ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΕΥΦΥΗΣ ΔΙΑΧΕΙΡΙΣΗ ΦΙΛΤΡΩΝ ΓΙΑ ΑΥΤΟΝΟΜΟΥΣ ΚΟΜΒΟΥΣ

ΣΑΚΑΡΕΛΟΣ ΔΗΜΗΤΡΗΣ ΤΟΥ ΟΔΥΣΣΕΑ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΥΠΕΥΘΥΝΟΣ

Κολομβάτσος Κωνσταντίνος
Αναπληρωτής Καθηγητής

Λαμία 17 Ιουνίου έτος 2026

ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΕΥΦΥΗΣ ΔΙΑΧΕΙΡΙΣΗ ΦΙΛΤΡΩΝ ΓΙΑ ΑΥΤΟΝΟΜΟΥΣ ΚΟΜΒΟΥΣ

ΣΑΚΑΡΕΛΟΣ ΔΗΜΗΤΡΙΟΣ ΤΟΥ ΟΔΥΣΣΕΑ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΥΠΕΥΘΥΝΟΣ

Κολομβάτσος Κωνσταντίνος
Αναπληρωτής Καθηγητής

Λαμία 17 Ιουνίου έτος 2026

UNIVERSITY OF
THESSALY

SCHOOL OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE & TELECOMMUNICATIONS

INTELLIGENT MANAGEMENT OF FILTERS IN AUTONOMUS NODES

SAKARELOS DIMITRIOS

FINAL THESIS

ADVISOR

Kolomvatsos Konstantinos
Associate Professor

Lamia June 17, year 2026

«Με ατομική μου ευθύνη και γνωρίζοντας τις κυρώσεις [(1)], που προβλέπονται από της διατάξεις της παρ. 6 του άρθρου 22 του Ν. 1599/1986, δηλώνω ότι:

*1. Δεν παραθέτω κομμάτια βιβλίων ή άρθρων ή εργασιών άλλων αυτολεξεί **χωρίς να τα περικλείω σε εισαγωγικά** και χωρίς να αναφέρω το συγγραφέα, τη χρονολογία, τη σελίδα. Η αυτολεξεί παράθεση χωρίς εισαγωγικά χωρίς αναφορά στην πηγή, είναι λογοκλοπή. Πέραν της αυτολεξεί παράθεσης, λογοκλοπή θεωρείται και η παράφραση εδαφίων από έργα άλλων, συμπεριλαμβανομένων και έργων συμφοιτητών μου, καθώς και η παράθεση στοιχείων που άλλοι συνέλεξαν ή επεξεργάσθηκαν, χωρίς αναφορά στην πηγή. Αναφέρω πάντοτε με πληρότητα την πηγή κάτω από τον πίνακα ή σχέδιο, όπως στα παραθέματα.*

*2. Δέχομαι ότι η αυτολεξεί **παράθεση χωρίς εισαγωγικά**, ακόμα κι αν συνοδεύεται από αναφορά στην πηγή σε κάποιο άλλο σημείο του κειμένου ή στο τέλος του, είναι αντιγραφή. Η αναφορά στην πηγή στο τέλος π.χ. μιας παραγράφου ή μιας σελίδας, δεν δικαιολογεί συρραφή εδαφίων έργου άλλου συγγραφέα, έστω και παραφρασμένων, και παρουσίασή τους ως δική μου εργασία.*

3. Δέχομαι ότι υπάρχει επίσης περιορισμός στο μέγεθος και στη συχνότητα των παραθεμάτων που μπορώ να εντάξω στην εργασία μου εντός εισαγωγικών. Κάθε μεγάλο παράθεμα (π.χ. σε πίνακα ή πλαίσιο, κλπ.), προϋποθέτει ειδικές ρυθμίσεις, και όταν δημοσιεύεται προϋποθέτει την άδεια του συγγραφέα ή του εκδότη. Το ίδιο και οι πίνακες και τα σχέδια

4. Δέχομαι όλες τις συνέπειες σε περίπτωση λογοκλοπής ή αντιγραφής.

Ημερομηνία: 17/06/2026

Ο Δηλών

(1) «Όποιος εν γνώσει του δηλώνει ψευδή γεγονότα ή αρνείται ή αποκρύπτει τα αληθινά με έγγραφη υπεύθυνη δήλωση
του άρθρου 8 παρ. 4 Ν. 1599/1986 τιμωρείται με φυλάκιση τουλάχιστον τριών μηνών. Εάν ο υπαίτιος αυτών των πράξεων
σκόπευε να προσπορίσει στον εαυτόν του ή σε άλλον περιουσιακό όφελος βλάπτοντας τρίτον ή σκόπευε να βλάψει άλλον, τιμωρείται με κάθειρξη μέχρι 10 ετών.»

ΠΕΡΙΛΗΨΗ

Τα τελευταία χρόνια, η ραγδαία εξάπλωση του Διαδικτύου των Πραγμάτων (IoT) τόσο σε πλήθος όσο και σε είδος συσκευών καθώς και η γεωγραφική εξάπλωση και χρήση τους, έχει οδηγήσει στη δημιουργία τεράστιων οικοσυστημάτων συσκευών που παράγουν συνεχώς και με μεγάλο ρυθμό δεδομένα. Μεγάλος όγκος ποικιλόμορφων δεδομένων, παραγόμενα σε υψηλές ταχύτητες συσσωρεύεται. Η ανάγκη για επεξεργασία αυτών των δεδομένων, πολλές φορές σε πραγματικό χρόνο, καθιστά πλέον αδύνατη κι ασύμφορη τη μεταφορά όλου του όγκου των δεδομένων στο υπολογιστικό νέφος (Cloud) όπως παραδοσιακά γινόταν μέχρι πρότινος. Αισθητήρες περιβάλλοντος, βιομηχανικοί αισθητήρες κι ελεγκτές, φορητές συσκευές, έξυπνες κάμερες και πολλές άλλες συσκευές απαιτούν άμεση επεξεργασία δεδομένων και λήψη αποφάσεων. Οι προκλήσεις αυτές οδηγούν στη λύση της μεταφοράς δεδομένων, επεξεργασίας τους και λήψης άμεσα αποφάσεων, στις παρυφές του δικτύου (Edge Computing) ώστε η επεξεργασία να γίνει όσο το δυνατόν πιο κοντά στην πηγή παραγωγής των. Οι αυτόνομοι κόμβοι, έχοντας δική τους υπολογιστική ισχύ, αποθηκευτικό χώρο και δυνατότητα επεξεργασίας των δεδομένων μειώνουν σημαντικά το χρόνο λήψης αποφάσεων. Δυστυχώς όμως, οι αυτόνομοι κόμβοι υστερούν σε διαθέσιμη ενέργεια, όγκο δεδομένων που μπορούν να αποθηκεύσουν σε σχέση με το Cloud, αλλά και σε υπολογιστική ισχύ. Η σωστή διαχείριση και φιλτράρισμα δεδομένων αποτελούν επιτακτική ανάγκη. Τα φίλτρα επιτρέπουν στους κόμβους να μειώνουν τον όγκο δεδομένων που μεταδίδονται, να εντοπίζουν τυχόν ανωμαλίες και να διατηρούν μόνο χρήσιμες πληροφορίες ακόμη και σε περιπτώσεις δυναμικών περιβαλλόντων ώστε να επιτυγχάνεται αξιοπιστία κι ενεργειακή αποδοτικότητα του συστήματος. Στην παρούσα εργασία προτείνουμε μια διαφορετική προσέγγιση του τρόπου αναπαράστασης των αιτημάτων που καταφθάνουν στον κόμβο και των φίλτρων, μέσω γεωμετρίας στο δισδιάστατο χώρο, και χρήση διμετάβλητων κατανομών για την επιλογή

