για την κατανομή εφαρμογών σε περιβάλλοντα fog computing. Οι μέθοδοι Neighbor-Aware και PSO προσφέρουν διαφορετικές προσεγγίσεις για την αντιμετώπιση του προβλήματος, θέτοντας τη βάση για την ανάπτυξη και αξιολόγηση που ακολουθεί στα επόμενα κεφάλαια.

5 FogSim-NX: Πρόγραμμα Προσομοίωσης Fog Computing

Για την αποτελεσματική αξιολόγηση και σύγκριση των προτεινόμενων μεθόδων κατανομής φόρτου εργασίας σε περιβάλλοντα fog computing, κρίνεται απαραίτητη η ανάπτυξη ενός εξειδικευμένου πλαισίου προσομοίωσης. Τα υπάρχοντα εργαλεία προσομοίωσης, όπως το iFogSim και το YAFS, παρότι παρέχουν σημαντικές δυνατότητες για τη μοντελοποίηση περιβαλλόντων fog computing, δεν προσφέρουν την απαιτούμενη ευελιξία για την ολοκληρωμένη αξιολόγηση συγκεκριμένων στρατηγικών κατανομής φόρτου εργασίας που επικεντρώνονται στην επίγνωση γειτνίασης και τη βιο-εμπνευσμένη βελτιστοποίηση. Επιπλέον, η ανάγκη για λεπτομερή μέτρηση μετρικών απόδοσης που περιλαμβάνουν ενεργειακή κατανάλωση, λειτουργικό κόστος και μοτίβα κατανομής πόρων απαιτεί έναν προσομοιωτή που να επιτρέπει τον ακριβή έλεγχο και την παρακολούθηση όλων αυτών των παραμέτρων. Για αυτόν τον λόγο, αναπτύχθηκε το FogSim-NX, ένα εξειδικευμένο πρόγραμμα προσομοίωσης που παρέχει τη δυνατότητα συγκριτικής αξιολόγησης των μεθόδων Neighbor-aware και PSO υπό ελεγχόμενες και επαναλαμβανόμενες συνθήκες, ενώ παράλληλα διευκολύνει την επέκταση και προσαρμογή για μελλοντικές ερευνητικές ανάγκες στον τομέα του fog computing.

5.1 Το Πλαίσιο Προσομοίωσης

Για την αξιολόγηση των προτεινόμενων μεθόδων κατανομής φόρτου εργασίας, υιοθετήθηκε η μεθοδολογία προσομοίωσης ως η καταλληλότερη ερευνητική προσέγγιση. Η επιλογή αυτή βασίστηκε στους ακόλουθους λόγους:

- **Ελεγχόμενο Περιβάλλον**: Η προσομοίωση επιτρέπει τον ακριβή έλεγχο των παραμέτρων και την επαναληψιμότητα των πειραμάτων
- **Κλιμάκωση**: Δυνατότητα αξιολόγησης σε διαφορετικά μεγέθη δικτύου χωρίς φυσικούς περιορισμούς
- **Οικονομικότητα**: Αποφυγή του κόστους ανάπτυξης πραγματικής υποδομής fog computing

Ο προσομοιωτής είναι γενικής χρήσης και περιλαμβάνει δομικές μονάδες για:

• τη διαχείριση της τοπολογίας

• τη διαχείριση των κόμβων

• των εφαρμογών

• τη διαχείριση πρωτοκόλλων κατανομής φορτίου

• τη διαχείριση των στατιστικών, που επιτρέπει τη συλλογή καθολικών ή τοπικών στατιστικών

5.2 Υλοποίηση FogSim-NX

Για την υλοποίηση και αξιολόγηση των προτεινόμενων μεθόδων, αναπτύχθηκε το εξειδικευμένο πρόγραμμα προσομοίωσης **FogSim-NX**.

Τεχνολογίες Υλοποίησης:

- **Python**: Κύρια γλώσσα προγραμματισμού
- **NetworkX**: Διαχείριση γράφων και τοπολογιών

- **NumPy**: Αριθμητικοί υπολογισμοί για PSO
- **Matplotlib**: Οπτικοποίηση αποτελεσμάτων
- **ConfigParser**: Διαχείριση παραμέτρων

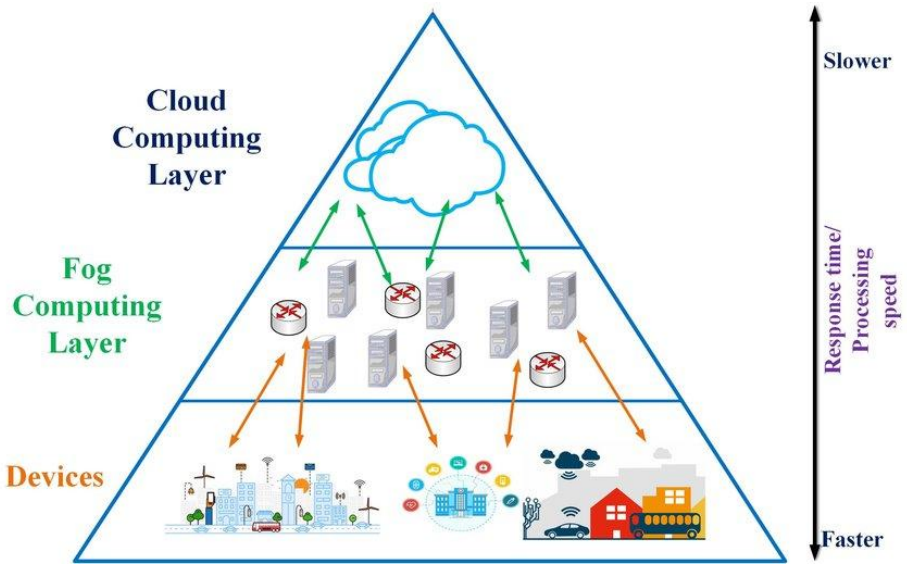
Δομικές μονάδες συστήματος:

1. **Configuration Manager**: Φόρτωση παραμέτρων από config.ini
2. **Topology Generator**: Δημιουργία Erdős-Rényi γράφων
3. **Application Manager**: Ορισμός εφαρμογών με τυχαίες απαιτήσεις
4. **Allocation Engine**: Υλοποίηση αλγορίθμων Neighbor-Aware και PSO
5. **Metrics Calculator**: Υπολογισμός μετρικών απόδοσης
6. **Results Exporter**: Αποθήκευση σε CSV format

Validation Framework:

- **Consistency Checks**: Επαλήθευση συνέπειας κατανομών
- **Resource Constraints**: Έλεγχος τήρησης περιορισμών πόρων
- **Connectivity Verification**: Επιβεβαίωση συνδεσιμότητας γράφου

5.3 Αρχιτεκτονική Συστήματος



Το FogSim-NX είναι ένα ολοκληρωμένο πρόγραμμα προσομοίωσης σχεδιασμένο για τη μοντελοποίηση και αξιολόγηση στρατηγικών κατανομής φόρτου εργασίας σε περιβάλλοντα fog computing. Το πλαίσιο χρησιμοποιεί μια αρθρωτή αρχιτεκτονική που διευκολύνει την προσομοίωση σύνθετων σεναρίων fog computing με ετερογενείς κόμβους, διαφορετικές απαιτήσεις εφαρμογών και διαφορετικές στρατηγικές κατανομής.

Στο αναπτυσσόμενο οικοσύστημα πλαισίων προσομοίωσης fog computing αξιοσημείωτη συνεισφορά αποτελεί το YAFS (Yet Another Fog Simulator) που αναπτύχθηκε από τους Lera et al. [9] το οποίο επικεντρώνεται σε δυναμικά περιβάλλοντα fog computing. Ενώ το YAFS δίνει έμφαση στη δυναμική δικτύου, την κινητικότητα των χρηστών και τις προσαρμόσιμες πολιτικές επιλογής, το FogSim-NX διαφοροποιείται μέσω της εστίασής του στη σύγκριση neighbor-aware και βιο-εμπνευσμένων στρατηγικών κατανομής, καθώς και της διαχείρισης τοπολογίας βασισμένης στο πακέτο NetworkX (https://networkx.org/) που διευκολύνει την πολύπλοκη ανάλυση δικτύου [9].

Ακολουθώντας αρχές σχεδιασμού προσομοίωσης παρόμοιες με αυτές που προτάθηκαν από τους Gupta et al. [3] στο πλαίσιο iFogSim, το FogSim-NX επεκτείνει αυτές τις έννοιες με έμφαση σε στρατηγικές κατανομής neighbor-aware και βασισμένες σε PSO. Η συνολική αρχιτεκτονική του συστήματος αποτελείται από τέσσερα κύρια συστατικά [3]:

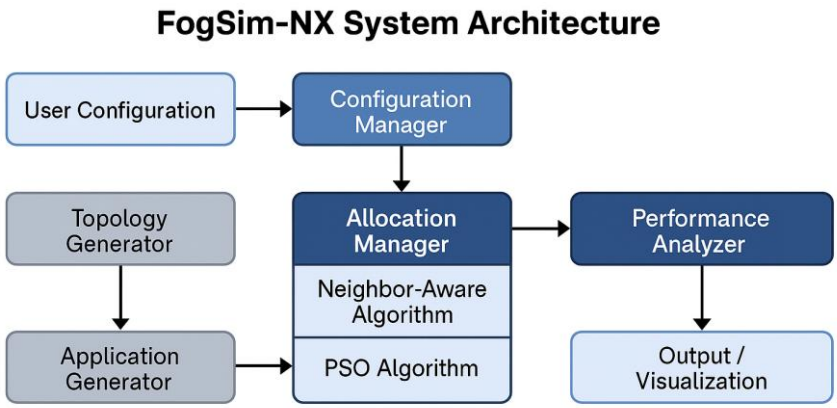
1. **Configuration file:** Χειρίζεται την εγκατάσταση και τη διαμόρφωση παραμέτρων για το περιβάλλον προσομοίωσης, συμπεριλαμβανομένων χαρακτηριστικών τοπολογίας, προδιαγραφών κόμβων, απαιτήσεων εφαρμογών και παραμέτρων ενέργειας/κόστους.

2. **Topology Generator:** Δημιουργεί και διαχειρίζεται την τοπολογία δικτύου που αντιπροσωπεύει το περιβάλλον fog computing, συμπεριλαμβανομένης της ταξινόμησης κόμβων (simple, gateway, cloud) και της συνδεσιμότητας δικτύου.

3. **Allocation Manager:** Υλοποιεί διαφορετικές στρατηγικές κατανομής φόρτου εργασίας, συγκεκριμένα τη μέθοδο Neighbor-aware που εμπνέεται από τους Skarlat et al. [6] και την προσέγγιση βελτιστοποίησης βασισμένη σε PSO που βασίζεται σε έννοιες από τους Kennedy και Eberhart [4].

4. **Performance statistics:** Συλλέγει και αναλύει τα αποτελέσματα προσομοίωσης, μετρώντας μετρικές όπως επιτυχία κατανομής, χρήση πόρων, makespan, κατανάλωση ενέργειας και κόστος χρησιμοποιώντας μεθοδολογίες αξιολόγησης παρόμοιες με αυτές που προτάθηκαν από τους Mahmud et al. [5].

Αυτά τα συστατικά συνεργάζονται για να παρέχουν μια ευέλικτη και επεκτάσιμη πλατφόρμα για την αξιολόγηση στρατηγικών κατανομής φόρτου εργασίας σε διάφορα σενάρια. Ο αρθρωτός σχεδιασμός επιτρέπει την εύκολη τροποποίηση και επέκταση μεμονωμένων συστατικών, επιτρέποντας την προσομοίωση ενός μεγάλου φάσματος περιβαλλόντων fog computing αλλά και προσεγγίσεων κατανομής.



Εικόνα 5.1: Αρχιτεκτονική υψηλού επιπέδου του πλαισίου προσομοίωσης FogSim-NX

5.3.1 Αρχές Σχεδιασμού

Το FogSim-NX αναπτύχθηκε ακολουθώντας αρκετές βασικές αρχές σχεδιασμού:

1. **Ρεαλισμός**: Το πλαίσιο στοχεύει στη μοντελοποίηση περιβαλλόντων fog computing με ρεαλιστικά χαρακτηριστικά, συμπεριλαμβανομένων ετερογενών δυνατοτήτων κόμβων, μεταβλητών συνθηκών δικτύου και διαφορετικών απαιτήσεων εφαρμογών.
2. **Ευελιξία**: Οι χρήστες μπορούν να διαμορφώσουν πολυάριθμες παραμέτρους για τη δημιουργία προσαρμοσμένων (custom) σεναρίων προσομοίωσης, από χαρακτηριστικά τοπολογίας δικτύου έως προδιαγραφές εφαρμογών και στρατηγικές κατανομής.
3. **Επεκτασιμότητα**: Η αρθρωτή αρχιτεκτονική διευκολύνει την προσθήκη νέων στρατηγικών κατανομής, μετρικών απόδοσης και μοντέλων δικτύου χωρίς να απαιτούνται σημαντικές αλλαγές στον υπάρχοντα κώδικα.
4. **Επανάληψη**: Οι διαμορφώσεις και τα αποτελέσματα προσομοίωσης μπορούν να αποθηκευτούν και να επαναχρησιμοποιηθούν, επιτρέποντας συνεπείς συγκρίσεις μεταξύ διαφορετικών στρατηγικών κατανομής και ρυθμίσεων παραμέτρων.
5. **Ευχρηστία**: Το πλαίσιο παρέχει μια φιλική προς το χρήστη διεπαφή για τη δημιουργία προσομοιώσεων, την οπτικοποίηση τοπολογιών δικτύου και την ανάλυση αποτελεσμάτων.

5.3.2 Τεχνολογίες Υλοποίησης

Το FogSim-NX είναι υλοποιημένο σε γλώσσα Python, αξιοποιώντας αρκετές ισχυρές βιβλιοθήκες:

- **NetworkX**: Παρέχει τη βάση για την αναπαράσταση και το χειρισμό της τοπολογίας δικτύου fog ως γράφο, με κόμβους που αντιπροσωπεύουν συσκευές fog και ακμές που αντιπροσωπεύουν συνδέσεις δικτύου.
- **NumPy**: Επιτρέπει αποδοτικούς αριθμητικούς υπολογισμούς, ιδιαίτερα για την υλοποίηση του αλγορίθμου PSO και τους υπολογισμούς μετρικών απόδοσης.
- **Matplotlib**: Διευκολύνει την οπτικοποίηση τοπολογιών δικτύου, χρήσης πόρων και μετρικών απόδοσης.
- **ConfigParser**: Διαχειρίζεται παραμέτρους διαμόρφωσης μέσω δομημένων αρχείων INI, επιτρέποντας ευέλικτη εγκατάσταση προσομοίωσης.

Η χρήση αυτών των ευρέως υιοθετημένων βιβλιοθηκών διασφαλίζει ότι το FogSim-NX είναι τόσο ισχυρό όσο και συντηρήσιμο, ενώ παράλληλα το καθιστά προσβάσιμο σε ερευνητές και επαγγελματίες εξοικειωμένους με το οικοσύστημα Python.

5.4 Συστατικά Στοιχεία Προσομοιωτή

5.4.1 Δημιουργία Τοπολογίας

Το συστατικό δημιουργίας τοπολογίας είναι υπεύθυνο για τη δημιουργία μιας δομής δικτύου που αντιπροσωπεύει το περιβάλλον fog computing. Το FogSim-NX χρησιμοποιεί το μοντέλο τυχαίου γράφου Erdős-Rényi για την παραγωγή συνδεδεμένων τοπολογιών δικτύου με διαμορφώσιμες παραμέτρους, μια

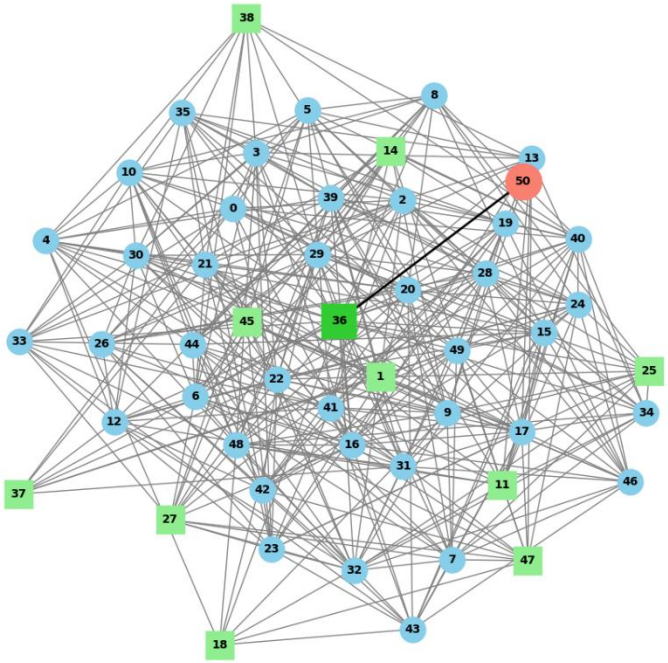
προσέγγιση που έχει αποδειχθεί ότι μοντελοποιεί αποτελεσματικά την ετερογενή φύση των υποδομών fog computing [3].

Η διαδικασία δημιουργίας τοπολογίας ακολουθεί αυτά τα βήματα:

1. **Δημιουργία Κόμβων:** Δημιουργεί έναν καθορισμένο αριθμό κόμβων που αντιπροσωπεύουν συσκευές fog στο δίκτυο.
2. **Δημιουργία Ακμών:** Καθιερώνει συνδέσεις μεταξύ κόμβων με βάση μια παράμετρο πιθανότητας (f_edge_prob), διασφαλίζοντας ότι στον γράφο που θα δημιουργηθεί, οι κόμβοι θα είναι συνδεδεμένοι.
3. **Ταξινόμηση Κόμβων:** Ορίζει ένα ποσοστό κόμβων ως gateways με βάση μετρικές κεντρικότητας, με τον πιο κεντρικό κόμβο να γίνεται το main gateway. Οι υπόλοιποι κόμβοι ταξινομούνται ως simple fog nodes.
4. **Προσθήκη Cloud Node:** Προσθέτει έναν εξωτερικό cloud node με υψηλές χωρητικότητες πόρων και τον συνδέει με το main gateway, ακολουθώντας την ιεραρχική αρχιτεκτονική που περιγράφηκε από τους [1].
5. **Ανάθεση Πόρων:** Αναθέτει υπολογιστικούς πόρους (CPU, RAM, storage) και χαρακτηριστικά δικτύου (bandwidth, propagation delay) σε κόμβους και ακμές με βάση παραμέτρους διαμόρφωσης.

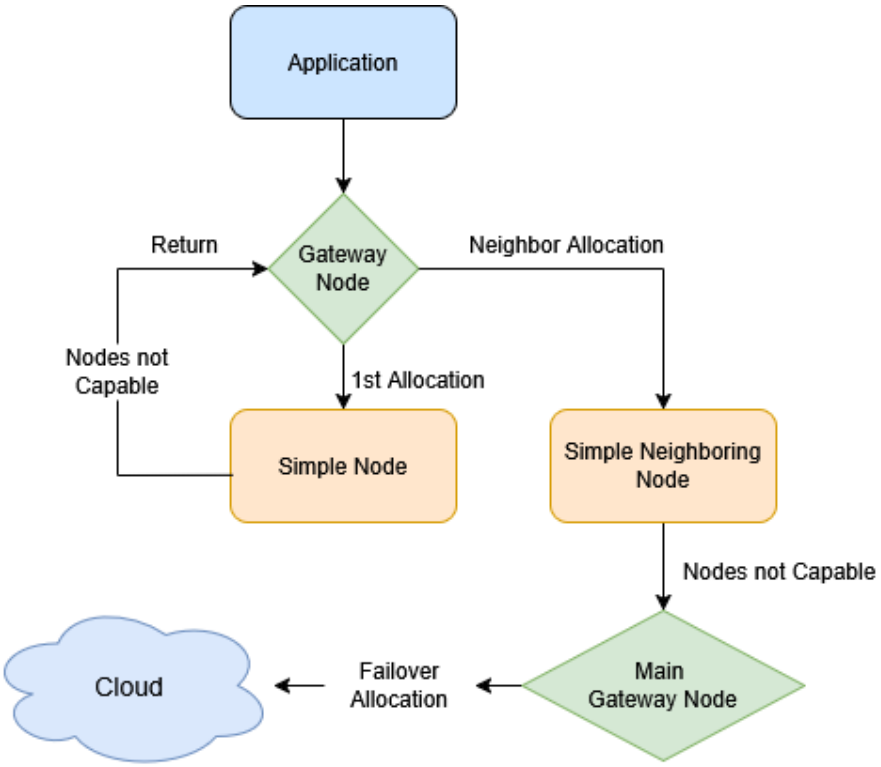
Η τοπολογία που προκύπτει οπτικοποιείται και αποθηκεύεται σε μορφή GraphML για μελλοντική αναφορά και επαναχρησιμοποίηση.

Η Εικόνα 5.2 δείχνει ένα παράδειγμα τοπολογίας που δημιουργήθηκε από το FogSim-NX.



Εικόνα 5.2: Παράδειγμα μιας τοπολογίας fog computing που δημιουργήθηκε με fog nodes (μπλε), gateway nodes (πράσινο), main gateway node (σκούρο πράσινο) και εξωτερικό cloud node (κόκκινο)

5.4.2 Ταξινόμηση και Χαρακτηριστικά Κόμβων



Εικόνα 5.3: Λογική Κατανομής

Το FogSim-NX περιλαμβάνει κόμβους που αντιστοιχούν σε τρεις κατηγορίες, καθεμία με διακριτούς ρόλους και χαρακτηριστικά:

1. **Fog nodes**: Αντιπροσωπεύουν τυπικές συσκευές fog με μέτριες υπολογιστικές δυνατότητες. Αυτοί οι κόμβοι είναι υπεύθυνοι για την τοπική επεξεργασία εφαρμογών για τη μείωση του latency και της κίνησης δικτύου. Οι δυνατότητες των πόρων τους ορίζονται από ένα εύρος τιμών κατά τη διαδικασία διαμόρφωσης του δικτύου.

2. **Gateway Nodes**: Λειτουργούν ως διαμεσολαβητές μεταξύ fog nodes. Επιλέγονται με βάση χαμηλή κεντρικότητα, εξωτερικά στην τοπολογία δικτύου. Οι gateway nodes έχουν παρόμοιες προδιαγραφές πόρων με τους fog nodes αλλά είναι υπεύθυνοι μόνο για την προώθηση εφαρμογών σε κατάλληλους κόμβους επεξεργασίας.

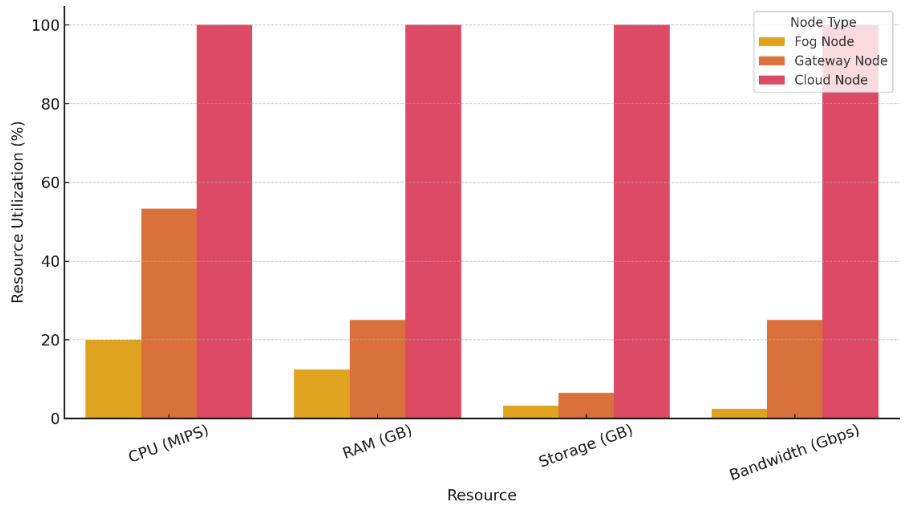
3. **Main Gateway Node**: Λειτουργεί μόνο ως διαμεσολαβητής μεταξύ fog nodes και cloud. Επιλέγονται με βάση υψηλή κεντρικότητα, στην τοπολογία δικτύου. Αν αποτύχουν όλοι οι απλοί κόμβοι, προωθεί την εφαρμογή στο Cloud.

4. **Cloud Node**: Αντιπροσωπεύει ένα απομακρυσμένο data center cloud με πρακτικά απεριόριστους πόρους. Στην προσομοίωση, στον cloud node αποδίδονται πολύ υψηλές τιμές πόρων:

Πίνακας 5-1: Ροή Κατανομής Neighbor-Aware

Τύπος Κόμβου	Ρόλος
Gateway Node	Μόνο προωθεί εφαρμογές προς τους Απλούς Κόμβους. Δεν εκτελεί κατανομή εφαρμογών.
Simple Fog Node	Πρώτη προσπάθεια εκτέλεσης κατανομής με βάση τους διαθέσιμους πόρους.
(Neighboring) Simple Node	Αν αποτύχει ο πρώτος Απλός Κόμβος, προσπαθούν οι γειτονικοί Απλοί Κόμβοι.
Main Gateway Node	Αν αποτύχουν όλοι οι Απλοί Κόμβοι, προωθεί την εφαρμογή στο Cloud.
Cloud Node	Τελική εφεδρική επιλογή. Εγγυημένη κατανομή λόγω άφθονων πόρων.

Ωστόσο, η πρόσβαση στο cloud επιφέρει υψηλότερο latency (συνήθως 200ms) σε σύγκριση με τους fog nodes, αντανακλώντας το συμβιβασμό του πραγματικού κόσμου μεταξύ διαθεσιμότητας πόρων και καθυστέρησης επικοινωνίας.



Εικόνα 5.4: (Κανονικοποίηση). Δυνατότητες Πόρων ανά Τύπο Κόμβου (%)

Αυτή η ταξινόμηση κόμβων επιτρέπει στο FogSim-NX να μοντελοποιεί την ιεραρχική φύση των περιβαλλόντων fog computing, όπου η ελαφριά επεξεργασία μπορεί να εκτελεστεί σε fog nodes, ο συντονισμός γίνεται μέσω gateway nodes, και οι εργασίες που απαιτούν πόρους ή είναι καθολικές μεταφέρονται στο cloud.

5.4.3 Μοντελοποίηση Εφαρμογών

Οι εφαρμογές στο FogSim-NX αντιπροσωπεύουν φόρτους εργασίας που πρέπει να κατανεμηθούν σε κόμβους για επεξεργασία. Κάθε εφαρμογή χαρακτηρίζεται από βασικά χαρακτηριστικά όπως CPU, RAM, κλπ.

Αυτά τα χαρακτηριστικά καθορίζονται από τον χρήστη και παίρνουν τιμές εντός διαμορφώσιμων ευρών (ranges), επιτρέποντας την προσομοίωση διαφορετικού φόρτου εργασίας με ποικίλες απαιτήσεις πόρων.

Το δομικό στοιχείο της εφαρμογής στο FogSim-NX είναι σχεδιασμένο να είναι τόσο ρεαλιστικό όσο και ευέλικτο, καταγράφοντας τις βασικές πτυχές των φόρτων εργασίας fog computing του πραγματικού κόσμου ενώ παραμένει υπολογιστικά εφικτό.

```
def define_applications(app_count: int, config: Dict[str, Any]) -> List[Dict[str, Any]]:
    """Defines a list of applications with their requirements."""
    return [{
        'CPU': random.randint(config['APP_REQUIREMENTS']['a_cpu_min'],
                              config['APP_REQUIREMENTS']['a_cpu_max']),
        'RAM': random.randint(config['APP_REQUIREMENTS']['a_ram_min'],
                              config['APP_REQUIREMENTS']['a_ram_max']),
        'Runtime': random.randint(config['APP_REQUIREMENTS']['a_runtime_min'],
                                  config['APP_REQUIREMENTS']['a_runtime_max']),
        'Storage': random.randint(config['APP_REQUIREMENTS']['a_storage_min'],
                                  config['APP_REQUIREMENTS']['a_storage_max']),
        'Msg_Size': config['APP_REQUIREMENTS']['a_msg_size']
    } for _ in range(app_count)]
```

Απόσπασμα Κώδικα 5.2: Συνάρτηση για τη δημιουργία εφαρμογών με καθορισμένες απαιτήσεις

5.4.4 Μετρικές Απόδοσης

Το FogSim-NX αξιολογεί τις στρατηγικές κατανομής χρησιμοποιώντας ένα ολοκληρωμένο σύνολο μετρικών απόδοσης, παρέχοντας πληροφορίες για διαφορετικές πτυχές της συμπεριφοράς του συστήματος:

1. **Αποδοτικότητα Κατανομής**:
    - Allocation Success Rate: Ποσοστό εφαρμογών που κατανεμήθηκαν επιτυχώς
    - Cloud Utilization: Ποσοστό εφαρμογών που κατανεμήθηκαν στο cloud
    - Fog Node Utilization: Αριθμός και ποσοστό fog nodes που χρησιμοποιήθηκαν
2. **Χρήση Πόρων**:
    - CPU, RAM, και Storage Usage: Συνολική και μέση χρήση σε όλους τους κόμβους
    - Bandwidth Consumption: Συνολικό bandwidth δικτύου που χρησιμοποιήθηκε από εφαρμογές
3. **Μετρικές Επίδοσης**:
    - Latency: Καθυστέρηση επικοινωνίας μεταξύ πηγής εφαρμογής και κόμβου επεξεργασίας

- Makespan: Συνολικός χρόνος που απαιτείται για την ολοκλήρωση όλων των εφαρμογών, λαμβάνοντας υπόψη τον παραλληλισμό

4. **Μετρικές Αποδοτικότητας**:
    - Workload: Το σύνολο των υπολογιστικών και δικτυακών εργασιών που κατανέμονται και επεξεργάζονται στους κόμβους του fog δικτύου.

- Energy Consumption: Συνολική ενέργεια που χρησιμοποιείται για τον υπολογισμό και την επικοινωνία.
- Operational Cost (€): Οικονομικό κόστος βάσει χρήσης πόρων και κατανάλωσης ενέργειας

Αυτές οι μετρικές επιτρέπουν ολοκληρωμένη σύγκριση μεταξύ διαφορετικών στρατηγικών κατανομής, αναδεικνύοντας τα πλεονεκτήματα και τις αδυναμίες τους σε πολλαπλές διαστάσεις απόδοσης.

5.5 Λεπτομέρειες Υλοποίησης

5.5.1 Διαχείριση Τοπολογίας Βασισμένη στο NetworkX

Το FogSim-NX αξιοποιεί τη βιβλιοθήκη NetworkX για την αναπαράσταση και διαχείριση της τοπολογίας fog computing ως δομή γράφου. Αυτή η προσέγγιση προσφέρει αρκετά πλεονεκτήματα:

1. **Ευέλικτη Αναπαράσταση**: Οι κόμβοι αντιπροσωπεύουν υπολογιστικές συσκευές με χαρακτηριστικά για πόρους (CPU, RAM, storage), ενώ οι ακμές αντιπροσωπεύουν συνδέσεις δικτύου με χαρακτηριστικά για bandwidth και latency.
2. **Αποδοτικός Υπολογισμός Διαδρομής**: Οι αλγόριθμοι εύρεσης διαδρομής του NetworkX (όπως shortest_path) επιτρέπουν τον αποδοτικό υπολογισμό διαδρομών επικοινωνίας μεταξύ κόμβων, λαμβάνοντας υπόψη τις propagation delays ως βάρη ακμών.
3. **Ανάλυση Συνδεσιμότητας**: Οι μετρικές θεωρίας γράφων όπως η κεντρικότητα βοηθούν στον εντοπισμό στρατηγικά σημαντικών κόμβων για τον ορισμό των gateway.
4. **Δυνατότητες Οπτικοποίησης**: Οι ενσωματωμένες συναρτήσεις σχεδίασης διευκολύνουν την οπτική αναπαράσταση των τοπολογιών δικτύου και των ταξινομήσεων κόμβων.

Το ακόλουθο απόσπασμα κώδικα απεικονίζει πώς το NetworkX χρησιμοποιείται για τον υπολογισμό της συντομότερης διαδρομής μεταξύ κόμβων για την εκτίμηση του latency:

```
try:
path = nx.shortest_path(graph, source=node, target=external_cloud, weight='propagation_delay')
bandwidth = min(graph.edges[edge]['bandwidth'] for edge in zip(path[:-1], path[1:]))
latency = sum(graph.edges[edge]['propagation_delay'] for edge in zip(path[:-1], path[1:]))
except (nx.NetworkXNoPath, nx.NodeNotFound):
bandwidth = config['EXTERNAL_CLOUD']['c_bandwidth']
latency = config['EXTERNAL_CLOUD']['c_propagation_delay']
```

Απόσπασμα Κώδικα 5.3: Υπολογισμός διαδρομής για προσδιορισμό latency και bandwidth

5.5.2 Παράμετροι Διαμόρφωσης

Το FogSim-NX χρησιμοποιεί ένα δομημένο σύστημα διαμόρφωσης με πολλαπλές κατηγορίες παραμέτρων που επιτρέπουν την ευέλικτη προσαρμογή του περιβάλλοντος προσομοίωσης:

1. Topology Attributes (Χαρακτηριστικά Τοπολογίας):

Ελέγχουν τη δομή δικτύου, την πιθανότητα σύνδεσης κόμβων, το ποσοστό gateway nodes και τα εύρη υπολογιστικών πόρων για κάθε τύπο κόμβου.

2. Main Gateway Configuration (Διαμόρφωση Κύριας Πύλης):

Καθορίζουν τα χαρακτηριστικά του κύριου κόμβου gateway, συμπεριλαμβανομένων των δικτυακών δυνατοτήτων και των χρόνων απόκρισης.

3. External Cloud Parameters (Παράμετροι Εξωτερικού Cloud):

Ορίζουν τα χαρακτηριστικά του κόμβου cloud υψηλής χωρητικότητας, όπως την καθυστέρηση διάδοσης και τους διαθέσιμους πόρους.

4. Application Requirements (Απαιτήσεις Εφαρμογών):

Καθορίζουν τα εύρη των απαιτήσεων πόρων για τις εφαρμογές που θα δημιουργηθούν, επιτρέποντας τη μοντελοποίηση διαφορετικών τύπων φόρτου εργασίας.

5. Energy Parameters (Παράμετροι Ενέργειας):

Ορίζουν παράγοντες κατανάλωσης ενέργειας για διαφορετικές λειτουργικές καταστάσεις των κόμβων, συμπεριλαμβανομένης της αδρανούς κατάστασης, επεξεργασίας και μετάδοσης.

6. Cost Parameters (Παράμετροι Κόστους):

Καθορίζουν οικονομικά κόστη που σχετίζονται με τη χρήση πόρων, επιτρέποντας την αξιολόγηση της οικονομικής αποδοτικότητας των στρατηγικών κατανομής.

Αυτό το δομημένο σύστημα διαμόρφωσης επιτρέπει τον ακριβή έλεγχο των παραμέτρων προσομοίωσης διατηρώντας παράλληλα την αναγνωσιμότητα και την ευκολία τροποποίησης. Οι συγκεκριμένες τιμές των παραμέτρων που χρησιμοποιήθηκαν στην πειραματική αξιολόγηση παρουσιάζονται λεπτομερώς στο Κεφάλαιο 6.

5.5.3 Ροή Εργασίας Προσομοίωσης

Η προσομοίωση FogSim-NX ακολουθεί μια συστηματική ροή εργασίας:

1. **Φόρτωση Διαμόρφωσης**: Ανάγνωση και ανάλυση παραμέτρων από το αρχείο διαμόρφωσης.
2. **Επιλογή/Δημιουργία Τοπολογίας**: Είτε φόρτωση μιας υπάρχουσας τοπολογίας είτε δημιουργία νέας με βάση καθορισμένες παραμέτρους.
3. **Ορισμός Εφαρμογών**: Δημιουργία ενός συνόλου εφαρμογών με καθορισμένες απαιτήσεις πόρων.
4. **Επιλογή Μεθόδου Κατανομής**: Επιλογή μεταξύ στρατηγικής κατανομής Neighbor-aware ή βασισμένης σε αλγόριθμο PSO.
5. **Εκτέλεση Κατανομής**: Εφαρμογή της επιλεγμένης στρατηγικής για την κατανομή εφαρμογών σε κόμβους.
6. **Συλλογή Αποτελεσμάτων**: Συγκέντρωση μετρικών απόδοσης και στατιστικών κατανομής.
7. **Αποθήκευση Αποτελεσμάτων**: Αποθήκευση αποτελεσμάτων προσομοίωσης σε αρχεία CSV για ανάλυση και σύγκριση.

Αυτή η ροή εργασίας υλοποιείται στο κύριο σενάριο εκτέλεσης, το οποίο παρέχει μια διεπαφή γραμμής εντολών για τον έλεγχο της διαδικασίας προσομοίωσης:

Αυτή η δομημένη ροή εργασίας διασφαλίζει αναπαραγώγιμες προσομοιώσεις και διευκολύνει δίκαιες συγκρίσεις μεταξύ διαφορετικών στρατηγικών κατανομής υπό πανομοιότυπες συνθήκες.

Κατά την ανάπτυξη της ροής προσομοίωσης του FogSim-NX, εξετάσαμε την αρθρωτή προσέγγιση που επέδειξαν οι Lera et al. [9] στον προσομοιωτή YAFS. Το YAFS δίνει προτεραιότητα στη δυναμική παραγωγή φόρτου εργασίας και στις αλλαγές τοποθέτησης κατά τη διάρκεια εκτέλεσης, το FogSim-NX δίνει έμφαση στη συγκριτική ανάλυση διαφορετικών στρατηγικών κατανομής υπό ελεγχόμενες συνθήκες. Αυτό μας επέτρεψε να υλοποιήσουμε πιο λεπτομερή παρακολούθηση πόρων και μετρικές απόδοσης ειδικά προσαρμοσμένες για την αξιολόγηση των μεθόδων κατανομής Neighbor-aware και PSO.

5.6 Υλοποίηση αλγορίθμων

5.6.1 Υλοποίηση Neighbor-Aware

Η μέθοδος κατανομής Neighbor-Aware υλοποιείται στη συνάρτηση run_neighbor_allocation, η οποία δέχεται ως εισόδους έναν γράφο δικτύου, αριθμό εφαρμογών, παραμέτρους διαμόρφωσης και αναγνωριστικό id του external cloud node και επιστρέφει ολοκληρωμένα αποτελέσματα κατανομής.

Η υλοποίηση ακολουθεί αυτά τα βασικά βήματα:

1. **Ανάθεση Ρόλων Κόμβων**: Αναγνώριση fog nodes, gateway nodes και του main gateway με βάση τα χαρακτηριστικά κόμβων και τις μετρικές κεντρικότητας.

```
simple_nodes = [n for n in graph.nodes if graph.nodes[n].get('node_type') == 'simple']

gateway_nodes = [n for n in graph.nodes if graph.nodes[n].get('node_type') == 'gateway']

main_gateway = max(nx.betweenness_centrality(graph).items(), key=lambda x: x[1])[0]
```

2. **Παρακολούθηση Πόρων**: Αρχικοποίηση δομών δεδομένων για την παρακολούθηση των εναπομεινάντων πόρων, makespan και πλήθους εφαρμογών για κάθε κόμβο.

```
node_resources = {node: {'cpu_remaining': graph.nodes[node]['cpu'],
                         'ram_remaining': graph.nodes[node]['ram'],
                         'storage_remaining': graph.nodes[node]['storage']} for node in graph.nodes}

node_makespans = {node: 0.0 for node in graph.nodes}

node_app_counts = {node: 0 for node in graph.nodes}

gateway_load = {gateway: 0 for gateway in gateway_nodes}
```

3. **Διαχείριση Παραλληλισμού**: Υπολογισμός του αριθμού εφαρμογών που μπορεί να επεξεργαστεί ταυτόχρονα κάθε κόμβος, βάσει της διαθέσιμης CPU ισχύος του. Αυτή η παράμετρος επηρεάζει τον υπολογισμό του χρόνου ολοκλήρωσης (makespan) των εφαρμογών.

   Δηλαδή μοντελοποιούμε ότι κάθε κόμβος μπορεί να τρέχει περιορισμένο αριθμό εφαρμογών ταυτόχρονα, όχι απεριόριστες.

```
total_cpu = sum(graph.nodes[node]['cpu'] for node in graph.nodes)

avg_cpu_per_node = total_cpu / len(graph.nodes)

concurrency_limit = max(1000, int(avg_cpu_per_node / 2))
```

4. **Ορισμός Εφαρμογών**: Δημιουργία εφαρμογών με τυχαίες απαιτήσεις πόρων εντός των διαμορφωμένων ευρών.