του χρόνου ανανέωσης των φίλτρων αλλά και τον τρόπο υπολογισμού τους. Εφαρμόζουμε την προτεινόμενη μέθοδο σε γλώσσα Python κάνοντας προσομοίωση διαχείρισης αιτημάτων, φίλτρων και δεδομένων σε έναν αυτόνομο κόμβο. Τα αποτελέσματα δείχνουν ότι το μοντέλο που υιοθετήθηκε έχει πολύ καλή συμπεριφορά στο να διαχειρίζεται μελλοντικά αιτήματα που καταφθάνουν στο κόμβο αλλά και στη διαχείριση των δεδομένων.

Λέξεις – κλειδιά: Διαδίκτυο των πραγμάτων,IoT, Cloud computing, Edge computing, data streaming, real time processing, φίλτρα, filter.

ABSTRACT

In recent years, the rapid expansion of the Internet of Things (IoT), both in terms of the number and types of devices as well as their geographical spread and usage, has led to the creation of vast ecosystems of devices that continuously generate data at a high rate. A large volume of diverse data, produced at high speeds, accumulates. The need to process this data, often in real time, now makes the transfer of the entire volume of data to the Cloud, as traditionally done until recently, both impossible and impractical. Environmental sensors, industrial sensors and controllers, portable devices, smart cameras, and many other devices require immediate data processing and decision-making. These challenges lead to the solution of transferring data, processing it, and making immediate decisions at the network Edge (Edge Computing) so that processing can occur as close as possible to the source of production. Autonomous nodes, having their own computing power, storage space, and data processing capability, significantly reduce decision-making time. Unfortunately, however, autonomous nodes lag in available energy, the volume of data they can store compared to the Cloud, and computing power. Proper data management and filtering are an urgent necessity. Filters allow nodes to reduce the volume of transmitted data, detect any anomalies, and retain only useful information even in dynamic environments to achieve system reliability and energy efficiency. In this paper, we propose a different approach to the way requests arriving at the node and filters are represented through geometry in the two-dimensional space, and the use of bivariate distributions for selecting the filter refresh time as well as the method of their calculation. We implement the proposed method in Python by simulating the management of requests, filters, and data in a standalone node. The results show that the adopted model performs

very well in handling future requests that arrive at the node as well as in managing the data.

Keywords: Internet of Things, Cloud computing, Edge computing, data streaming, real-time processing, filters.

Table of Contents

ΚΕΦΑΛΑΙΟ 1 Εισαγωγή

1.1 Ορισμοί

1.1.1 Internet of Things

Ο όρος Internet of Things (IoT) περιγράφει ένα εκτεταμένο οικοσύστημα επικοινωνίας, στο οποίο πλήθος ηλεκτρονικών συσκευών – τα λεγόμενα *things* – διαθέτουν αισθητήρες, λογισμικό και δυνατότητα σύνδεσης στο διαδίκτυο, επιτρέποντας την αυτόματη ανταλλαγή δεδομένων χωρίς την ανάγκη ανθρώπινης παρέμβασης [1], [2]. Ως "Thing" μπορεί να θεωρηθεί οποιαδήποτε συσκευή ή οντότητα που μπορεί να συλλέγει και να μεταδίδει πληροφορίες: από smartphones και smartwatches, μέχρι αισθητήρες θερμοκρασίας ή φωτός, αλλά και πιο εξειδικευμένες περιπτώσεις όπως ιατρικά εμφυτεύματα ή ζώα με ενσωματωμένα αναγνωριστικά κυκλώματα (biochip transponders) [3].

Η ιδέα της ενσωμάτωσης αισθητήρων σε καθημερινά αντικείμενα είχε ήδη διατυπωθεί από τη δεκαετία του 1980, όμως η τεχνολογία της εποχής δεν επέτρεπε την πρακτική υλοποίησή της [4]. Η απουσία οικονομικών και ενεργειακά αποδοτικών επεξεργαστών καθιστούσε αδύνατη τη μαζική διασύνδεση εκατομμυρίων συσκευών. Η κατάσταση άρχισε να αλλάζει με την εμφάνιση των RFID tags, τα οποία προσέφεραν ασύρματη επικοινωνία με ελάχιστη κατανάλωση ενέργειας [5], καθώς και με την καθιέρωση του IPv6, που επέτρεψε την τεράστια διεύρυνση του διαθέσιμου χώρου διευθύνσεων [6]. Το 1999, ο Kevin Ashton, συνιδρυτής του Auto-ID Centre στο MIT, χρησιμοποίησε για πρώτη φορά τον όρο *Internet of Things*, περιγράφοντας το όραμα ενός κόσμου όπου τα αντικείμενα θα επικοινωνούν αυτόνομα μέσω του διαδικτύου [7].

Αρχικά, οι εφαρμογές του IoT περιορίζονταν στην απλή παρακολούθηση πολύτιμων αντικειμένων μέσω RFID. Με την πάροδο

των χρόνων, όμως, το κόστος των αισθητήρων και των ενσωματωμένων κυκλωμάτων μειώθηκε δραστικά, ενώ τα ασύρματα δίκτυα έγιναν πιο αξιόπιστα και προσβάσιμα [8]. Σήμερα, η βασική λειτουργικότητα μιας IoT συσκευής μπορεί να κοστίζει ελάχιστα, επιτρέποντας τη διασύνδεση σχεδόν οποιουδήποτε αντικειμένου.