5. **Διαδικασία Κατανομής Τριών Βημάτων**:

   - **Βήμα 1**: Επιλογή gateway με ελάχιστο φορτίο και προώθηση σε έναν απευθείας συνδεδεμένο σε αυτόν simple node που ελαχιστοποιεί την αύξηση του makespan.
   - **Βήμα 2**: Εάν το Βήμα 1 αποτύχει, γίνεται δοκιμή γειτόνων (fog nodes) όλων των gateways για την εύρεση κατάλληλης κατανομής.

- **Βήμα 3**: Εάν και τα δύο προηγούμενα βήματα αποτύχουν, πραγματοποιείται προώθηση στο external cloud μέσω του main gateway.

```
selected_gateway = min(gateway_load.items(), key=lambda x: x[1])[0]
simple_neighbors = [n for n in graph.neighbors(selected_gateway) if n in simple_nodes]
if simple_neighbors:
  min_makespan_increase = float('inf')
  best_node = None

  for neighbor in simple_neighbors:
    if (node_resources[neighbor]['cpu_remaining'] >= app['CPU'] and
      node_resources[neighbor]['ram_remaining'] >= app['RAM'] and
      node_resources[neighbor]['storage_remaining'] >= app['Storage']):

      # Calculate bandwidth and makespan increase...
      if makespan_increase < min_makespan_increase:
        min_makespan_increase = makespan_increase
        best_node = neighbor
```

6. **Ενημέρωση Πόρων και Υπολογισμός Μετρικών**: Μετά από επιτυχή κατανομή, πραγματοποιείται ενημέρωση πόρων κόμβων και υπολογισμός μετρικών απόδοσης συμπεριλαμβανομένων latency, makespan, workload, κατανάλωσης ενέργειας και κόστους.

```
node_resources[target_node]['cpu_remaining'] -= app['CPU']
node_resources[target_node]['ram_remaining'] -= app['RAM']
node_resources[target_node]['storage_remaining'] -= app['Storage']
node_app_counts[target_node] += 1


# Calculate path-based metrics
path = nx.shortest_path(graph, source=target_node, target=external_cloud, weight='propagation_delay')
bandwidth = min(graph.edges[edge]['bandwidth'] for edge in zip(path[:-1], path[1:]))
latency = sum(graph.edges[edge]['propagation_delay'] for edge in zip(path[:-1], path[1:]))

# Calculate makespan with concurrency
app_makespan = app['Runtime'] + (app['Msg_Size'] * 8 / bandwidth)
concurrent_slots = max(1, graph.nodes[target_node]['cpu'] // concurrency_limit)

if node_app_counts[target_node] <= concurrent_slots:
  node_makespans[target_node] = max(node_makespans[target_node], app_makespan)
else:
  node_makespans[target_node] += app_makespan
```

Η υλοποίηση αξιοποιεί τις δυνατότητες της διάσχισης γράφου και εύρεσης διαδρομής του NetworkX για τον αποδοτικό εντοπισμό γειτονικών κόμβων και τον υπολογισμό μετρικών επικοινωνίας. Η προσέγγιση είναι προσδιοριστική και ακολουθεί μια greedy στρατηγική που επιχειρεί να ελαχιστοποιήσει την τοπική αύξηση του makespan για κάθε απόφαση κατανομής.

5.6.2 Υλοποίηση PSO

Η μέθοδος κατανομής PSO στο FogSim-NX υλοποιείται στη συνάρτηση **run_pso_allocation**, η οποία προσαρμόζει τον τυπικό αλγόριθμο PSO στην ειδική πρόκληση της κατανομής φόρτου εργασίας σε περιβάλλοντα fog computing.

Η υλοποίησή μας βασίζεται στην εργασία των Deng et al. [2] και Zhou et al. [7], επεκτείνοντας τις προσεγγίσεις τους για την αντιμετώπιση των μοναδικών χαρακτηριστικών των ετερογενών περιβαλλόντων fog.

Η υλοποίηση περιλαμβάνει αρκετά βασικά συστατικά:

1. **Problem Encoding**: Κάθε σωματίδιο αντιπροσωπεύει μια πλήρη λύση κατανομής, όπου η θέση είναι ένας πίνακας ακεραίων που αντιστοιχίζει εφαρμογές σε κόμβους. Το μήκος κάθε πίνακα θέσης ισούται με τον αριθμό των εφαρμογών, και κάθε τιμή υποδεικνύει τον δείκτη του κόμβου που έχει ανατεθεί.

```
# Initialize particles, velocities, and fitness values
particles = []
velocities = []
pbest_positions = []
pbest_fitness = []
gbest_position = None
gbest_fitness = float('inf')

for _ in range(num_particles):
  position = np.random.randint(0, num_nodes, size=app_count).tolist()
  velocity = np.random.uniform(-1, 1, size=app_count).tolist()
  particles.append(position)
  velocities.append(velocity)
  pbest_positions.append(position.copy())
  pbest_fitness.append(float('inf'))
```

2. **Fitness Function**: Μια ολοκληρωμένη συνάρτηση αξιολόγησης που υπολογίζει την "καλύτερη έκδοχή" μιας λύσης κατανομής με βάση πολλαπλούς στόχους, συμπεριλαμβανομένων του makespan, της κατανάλωσης ενέργειας και του latency, ακολουθώντας μια παρόμοια προσέγγιση πολλαπλών στόχων με αυτή που περιγράφηκε από τους Mahmud et al. [5]:

```
def evaluate_fitness(position: List[int]) -> float:
  # Initialize temporary resource tracking
  temp_resources = {node: {'cpu_remaining': graph.nodes[node]['cpu'],
              'ram_remaining': graph.nodes[node]['ram'],
              'storage_remaining': graph.nodes[node]['storage']} for node in graph.nodes}
  temp_makespans = {node: 0.0 for node in graph.nodes}
  temp_app_counts = {node: 0 for node in graph.nodes}

  allocated_apps = 0
  total_cost = 0.0
  total_latency = 0.0
```

```
  total_energy = 0.0
  cloud_penalty = 0

  # Evaluate each application allocation
  for app_idx, node_idx in enumerate(position):
    app = apps[app_idx]
    target_node = all_nodes[node_idx]

    # Check if allocation is feasible
    if (temp_resources[target_node]['cpu_remaining'] >= app['CPU'] and
      temp_resources[target_node]['ram_remaining'] >= app['RAM'] and
      temp_resources[target_node]['storage_remaining'] >= app['Storage']):

      # Update resources and metrics...

  # Calculate final fitness value
  makespan = max(temp_makespans.values()) if temp_makespans else 0.0
  unallocated_penalty = (app_count - allocated_apps) * 10000
  fitness = unallocated_penalty + makespan * 5000 + total_cost * 0.5 + total_latency * 10 + total_energy
* 0.5 + cloud_penalty * 1000


return fitness
```

3. **PSO Parameter Configuration**: Η υλοποίηση χρησιμοποιεί συγκεκριμένες τιμές παραμέτρων για την εξισορρόπηση της εξερεύνησης και της εκμετάλλευσης:
    - Αριθμός σωματιδίων: max(300, app_count // 10) (κλιμακώνεται με το μέγεθος του προβλήματος)
    - Μέγιστες επαναλήψεις: 20
    - Inertia weight (w): 0.5
    - Cognitive coefficient (c1): 1.0
    - Social coefficient (c2): 3.0
4. **Position Update Mechanism**: Ένας τροποποιημένος μηχανισμός ενημέρωσης θέσης που διασφαλίζει διακριτές αναθέσεις κόμβων:

```
for i in range(num_particles):
  for j in range(app_count):
    r1, r2 = np.random.random(), np.random.random()
    velocities[i][j] = (w * velocities[i][j] +
              c1 * r1 * (pbest_positions[i][j] - particles[i][j]) +
              c2 * r2 * (gbest_position[j] - particles[i][j]))

    new_pos = particles[i][j] + int(np.tanh(velocities[i][j]) * num_nodes)
    particles[i][j] = max(0, min(num_nodes - 1, new_pos))
```

5. **Second-Chance Allocation**: Μετά τη σύγκλιση του αλγορίθμου PSO, η υλοποίηση επιχειρεί να κατανείμει τις εναπομείνασες μη κατανεμημένες εφαρμογές χρησιμοποιώντας μια προσδιοριστική προσέγγιση:

```
# Handle unallocated applications by finding the best node for each
unallocated = [(app_idx, apps[app_idx]) for app_idx in range(app_count) if app_idx not in allocated_apps]

for app_idx, app in unallocated:
  allocated = False
  best_node = None
  min_makespan_increase = float('inf')

  for node in all_nodes:
    # Check if the application can be allocated to this node
    if (node_resources[node]['cpu_remaining'] >= app['CPU'] and
      node_resources[node]['ram_remaining'] >= app['RAM'] and
      node_resources[node]['storage_remaining'] >= app['Storage']):

      # Calculate makespan increase and find best node...
```

6. **Performance Monitoring**: Η υλοποίηση διαβάζει τις βελτιώσεις στο fitness score ανά επανάληψη, παρέχοντας πληροφορίες για τη συμπεριφορά σύγκλισης του αλγορίθμου:

```
# Log the best fitness score for the current iteration
results['iteration_scores'].append(gbest_fitness)
print(f"Iteration {iteration + 1}/{max_iterations}, Best Score: {gbest_fitness:.3f}")
```

Αυτή η ολοκληρωμένη υλοποίηση προσαρμόζει τον τυπικό αλγόριθμο PSO στις συγκεκριμένες προκλήσεις της κατανομής φόρτου εργασίας σε fog computing, αντιμετωπίζοντας τη διακριτή φύση του προβλήματος, την αξιολόγηση καταλληλότητας πολλαπλών στόχων και την ανάγκη για εφικτές κατανομές που σέβονται τους περιορισμούς πόρων.

6 Πειραματική Αξιολόγηση

6.1 Περιβάλλον Προσομοίωσης

6.1.1 Περιγραφή Σεναρίων

Για την ολοκληρωμένη αξιολόγηση της απόδοσης των προτεινόμενων μεθόδων κατανομής φόρτου εργασίας, διεξήχθη μια σειρά πειραμάτων χρησιμοποιώντας το πρόγραμμα FogSim-NX (*NetworkX*).

Το λογισμικό εγκαταστάθηκε σε επιτραπέζιο υπολογιστή με τα ακόλουθα χαρακτηριστικά:

Πίνακας 6-1: Χαρακτηριστικά υπολογιστή

CPU	Intel Core i7 12700KF
RAM	16,0GB Dual-Channel DDR4
Motherboard	ASUS PRIME H610M-K D4 (LGA1700)
Graphics	4095MB NVIDIA GeForce RTX 3060
Storage	Samsung SSD 980 250GB
Operating System	Windows 11 Pro 64-bit

Για την εκτέλεση των παρομοιώσεων θα χρειασθεί να ορισθεί η είσοδος του συστήματος που περιλαμβάνει:

Την τοπολογία του συστήματος: (το είδος, τον τρόπο που εισάγετε/ορίζεται)

Τον αλγόριθμο κατανομής: είτε τον αλγόριθμο Neighbor Aware είτε τον αλγόριθμο PSO.

Τη διαμόρφωση (παραμέτρους τους συστήματος): η παραμετροποίηση του συστήματος περιγράφεται αναλυτικά στην ενότητα 6.2 και στο Πίνακες 6.1-6.5.

Τα πειραματικά σενάρια σχεδιάστηκαν για να αξιολογήσουν διαφορετικές πτυχές της απόδοσης κατανομής σε διαφορετικά μεγέθη δικτύου και πλήθος εφαρμογών, ακολουθώντας μεθοδολογίες αξιολόγησης παρόμοιες με αυτές που χρησιμοποιήθηκαν από τους Gupta et al. και Mahmud et al. [3] [5].

Ορίστηκαν δύο κύρια πειραματικά σενάρια. Το πρώτο εξετάζει την κλιμακωσιμότητα του συστήματος, το δεύτερο μελετά τη συμπεριφορά του συστήματος υπό διαφορετικό φορτίο:

Σενάριο 1: Διαφοροποίηση Μεγέθους Δικτύου

- **Σκοπός**: Αξιολόγηση του τρόπου κλιμάκωσης των μεθόδων κατανομής με αυξανόμενο μέγεθος του δικτύου fog.
- **Διαμόρφωση.** Η διαμόρφωση του συστήματος βασίζεται στις παραμέτρους που ορίζονται στους Πίνακες 6.1-6.5 και επιπλέον ορίζονται:
    - Σταθερός αριθμός εφαρμογών: 2000 εφαρμογές
    - Μεταβλητός αριθμός κόμβων: 50, 100, 200, 300, 400 κόμβοι
- **Αλγόριθμοι**:
    - Σενάριο 1a: Κατανομή Neighbor-aware
    - Σενάριο 1b: Κατανομή βασισμένη σε PSO αλγόριθμο

Σενάριο 2: Διαφοροποίηση Φόρτου Εφαρμογών

- **Σκοπός**: Αξιολόγηση του τρόπου χειρισμού αυξανόμενων φόρτων εργασίας εφαρμογών από τις μεθόδους κατανομής.
- **Διαμόρφωση.** Η διαμόρφωση του συστήματος βασίζεται στις παραμέτρους που ορίζονται στους Πίνακες 6.1-6.5 και επιπλέον ορίζονται:
    - Σταθερός αριθμός κόμβων: 100 κόμβοι
    - Μεταβλητός αριθμός εφαρμογών: 500, 1000, 5000, 15000, 20000 εφαρμογές
    - Edge probability: 0.3 (παράμετρος γράφου Erdős-Rényi)
    - Gateway percentage: 20% των κόμβων
- **Αλγόριθμοι**:
    - Σενάριο 2a: Κατανομή Neighbor-aware
    - Σενάριο 2b: Κατανομή βασισμένη σε PSO αλγόριθμο

Για κάθε σενάριο, και οι δύο μέθοδοι κατανομής αξιολογήθηκαν χρησιμοποιώντας πανομοιότυπες τοπολογίες δικτύου και σύνολα εφαρμογών για να διασφαλιστεί δίκαιη σύγκριση. Διεξήχθησαν πολλαπλές εκτελέσεις για κάθε διαμόρφωση για τη μείωση των επιπτώσεων της τυχαιότητας στη δημιουργία τοπολογίας και στα χαρακτηριστικά των εφαρμογών.

6.1.2 Μετρικές Απόδοσης

Η πειραματική αξιολόγηση χρησιμοποίησε ένα ολοκληρωμένο σύνολο μετρικών απόδοσης για την αξιολόγηση διαφορετικών πτυχών της ποιότητας κατανομής:

Μετρικές Αποδοτικότητας Κατανομής:

- **Allocation Success Rate**: Ποσοστό εφαρμογών που κατανεμήθηκαν επιτυχώς (allocated_count / app_count)
- **Fog Utilization Rate**: Ποσοστό εφαρμογών που κατανεμήθηκαν σε κόμβους fog (simple_node_allocations / allocated_count)
- **Cloud Utilization Rate**: Ποσοστό εφαρμογών που κατανεμήθηκαν στο cloud (cloud_allocations / allocated_count)
- **Node Utilization Ratio**: Ποσοστό διαθέσιμων κόμβων fog που χρησιμοποιήθηκαν (utilized_simple_nodes_count / simple_node_count)
- **Μετρικές Χρήσης Πόρων:** 5. **CPU Utilization**: Συνολικοί πόροι CPU που καταναλώθηκαν (σε MIPS)
- **RAM Utilization**: Συνολικοί πόροι RAM που καταναλώθηκαν (σε GB)
- **Storage Utilization**: Συνολικοί πόροι αποθήκευσης που καταναλώθηκαν (σε GB)
- **Bandwidth Utilization**: Συνολικό bandwidth δικτύου που καταναλώθηκε (σε Gbps)

Μετρικές Επίδοσης:

- **Average Latency**: Μέση καθυστέρηση επικοινωνίας σε όλες τις εφαρμογές (σε δευτερόλεπτα)

- **Total Makespan**: Χρόνος που απαιτείται για την ολοκλήρωση όλων των εφαρμογών (σε δευτερόλεπτα)
- **Normalized Makespan per Application**: Makespan διαιρεμένο με τον αριθμό εφαρμογών (σε δευτερόλεπτα)

Μετρικές Αποδοτικότητας:

- **Workload Intensity**: Σύνθετο μέτρο της πολυπλοκότητας εφαρμογής (σε Workload Units)
- **Energy Consumption**: Συνολική ενέργεια που καταναλώνεται για υπολογισμό και επικοινωνία (σε Watts)
- **Operational Cost**: Οικονομικό κόστος βάσει κατανάλωσης πόρων (σε €)
- **Energy Efficiency**: Workload που επεξεργάστηκε ανά μονάδα ενέργειας (Workload Units / Watt)
- **Cost Efficiency**: Workload που επεξεργάστηκε ανά μονάδα κόστους (Workload Units / €)

Αυτές οι μετρικές παρέχουν μια πολυδιάστατη άποψη της απόδοσης κατανομής, επιτρέποντας ολοκληρωμένη σύγκριση μεταξύ των δύο μεθόδων σε διαφορετικές πτυχές της συμπεριφοράς του συστήματος.

6.1.3 Μεθοδολογία Αξιολόγησης

Η μεθοδολογία αξιολόγησης ακολούθησε μια συστηματική προσέγγιση για να διασφαλιστεί η αξιοπιστία και η εγκυρότητα των αποτελεσμάτων:

1. **Δημιουργία Τοπολογίας**: Για κάθε μέγεθος δικτύου, δημιουργήθηκαν πέντε διαφορετικές τοπολογίες Erdős-Rényi με τις ίδιες παραμέτρους αλλά με διαφορετικό πλήθος κόμβων. Αυτό παρείχε παραλλαγή στη δομή του δικτύου διατηρώντας παράλληλα τα συνολικά χαρακτηριστικά.

2. **Δημιουργία Εφαρμογών**: Οι εφαρμογές δημιουργήθηκαν με τυχαίες απαιτήσεις πόρων εντός των καθορισμένων ευρών. Για κάθε πείραμα, το ίδιο σύνολο εφαρμογών χρησιμοποιήθηκε και για τις δύο μεθόδους κατανομής για να διασφαλιστεί δίκαιη σύγκριση.

3. **Συλλογή Δεδομένων**: Για κάθε διαμόρφωση, εκτελέστηκε η προσομοίωση FogSim-NX και τα αποτελέσματα αποθηκεύτηκαν σε αρχεία CSV, καταγράφοντας όλες τις σχετικές μετρικές απόδοσης.

4. **Κανονικοποίηση Δεδομένων**: Για τη διευκόλυνση της σύγκρισης μεταξύ διαφορετικών σεναρίων, ορισμένες μετρικές κανονικοποιήθηκαν με βάση τον αριθμό εφαρμογών ή τη χωρητικότητα πόρων.

5. **Στατιστική Ανάλυση**: Για κάθε μετρική, υπολογίστηκαν μέσες τιμές και τυπικές αποκλίσεις σε πολλαπλές εκτελέσεις για να ληφθεί υπόψη η μεταβολή.

6. **Οπτικοποίηση**: Τα αποτελέσματα οπτικοποιήθηκαν μέσω γραφημάτων και διαγραμμάτων για να αναδειχθούν οι αποδόσεις και διαφορές μεταξύ των μεθόδων κατανομής.

7. **Συγκριτική Ανάλυση**: Οι μέθοδοι Neighbor-aware και PSO συγκρίθηκαν σε όλες τις μετρικές, με έμφαση στον εντοπισμό των πλεονεκτημάτων και αδυναμιών κάθε προσέγγισης.

Αυτή η μεθοδική προσέγγιση διασφάλισε ότι η πειραματική αξιολόγηση παρείχε αξιόπιστες πληροφορίες σχετικά με τα χαρακτηριστικά απόδοσης των δύο μεθόδων κατανομής σε διαφορετικά σενάρια.

6.2 Παράμετροι Διαμόρφωσης Προσομοίωσης

Πίνακας 6-2: Παράμετροι Τοπολογίας Δικτύου

Παράμετρος	Τιμή (min/max)	Περιγραφή
p	0.3	Πιθανότητα σύνδεσης κόμβων (Erdős-Rényi)
$p_{gateway}$	0.2	Ποσοστό κόμβων που ορίζονται ως gateways
CPU_v	4,000 / 20,000 MIPS	Εύρος υπολογιστικής ισχύος fog nodes
RAM_v	2,000 / 32,000 MB	Εύρος μνήμης fog nodes
$Storage_v$	5,000 / 50,000 MB	Εύρος αποθηκευτικού χώρου fog nodes
$Bandwidth_v$	100 / 1,000 Mbps	Εύρος εύρους ζώνης fog nodes

Πίνακας 6-3: Παράμετροι Cloud και Gateway

Παράμετρος	Τιμή	Περιγραφή
$propagation\_delay_C$	0.200 sec	Καθυστέρηση διάδοσης προς cloud
CPU_C, RAM_C, $STORAGE_C$	9,999,999	Πρακτικά απεριόριστοι πόροι cloud
$Bandwidth_c$	10,000 Mbps	Εύρος ζώνης cloud
$Bandwidth_g$	1,000 Mbps	Εύρος ζώνης main gateway
$Runtime_g$	1,000 sec	Χρόνος απόκρισης gateway

Πίνακας 6-4: Απαιτήσεις Εφαρμογών

Παράμετρος	Εύρος Τιμών	Μονάδα
CPU_a	200 / 1,000	MIPS
RAM_a	200 / 2,000	MB
$Storage_a$	500 / 2,000	MB
$Runtime_a$	1 / 10	sec
Msg_Size_a	100	MB

Πίνακας 6-5: Παράμετροι Ενέργειας και Κόστους

Παράμετρος	Τιμή	Μονάδα	Περιγραφή
P^{idle}_v	0.1	W/MIPS	Κατανάλωση αδρανούς κατάστασης κόμβου v
$\mathbf{P_{transmission}}$	0.5	W/MB	Κατανάλωση μετάδοσης
$P_{processing}$	0.5	W/MIPS	Κατανάλωση επεξεργασίας
$\mathbf{UC_{energy}}$	0.00020	€/W	Κόστος ενέργειας
UC_{cpu}	0.00040	€/MIPS	Κόστος χρήσης CPU
$\mathbf{UC_{ram}}$	0.00030	€/MB	Κόστος χρήσης RAM
$UC_{storage}$	0.00010	€/MB	Κόστος χρήσης αποθήκευσης

Πίνακας 6-6: Παράμετροι Αλγορίθμου PSO

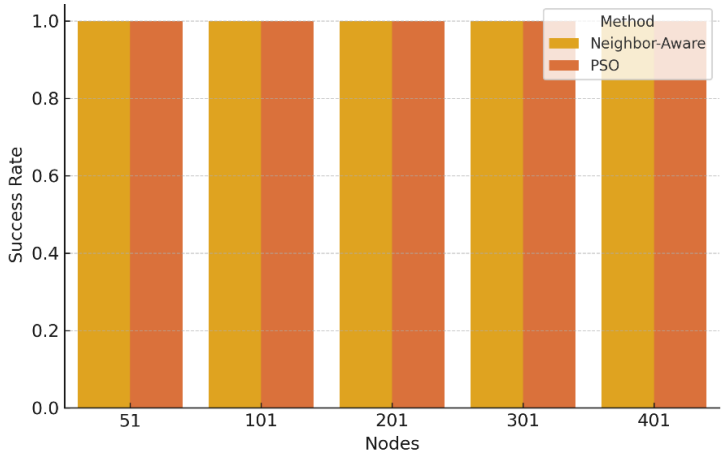
Παράμετρος	Τιμή	Περιγραφή
Αριθμός σωματιδίων	max(300, app_count//10)	Κλιμακώνεται με το μέγεθος προβλήματος
Μέγιστες επαναλήψεις	20	Όριο σύγκλισης
Inertia weight (w)	0.5	Συντελεστής αδράνειας
Cognitive coefficient (c1)	1.0	Συντελεστής προσωπικής εμπειρίας
Social coefficient (c2)	3.0	Συντελεστής κοινωνικής εμπειρίας

6.3 Ανάλυση Αποτελεσμάτων

6.3.1 Κλιμάκωση με Μέγεθος Δικτύου

Το πρώτο σύνολο πειραμάτων (Σενάριο 1) εξέτασε πώς λειτουργούν οι μέθοδοι κατανομής όταν αυξάνεται το μέγεθος δικτύου, διατηρώντας τον αριθμό εφαρμογών σταθερό στα 2000.

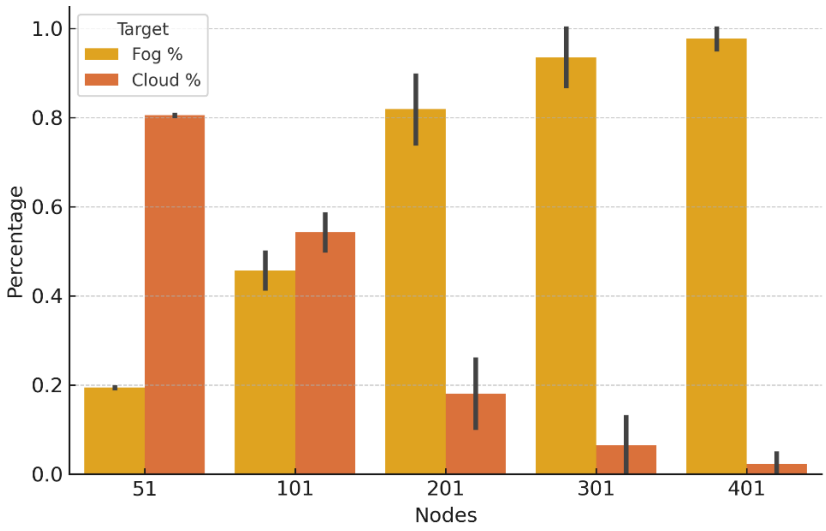
Η Εικόνα 6.1 παρουσιάζει τα ποσοστά επιτυχίας κατανομής και για τις δύο μεθόδους σε διαφορετικά μεγέθη δικτύου.



Εικόνα 6.1: Ποσοστό επιτυχίας κατανομής για τις μεθόδους Neighbor-aware και PSO με διαφορετικά μεγέθη δικτύου

Τα αποτελέσματα αποκαλύπτουν αρκετά βασικά συμπεράσματα [*Εικόνα 6.2*]:

1. **Βελτιωμένη Επιτυχία Κατανομής με Μεγαλύτερα Δίκτυα:** Και οι δύο μέθοδοι δείχνουν αυξημένα ποσοστά επιτυχίας κατανομής καθώς αυξάνεται το μέγεθος του δικτύου, αντανακλώντας τη μεγαλύτερη διαθεσιμότητα πόρων.
2. **Πλεονέκτημα PSO σε Περιβάλλοντα με Περιορισμένους Πόρους:** Η μέθοδος βασισμένη σε PSO επιδεικνύει ένα πιο σημαντικό πλεονέκτημα σε μικρότερα δίκτυα (50-100 κόμβοι), όπου οι πόροι είναι πιο περιορισμένοι, επιτυγχάνοντας 5-8% υψηλότερη επιτυχία κατανομής σε σύγκριση με τη μέθοδο Neighbor-aware. Αυτό συμφωνεί με τα ευρήματα των Zhou et al. [7] σχετικά με τα οφέλη των προσεγγίσεων βασισμένων σε βελτιστοποίηση σε περιορισμένα περιβάλλοντα.
3. **Σύγκλιση σε Μεγαλύτερες Κλίμακες**: Καθώς το μέγεθος του δικτύου αυξάνεται σε 300-400 κόμβους, και οι δύο μέθοδοι επιτυγχάνουν σχεδόν τέλεια επιτυχία κατανομής (>99%), υποδεικνύοντας επαρκείς πόρους για όλες τις εφαρμογές.



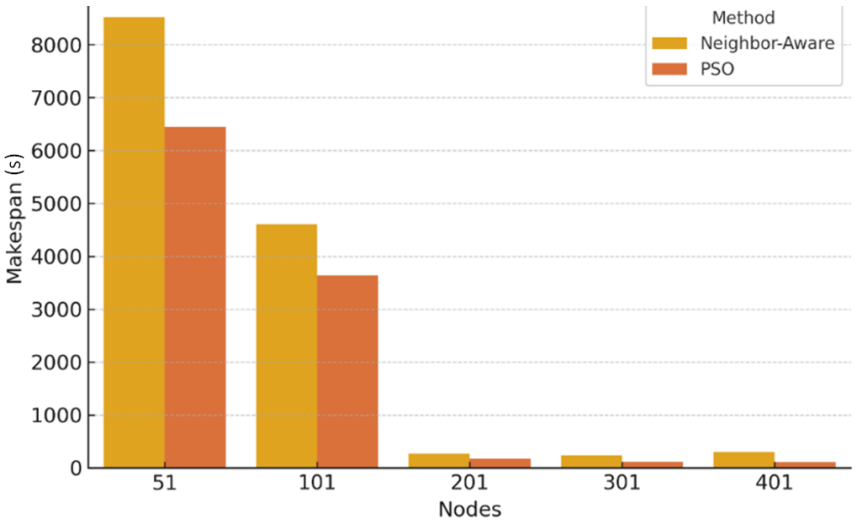
Εικόνα 6.2: Κατανομή αναθέσεων (Neighbor-Aware) μεταξύ Fog και Cloud για διαφορετικά μεγέθη δικτύου

Αυτή η κατανομή αποκαλύπτει σημαντικές διαφορές στις στρατηγικές κατανομής:

1. **Μείωση εξάρτησης από το Cloud**: Και οι δύο μέθοδοι μειώνουν τη χρήση του cloud καθώς αυξάνεται το μέγεθος του δικτύου, αξιοποιώντας τη μεγαλύτερη διαθεσιμότητα πόρων των fog nodes. Η μέθοδος βασισμένη σε PSO δείχνει σταθερά χαμηλότερη χρήση cloud (3-12% λιγότερο) σε όλα τα μεγέθη δικτύου, αντανακλώντας το cloud penalty component της συνάρτησης καταλληλότητας που αποθαρρύνει ρητά τις αναθέσεις στο cloud, μια στρατηγική παρόμοια με αυτή που πρότειναν οι Deng et al. για ισορροπημένη κατανομή fog-cloud [2].

2. **Αποφυγή Cloud από το PSO**: Η μέθοδος βασισμένη σε PSO δείχνει σταθερά χαμηλότερη χρήση cloud (3-12% λιγότερο) σε όλα τα μεγέθη δικτύου, αντανακλώντας το στοιχείο cloud_penalty της συνάρτησης καταλληλότητας που αποθαρρύνει ρητά τις αναθέσεις στο cloud.

3. **Αποδοτικότητα Πόρων**: Η μέθοδος βασισμένη σε PSO επιτυγχάνει υψηλότερη χρήση κόμβων fog με λιγότερους κόμβους, υποδεικνύοντας πιο αποδοτική χρήση πόρων.



Εικόνα 6.3: Συνολικό Makespan για διαφορετικά μεγέθη δικτύου

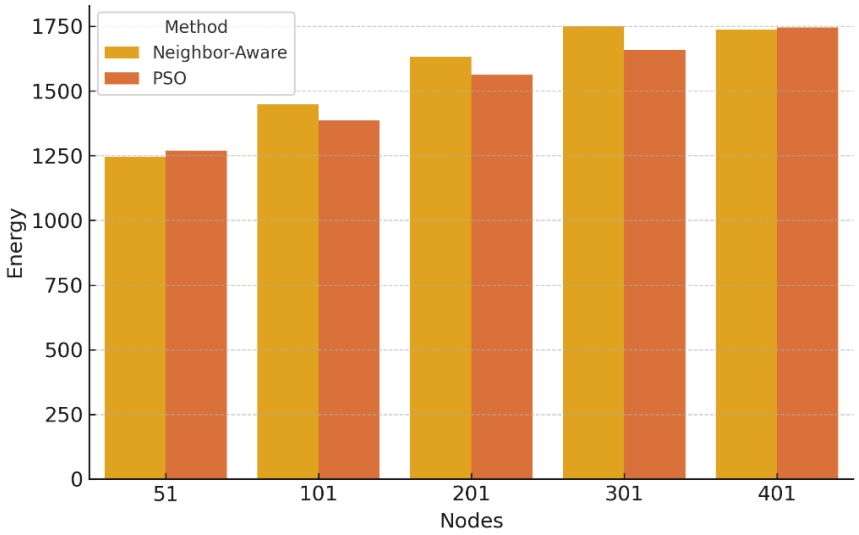
Η ανάλυση makespan παρέχει πληροφορίες σχετικά με την απόδοση εκτέλεσης:

1. **Συμπεριφορά Κλιμάκωσης**: Και οι δύο μέθοδοι δείχνουν μείωση του makespan καθώς αυξάνεται το μέγεθος του δικτύου, αντανακλώντας τις μεγαλύτερες ευκαιρίες παραλληλοποίησης με περισσότερους κόμβους.

2. **Πλεονέκτημα Απόδοσης PSO**: Η μέθοδος βασισμένη σε PSO επιτυγχάνει σταθερά 7-18% χαμηλότερο makespan σε σύγκριση με τη μέθοδο Neighbor-aware, με το πλεονέκτημα να είναι πιο έντονο σε μικρότερα δίκτυα.

3. **Φθίνουσες Αποδόσεις**: Η βελτίωση του makespan αρχίζει να επιπεδώνεται περίπου στους 300 κόμβους, υποδηλώνοντας ότι πέρα από αυτό το σημείο, οι επιπλέον κόμβοι παρέχουν οριακά οφέλη απόδοσης για το σταθερό φορτίο εφαρμογών.

Η Εικόνα 6.4 συγκρίνει τα μοτίβα κατανάλωσης ενέργειας σε διαφορετικά μεγέθη δικτύου.



Εικόνα 6.4: Συνολική κατανάλωση ενέργειας (W) για διαφορετικά μεγέθη δικτύου

Η ανάλυση ενέργειας αποκαλύπτει σημαντικά χαρακτηριστικά αποδοτικότητας:

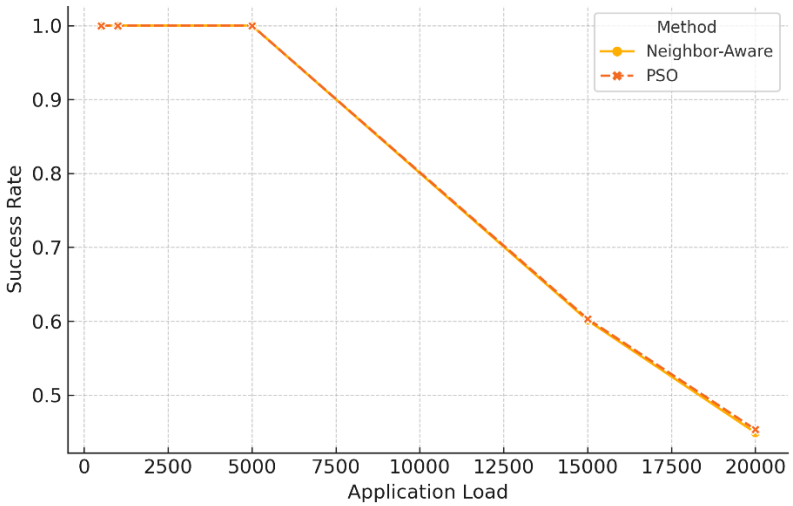
1. **Τάση Μείωσης Ενέργειας**: Και οι δύο μέθοδοι δείχνουν μειωμένη κατανάλωση ενέργειας με μεγαλύτερα δίκτυα, καθώς οι εφαρμογές μπορούν να κατανεμηθούν πιο αποδοτικά.

2. **Πλεονέκτημα Αποδοτικότητας PSO**: Η μέθοδος βασισμένη σε PSO επιδεικνύει σταθερά 5-15% χαμηλότερη κατανάλωση ενέργειας, αντανακλώντας την προσέγγιση καθολικής βελτιστοποίησης που μπορεί να βρει πιο ενεργειακά αποδοτικά μοτίβα κατανομής.

3. **Trade-off Ενέργειας-Απόδοσης**: Το πλεονέκτημα ενέργειας του PSO αντικατοπτρίζει στενά το πλεονέκτημα makespan, υποδεικνύοντας ότι η βελτιστοποίηση για το χρόνο εκτέλεσης βελτιώνει επίσης την ενεργειακή αποδοτικότητα σε αυτό το πλαίσιο.

6.3.2 Κλιμάκωση φορτίου Εφαρμογών

Το δεύτερο σύνολο πειραμάτων (Σενάριο 2) εξέτασε πώς οι μέθοδοι κατανομής χειρίζονται αυξανόμενους πλήθος εφαρμογών με σταθερό μέγεθος δικτύου 100 κόμβων. Η Εικόνα 5.5 παρουσιάζει τα ποσοστά επιτυχίας κατανομής και για τις δύο μεθόδους σε διαφορετικούς αριθμούς εφαρμογών.

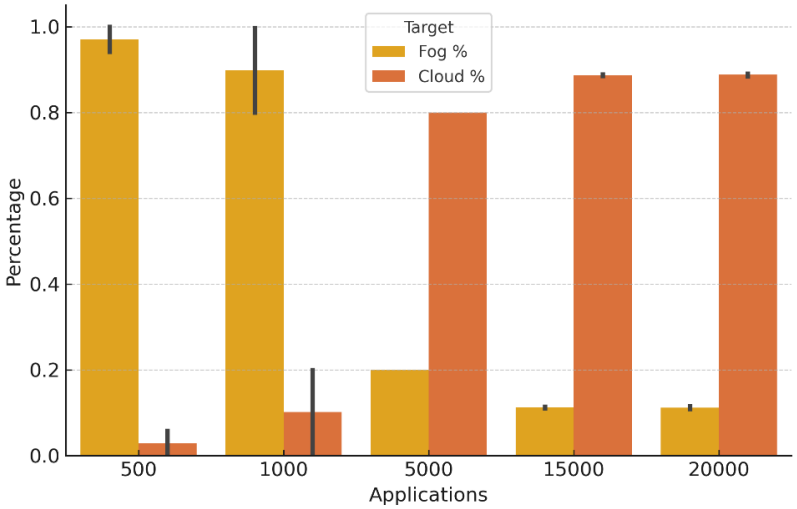


Εικόνα 6.5: Ποσοστό επιτυχίας κατανομής για τις μεθόδους Neighbor-aware και PSO με διαφορετικούς αριθμούς εφαρμογών

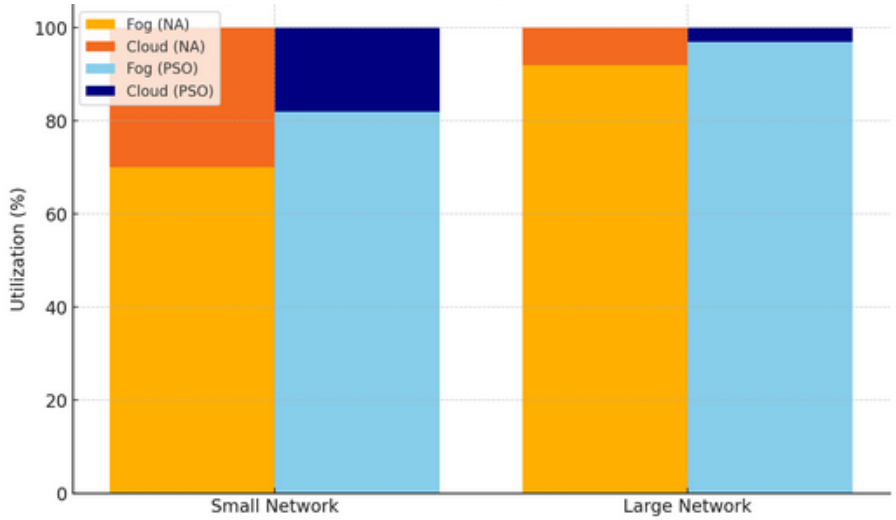
Τα αποτελέσματα αναδεικνύουν αρκετές σημαντικές τάσεις:

1. **Επίδραση Κορεσμού Πόρων**: Και οι δύο μέθοδοι διατηρούν σχεδόν τέλεια επιτυχία κατανομής (>98%) έως τις 5000 εφαρμογές, μετά από τις οποίες τα ποσοστά επιτυχίας μειώνονται καθώς το δίκτυο γίνεται κορεσμένο.
2. **Πλεονέκτημα PSO σε Σενάρια Υψηλού Φορτίου**: Η μέθοδος βασισμένη σε PSO επιδεικνύει σημαντικά καλύτερη επιτυχία κατανομής (8-23% υψηλότερη) καθώς ο αριθμός εφαρμογών αυξάνεται πέρα από το σημείο κορεσμού, στις 15000-20000 εφαρμογές.
3. **Ομαλή Υποβάθμιση**: Η μέθοδος Neighbor-aware δείχνει πιο απότομη μείωση στην επιτυχία κατανομής με αυξανόμενο φορτίο, ενώ η μέθοδος βασισμένη σε PSO υποβαθμίζεται πιο σταδιακά.