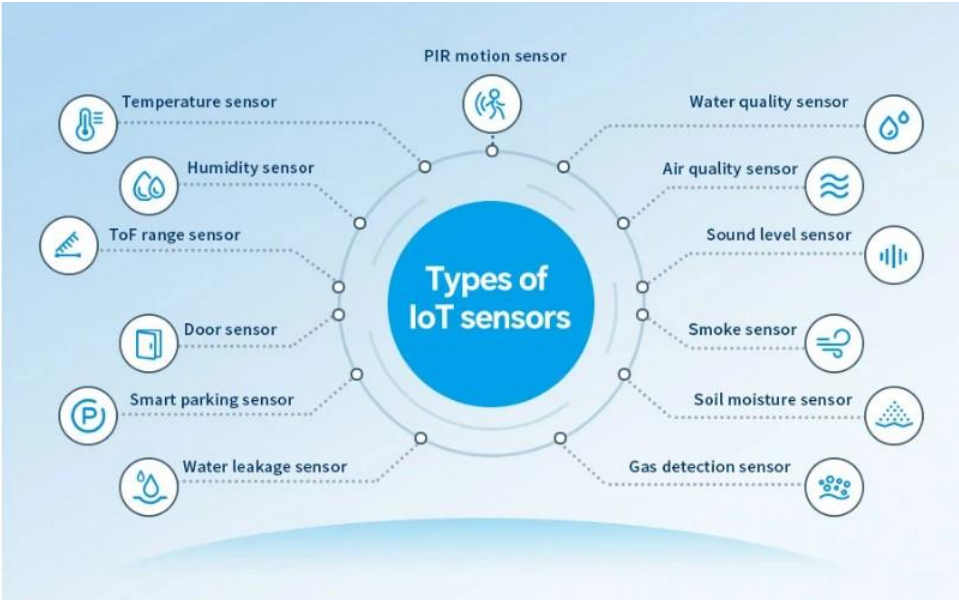
Οι σύγχρονες εφαρμογές του IoT καλύπτουν ένα ευρύ φάσμα τομέων όπως ενδεικτικά:

- **Wearables**: Έξυπνα ρολόγια και βραχιόλια που παρακολουθούν βιομετρικά δεδομένα, δραστηριότητα, ύπνο και προσφέρουν δυνατότητες επικοινωνίας [9].
- **Έξυπνα σπίτια (Smart Homes**): Διαχείριση φωτισμού, συσκευών, συστημάτων ασφαλείας και αυτοματισμών μέσω κινητού τηλεφώνου [10].
- **Γεωργία ακριβείας**: Αισθητήρες που μετρούν υγρασία, θερμοκρασία και φωτεινότητα για βελτιστοποίηση της παραγωγής [11].
- **Υγεία:** Διασύνδεση ιατρικών μηχανημάτων για συνεχή παρακολούθηση ασθενών και υποστήριξη διαγνωστικών διαδικασιών [12].
- **Βιομηχανία**: Αυτοματισμοί, έλεγχος αποθεμάτων, προληπτική συντήρηση και βελτιστοποίηση παραγωγικών διαδικασιών [13].

Η σύνδεση συσκευών, ανθρώπων και περιβάλλοντος μέσω IoT συστημάτων προσφέρει σημαντικά πλεονεκτήματα: δυνατότητα ανάλυσης μεγάλων ποσοτήτων δεδομένων για πρόβλεψη και λήψη αποφάσεων, μείωση λαθών, απομακρυσμένη παρακολούθηση σε πραγματικό χρόνο και αύξηση της αποδοτικότητας μέσω αυτοματισμών [14].

Παρά τα οφέλη, το IoT εξακολουθεί να αντιμετωπίζει σημαντικές προκλήσεις. Ο τεράστιος όγκος δεδομένων που παράγεται συνεχώς

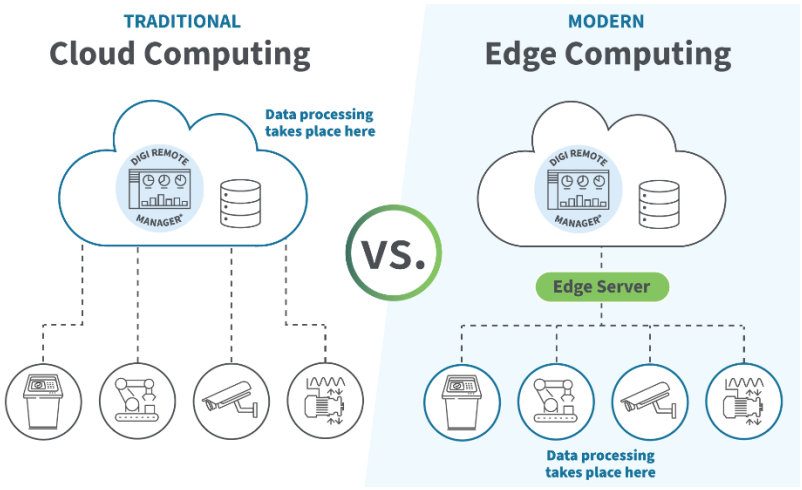
απαιτεί μεγάλους αποθηκευτικούς πόρους και εξειδικευμένες τεχνικές επεξεργασίας [15]. Επιπλέον, ζητήματα ασφάλειας και προστασίας δεδομένων παραμένουν κρίσιμα, καθώς κακόβουλοι χρήστες μπορούν να υποκλέψουν πληροφορίες ή να αποκτήσουν πρόσβαση σε συστήματα [16]. Η έλλειψη ενιαίων διεθνών προτύπων δυσκολεύει επίσης τη διαλειτουργικότητα συσκευών διαφορετικών κατασκευαστών [17].



Εικόνα 1: Διάφοροι τύποι αισθητήρων IoT

1.1.2 Edge Computing

Το Edge computing αποτελεί ένα σύγχρονο υπολογιστικό παράδειγμα που μεταφέρει την επεξεργασία δεδομένων από τα κεντρικά υπολογιστικά νέφη (Cloud datacenters) προς το άκρο του δικτύου, δηλαδή κοντά στις συσκευές που παράγουν τα δεδομένα [17], [18]. Η προσέγγιση αυτή ήταν η απάντηση στην εκθετική αύξηση των IoT συσκευών και του όγκου δεδομένων που παράγουν, καθιστώντας μη βιώσιμη την αποκλειστική εξάρτηση από το Cloud για επεξεργασία και αποθήκευση [18].



Εικόνα 2: Διαφορές Edge computing και Cloud computing

Σε αντίθεση με το παραδοσιακό Cloud computing, όπου τα δεδομένα μεταφέρονται σε απομακρυσμένα datacenters, το Edge computing επιτρέπει την τοπική επεξεργασία, μειώνοντας την καθυστέρηση (latency), το απαιτούμενο εύρος ζώνης και την εξάρτηση από τη συνδεσιμότητα.