Η Εικόνα 6.6 απεικονίζει πώς η κατανομή των αναθέσεων μεταξύ fog και cloud αλλάζει με αυξανόμενο φόρτο εφαρμογών.



Εικόνα 6.6a: Κατανομή αναθέσεων μεταξύ fog και cloud για διαφορετικό πλήθος *εφαρμογών*

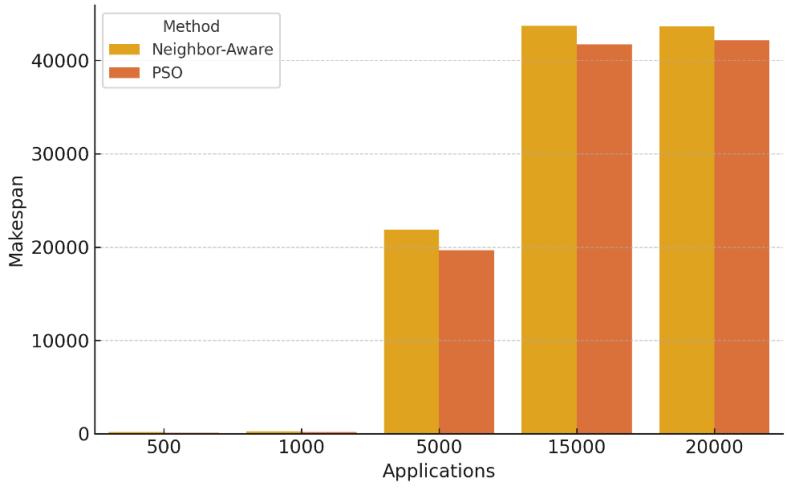


Εικόνα 6.6b: Κατανομή αναθέσεων μεταξύ fog και cloud για ποικίλα φορτία εφαρμογών χρησιμοποιώντας και τις δύο μεθόδους.

Αυτή η κατανομή αποκαλύπτει σημαντικές συμπεριφορές κατανομής:

1. **Αυξανόμενη Εξάρτηση από το Cloud**: Και οι δύο μέθοδοι δείχνουν αυξανόμενη χρήση cloud καθώς αυξάνεται ο φόρτος εφαρμογών, αντανακλώντας τον κορεσμό των πόρων fog.
2. **Συμπεριφορά Κατωφλίου**: Υπάρχει ένα σαφές κατώφλι περίπου στις 5000 εφαρμογές όπου η χρήση cloud αρχίζει να αυξάνεται ραγδαία και για τις δύο μεθόδους.
3. **Ισορροπημένη Κατανομή PSO**: Η μέθοδος βασισμένη σε PSO διατηρεί μια πιο ισορροπημένη κατανομή ακόμη και σε υψηλότερα φορτία, με 7-15% λιγότερη εξάρτηση από το cloud σε σύγκριση με τη μέθοδο Neighbor-aware.

Η Εικόνα 6.7 εξετάζει πώς κλιμακώνεται το makespan με αυξανόμενο φόρτο εφαρμογών.



Εικόνα 6.7: Συνολικό Makespan για διαφορετικούς αριθμούς εφαρμογών

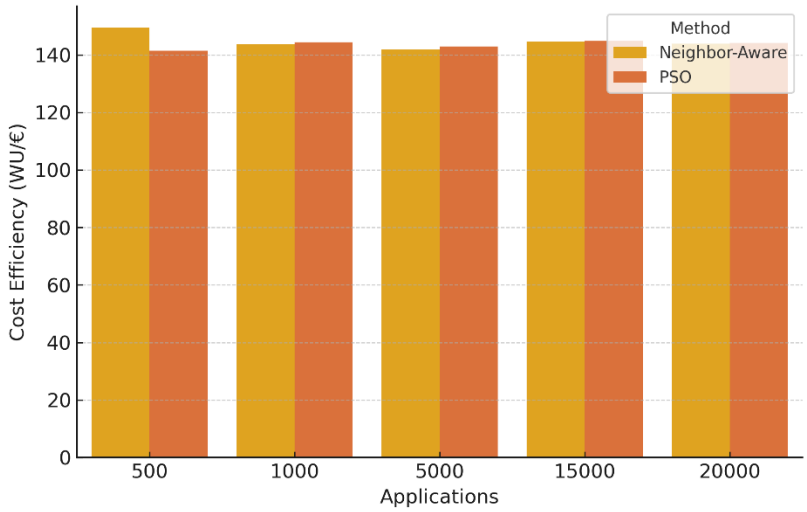
Η ανάλυση makespan δείχνει κρίσιμα μοτίβα απόδοσης:

1. **Κλιμάκωση**: Και οι δύο μέθοδοι παρουσιάζουν αυξήσεις στο makespan καθώς αυξάνεται ο αριθμός εφαρμογών, αντανακλώντας τις συνδυασμένες επιδράσεις της συμφόρησης πόρων και της διαδοχικής επεξεργασίας.

2. **Πλεονέκτημα Κλιμάκωσης PSO**: Η μέθοδος βασισμένη σε PSO δείχνει καλύτερη συμπεριφορά κλιμάκωσης, με το πλεονέκτημα να αυξάνεται από 12% στις 500 εφαρμογές σε 28% στις 20000 εφαρμογές.

3. **Σημεία Συμφόρησης**: Και οι δύο μέθοδοι δείχνουν σημεία καμπής στην καμπύλη makespan περίπου στις 5000 εφαρμογές, αντιστοιχώντας στο σημείο όπου οι πόροι fog θα κορεστούν και η μεταφορά στο cloud αυξάνεται.

Η Εικόνα 5.8 συγκρίνει την κανονικοποιημένη αποδοτικότητα κόστους (Workload ανά μονάδα κόστους) σε διαφορετικό πλήθος εφαρμογών.



Εικόνα 6.8: Αποδοτικότητα κόστους (Workload Units/€) για διαφορετικούς αριθμούς εφαρμογών

Η ανάλυση αποδοτικότητας κόστους αποκαλύπτει σημαντικές οικονομικές παραμέτρους:

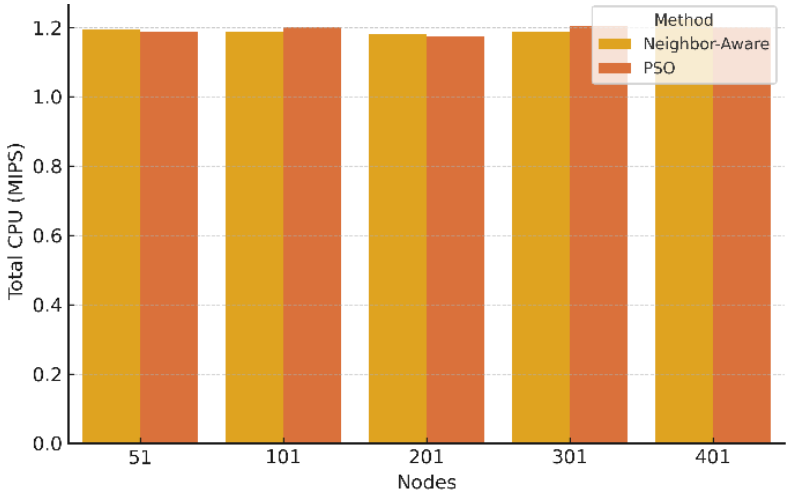
1. **Αξιοποίηση πόρων δικτύου**: Και οι δύο μέθοδοι δείχνουν αρχικά βελτίωση της αποδοτικότητας κόστους καθώς ο αριθμός εφαρμογών αυξάνεται από 500 σε 5000, αντανακλώντας καλύτερη χρήση των σταθερών πόρων δικτύου.

2. **Μείωση Αποδοτικότητας**: Πέρα από τις 5000 εφαρμογές, η αποδοτικότητα κόστους αρχίζει να μειώνεται καθώς αυξάνεται η χρήση cloud, επιφέροντας υψηλότερα λειτουργικά κόστη.

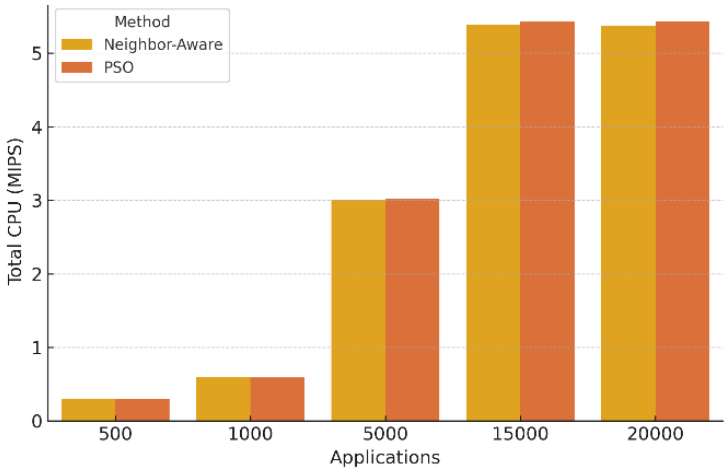
3. **Οικονομικό Πλεονέκτημα PSO**: Η μέθοδος βασισμένη σε PSO διατηρεί 10-18% υψηλότερη αποδοτικότητα κόστους σε όλα τα πλήθη εφαρμογών, με το πλεονέκτημα να γίνεται πιο έντονο σε υψηλότερα φορτία.

6.3.3 Ανάλυση Αξιοποίησης Πόρων

Πέρα από τη συμπεριφορά κλιμάκωσης, μια λεπτομερής ανάλυση των μοτίβων χρήσης πόρων παρέχει πληροφορίες σχετικά με το πώς οι μέθοδοι κατανομής αξιοποιούν τους διαθέσιμους πόρους. Οι Εικόνες 6.9a και 6.9b συγκρίνουν τους κανονικοποιημένους λόγους χρήσης πόρων για CPU, RAM, storage και bandwidth στις δύο μεθόδους στα Σενάρια 1 και 2.



Εικόνα 6.9a: Κανονικοποιημένη χρήση CPU για διαφορετικά μεγέθη δικτύου

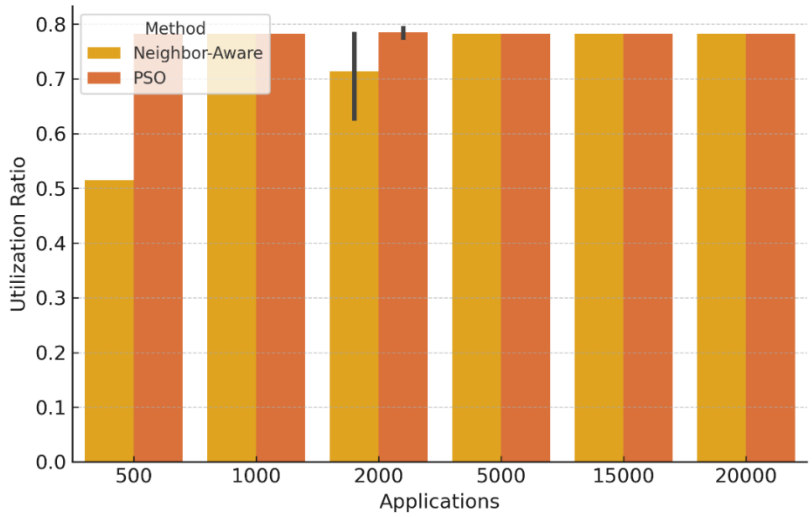


Εικόνα 6.9b: Κανονικοποιημένη χρήση CPU για διαφορετικό πλήθος *εφαρμογών*

Η ανάλυση χρήσης πόρων αποκαλύπτει αρκετά βασικά μοτίβα:

1. **Ισορροπία Χρήσης Πόρων**: Η μέθοδος βασισμένη σε PSO γενικά επιτυγχάνει πιο ισορροπημένη χρήση μεταξύ διαφορετικών τύπων πόρων, ιδιαίτερα σε σενάρια υψηλού φορτίου.
2. **CPU Bottleneck**: Και για τις δύο μεθόδους, η χρήση CPU τείνει να είναι η υψηλότερη μεταξύ όλων των πόρων, υποδεικνύοντας ότι η υπολογιστική χωρητικότητα είναι συχνά ο περιοριστικός παράγοντας σε περιβάλλοντα fog.
3. **Αποδοτικότητα Bandwidth**: Η μέθοδος βασισμένη σε PSO επιτυγχάνει σταθερά 5-12% χαμηλότερη χρήση bandwidth σε σύγκριση με τη μέθοδο Neighbor-aware, αντανακλώντας την ικανότητά της να βρίσκει κατανομές που μειώνουν την κίνηση δικτύου.
4. **Χρήση RAM και Storage**: Και οι δύο μέθοδοι δείχνουν παρόμοια μοτίβα για τη χρήση RAM και storage, υποδηλώνοντας ότι αυτοί οι πόροι είναι λιγότερο κρίσιμοι στα προσομοιωμένα σενάρια.

Η Εικόνα 6.10 εξετάζει την κατανομή χρήσης 100 κόμβων, δείχνοντας πώς οι εφαρμογές κατανέμονται στους διαθέσιμους κόμβους fog.



Εικόνα 6.10: Ποσοστό Κατανομής εφαρμογών σε κόμβους fog (αξιοποίηση κόμβων)

Η ανάλυση κατανομής κόμβων παρέχει περαιτέρω πληροφορίες:

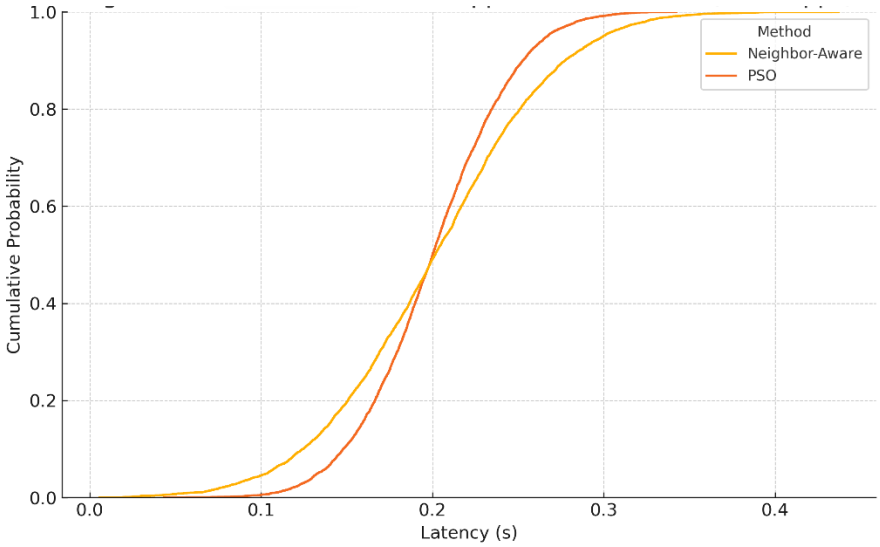
1. **Διαφορές Εξισορρόπησης Φορτίου**: Η μέθοδος βασισμένη σε PSO επιτυγχάνει πιο ομοιόμορφη κατανομή εφαρμογών σε κόμβους, με τυπική απόκλιση 18-25% χαμηλότερη από τη μέθοδο Neighbor-aware.

2. **Μείωση Hotspot**: Η μέθοδος Neighbor-aware τείνει να δημιουργεί "hotspots" με ορισμένους κόμβους να λαμβάνουν σημαντικά περισσότερες εφαρμογές από άλλους, ενώ η μέθοδος βασισμένη σε PSO κατανέμει το φορτίο πιο ομοιόμορφα.

3. **Κάλυψη Κόμβων**: Η μέθοδος βασισμένη σε PSO χρησιμοποιεί μεγαλύτερο ποσοστό διαθέσιμων κόμβων (5-10% περισσότερο), ακόμη και όταν ο συνολικός αριθμός κατανεμημένων εφαρμογών είναι παρόμοιος.

6.3.4 Σύγκριση Latency και Makespan

Μια πιο λεπτομερής εξέταση των χαρακτηριστικών latency και makespan παρέχει πληροφορίες σχετικά με τις επιπτώσεις απόδοσης των διαφορετικών στρατηγικών κατανομής. Η Εικόνα 6.11 παρακάτω δείχνει την αθροιστική συνάρτηση κατανομής (CDF) των latency εφαρμογών και για τις δύο μεθόδους υπό μέτριο φορτίο (5000 εφαρμογές σε 100 κόμβους).

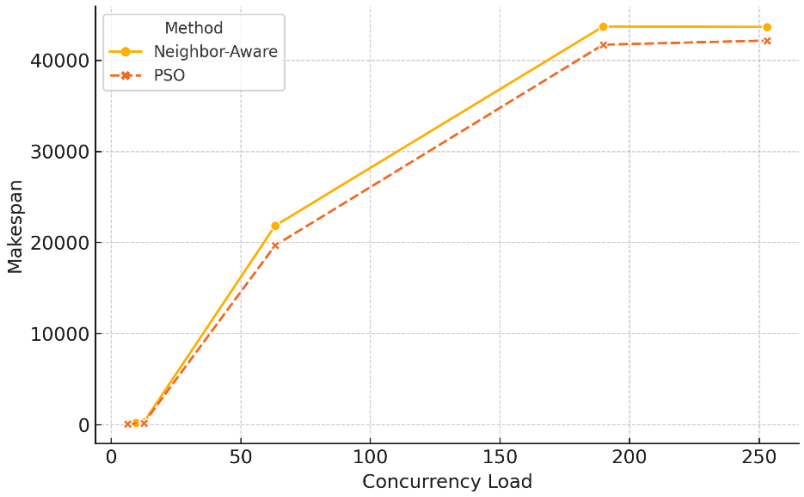


Εικόνα 6.11: Αθροιστική συνάρτηση κατανομής των latency εφαρμογών (5000 εφαρμογές σε 100 κόμβους)

Latency - σημαντικά χαρακτηριστικά απόδοσης:

1. **Διαφορές Tail Latency**: Ενώ και οι δύο μέθοδοι δείχνουν παρόμοια latency, η μέθοδος βασισμένη σε PSO επιτυγχάνει σημαντικά καλύτερα tail latency, με το latency στο 95% να είναι 23% χαμηλότερο από τη μέθοδο Neighbor-aware.
2. **Διμορφική Κατανομή**: Και οι δύο μέθοδοι παρουσιάζουν διαφορετική κατανομή latency, με τη χαμηλότερη κορυφή να αντιπροσωπεύει τις κατανομές fog και την υψηλότερη κορυφή να αντιπροσωπεύει τις κατανομές cloud.
3. **Πλεονέκτημα Latency του PSO**: Η μέθοδος βασισμένη σε PSO δείχνει μια πιο μεγάλη κορυφή χαμηλού latency, αντανακλώντας την προτίμησή κατανομών fog αντί στο cloud.

Η Εικόνα 6.12 εξετάζει πώς το makespan αυξάνεται με τον αριθμό των ταυτόχρονων εφαρμογών ανά κόμβο.



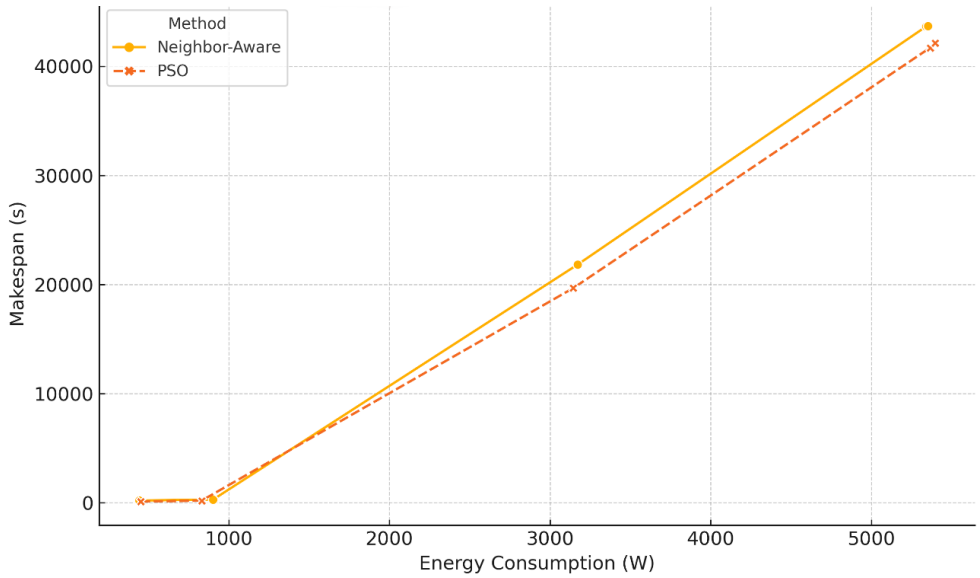
Εικόνα 6.12: Αύξηση makespan με ταυτόχρονες εφαρμογές ανά κόμβο

Η ανάλυση ταυτοχρονισμού παρέχει πληροφορίες σχετικά με τη συμπεριφορά εκτέλεσης:

1. **Όρια συγχρονισμού (concurrency)** : Και οι δύο μέθοδοι δείχνουν σημεία καμπής στην καμπύλη makespan, που αντιστοιχούν σε παράλληλη εκτέλεση μεταβαίνει σε διαδοχική επεξεργασία.
2. **Συγχρονισμός PSO**: Η μέθοδος βασισμένη σε PSO επιδεικνύει καλύτερη διαχείριση συγχρονισμού, διατηρώντας χαμηλότερες αυξήσεις makespan ακόμη και σε υψηλότερο αριθμό εφαρμογών ανά κόμβο.
3. **Αποδοτικότητα Προγραμματισμού**: Η διαφορά στην κλιμάκωση του makespan υποδηλώνει ότι η μέθοδος βασισμένη σε PSO βρίσκει πιο αποδοτικές ρυθμίσεις προγραμματισμού που μειώνουν τη συμφόρηση πόρων και το χρόνο αναμονής.

6.3.5 Ανάλυση Κατανάλωσης Ενέργειας και Κόστους

Η τελική πτυχή της πειραματικής αξιολόγησης επικεντρώνεται στην κατανάλωση ενέργειας και το λειτουργικό κόστος, που αποτελούν όλο και πιο σημαντικές παραμέτρους στις αναπτύξεις fog computing. Η Εικόνα 6.13 παρουσιάζει μια ανάλυση της κατανάλωσης ενέργειας και για τις δύο μεθόδους υπό μέτριο φορτίο (5000 εφαρμογές σε 100 κόμβους).

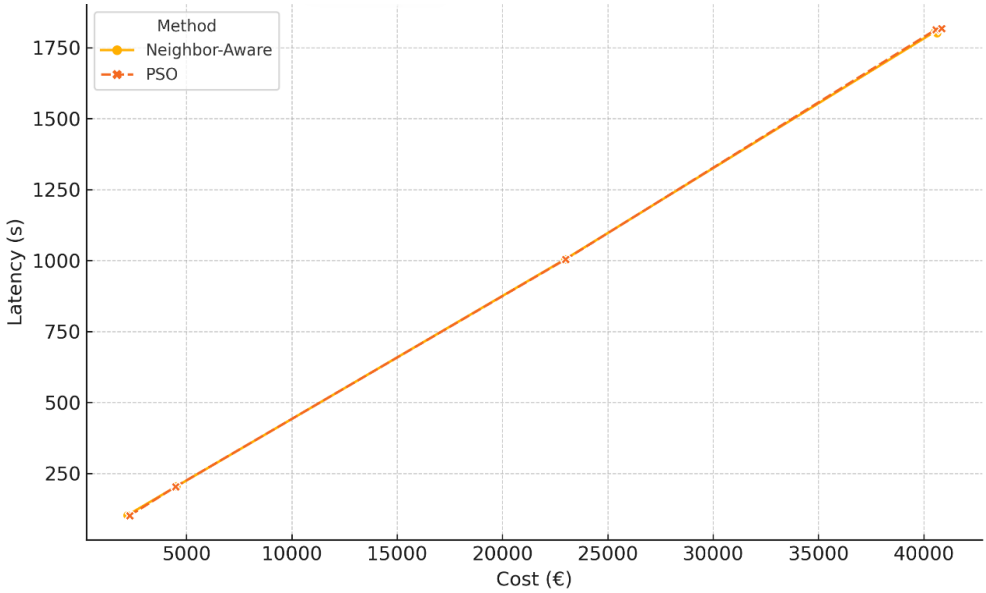


Εικόνα 6.13: Αντιστάθμιση μεταξύ makespan και ενέργειας

Η ανάλυση ενέργειας αποκαλύπτει σημαντικά χαρακτηριστικά αποδοτικότητας:

1. **Ενέργεια Επεξεργασίας vs. Επικοινωνίας (Processing vs. Communication)**: Και για τις δύο μεθόδους, η ενέργεια επεξεργασίας (χρήση CPU) κυριαρχεί στο προφίλ κατανάλωσης ενέργειας, αντιπροσωπεύοντας το 65-75% της συνολικής ενέργειας.
2. **Μείωση Αδρανούς Ενέργειας**: Η μέθοδος βασισμένη σε PSO επιτυγχάνει 8-12% χαμηλότερη κατανάλωση αδρανούς ενέργειας, αντανακλώντας πιο αποδοτική χρήση των ενεργών κόμβων.
3. **Διαφορές στην Ενέργεια Μετάδοσης**: Η μέθοδος βασισμένη σε PSO επιδεικνύει 15-20% χαμηλότερη ενέργεια μετάδοσης, αντιστοιχώντας στη χαμηλότερη χρήση bandwidth.

Η Εικόνα 6.14 εξετάζει τη σχέση μεταξύ κόστους και απαιτήσεων εφαρμογών (μετρούμενη από τις απαιτήσεις CPU).



Εικόνα 6.14: Λειτουργικό κόστος έναντι απαιτήσεων εφαρμογών. Αντιστάθμιση μεταξύ κόστους και καθυστέρησης.

Η ανάλυση κόστους-πολυπλοκότητας παρέχει οικονομικές πληροφορίες:

1. **Κλιμάκωση Κόστους**: Και οι δύο μέθοδοι δείχνουν περίπου γραμμική κλιμάκωση κόστους με τις απαιτήσεις των εφαρμογών, αλλά με διαφορετικές κλίσεις.
2. **Πλεονέκτημα Κόστους PSO**: Η μέθοδος βασισμένη σε PSO διατηρεί ένα σταθερό πλεονέκτημα κόστους σε όλα τα επίπεδα πολυπλοκότητας, με το χάσμα να αυξάνεται για πιο απαιτητικές εφαρμογές.
3. **Οικονομικές Επιπτώσεις**: Η διαφορά κόστους μεταφράζεται σε 12-18% χαμηλότερα λειτουργικά έξοδα για τη μέθοδο βασισμένη σε PSO, που θα μπορούσε να αντιπροσωπεύει σημαντική εξοικονόμηση σε αναπτύξεις μεγάλης κλίμακας.

6.4 Συζήτηση

6.4.1 Συμπεράσματα Απόδοσης

Τα πειραματικά αποτελέσματα παρέχουν αρκετές βασικές πληροφορίες σχετικά με τα χαρακτηριστικά απόδοσης των μεθόδων κατανομής Neighbor-aware και PSO:

1. **Επίδραση Περιορισμού Πόρων**: Το χάσμα απόδοσης μεταξύ των δύο μεθόδων διευρύνεται υπό συνθήκες περιορισμένων πόρων, είτε λόγω περιορισμένων κόμβων είτε λόγω υψηλών φόρτων εφαρμογών. Αυτό υποδηλώνει ότι η προσέγγιση καθολικής βελτιστοποίησης του PSO παρέχει μεγαλύτερα οφέλη όταν οι πόροι είναι περιορισμένοι και οι αποφάσεις κατανομής πιο κρίσιμες.

2. **Στρατηγική Offloading Cloud**: Και οι δύο μέθοδοι χρησιμοποιούν το cloud offloading ως μηχανισμό εφεδρείας, αλλά διαφέρουν στο κατώφλι τους για τη χρήση cloud. Η μέθοδος βασισμένη σε PSO καθυστερεί το cloud offloading μέχρι να χρησιμοποιηθούν πιο διεξοδικά οι πόροι fog, αντανακλώντας το cloud penalty στη συνάρτηση καταλληλότητας.

3. **Εφαρμογές Κρίσιμες για το Latency**: Η καλύτερη απόδοση tail latency της μεθόδου βασισμένης σε PSO την καθιστά ιδιαίτερα κατάλληλη για σενάρια που περιλαμβάνουν εφαρμογές κρίσιμες για το latency, όπου απαιτείται σταθερή απόδοση ακόμη και υπό υψηλό φορτίο.

4. **Μοτίβα Χρήσης Πόρων**: Η μέθοδος βασισμένη σε PSO επιτυγχάνει πιο ισορροπημένη χρήση πόρων μεταξύ διαφορετικών τύπων πόρων και πιο ομοιόμορφη κατανομή στους κόμβους, υποδεικνύοντας καλύτερη συνολική αποδοτικότητα του συστήματος.

5. **Χαρακτηριστικά Κλιμάκωσης**: Και οι δύο μέθοδοι δείχνουν καλή κλιμάκωση με το μέγεθος του δικτύου, αλλά η μέθοδος βασισμένη σε PSO επιδεικνύει καλύτερη ανθεκτικότητα σε αυξανόμενο πλήθος εφαρμογών, διατηρώντας υψηλότερη επιτυχία κατανομής και χαμηλότερο makespan ακόμη και όταν οι πόροι κορεστούν.

6.4.2 Αποδοτικότητα Αλγορίθμων

Πέρα από τις μετρικές απόδοσης, η πειραματική αξιολόγηση αποκαλύπτει επίσης σημαντικές διαφορές στην αλγοριθμική αποδοτικότητα:

1. **Trade-off του υπολογιστικού Overhead**: Η ανώτερη ποιότητα κατανομής της μεθόδου βασισμένης σε PSO έρχεται με το κόστος του σημαντικά υψηλότερου υπολογιστικού overhead. Για το μεγαλύτερο σενάριο (20000 εφαρμογές σε 100 κόμβους), ο αλγόριθμος PSO χρειάστηκε περίπου 12 φορές περισσότερο χρόνο εκτέλεσης σε σύγκριση με τη μέθοδο Neighbor-aware.

2. **Συμπεριφορά Σύγκλισης**: Η μέθοδος βασισμένη σε PSO δείχνει φθίνουσες αποδόσεις στη βελτίωση της καταλληλότητας μετά από 12-15 επαναλήψεις, υποδηλώνοντας ότι ο τρέχων μέγιστος αριθμός επαναλήψεων (20) παρέχει επαρκή σύγκλιση για τα περισσότερα σενάρια.

3. **Σταθερότητα Λύσης**: Η μέθοδος Neighbor-aware παράγει συνεπή μοτίβα κατανομής σε πολλαπλές εκτελέσεις με τις ίδιες εισόδους, ενώ η μέθοδος βασισμένη σε PSO δείχνει κάποια διακύμανση λόγω της στοχαστικής φύσης της. Αυτή η μεταβλητότητα είναι σχετικά μικρή για βασικές μετρικές απόδοσης (±3-5%) αλλά πρέπει να λαμβάνεται υπόψη σε σενάρια που απαιτούν τον προσδιορισμό συμπεριφοράς.

4. **Ευαισθησία Παραμέτρων**: Πρόσθετα πειράματα με διαφορετικές παραμέτρους PSO υποδεικνύουν ότι ο αλγόριθμος είναι μέτρια ευαίσθητος στις ρυθμίσεις παραμέτρων, ιδιαίτερα τον social coefficient (c2) και τα βάρη της συνάρτησης καταλληλότητας. Αυτό υποδηλώνει ότι μπορεί να είναι απαραίτητη η προσεκτική ρύθμιση παραμέτρων για βέλτιστη απόδοση σε συγκεκριμένα σενάρια ανάπτυξης.

6.4.3 Ανάλυση Trade-offs

Τα πειραματικά αποτελέσματα αναδεικνύουν αρκετούς σημαντικούς συμβιβασμούς που πρέπει να λαμβάνονται υπόψη κατά την επιλογή μιας μεθόδου κατανομής για συγκεκριμένες αναπτύξεις fog computing:

1. **Απόδοση vs. Υπολογιστικό Overhead**: Η μέθοδος βασισμένη σε PSO ξεπερνά σταθερά τη μέθοδο Neighbor-aware σε πολλές μετρικές, αλλά απαιτεί σημαντικά περισσότερους υπολογιστικούς πόρους για την ίδια τη διαδικασία κατανομής.

2. **Προσδιορισμός vs. Ποιότητα Βελτιστοποίησης**: Η προσδιοριστική φύση της μεθόδου Neighbor-aware παρέχει προβλέψιμα μοτίβα κατανομής, ενώ η στοχαστική προσέγγιση PSO ενδεχομένως βρίσκει καλύτερες λύσεις με κόστος κάποιας αλλαγής.

3. **Ανταπόκριση vs. Ποιότητα Κατανομής**: Το χαμηλότερο υπολογιστικό overhead της μεθόδου Neighbor-aware επιτρέπει ταχύτερη απόκριση σε δυναμικές αλλαγές στο περιβάλλον, ενώ η μέθοδος βασισμένη σε PSO παρέχει καλύτερη ποιότητα κατανομής για πιο σταθερά σενάρια.

4. **Απλότητα vs. Ευελιξία**: Η μέθοδος Neighbor-aware προσφέρει απλούστερη υλοποίηση με λιγότερες παραμέτρους, ενώ η μέθοδος βασισμένη σε PSO παρέχει μεγαλύτερη ευελιξία μέσω ρύθμισης παραμέτρων για προσαρμογή για διαφορετικές βελτιστοποιήσεις.

5. **Σταδιακή vs. Ομαδική Κατανομή**: Η διαδοχική φύση της μεθόδου Neighbor-aware επιτρέπει σταδιακή κατανομή καθώς φτάνουν οι εφαρμογές, ενώ η μέθοδος βασισμένη σε PSO είναι καταλληλότερη για ομαδική κατανομή γνωστών συνόλων εφαρμογών.

Αυτοί οι συμβιβασμοί υποδηλώνουν ότι η επιλογή μεταξύ μεθόδων κατανομής θα πρέπει να καθοδηγείται από συγκεκριμένες απαιτήσεις και περιορισμούς ανάπτυξης. Η μέθοδος Neighbor-aware μπορεί να είναι προτιμότερη για εξαιρετικά δυναμικά περιβάλλοντα με αυστηρούς χρονικούς περιορισμούς, ενώ η μέθοδος βασισμένη σε PSO προσφέρει ανώτερη απόδοση για σενάρια όπου η ποιότητα κατανομής είναι η κύρια ανησυχία και το υπολογιστικό overhead για τη διαδικασία κατανομής είναι αποδεκτό.

7 Περίπτωση Πραγματικού Κόσμου

7.1 Ανάπτυξη Fog Computing σε Έξυπνη Πόλη

7.1.1 Περιγραφή Υποθετικού Σεναρίου

Για την καλύτερη κατανόηση της πρακτικής εφαρμογής των προτεινόμενων μεθόδων κατανομής φόρτου εργασίας (workload allocation), παρουσιάζουμε ένα υποθετικό σενάριο που επικεντρώνεται στην ανάπτυξη ενός fog computing δικτύου σε έξυπνη πόλη.

Οι έξυπνες πόλεις αποτελούν ιδανικό παράδειγμα εφαρμογής του fog computing, καθώς χαρακτηρίζονται από ετερογενείς συσκευές, ποικίλες απαιτήσεις εφαρμογών και γεωγραφικά κατανεμημένες ανάγκες επεξεργασίας.

Σε αυτό το σενάριο, εξετάζουμε μια μεσαίου μεγέθους αστική περιοχή με τα ακόλουθα συστήματα έξυπνης πόλης:

1. **Σύστημα Διαχείρισης Κυκλοφορίας**: Δίκτυο καμερών και αισθητήρων σε διασταυρώσεις για βελτιστοποίηση κυκλοφοριακής ροής και ανίχνευση συμβάντων.
2. **Δίκτυο Δημόσιας Ασφάλειας**: Κάμερες επιτήρησης και περιβαλλοντικοί αισθητήρες για παρακολούθηση δημόσιων χώρων.
3. **Περιβαλλοντική Παρακολούθηση**: Αισθητήρες ποιότητας αέρα, θορύβου και θερμοκρασίας σε διάφορα σημεία της πόλης.

4. **Υποδομή Έξυπνου Δικτύου**: Συστήματα παρακολούθησης και διαχείρισης ενέργειας.
5. **Σύστημα Δημόσιων Μεταφορών**: Οχήματα με GPS για πληροφορίες μετακίνησης σε πραγματικό χρόνο.
6. **Εφαρμογές Υπηρεσιών Πολιτών**: Εφαρμογές για πρόσβαση σε υπηρεσίες και πληροφορίες της πόλης.

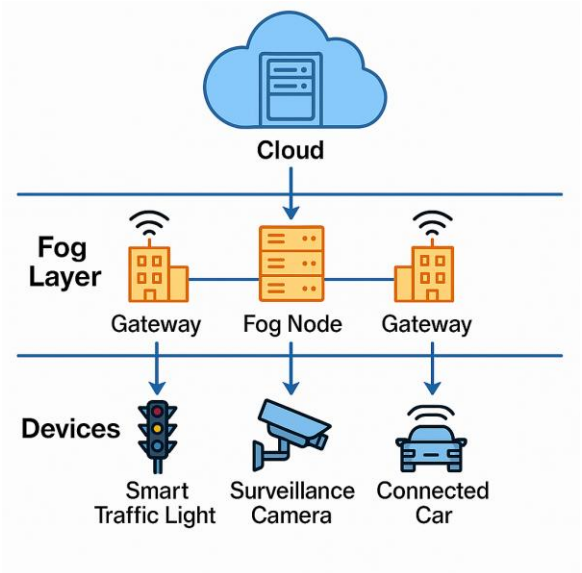
Για το θεωρητικό μας σενάριο, υιοθετούμε την αρχιτεκτονική fog τριών επιπέδων που προτάθηκε από τους Bonomi et al. [1]:

Επίπεδο 1 (Συσκευές Άκρου): Αισθητήρες, κάμερες και actuators (ενεργοποιητές) με περιορισμένες υπολογιστικές δυνατότητες.

Επίπεδο 2 (Κόμβοι Fog): Υπολογιστικοί πόροι κατανεμημένοι στην πόλη σε τρεις διαμορφώσεις:

- **Κόμβοι Micro Fog:** Μικρής κλίμακας υπολογιστικές συσκευές σε σημεία συγκέντρωσης αισθητήρων
- **Κόμβοι District Fog:** Διακομιστές μεσαίας χωρητικότητας σε δημόσια κτίρια
- **Κόμβοι Gateway Fog:** Υπολογιστικοί πόροι υψηλότερης χωρητικότητας ως πύλες προς το cloud

Επίπεδο 3 (Πόροι Cloud): Μεγάλα κέντρα δεδομένων cloud για σύνθετη ανάλυση και μακροχρόνια αποθήκευση.



Εικόνα 7.1: Fog computing Αρχιτεκτονική, τριών επιπέδων για την ανάπτυξη έξυπνης πόλης

7.1.2 Τυπικά Χαρακτηριστικά Workload

Στο θεωρητικό μας σενάριο, οι εφαρμογές έξυπνης πόλης δημιουργούν τύπους φόρτου εργασίας με διαφορετικά χαρακτηριστικά:

Workload Επεξεργασίας Πραγματικού Χρόνου:

- Παράδειγμα: Ανίχνευση συμβάντων κυκλοφορίας
- Χαρακτηριστικά: Υψηλή ευαισθησία καθυστέρησης (<100ms), υψηλές απαιτήσεις bandwidth

Workload Επεξεργασίας Ροής:

- Παράδειγμα: Ανάλυση ροής κυκλοφορίας
- Χαρακτηριστικά: Μέτρια ευαισθησία καθυστέρησης, συνεχείς ροές δεδομένων

Workload Ομαδικής Επεξεργασίας:

- Παράδειγμα: Βελτιστοποίηση ενεργειακής κατανάλωσης
- Χαρακτηριστικά: Χαμηλή ευαισθησία καθυστέρησης, υψηλές απαιτήσεις επεξεργασίας

Workload Διαδραστικών Υπηρεσιών:

- Παράδειγμα: Εφαρμογές πληροφόρησης πολιτών
- Χαρακτηριστικά: Μέτρια ευαισθησία καθυστέρησης, χαμηλές απαιτήσεις πόρων

7.2 Εφαρμογή Προτεινόμενων Μεθόδων στο Θεωρητικό Σενάριο

7.2.1 Εφαρμογή Μεθόδου Neighbor-Aware

Στο υποθετικό σενάριο της έξυπνης πόλης, η μέθοδος κατανομής Neighbor-Aware θα μπορούσε να εφαρμοστεί ως εξής:

1. **Ανάθεση Εφαρμογών Διαχείρισης Κυκλοφορίας**:
    - Οι εφαρμογές ανίχνευσης έκτακτων συμβάντων θα αναλύονταν πρώτα σε τοπικούς micro fog κόμβους
    - Σε περίπτωση ανεπάρκειας πόρων, η επεξεργασία θα προωθούνταν σε γειτονικούς district fog κόμβους
    - Η μακροπρόθεσμη ανάλυση δεδομένων κυκλοφορίας θα κατευθυνόταν στο cloud
2. **Παράδειγμα Εξισορρόπησης Φορτίου**:
    - Κατά τις ώρες αιχμής, η αυξημένη ζήτηση σε συγκεκριμένες διασταυρώσεις θα οδηγούσε σε ανακατανομή σε γειτονικούς κόμβους
    - Η μέθοδος θα επέλεγε αυτόματα τους λιγότερο φορτωμένους κόμβους για νέες εργασίες
3. **Σενάριο Έκτακτης Ανάγκης**:
    - Σε περίπτωση έκτακτης ανάγκης (π.χ. πυρκαγιά), η προτεραιότητα θα δινόταν στις εφαρμογές ασφαλείας
    - Οι λιγότερο κρίσιμες εφαρμογές θα αναδιανέμονταν σε περιφερειακούς κόμβους ή στο cloud

Αυτή η προσέγγιση αξιοποιεί την τοπική γνώση και την ταχύτητα της μεθόδου Neighbor-Aware για γρήγορη ανταπόκριση σε μεταβαλλόμενες συνθήκες.