Η ανάγκη για Edge computing προέκυψε από λειτουργικές και τεχνολογικές προκλήσεις όπως:

- Εκρηκτική αύξηση IoT συσκευών: Ο αριθμός των IoT συσκευών αυξάνεται με ρυθμούς που υπερβαίνουν τις δυνατότητες των παραδοσιακών δικτύων και Cloud υποδομών [8]. Οι συσκευές αυτές παράγουν συνεχώς δεδομένα υψηλής συχνότητας, τα οποία δεν είναι πρακτικό να μεταφέρονται όλα στο Cloud.
- Απαιτήσεις real-time εφαρμογών: Εφαρμογές όπως αυτόνομα οχήματα, βιομηχανικός αυτοματισμός και συστήματα υγείας απαιτούν απόκριση σε χιλιοστά του δευτερολέπτου, πολλές φορές και λιγότερο, κάτι που δεν μπορεί να επιτευχθεί με την καθυστέρηση που εισάγει το Cloud [17].
- Περιορισμοί εύρους ζώνης: Η συνεχής μεταφορά δεδομένων από εκατομμύρια συσκευές προς το Cloud δημιουργεί συμφόρηση και υψηλό κόστος δικτύου [18].

- Ζητήματα ιδιωτικότητας και ασφάλειας: Η τοπική επεξεργασία μειώνει την ανάγκη μεταφοράς ευαίσθητων δεδομένων, ενισχύοντας την προστασία ιδιωτικότητας [16].

Η αρχιτεκτονική του αποτελείται από τα παρακάτω επίπεδα:

1. Επίπεδο Συσκευών (Device Layer): Περιλαμβάνει IoT αισθητήρες, actuators και constrained devices με περιορισμένη υπολογιστική ισχύ [7]. Εδώ γίνεται βασική προ-επεξεργασία, φιλτράρισμα και event detection.
2. Επίπεδο Edge Nodes: Αποτελείται από gateways, routers, micro-servers ή τοπικούς κόμβους που διαθέτουν μεγαλύτερη υπολογιστική ισχύ. Εκτελούν: Aggregation, Analytics, ML inference, προσωρινή αποθήκευση (Edge caching) [17].
3. Επίπεδο Fog / MEC (Multi-access Edge Computing): Ενδιάμεσο επίπεδο μεταξύ Edge και Cloud, συχνά ενσωματωμένο σε τηλεπικοινωνιακές υποδομές (π.χ. 5G base stations).
4. Επίπεδο Cloud: Παραμένει υπεύθυνο για μακροχρόνια αποθήκευση, εκπαίδευση ML μοντέλων, μεγάλης κλίμακας analytics.

Το Edge δεν αντικαθιστά το Cloud αλλά το συμπληρώνει. Επίσης έχει πλεονεκτήματα απέναντι στην αμιγή χρήση Cloud όπως:

- **Μείωση καθυστέρησης (Low Latency)**: Η επεξεργασία κοντά στη συσκευή μειώνει τον χρόνο απόκρισης από εκατοντάδες ms σε λίγα ms [17].
- **Μείωση απαιτήσεων bandwidth**: Στο Cloud αποστέλλονται μόνο συμπιεσμένα ή φιλτραρισμένα δεδομένα [18].
- **Αξιοπιστία σε ασταθή δίκτυα**: Οι συσκευές συνεχίζουν να λειτουργούν ακόμη και με περιορισμένη συνδεσιμότητα.

- **Ενίσχυση ιδιωτικότητας**: Τα δεδομένα παραμένουν τοπικά, μειώνοντας τον κίνδυνο διαρροών [16].
- **Υποστήριξη real-time εφαρμογών**: Απαραίτητο για αυτόνομα οχήματα, βιομηχανικούς αυτοματισμούς, έξυπνες πόλεις και ιατρική παρακολούθηση.

Το Edge computing αποτελεί κρίσιμο κομμάτι της σύγχρονης ψηφιακής υποδομής, επιτρέποντας την αποδοτική λειτουργία των IoT συστημάτων, την υποστήριξη real-time εφαρμογών και τη μείωση του φόρτου στο Cloud. Η στενή του σχέση με τα constrained devices και τις τεχνικές τοπικής επεξεργασίας το καθιστά απαραίτητο για την ανάπτυξη έξυπνων, κλιμακούμενων και ενεργειακά αποδοτικών συστημάτων.

1.1.3 Constrained Devices

Τα constrained devices αποτελούν τον θεμέλιο λίθο των συστημάτων IoT και των αρχιτεκτονικών Edge computing, καθώς είναι οι συσκευές που βρίσκονται στο άκρο του δικτύου και αλληλοεπιδρούν άμεσα με το φυσικό περιβάλλον. Πρόκειται για συσκευές χαμηλού κόστους και χαμηλής κατανάλωσης ενέργειας, οι οποίες διαθέτουν σημαντικούς περιορισμούς σε υπολογιστικούς και λειτουργικούς πόρους [7], [11].

Οι περισσότερες constrained συσκευές βασίζονται σε μικροελεγκτές (MCUs) αντί για πλήρεις επεξεργαστές, γεγονός που περιορίζει σημαντικά τις δυνατότητές τους. Συνήθως χρησιμοποιούν πυρήνες όπως ARM Cortex-M0/M3/M4, με συχνότητες λειτουργίας από 16 MHz έως 200 MHz [7].

Η διαθέσιμη μνήμη RAM είναι εξαιρετικά περιορισμένη, κυμαινόμενη συνήθως μεταξύ 16 KB και 512 KB, ενώ ο αποθηκευτικός χώρος (flash) κυμαίνεται από 256 KB έως 4 MB, επαρκής μόνο για βασικό firmware και μικρές ποσότητες δεδομένων [12].

Αυτοί οι περιορισμοί καθιστούν αδύνατη την εκτέλεση βαριών αλγορίθμων, απαιτώντας ελαφρά πρωτόκολλα, απλοποιημένες δομές δεδομένων και βέλτιστη διαχείριση μνήμης [1].

Εικόνα 3: Μειονεκτήματα constrained συσκευών

Πολλά constrained devices λειτουργούν με μπαταρία ή με energy harvesting, γεγονός που επιβάλλει αυστηρούς περιορισμούς στην κατανάλωση ενέργειας. Η επικοινωνία μέσω ασύρματων δικτύων αποτελεί μία από τις πιο ενεργοβόρες λειτουργίες, γι' αυτό και τα πρωτόκολλα που χρησιμοποιούνται είναι σχεδιασμένα για ελάχιστη κατανάλωση [9].

Η ανάγκη για πολυετή αυτονομία οδηγεί σε τεχνικές όπως:

- duty cycling (π.χ. ContikiMAC) [5]
- event-driven ενεργοποίηση
- τοπική επεξεργασία αντί για αποστολή δεδομένων

Η ενέργεια αποτελεί συχνά τον κυρίαρχο περιορισμό, περισσότερο ακόμη και από την υπολογιστική ισχύ [11] .