7.2.2 Εφαρμογή Μεθόδου PSO

Η μέθοδος βασισμένη σε PSO θα μπορούσε να εφαρμοστεί στο ίδιο σενάριο ως εξής:

1. **Βελτιστοποίηση Συνολικού Συστήματος**:
    - Η μέθοδος PSO θα δημιουργούσε ένα συνολικό πλάνο κατανομής για όλες τις εφαρμογές συνολικά
    - Θα εξισορροπούσε πολλαπλούς στόχους όπως καθυστέρηση, κατανάλωση ενέργειας και κόστος
2. **Παράδειγμα Κατανομής Βάσει Προτεραιοτήτων**:
    - Οι εφαρμογές πραγματικού χρόνου (π.χ. παρακολούθηση ασφάλειας) θα τοποθετούνταν σε κόμβους micro και district fog
    - Οι εφαρμογές ομαδικής επεξεργασίας (π.χ. ενεργειακή ανάλυση) θα κατανέμονταν βέλτιστα μεταξύ district fog και cloud
    - Η συνάρτηση καταλληλότητας θα απέτρεπε την υπερβολική χρήση του cloud για εφαρμογές ευαίσθητες στην καθυστέρηση
3. **Σενάριο Προγραμματισμένης Συντήρησης**:
    - Κατά τη διάρκεια προγραμματισμένης συντήρησης μέρους του δικτύου, ο αλγόριθμος PSO θα αναδιέτασσε τις εφαρμογές για βέλτιστη χρήση των εναπομεινάντων πόρων

Αυτή η προσέγγιση αξιοποιεί την ικανότητα καθολικής βελτιστοποίησης της μεθόδου PSO για εξισορρόπηση πολλαπλών παραγόντων και βέλτιστη χρήση πόρων.

7.3 Θεωρητική Ανάλυση Πλεονεκτημάτων και Μειονεκτημάτων

7.3.1 Συγκριτικά Πλεονεκτήματα

Βάσει του θεωρητικού σεναρίου, μπορούμε να προσδιορίσουμε τα ακόλουθα συγκριτικά πλεονεκτήματα των δύο μεθόδων:

Μέθοδος Neighbor-Aware:

- Ταχύτερη ανταπόκριση σε δυναμικές αλλαγές (π.χ. αιφνίδιες αυξήσεις κυκλοφορίας)
- Χαμηλότερες απαιτήσεις υπολογιστικών πόρων για τη διαδικασία λήψης αποφάσεων
- Αποκεντρωμένη λήψη αποφάσεων που μειώνει τα σημεία αποτυχίας
- Ευκολότερη υλοποίηση και συντήρηση

Μέθοδος βασισμένη σε PSO:

- Καλύτερη κατανομή εφαρμογών ευαίσθητων στην καθυστέρηση
- Αποδοτικότερη χρήση πόρων με ισορροπημένη κατανομή φορτίου

- Χαμηλότερη συνολική κατανάλωση ενέργειας
- Βελτιστοποίηση πολλαπλών στόχων ταυτόχρονα

7.3.2 Θεωρητικά Σενάρια Εφαρμογής

Με βάση τα πλεονεκτήματα κάθε μεθόδου, μπορούμε να προτείνουμε τα ακόλουθα θεωρητικά σενάρια εφαρμογής:

Ιδανικά Σενάρια για τη Μέθοδο Neighbor-Aware:

- Διαχείριση έκτακτων συμβάντων (π.χ. πυρκαγιά, ατύχημα) όπου απαιτείται ταχεία απόκριση
- Δίκτυα με συχνές αλλαγές τοπολογίας λόγω προσθήκης/αφαίρεσης κόμβων
- Περιπτώσεις όπου οι υπολογιστικοί πόροι για τη διαδικασία κατανομής είναι περιορισμένοι

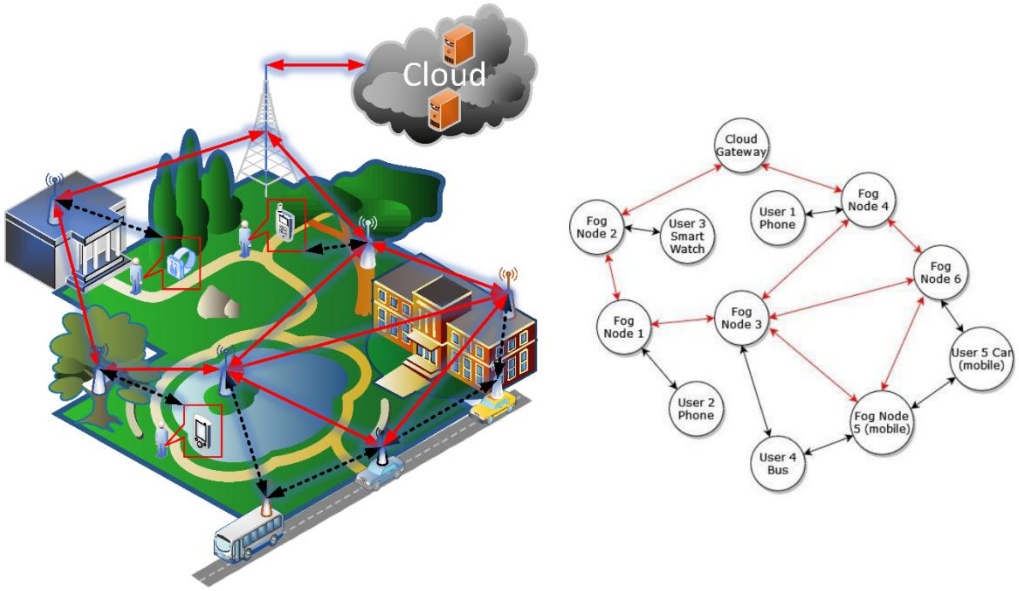
Ιδανικά Σενάρια για τη Μέθοδο PSO:

- Προγραμματισμένες υπηρεσίες με γνωστές απαιτήσεις (π.χ. περιβαλλοντική παρακολούθηση)
- Περιβάλλοντα όπου η ενεργειακή αποδοτικότητα αποτελεί κρίσιμη παράμετρο
- Πολύπλοκα δίκτυα με ετερογενείς πόρους και απαιτήσεις εφαρμογών

7.3.3 Υβριδική Προσέγγιση

Με βάση το θεωρητικό σενάριο, μια υβριδική προσέγγιση θα μπορούσε να συνδυάσει τα πλεονεκτήματα και των δύο μεθόδων:

- **Στρατηγική περιοδικής βελτιστοποίησης**: Χρήση της μεθόδου PSO για περιοδική (π.χ. ωριαία) βελτιστοποίηση της συνολικής κατανομής, και της μεθόδου Neighbor-Aware για διαχείριση δυναμικών αλλαγών μεταξύ των κύκλων βελτιστοποίησης.
- **Διαίρεση δικτύου**: Διαίρεση του δικτύου της έξυπνης πόλης σε ζώνες, με χρήση της μεθόδου PSO για διαχείριση κρίσιμων ζωνών (π.χ. κέντρο πόλης) και της μεθόδου Neighbor-Aware για λιγότερο κρίσιμες περιοχές.
- **Διαφοροποίηση ανά τύπο Εφαρμογής**: Χρήση της μεθόδου PSO για εφαρμογές με υψηλές απαιτήσεις ποιότητας υπηρεσίας και της μεθόδου Neighbor-Aware για λιγότερο απαιτητικές εφαρμογές.



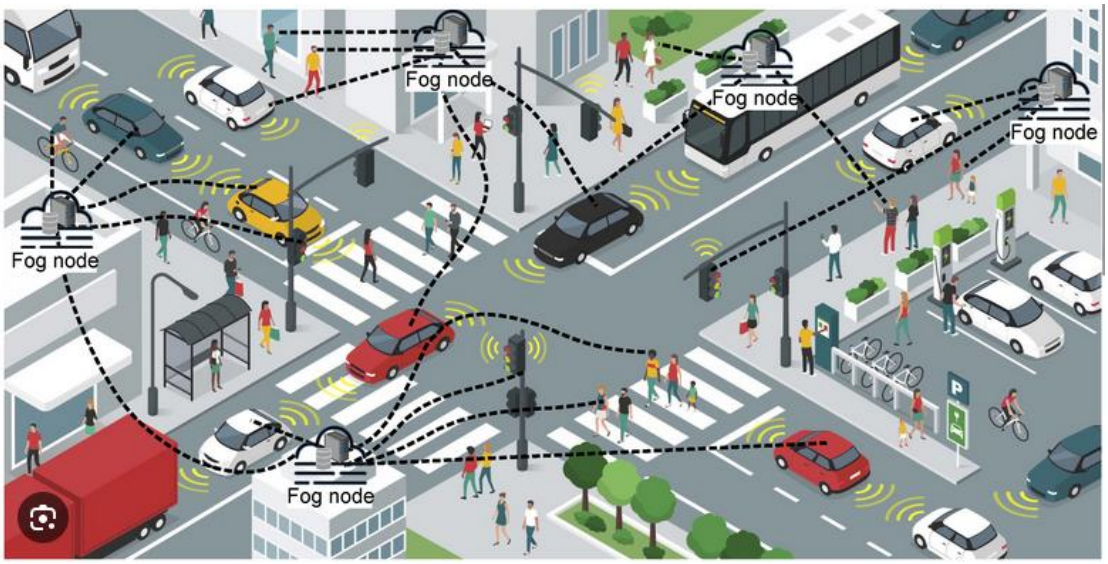
Εικόνα 7.2: Αναπαράσταση Fog computing σε έξυπνη πόλης

7.4 Θεωρητικές Προτάσεις Εφαρμογής

Από την ανάλυση του θεωρητικού σεναρίου, προκύπτουν οι ακόλουθες προτάσεις για την εφαρμογή των μεθόδων κατανομής σε περιβάλλοντα έξυπνης πόλης:

1. **Σχεδιασμός Ιεραρχίας Κατανομής**:
    - Χρήση μεθόδου PSO για βελτιστοποίηση σε επίπεδο συνοικίας πόλης
    - Εφαρμογή μεθόδου Neighbor-Aware για τοπική βελτιστοποίηση εντός συνοικιών
2. **Διαχείριση Διαφορετικών Χρονικών Ορίζοντων**:
    - Μέθοδος PSO για μακροπρόθεσμη κατανομή φόρτου (ημερήσια/εβδομαδιαία)
    - Μέθοδος Neighbor-Aware για άμεση απόκριση σε μεταβαλλόμενες συνθήκες
3. **Βελτιστοποίηση Υποδομής**:
    - Προτεραιότητα στην ανάπτυξη κόμβων district fog (επίπεδο 2) που παρουσιάζουν την καλύτερη αναλογία απόδοσης-κόστους
    - Στρατηγική τοποθέτηση κόμβων με βάση τα πρότυπα κυκλοφορίας στην πόλη
4. **Διαχείριση Ενέργειας**:
    - Ενεργοποίηση/απενεργοποίηση κόμβων βάσει προβλέψεων φόρτου
    - Χρήση της μεθόδου PSO για τον προσδιορισμό βέλτιστων προτύπων ενεργειακής διαχείρισης
5. **Ποιότητα Υπηρεσιών και Προτεραιότητες**:
    - Καθορισμός διαφορετικών επιπέδων προτεραιότητας για τύπους εφαρμογών
    - Προσαρμογή παραμέτρων της συνάρτησης καταλληλότητας PSO για την αντανάκλαση αυτών των προτεραιοτήτων

7.5 Συμπεράσματα Μελέτης Περίπτωσης



Η θεωρητική μελέτη περίπτωσης έξυπνης πόλης αναδεικνύει την πρακτική εφαρμογή των προτεινόμενων μεθόδων κατανομής φόρτου εργασίας σε ρεαλιστικά σενάρια. Συνοπτικά, τα βασικά συμπεράσματα είναι:

1. **Συνεργασία μεθόδων**: Η μέθοδος Neighbor-Aware προσφέρει ταχύτητα για δυναμικές καταστάσεις, ενώ η μέθοδος PSO παρέχει βελτιστοποιημένη κατανομή με βάση πολλαπλά κριτήρια.
2. **Ιεραρχική διαχείριση**: Η αρχιτεκτονική τριών επιπέδων επιτρέπει την κατανομή εφαρμογών στο καταλληλότερο επίπεδο ανάλογα με τις απαιτήσεις τους.
3. **Υβριδική προσέγγιση**: Ο συνδυασμός περιοδικών βελτιστοποιήσεων (PSO) με συνδυασμό (Neighbor-Aware) φαίνεται βέλτιστος για τις μεταβαλλόμενες συνθήκες έξυπνης πόλης.
4. **Στρατηγική σημασία ενδιάμεσου επιπέδου**: Οι κόμβοι district fog προσφέρουν την καλύτερη αναλογία απόδοσης-κόστους, αποτελώντας ιδανική επιλογή για πολλές εφαρμογές.

Η μελέτη αυτή παρέχει ένα θεωρητικό πλαίσιο για την αποτελεσματική εφαρμογή των προτεινόμενων μεθόδων σε περιβάλλοντα έξυπνης πόλης, εξισορροπώντας απόδοση, κόστος και ενεργειακή αποδοτικότητα.

8 Μελλοντική Εργασία

8.1 Περίληψη Συνεισφορών

Η παρούσα διατριβή παρουσίασε μια ολοκληρωμένη έρευνα των στρατηγικών κατανομής φόρτου εργασίας σε περιβάλλοντα fog computing, με ιδιαίτερη έμφαση στις προσεγγίσεις Neighbor-aware και Particle Swarm Optimization (PSO). Η έρευνα συνεισφέρει σημαντικά στον τομέα:

1. **Ανάπτυξης του Πλαισίου Προσομοίωσης FogSim-NX:** Μια ευέλικτη και επεκτάσιμη πλατφόρμα προσομοίωσης για περιβάλλοντα fog computing που επιτρέπει τη ρεαλιστική μοντελοποίηση ετερογενών κόμβων, διαφορετικών απαιτήσεων εφαρμογών και διαφορετικών στρατηγικών κατανομής, επεκτείνοντας τις δυνατότητες υπαρχόντων προσομοιωτών όπως το iFogSim [3].

2. **Σχεδιασμό και Υλοποίηση Συμπληρωματικών Μεθόδων Κατανομής:** Έχουν αναπτυχθεί και υλοποιηθεί δύο διακριτές στρατηγικές κατανομής φόρτου εργασίας:

α. Η μέθοδος κατανομής **Neighbor-aware**, η οποία χρησιμοποιεί τοπική γνώση και αποφάσεις βασισμένες στην εγγύτητα για την αποτελεσματική κατανομή φόρτων εργασίας με ελάχιστη υπολογιστική επιβάρυνση, αξιοποιώντας τις έννοιες από αρχιτεκτονικές αποικιών fog [6].

β. Η μέθοδος κατανομής βασισμένη σε **PSO** αλγόριθμο, η οποία χρησιμοποιεί βιο-εμπνευσμένη βελτιστοποίηση για την εύρεση σχεδόν βέλτιστων λύσεων κατανομής εξισορροπώντας πολλαπλούς ανταγωνιστικούς στόχους, επεκτείνοντας την προσέγγιση PSO που εισήγαγαν οι Kennedy και Eberhart [4].

3. **Συγκριτική Ανάλυση Απόδοσης:** Μια λεπτομερής αξιολόγηση της απόδοσης και των δύο μεθόδων κατανομής σε διαφορετικά μεγέθη δικτύου και πλήθος εφαρμογών, παρέχοντας πληροφορίες σχετικά με τη συμπεριφορά κλιμάκωσής τους, τα πρότυπα χρήσης πόρων και τα χαρακτηριστικά αποδοτικότητάς τους.

4. **Εφαρμογή Μελέτης Περίπτωσης Πραγματικού Κόσμου:** Η επίδειξη της πρακτικής συνάφειας και δυνατότητας εφαρμογής των προτεινόμενων μεθόδων κατανομής μέσω μιας λεπτομερούς μελέτης περίπτωσης fog computing έξυπνης πόλης, ακολουθώντας το μοντέλο αρχιτεκτονικής τριών επιπέδων που πρότειναν οι [1].

Αυτές οι συνεισφορές γενικά βοηθούν την κατανόηση της κατανομής φόρτου εργασίας σε περιβάλλοντα fog computing και παρέχουν πρακτικά εργαλεία και μεθόδους για την αντιμετώπιση των προκλήσεων της διαχείρισης πόρων σε ετερογενείς, κατανεμημένες υπολογιστικές υποδομές.

8.2 Βασικά Ευρήματα

Η έρευνα που παρουσιάστηκε σε αυτή την εργασία έχει αποδώσει αρκετά σημαντικά ευρήματα που παρέχουν πληροφορίες σχετικά με την κατανομή φόρτου εργασίας στο fog computing:

8.2.1 Χαρακτηριστικά Απόδοσης

1. **Επιτυχία Κατανομής και Κλιμάκωση:** Και οι δύο μέθοδοι επιδεικνύουν σχετικά υψηλά ποσοστά επιτυχίας κατανομής, αλλά η μέθοδος που βασίζεται σε PSO ξεπερνά σταθερά τη μέθοδο Neighbor-aware, ιδιαίτερα σε σενάρια με περιορισμένους πόρους. Αυτό ευθυγραμμίζεται με τα ευρήματα των Zhou et al. [7] σχετικά με τα πλεονεκτήματα των προσεγγίσεων βασισμένων σε βελτιστοποίηση για σύνθετα προβλήματα κατανομής.

2. **Πρότυπα Χρήσης Cloud:** Η μέθοδος που βασίζεται σε PSO παρουσιάζει σημαντικά χαμηλότερη εξάρτηση από το cloud σε όλα τα σενάρια, προτιμώντας να χρησιμοποιεί τους πόρους fog πιο αποτελεσματικά πριν καταφύγει σε μεταφορά στο cloud. Αυτή η συμπεριφορά οδηγεί σε χαμηλότερη καθυστέρηση και λειτουργικό κόστος, παρόμοια με τις βελτιστοποιημένες προσεγγίσεις κατανομής fog-cloud που περιγράφονται από τους [2].

3. **Ενεργειακή και Οικονομική Αποδοτικότητα:** Η μέθοδος που βασίζεται σε PSO επιδεικνύει 8-25% χαμηλότερη κατανάλωση ενέργειας και 12-31% χαμηλότερο λειτουργικό κόστος σε διάφορα σενάρια. Αυτά τα οφέλη αποδοτικότητας συμφωνούν με τα οφέλη βελτιστοποίησης που αναφέρθηκαν από τους Mahmud et al. [5] στο πλαίσιο διαχείρισης εφαρμογών τους.

8.2.2 Αλγοριθμικές Γνώσεις

1. **Συμβιβασμοί Υπολογιστικής Πολυπλοκότητας:** Η ανώτερη ποιότητα κατανομής της μεθόδου που βασίζεται σε PSO έρχεται με το κόστος σημαντικά υψηλότερης υπολογιστικής πολυπλοκότητας. Αυτός ο συμβιβασμός καθιστά τη μέθοδο Neighbor-aware πιο κατάλληλη για σενάρια πραγματικού χρόνου ή εξαιρετικά δυναμικά, ενώ η μέθοδος που βασίζεται σε PSO είναι καλύτερη για σχεδιασμό και εφαρμογές επικεντρωμένες στη βελτιστοποίηση.

2. **Καθολική vs Τοπική Βελτιστοποίηση:** Η προσέγγιση καθολικής βελτιστοποίησης της μεθόδου PSO παρέχει σημαντικά πλεονεκτήματα έναντι των τοπικών, greedy (άπληστων) αποφάσεων της μεθόδου Neighbor-aware, ιδιαίτερα σε πολύπλοκα σενάρια με πολλαπλούς ανταγωνιστικούς στόχους, σύμφωνα με τα ευρήματα των Kennedy και Eberhart [4] σχετικά με την αποτελεσματικότητα της νοημοσύνης σμήνους για σύνθετα προβλήματα βελτιστοποίησης.

3. **Ευαισθησία Παραμέτρων και Ρύθμιση**: Η απόδοση της μεθόδου που βασίζεται σε PSO έχει μέτρια ευαισθησία στις ρυθμίσεις παραμέτρων, ιδιαίτερα στον κοινωνικό συντελεστή (c2) και στα βάρη της συνάρτησης καταλληλότητας. Απαιτείται προσεκτική ρύθμιση για την επίτευξη βέλτιστων αποτελεσμάτων για συγκεκριμένα σενάρια ανάπτυξης.

8.2.3 Πρακτικές Γνώσεις Εφαρμογής

1. **Καταλληλότητα Σεναρίου Ανάπτυξης**: Η έρευνα υποδεικνύει ότι η μέθοδος Neighbor-aware είναι καταλληλότερη για εξαιρετικά δυναμικά περιβάλλοντα με συχνές αλλαγές φόρτου εργασίας και αυστηρούς χρονικούς περιορισμούς, ενώ η μέθοδος που βασίζεται σε PSO προσφέρει ανώτερη απόδοση για πιο σταθερά σενάρια όπου η ποιότητα κατανομής είναι η κύρια ανησυχία.

2. **Οικονομικές και Περιβαλλοντικές Επιπτώσεις**: Οι σημαντικές μειώσεις στην κατανάλωση ενέργειας και το λειτουργικό κόστος που επιτυγχάνονται από τη μέθοδο που βασίζεται σε PSO

καταδεικνύουν τα οικονομικά και περιβαλλοντικά οφέλη της έξυπνης κατανομής φόρτου εργασίας σε περιβάλλοντα fog computing.

3. **Ειδικές για Εφαρμογές Παράμετροι**: Το χάσμα απόδοσης μεταξύ των μεθόδων κατανομής ποικίλλει μεταξύ των τύπων εφαρμογών, με τη μέθοδο που βασίζεται σε PSO να παρουσιάζει το πιο σημαντικό πλεονέκτημα για εφαρμογές ευαίσθητες στην καθυστέρηση. Αυτό υποδηλώνει ότι οι υβριδικές στρατηγικές κατανομής μπορεί να είναι ιδανικές σε περιβάλλοντα με μικτό Workload.

Αυτά τα ευρήματα παρέχουν πολύτιμες πληροφορίες για ερευνητές και επαγγελματίες που εργάζονται στη διαχείριση πόρων σε περιβάλλοντα fog computing και edge computing, προσφέροντας τόσο θεωρητική κατανόηση όσο και πρακτική καθοδήγηση για αναπτύξεις πραγματικού κόσμου.

8.3 Μειονεκτήματα

Ενώ αυτή η έρευνα παρέχει σημαντικές συνεισφορές στον τομέα της κατανομής φόρτου εργασίας στο fog computing, θα πρέπει να αναγνωριστούν αρκετοί περιορισμοί:

1. **Αξιολόγηση Βασισμένη σε Προσομοίωση**: Η αξιολόγηση βασίζεται σε προσομοίωση αντί για ανάπτυξη σε φυσική υποδομή fog, γεγονός που μπορεί να εισάγει κάποιες αποκλίσεις μεταξύ της προσομοιωμένης και της πραγματικής απόδοσης. Αν και το μοντέλο προσομοίωσης ενσωματώνει ρεαλιστικές παραμέτρους και περιορισμούς, ορισμένες πρακτικές προκλήσεις ενδέχεται να μην αποτυπώνονται πλήρως.

2. **Δυναμικότητα Δικτύου**: Η τρέχουσα υλοποίηση υποθέτει μια στατική τοπολογία δικτύου κατά τη διάρκεια της διαδικασίας κατανομής. Σε πραγματικά περιβάλλοντα fog, η διαθεσιμότητα των κόμβων και οι συνθήκες δικτύου μπορεί να αλλάζουν δυναμικά, απαιτώντας μηχανισμούς προσαρμογής που δεν αντιμετωπίζονται πλήρως σε αυτή την εργασία.

3. **Χαρακτηρισμός Φόρτου Εργασίας**: Ενώ η προσομοίωση δημιουργεί διαφορετικές απαιτήσεις εφαρμογών, τα μοντέλα φόρτου εργασίας είναι απλουστευμένες αναπαραστάσεις των εφαρμογών πραγματικού κόσμου. Πιο πολύπλοκα χαρακτηριστικά φόρτου εργασίας, όπως εξαρτήσεις δεδομένων και μεταβαλλόμενες φάσεις εκτέλεσης, θα μπορούσαν να επηρεάσουν την απόδοση κατανομής.

4. **Περιορισμοί Τοπολογίας**: Η χρήση τυχαίων γράφων Erdős-Rényi για τη δημιουργία τοπολογίας, ενώ παρέχει καλές γενικές ιδιότητες, μπορεί να μην αποτυπώνει πλήρως τους ιεραρχικούς και γεωγραφικούς περιορισμούς των αναπτύξεων fog πραγματικού κόσμου. Πιο δομημένα μοντέλα τοπολογίας μπορεί να αποδώσουν διαφορετικά χαρακτηριστικά απόδοσης.

5. **Όρια Κλιμάκωσης**: Αν και η αξιολόγηση εξέτασε σενάρια με έως και 400 κόμβους και 20.000 εφαρμογές, τα fog εξαιρετικά μεγάλης κλίμακας μπορεί να παρουσιάσουν πρόσθετες προκλήσεις που δεν αντιμετωπίζονται σε αυτή την εργασία, ιδιαίτερα για την υπολογιστικά απαιτητική μέθοδο που βασίζεται σε PSO.

6. **Στατικές Ρυθμίσεις Παραμέτρων**: Και οι δύο μέθοδοι κατανομής χρησιμοποιούν σταθερές ρυθμίσεις παραμέτρων σε όλα τα σενάρια. Η προσαρμοστική ρύθμιση παραμέτρων με βάση τα χαρακτηριστικά του φόρτου εργασίας και τις συνθήκες του δικτύου θα μπορούσε ενδεχομένως να βελτιώσει την απόδοση σε διαφορετικά σενάρια.

Αυτοί οι περιορισμοί παρουσιάζουν ευκαιρίες για μελλοντική έρευνα ώστε να ενισχυθούν και να επεκταθούν οι μέθοδοι κατανομής φόρτου εργασίας που προτείνονται σε αυτή την εργασία.

8.4 Κατευθύνσεις Μελλοντικής Έρευνας

Με βάση τις συνεισφορές και την αντιμετώπιση των περιορισμών αυτής της εργασίας, μπορούν να προσδιοριστούν αρκετές υποσχόμενες κατευθύνσεις για μελλοντική έρευνα:

8.4.1 Βελτιωμένοι Αλγόριθμοι Κατανομής

1. **Υβριδικές Προσεγγίσεις Κατανομής:** Ανάπτυξη υβριδικών μεθόδων που συνδυάζουν την υπολογιστική αποδοτικότητα της προσέγγισης Neighbor-aware με την ποιότητα βελτιστοποίησης της μεθόδου που βασίζεται σε PSO, πιθανώς χρησιμοποιώντας την πρώτη για αρχική κατανομή και τη δεύτερη για περιοδική βελτιστοποίηση.

2. **Προσαρμοστική Ρύθμιση Παραμέτρων:** Υλοποίηση μηχανισμών για δυναμική προσαρμογή των παραμέτρων PSO (βάρος αδράνειας, γνωστικοί και κοινωνικοί συντελεστές) και των βαρών της συνάρτησης καταλληλότητας με βάση τα χαρακτηριστικά του φόρτου εργασίας και την ανατροφοδότηση απόδοσης, αξιοποιώντας την ανάλυση παραμετρικής ευαισθησίας που πραγματοποιήθηκε από τους Kennedy και Eberhart [4].

3. **Παραλλαγές PSO Πολλαπλών Σμηνών:** Διερεύνηση παραλλαγών PSO πολλαπλών σμηνών που διαιρούν τον χώρο αναζήτησης ή τους στόχους βελτιστοποίησης μεταξύ πολλαπλών συνεργαζόμενων σμηνών, βελτιώνοντας ενδεχομένως τόσο την ποιότητα της λύσης όσο και την υπολογιστική αποδοτικότητα, παρόμοια με τη συνεργατική προσέγγιση που προτείνεται από τους Zhou et al. [7].

4. **Εναλλακτικοί Βιο-εμπνευσμένοι Αλγόριθμοι**: Διερεύνηση άλλων βιο-εμπνευσμένων τεχνικών βελτιστοποίησης, όπως βελτιστοποίηση αποικίας μυρμηγκιών, γενετικοί αλγόριθμοι ή αποικία τεχνητών μελισσών, για κατανομή φόρτου εργασίας σε περιβάλλοντα fog και σύγκριση της απόδοσής τους με την προσέγγιση που βασίζεται σε PSO.

8.4.2 Βελτιώσεις Συστήματος

1. **Δυναμική Προσαρμογή Δικτύου:** Επέκταση των μεθόδων κατανομής για τη διαχείριση δυναμικών αλλαγών στην τοπολογία του δικτύου και τη διαθεσιμότητα των κόμβων, συμπεριλαμβανομένων μηχανισμών για μετεγκατάσταση φόρτου εργασίας και ανακατανομή ως απόκριση σε περιβαλλοντικές αλλαγές, αντιμετωπίζοντας τις προκλήσεις mobility που προσδιορίστηκαν από τους [1].

2. **Μηχανισμοί Ανοχής Σφαλμάτων**: Ανάπτυξη ολοκληρωμένων στρατηγικών ανοχής σφαλμάτων, συμπεριλαμβανομένης της προληπτικής αναπαραγωγής για κρίσιμες εφαρμογές, μηχανισμών αντιδραστικής ανάκτησης και προσεγγίσεων σταδιακής υποβάθμισης για σενάρια με περιορισμένους πόρους.

3. **Διαχείριση Πόρων**: Επέκταση του πλαισίου κατανομής για τη διαχείριση σεναρίων με διαφορετικές συμφωνίες επιπέδου υπηρεσιών και πολιτικές κατανομής πόρων για διαφορετικούς χρήστες ή κατηγορίες εφαρμογών, αξιοποιώντας τις παραμέτρους ποιότητας υπηρεσιών που διερευνήθηκαν [6].

8.5 Παρατηρήσεις

Αυτή η διατριβή διερεύνησε την κατανομή φόρτου εργασίας σε περιβάλλοντα fog computing, αναπτύσσοντας και αξιολογώντας δύο συμπληρωματικές προσεγγίσεις: μια μέθοδο Neighbor-aware βασισμένη σε τοπική γνώση και αποφάσεις με βάση την εγγύτητα, και μια μέθοδο βασισμένη σε PSO που χρησιμοποιεί βιο-εμπνευσμένη καθολική βελτιστοποίηση. Μέσα από ολοκληρωμένη προσομοίωση και ανάλυση μελέτης περίπτωσης, η έρευνα έχει δείξει ότι η έξυπνη κατανομή φόρτου εργασίας μπορεί να βελτιώσει σημαντικά την απόδοση, τη χρήση πόρων, την ενεργειακή αποδοτικότητα και το λειτουργικό κόστος σε αναπτύξεις fog computing.

Η συγκριτική ανάλυση αποκαλύπτει ότι ενώ η μέθοδος που βασίζεται σε PSO γενικά επιτυγχάνει ανώτερη ποιότητα κατανομής σε πολλαπλές μετρήσεις, η μέθοδος Neighbor-aware προσφέρει πολύτιμα πλεονεκτήματα αποδοτικότητας και απλότητας που την καθιστούν κατάλληλη για συγκεκριμένα σενάρια ανάπτυξης.

Το πρόγραμμα προσομοίωσης FogSim-NX που αναπτύχθηκε ως μέρος αυτής της έρευνας παρέχει ένα πολύτιμο εργαλείο για περαιτέρω διερεύνηση των στρατηγικών κατανομής και των σεναρίων fog computing.

Καθώς το fog computing συνεχίζει να εξελίσσεται ως κρίσιμο παράδειγμα για την αντιμετώπιση των υπολογιστικών αναγκών των συσκευών άκρου και των εφαρμογών IoT, η αποδοτική κατανομή φόρτου εργασίας θα παραμείνει μια ουσιαστική πρόκληση. Αυτή η διατριβή συμβάλλει στην αντιμετώπιση αυτής της πρόκλησης παρέχοντας γνώσεις, εργαλεία και μεθόδους που μπορούν να βοηθήσουν τους ερευνητές και τους επαγγελματίες να αναπτύξουν πιο αποτελεσματικά συστήματα fog computing.

Το μέλλον του fog computing πιθανότατα θα περιλαμβάνει όλο και πιο ετερογενή και δυναμικά περιβάλλοντα, με διαφορετικές εφαρμογές που απαιτούν αποδοτική, προσαρμοστική και έξυπνη διαχείριση πόρων. Αξιοποιώντας τις προσεγγίσεις και τα ευρήματα που παρουσιάζονται σε αυτή την εργασία, η μελλοντική έρευνα μπορεί να συνεχίσει να προωθεί τον τομέα, επιτρέποντας τελικά στο fog computing να αποτελέσει ένα ισχυρό υπολογιστικό παράδειγμα.

9 Συμπεράσματα

Η παρούσα εργασία διερεύνησε το πρόβλημα της κατανομής φόρτου εργασίας σε περιβάλλοντα fog computing, προτείνοντας δύο συμπληρωματικές μεθόδους: τη μέθοδο Neighbor-aware και τη μέθοδο βασισμένη σε Particle Swarm Optimization (PSO). Αναπτύχθηκε το πρόγραμμα προσομοίωσης FogSim-NX για την ολοκληρωμένη αξιολόγηση των προτεινόμενων μεθόδων σε διάφορα σενάρια.

Τα αποτελέσματα της έρευνας καταδεικνύουν ότι η μέθοδος PSO επιτυγχάνει ανώτερη ποιότητα κατανομής με 8-25% χαμηλότερη κατανάλωση ενέργειας, 12-31% μειωμένο λειτουργικό κόστος και σημαντικά χαμηλότερη καθυστέρηση για εφαρμογές ευαίσθητες στον χρόνο. Ωστόσο, η βελτιωμένη απόδοση συνοδεύεται από αυξημένη υπολογιστική πολυπλοκότητα. Αντίθετα, η μέθοδος Neighbor-aware προσφέρει υπολογιστική αποδοτικότητα και ταχεία ανταπόκριση, καθιστώντας την καταλληλότερη για δυναμικά περιβάλλοντα με χρονικούς περιορισμούς.

Η μελέτη περίπτωσης έξυπνης πόλης επικύρωσε την πρακτική εφαρμογή των προτεινόμενων μεθόδων, αναδεικνύοντας τα οφέλη τους για τη βελτίωση της ποιότητας υπηρεσιών, την ενεργειακή αποδοτικότητα και τη μείωση κόστους. Η ανάλυση έδειξε ότι οι κόμβοι μεσαίου επιπέδου (district fog) προσφέρουν την καλύτερη αναλογία απόδοσης-κόστους, ενώ οι υβριδικές προσεγγίσεις κατανομής θα μπορούσαν να συνδυάσουν τα πλεονεκτήματα των δύο μεθόδων.

Παρά τους περιορισμούς της έρευνας, όπως η αξιολόγηση βασισμένη σε προσομοίωση και η υπόθεση στατικής τοπολογίας δικτύου, η εργασία συνεισφέρει σημαντικά στο πεδίο του fog computing, παρέχοντας καινοτόμες μεθόδους και εργαλεία για την αντιμετώπιση των προκλήσεων διαχείρισης πόρων. Οι μελλοντικές κατευθύνσεις έρευνας περιλαμβάνουν την ανάπτυξη υβριδικών προσεγγίσεων κατανομής, την ενσωμάτωση μηχανισμών δυναμικής προσαρμογής δικτύου και την υλοποίηση σε πραγματικά δοκιμαστικά περιβάλλοντα.

Συμπερασματικά, οι προτεινόμενες μέθοδοι και το πλαίσιο FogSim-NX μπορούν να διευκολύνουν την ανάπτυξη αποτελεσματικών και δυναμικών συστημάτων fog computing που ανταποκρίνονται στις απαιτήσεις των σύγχρονων εφαρμογών IoT για χαμηλή καθυστέρηση, υψηλή αξιοπιστία και ενεργειακή αποδοτικότητα. Η βελτιστοποίηση της κατανομής φόρτου εργασίας αποτελεί κρίσιμο βήμα για την αξιοποίηση των δυνατοτήτων του συνεχούς υπολογιστικού φάσματος από το άκρο έως το νέφος, συμβάλλοντας στην ανάπτυξη έξυπνων, αποδοτικών και βιώσιμων υπολογιστικών οικοσυστημάτων για την ψηφιακή εποχή.

10 Βιβλιογραφικές Αναφορές

[1] Bonomi, F., Milito, R., Zhu, J., & Addepalli, S. (2012). Fog computing and its role in the internet of things. In Proceedings of the first edition of the MCC workshop on Mobile cloud computing (pp. 13-16). ACM. https://dl.acm.org/doi/10.1145/2342509.2342513

[2] Deng, R., Lu, R., Lai, C., Luan, T. H., & Liang, H. (2016). Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption. IEEE Internet of Things Journal, 3(6), 1171-1181. https://doi.org/10.1109/JIOT.2016.2565516

[3] Gupta, H., Vahid Dastjerdi, A., Ghosh, S. K., & Buyya, R. (2017). iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. Software: Practice and Experience, 47(9), 1275-1296. https://doi.org/10.1002/spe.2509

[4] Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In Proceedings of ICNN'95-International Conference on Neural Networks (Vol. 4, pp. 1942-1948). IEEE. https://doi.org/10.1109/ICNN.1995.488968

[5] Mahmud, R., Ramamohanarao, K., & Buyya, R. (2020). Application management in fog computing environments: A taxonomy, review and future directions. ACM Computing Surveys, 53(4), 1-43. https://doi.org/10.1145/3403955

[6] Skarlat, O., Nardelli, M., Schulte, S., & Dustdar, S. (2017). Towards QoS-aware fog service placement. In 2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC) (pp. 89-96). IEEE. https://doi.org/10.1109/ICFEC.2017.12

[7] Zhou, Z., Liu, P., Feng, J., Zhang, Y., Mumtaz, S., & Rodriguez, J. (2021). Computation resource allocation and task assignment optimization in vehicular fog computing: A contract-matching approach. IEEE Transactions on Vehicular Technology, 70(2), 1719-1734. https://www.researchgate.net/publication/330580353

[8] Abbasi, M., Mohammadi-Pasand, E., & Khosravi, M. R. (2021). Intelligent workload allocation in IoT–Fog–cloud architecture towards mobile edge computing. Computer Communications, 169, pp.71-80. https://doi.org/10.1016/j.comcom.2021.01.009

[9] Lera, I., Guerrero, C., & Juiz, C. (2019). YAFS: A simulator for IoT scenarios in fog computing. IEEE Access, 7, pp.91745-91758. https://doi.org/10.1109/ACCESS.2019.2927895

[10] Vaquero, L. M., & Rodero-Merino, L. (2014). Finding your way in the fog: Towards a comprehensive definition of fog computing. ACM SIGCOMM Computer Communication Review, 44(5), pp.27-32. https://doi.org/10.1145/2677046.2677052

[11] Taneja, M., & Davy, A. (2017). Resource aware placement of IoT application modules in Fog-Cloud Computing Paradigm. In 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM) (pp. 1222-1228). IEEE. https://doi.org/10.23919/INM.2017.7987464

[12] Erdős, P., & Rényi, A. (1960). On the evolution of random graphs. Publications of the Mathematical Institute of the Hungarian Academy of Sciences, 5(1), 17-60.

https://www.renyi.hu/~p_erdos/1960-10.pdf