Μερικά παραδείγματα constrained συσκευών είναι:

- ESP8266 / ESP32
- Arduino UNO / Nano / MKR

- Raspberry Pi Zero (σε ελαφριές εφαρμογές)
- Αισθητήρες θερμοκρασίας, υγρασίας, πίεσης
- Wearables [9]
- Βιομηχανικοί αισθητήρες [13]
- Έξυπνοι μετρητές ενέργειας

1.1.4 Σκοπός της παρούσας εργασίας

Σκοπός της παρούσας εργασίας είναι η ανάπτυξη μιας μεθοδολογίας διαχείρισης φίλτρων με ευφυή τρόπο, που περιλαμβάνει την απόφαση πότε θα γίνει ανανέωση (filter update) με ποιον τρόπο και πως θα υπολογιστεί το νέο φίλτρο. Επίσης μελετάται η διαδικασία επιλογής αυτών των δεδομένων που θα παραμείνουν στον κόμβο και αυτών που θα μεταναστεύσουν στο Cloud ή σε γειτονικούς – συνεργαζόμενους κόμβους.

Όπως αναπτύσσεται παρακάτω, υπάρχει πληθώρα εργασιών που έχουν εκπονηθεί πάνω σε αυτά τα θέματα. Κάποιες εστιάζουν στην αποθήκευση των δεδομένων, κάποιες στην επεξεργασία κι άλλες στη μετακίνησή δεδομένων κι υπηρεσιών. Οι περισσότερες από αυτές τις εργασίες μελετούν τα φίλτρα και τα δεδομένα με την προϋπάρχουσα μορφή τους κι εστιάζουν σε τρόπους επεξεργασίας μέσω αλγοριθμικής προσέγγισης αλλά και τη χρήσης μηχανικής μάθησης. Υπάρχουν και κάποιες εργασίες όμως που δίνουν μια άλλη μορφή – απεικόνιση τόσο των φίλτρων όσο και των δεδομένων, χρησιμοποιώντας Ευκλείδεια γεωμετρία επιπέδου για την αναπαράσταση εργασιών, αιτημάτων και φίλτρων. Ορμώμενοι από αυτή την σκέψη, στην παρούσα εργασία επεκτείνουμε τη χρήση επίπεδης γεωμετρίας ώστε η επεξεργασία εργασιών, φίλτρων και δεδομένων να αντιστοιχίζονται σε διαδικασίες που λαμβάνουν χώρα στο δισδιάστατο χώρο.

Σκοπός παραμένει πάντα η έξυπνη διαχείριση των φίλτρων ώστε να προβλεφθεί (όσο είναι δυνατόν) η μελλοντική κίνηση στο κόμβο, όσον

αφορά εργασίες που θα αφιχθούν αλλά και δεδομένα που θα τύχουν επεξεργασίας. Η καινοτομία που εισάγεται με την εργασία αυτή είναι ότι η απλότητα της επίπεδης γεωμετρίας, η οποία, ειρήσθω εν παρόδω, διδάσκεται από τις πρώτες τάξεις του γυμνασίου, χρησιμοποιείται για να γίνει πιο κατανοητή η όλη διαδικασία αλλά και για να εκμεταλλευτούμε τις ιδιότητές της. Όπως θα γίνει αντιληπτό σε επόμενο κεφάλαιο όπου θα αναπτυχθεί η όλη διαδικασία αναλυτικά, ο αναγνώστης αντί για φίλτρα, αιτήματα και δεδομένα, έρχεται αντιμέτωπος με σημεία του επιπέδου, τρίγωνα, ευθείες κ.α. Η διαχείριση των φίλτρων γίνεται με τη βοήθεια στατιστικών διμετάβλητων κατανομών που βρίσκουν άμεση εφαρμογή στο δισδιάστατο χώρο και χρησιμοποιούνται στη λήψη αποφάσεων τόσο χρονικά όσο και ποιοτικά.

Μιλάμε λοιπόν για μια πλήρως μαθηματικοποιημένη αντιμετώπιση του προβλήματος με χρήση Γεωμετρίας, Ανάλυσης και Στατιστικής στο πρόβλημα διαχείρισης φίλτρων σε αυτόνομους κόμβους. Παρά την απλότητα των μαθηματικών στοιχείων που χρησιμοποιούνται, οι ενέργειες και μέθοδοι που προτείνονται δίνουν μια ικανοποιητική λύση στο πρόβλημα. Πεποίθηση του συγγραφέα της παρούσας εργασίας είναι ότι σκοπός των μαθηματικών, εκτός από την επίλυση προβλημάτων, είναι και η απλοποίηση του τρόπου που μπορεί να αναπαρασταθούν πτυχές του περιβάλλοντός μας ώστε να γίνει πιο απτή η απεικόνιση και βαθύτερη η κατανόησή τους.

ΚΕΦΑΛΑΙΟ 2 Βιβλιογραφική Επισκόπηση

2.1 Εισαγωγικά στοιχεία

Το Edge computing έχει γίνει πλέον βασικό μοντέλο στο Internet of Things (IoT). Η επεξεργασία των δεδομένων κοντά στη πηγή δεδομένων μειώνει την ανάγκη επεξεργασίας σε διακομιστές Cloud [19]. Με την αλλαγή αυτή επιτυγχάνουμε μείωση του λανθάνοντος χρόνου (latency), τη χρήση εύρους ζώνης αλλά και των κινδύνων απορρήτου που σχετίζονται με αρχιτεκτονικές με επίκεντρο το Cloud [20]. Η αύξηση της πώλησης ηλεκτρονικών ευρείας κατανάλωσης, η ποικιλία συσκευών IoT που υπάρχουν και η εξάπλωσή τους σε όλο τον κόσμο, έχουν αυξήσει την ανάγκη για έξυπνα χαρακτηριστικά σε αυτόνομους μικρούς κόμβους με περιορισμένους πόρους [21]. Λόγω της σημαντικότητας, υπάρχει πληθώρα εργασιών και μελετών πάνω στο θέμα της διαχείρισης δεδομένων κι υπηρεσιών σε constrained devices. Έχουν μελετηθεί εις βάθος και ποικιλοτρόπως η αποθήκευση, η επεξεργασία αλλά και η μετακίνηση δεδομένων κι υπηρεσιών.

2.2. Αποθήκευση δεδομένων

Οι Zhijie Shen, Bowen Liu, Wanchun Dou [22] προτείνουν μια μέθοδο αποθήκευσης δεδομένων αρχικά αναλύοντας το χρόνο πρόσβασης και το κόστος τοποθέτησης δεδομένων σε ένα περιβάλλον συνεργατικό μεταξύ Edge και Cloud. Στη συνέχεια σχεδιάζουν μια στρατηγική ε και τη χαλάρωση Lagrangian. Συγκεκριμένα, έχοντας να αντιμετωπίσουν δύο αντικρουόμενους στόχους, την ελαχιστοποίηση του χρόνου πρόσβασης και την ελαχιστοποίηση του κόστους τοποθέτησης δεδομένων, χρησιμοποιείται ένα άνω φράγμα ε στον ένα στόχο και προσπάθεια ελαχιστοποίησης του άλλου στόχου. Παράλληλα χρησιμοποιείται Lagrangian relaxation για τη μείωση («χαλάρωμα»)

περιορισμών ενσωματώνοντάς τους στην αντικειμενική συνάρτηση με ποινή.

Σε μια άλλη περίπτωση επιστημονικών ροών εργασίας (Scientific Workflows) οι υπολογιστικές εργασίες πρέπει να εκτελεσθούν διαδοχικά ή παράλληλα και συχνά μοιράζονται μεγάλα σύνολα δεδομένων μεταξύ τους [23].Προτείνεται ένα μοντέλο που συνδυάζει πλεονεκτήματα τόσο του Edge computing όσο και του Cloud computing, και δημιουργεί ένα μοντέλο τοποθέτησης δεδομένων που εξισορροπεί χρόνο μεταφοράς και κόστος τοποθέτησης μέσω μιας στρατηγικής DE-DPSO-DPS η οποία βασίζεται σε έναν αλγόριθμο PSO (Particle Swarm Optimization) με διαφορική εξέλιξη (Differential Evolution).

Η χρήση IPFS (Inter Planetary File System) είναι ένα αποκεντρωμένο σύστημα αποθήκευσης, σαν ένα κατανεμημένο Cloud χωρίς κεντρικό server. Η χρήση του όμως δημιουργεί δυο νέα προβλήματα. Πρώτον απαιτεί σημαντικό χρόνο για τη καταγραφή δεδομένων και δεύτερον, τα CID's (Content Identifiers) δηλαδή οι μοναδικοί κωδικοί που αντιστοιχούν στα αρχεία του IPFS, δεν περιέχουν πληροφορίες για την πηγή προέλευσης ή μεταδεδομένα. Αυτό δημιουργεί προβλήματα στην ανάκτηση δεδομένων βάσει συγκεκριμένων συνθηκών.[24] Η προτεινόμενη λύση είναι μια υβριδική αρχιτεκτονική με χρήση της υποδομής που προσφέρει το MQTT (Message Queuing Telemetry Transport) για αποδοτική μεταφορά CID και μια βάση δεδομένων για αρχειοθέτηση των τιμών CIDS και σχετικών μεταδεδομένων. Με αυτό τον τρόπο η συσκευή ξέρει πότε να σβήσει τοπικά δεδομένα που έχουν ήδη αποθηκευτεί ασφαλώς στο IPFS.

Ένας άλλος παράγοντας που παίζει ρόλο στην αποθήκευση δεδομένων σε τοπικό επίπεδο είναι η σωστή γεωγραφική τοποθέτηση των κόμβων [25]. Μέσα από τη μελέτη 64 άρθρων για οκτώ χρόνια (2015-2022) μελετήθηκαν γεωγραφικοί μέθοδοι αποθήκευσης δεδομένων, χωρισμένοι σε πέντε κατηγορίες: Ακέραιο Προγραμματισμό, Ευρετικές και μεταευρετικές μεθόδους, υβριδικές μεθόδους μεταξύ των δύο

προαναφερθέντων κατηγοριών, μέθοδοι ομαδοποίησης κι ενισχυτική μάθηση.

Μεγάλα βήματα προόδου έχουν γίνει και στο θέμα της συνεργατικής αποθήκευσης δεδομένων. Στη συνεργασία δηλαδή μεταξύ κόμβων και διαμοιρασμού των δεδομένων που λαμβάνονται λόγω της περιορισμένης αποθηκευτικής χωρητικότητας. Δημιουργείται όμως το πρόβλημα ότι οι Edge servers μπορεί να υπάγονται σε διαφορετικούς πάροχους υποδομής. Δύο κεντρικά προβλήματα που δυσκολεύουν την συνεργατική αποθήκευση δεδομένων είναι η «εμπιστοσύνη» (trust) και τα «κίνητρα» (incentive) [26]. Προτείνεται λοιπόν το σύστημα CS-Edge, σύμφωνα με το οποίο οι Edge servers μπορούν να δημοσιεύουν αιτήματα μεταφοράς δεδομένων για τα οποία ανταγωνίζονται οι άλλοι κόμβοι, και οι νικητές επιλέγονται βάσει της «φήμης» τους. Αποθηκεύονται τα μεταφερόμενα δεδομένα και λαμβάνουν αμοιβές για την επιτυχή ολοκλήρωση των εργασιών. Μέσω κατανεμημένης συναίνεσης (distributed consensus) αυξάνεται η απόδοσή τους.

Σε άλλη μελέτη αντιμετωπίζεται το πρόβλημα Collaborative Edge Data Caching (CEDC), με στόχο την ελαχιστοποίηση του συνολικού κόστους που περιλαμβάνει κόστος αποθήκευσης, κόστος μετανάστευσης δεδομένων μεταξύ κόμβων και ποινή υποβάθμισης ποιότητα υπηρεσίας (QoS) [27]. Αποδεικνύεται ότι το πρόβλημα αυτό είναι NP-complete. Αυτό σημαίνει ότι δεν υπάρχει αλγόριθμος που να το λύνει σε πολυωνυμικό χρόνο, επομένως χρειάζεται έξυπνη προσέγγιση. Ο αλγόριθμος CEDC λειτουργεί online (CEDC-O) σε κάθε χρονική στιγμή, χωρίς να απαιτεί γνώση μελλοντικών πληροφοριών. Είναι βασισμένος στην Βελτιστοποίηση Lyapunov (Lyapunov Optimization). Σύμφωνα με αυτόν, εισάγονται εικονικές ουρές για να αναπαρασταθούν οι περιορισμοί του συστήματος, σε κάθε χρονική στιγμή (time slot) παίρνει αποφάσεις που ελαχιστοποιούν το στιγμιαίο κόστος χωρίς να ενδιαφέρεται για μελλοντική γνώση, αποφασίζει μόνο με βάση τα τρέχοντα δεδομένα.

Μεγάλο πρόβλημα στην συνεργατική αποθήκευση είναι η ετερογένεια κινητών συσκευών που συμμετέχουν στο Edge computing, η μεταβλητότητά τους (πχ κινητές συσκευές αποσυνδέονται, μετακινούνται, εξαντλούν μπαταρία), ο φόρτος δικτύου και η μη ύπαρξη κεντρικού ελέγχου [28]. Προτείνεται ο αλγόριθμος A2CS (Acceleration Algorithm for Collaborative Storage) βασισμένος στην αρχιτεκτονική Alternative Direction Method of Multipliers (ADMM), μια τεχνική κατανεμημένης βελτιστοποίησης, σύμφωνα με την οποία διασπάται το πρόβλημα σε μικρότερα υποπροβλήματα που μπορεί να λύσει κάθε κόμβος τοπικά και στη συνέχεια τα αποτελέσματα συντονίζονται μεταξύ τους. Η μέθοδος ADMM συνδυάζεται με επιτάχυνση Nesterov, μια τεχνική που αλλάζει το πώς επιλέγεται το επόμενο σημείο σύγκλισης κατά την βελτιστοποίηση.

Η αποθήκευση (caching) των απαραίτητων δεδομένων προγράμματος σε έναν κόμβο Edge, μειώνει αποτελεσματικά το χρόνο απόκρισης. Λόγω της αύξησης χρηστών συσκευών IoT, οι κόμβοι Edge δέχονται πλήθος αιτημάτων επιβαρύνοντας σημαντικά την επεξεργασία τους. Μια πρόταση είναι ένα συνεργατικό σύστημα αποθήκευσης προγραμμάτων (cooperative program caching) [29], στο οποίο διαφορετικοί κόμβοι συνεργάζονται για να αποθηκεύσουν δεδομένα και να δημιουργήσουν αντίγραφα (replicas) των δεδομένων που ζητούνται συχνά. Από νωρίς έχουν προταθεί αλγόριθμοι βέλτιστης τοποθέτησης αντιγράφων (Replica Placement) [30] [31] όπως η Δυναμική Αποκεντρωμένη Στρατηγική Τοποθέτησης Αντιγράφων σε Edge computing, (Dynamic Decentralized Strategy of Replica Placement on Edge Computing – DDRP) [32] ο οποίος μελετά τη τοποθεσία, τη συχνότητα πρόσβασης, τη βελτίωση καθυστέρησης και το κόστος τοποθέτησης αντιγράφων στους κόμβους. Άλλη προσέγγιση στο πρόβλημα είναι ο αλγόριθμος ITO (Intelligent Swarm Optimization) [33] σύμφωνα με τον οποίο η τοποθέτηση αντιγράφων δεδομένων μοντελοποιείται ως πρόβλημα ακέραιου προγραμματισμού 0-1, λαμβάνοντας υπόψη τη συνολική εξάρτηση των δεδομένων, την αξιοπιστία και τη συνεργασία των

χρηστών. Η αποθήκευση πολλαπλών αντιγράφων μπορεί να σπαταλήσει πολύτιμο χώρο αποθήκευσης. Η Κωδικοποίηση Διαγραφής (Erasure Coding) [34] αντιμετωπίζει αυτό το πρόβλημα χωρίζοντας κάθε αρχείο σε k τμήματα και στη συνέχεια δημιουργούνται $f = k \times n$ κωδικοποιημένα τμήματα (όπου n είναι το πλήθος των αρχείων), με τέτοιο τρόπο ώστε οποιοδήποτε υποσύνολο k από αυτά να επαρκεί για την ανακατασκευή του αρχείου. Η χρήση DHT (Distributed Hash Table) [35] είναι άλλος ένας τρόπος οργάνωσης των κόμβων που θα προτιμηθούν για την αποθήκευση αντιγράφων όπου κάθε κόμβος είναι υπεύθυνος για ένα τμήμα δεδομένων με βάση μια συνάρτηση κατακερματισμού.

Τέλος, έχει προταθεί η χρήση προηγμένων μεθόδων Μηχανικής Μάθησης όπως Ασαφής Λογική και αλγόριθμοι βελτιστοποίησης όπως Γενετικοί αλγόριθμοι (Genetic Algorithms) ή αλγόριθμοι Σμήνους (Swarm Intelligence) [36]. Με αυτές τις μεθόδους επιτρέπεται στο σύστημα να χειρίζεται την αβεβαιότητα και την ανακρίβεια (φόρτος δικτύου, χωρητικότητα αποθήκευσης κ.τ.λ.) και να βελτιστοποιείται η τοποθέτηση αντιγράφων, βελτιστοποιώντας μετρικές όπως ελαχιστοποίηση χρόνου πρόσβασης, μέγιστη διαθεσιμότητα και ισορρόπηση φορτίου (load balancing).

2.3 Επεξεργασία δεδομένων

Στο τομέα της επεξεργασίας των δεδομένων σε τοπικό επίπεδο, έχουν προταθεί αρκετές μέθοδοι. Μια από αυτές είναι το Oops (Optimizing Operation-mode Selection for IoT Edge Devices)[37]. Σύμφωνα με αυτό, οι συσκευές IoT υποστηρίζουν παράλληλες εργασίες όπως τοπική επεξεργασία, εκφόρτωση (offloading) και υβριδική κατάσταση. Το Oops, διασπά το πρόβλημα βελτιστοποίησης αυτών των εργασιών σε επιμέρους τμήματα ώστε να μην απαιτεί κεντρικούς υπολογιστικούς κόμβους. Σε κάθε νέο τμήμα δεδομένων υπολογίζεται η νέα κατάσταση λειτουργίας της συσκευής και η καθυστέρηση οριοθετείται από τη χειρότερη

περίπτωση, δηλαδή την λανθάνουσα εκτέλεση για όλο το τμήμα. Αποτέλεσμα είναι να μειώνεται δραστικά το runtime overhead και οι συσκευές να καταναλώνουν λιγότερους πόρους για να αποφασίσουν τι θα κάνουν.

Άλλη προσέγγιση στο πρόβλημα επεξεργασίας δεδομένων είναι η χρήση μηχανικής μάθησης [38]. Συγκεκριμένα προτείνεται ο συνδυασμός Ανάλυσης Κύριων Συνιστωσών (Principal Component Analysis - PCA), Δέντρο Αποφάσεων (Decision Tree - DT) και ταξινομητές Μηχανής Διανυσμάτων Στήριξης (Support Vector Machines - SVM). Η PCA μειώνει τον αριθμό μεταβλητών που πρέπει να επεξεργαστεί η συσκευή. Το DT αποδεικνύεται ότι λειτουργεί εξίσου καλά με πολύπλοκα νευρωνικά δίκτυα στη λήψη αποφάσεων και η SVM βρίσκει την «καλύτερη» γραμμή διαχωρισμού μεταξύ κατηγοριών δεδομένων, ώστε να αυξήσει την ακρίβεια εκεί που δε τα καταφέρνει το DT. Η εργασία έδειξε ότι δε χρειάζεται να στείλουμε τα δεδομένα στο Cloud ώστε να τύχουν επεξεργασίας από πανίσχυρους υπολογιστές αλλά με τον έξυπνο συνδυασμό PCA+DT+SVM μπορούμε να έχουμε αξιόπιστη μηχανική μάθηση και στις πιο αδύναμες συσκευές IoT.

Επίσης χρησιμοποιώντας μηχανική μάθηση, προτείνεται μια νέα αρχιτεκτονική που ονομάζεται MEC-AI HetFL (Multi-Edge Clustered and Edge AI Heterogeneous Federated Learning) [39]. Με αυτή την τεχνική μηχανικής μάθησης οι συσκευές εκπαιδεύονται τοπικά χωρίς να στέλνουν τα ακατέργαστα δεδομένα στον κεντρικό διακομιστή. Κάθε συσκευή μοιράζεται μόνο τα αποτελέσματα (παραμέτρους μοντέλου). Έτσι προστατεύεται η ιδιωτικότητα των χρηστών. Η μέθοδος MEC-AI HetFL ομαδοποιεί τις συσκευές IoT σε clusters ανάλογα με τις δυνατότητές τους. Επιλέγει δυναμικά τους πιο σημαντικούς κόμβους για συμμετοχή στην εκπαίδευση και συντονίζει τη συνεργασία με τη βοήθεια Edge AI. Σε σχέση με παρόμοιες υπάρχουσες λύσεις όπως EdgeFed, FedSA, FedMP και H-DDPG, βελτιώνει έως 5 φορές την απόδοση σε κατανεμημένα κι ετερογενή περιβάλλοντα.

Ένα διαφορετικό πλαίσιο εργασίας το MLADFC (Machine Learning Analytics-based Data Classification Framework) αποτελείται από δύο στάδια, το πρώτο είναι ένα μοντέλο παλινδρόμησης (regression model) κι ένα Υβριδικό Αλγόριθμο KNN περιορισμένων πόρων (HRCKNN) [40]. Το πρώτο στάδιο εκτελείται σε επίπεδο συσκευής. Αν αποφασισθεί ότι δεν αξίζει να τα επεξεργαστεί τοπικά, τα δεδομένα προωθούνται στο Edge για να εφαρμοστεί το δεύτερο στάδιο, ο αλγόριθμος HRCKNN. Το μοντέλο αυτό έδειξε αποδεδειγμένη αποτελεσματικότητα και μείωσε τη κατανάλωση ενέργειας του δικτύου, οδηγώντας σε παράταση της διάρκειας μπαταρίας των συνδεδεμένων κόμβων.

Γενικά έχουν προταθεί πολλοί αλγόριθμοι μηχανικής μάθησης και υπολογιστικής νοημοσύνης όπως K-means, DBSCAN, Agglomerative Clustering [41]. Η έρευνα εισάγει ένα καινοτόμο πλαίσιο που χρησιμοποιεί κάποιον αλγόριθμο ετερογενούς ομαδοποίησης από τους παραπάνω και στη συνέχεια διαχειρίζεται δυναμικά τους πόρους με χρήση Deep Q-Networks (DQN) και Deep Reinforcement Learning (DRL). Με τον τρόπο αυτό δημιουργήθηκε ένα έξυπνο, αυτόματο σύστημα που «μαθαίνει» πώς να διαχειρίζεται τα δεδομένα και τους πόρους των IoT συσκευών, με εφαρμογές στην γεωργία και τα έξυπνα δίκτυα αισθητήρων.

Μεγάλη έμφαση δίνεται στην επαναχρησιμοποίηση εργασιών. Πολλές φορές οι εργασίες σε συσκευές IoT εκπαιδεύουν ξεχωριστά μοντέλα ML, αλλά με παρόμοιες εργασίες. Προτείνεται λοιπόν ο εντοπισμός επαναχρησιμοποιήσιμων εργασιών και η βαθμονόμηση μοντέλων προκειμένου να βελτιωθεί η ικανότητά τους προβλέψεων όταν επαναχρησιμοποιούνται [42]. Προτείνεται ένα πλαίσιο δύο φάσεων Κατανεμημένης Πολυ-εργασιακής Μηχανικής Μάθησης (DMtL – Distributed Multitask Learning). Στη πρώτη φάση παρόμοιες εργασίες ομαδοποιούνται βάσει των χαρακτηριστικών επιδόσεών των τοπικά εκπαιδευμένων μοντέλων χρησιμοποιώντας Καμπύλες Μάθησης (LC – Learning Curves). Στη συνέχεια αξιοποιείται η μέθοδος PLC-driven DMtL κι ενισχύεται η απόδοση των υποψήφιων

επαναχρησιμοποιήσιμων μοντέλων ανά ομάδα εργασιών, εξασφαλίζοντας υψηλής ποιότητας analytics.

2.4 Μετακίνηση δεδομένων κι υπηρεσιών

Ίσως το μεγαλύτερο πρόβλημα όσον αφορά τις constrained συσκευές, είναι η μετακίνηση δεδομένων και υπηρεσιών από τις constrained devices στο Edge ή ακόμη και στο Cloud.

Μια πρόταση που έχει διερευνηθεί αναλυτικά είναι η υλοποίηση του Edge Mesh (EM) [43], μιας αρχιτεκτονικής όπου ένα οικοσύστημα αυτόνομων και συνεργαζόμενων κόμβων μπορεί να κατανοεί τη κατάσταση του περιβάλλοντος και να εξυπηρετεί αποδοτικά χρήστες κι εφαρμογές. Πέντε είναι οι βασικοί άξονες γύρω από τους οποίους δομείται η εργασία αυτή. Το υλικό που απαιτείται για να υποστηρίξει την επεξεργασία στο Edge, τα μοντέλα μηχανικής μάθησης και βελτιστοποίησης (ML/DL) που θα παίρνουν αποφάσεις τοπικά, χωρίς εξάρτηση από το κεντρικό Cloud, τη διαχείριση των δεδομένων στο Edge που περιλαμβάνει τεχνικές αποθήκευσης, caching και κατανεμημένες βάσεις δεδομένων, τη διαχείριση εργασιών και την μετανάστευση υπηρεσιών όταν αλλάζουν οι συνθήκες δικτύου ή φορτίου και τέλος την αυτόνομη διαχείριση κόμβων.

Άλλη τεχνολογία που προτείνεται είναι το MEC (Multi-Access Edge Computing) για δίκτυα 5ης γενιάς (5G) χρησιμοποιώντας το Kubernetes ως σύστημα με containers [44]. Προτείνεται για περιπτώσεις που η constrained συσκευή είναι εν κινήσει, όπως πχ όταν ο χρήστης κινείται με αυτοκίνητο. Σε τέτοιες περιπτώσεις έχουμε service migration. Η μέθοδος αυτή επιτυγχάνει χαμηλή καθυστέρηση. Μια άλλη ιδέα που προσπαθεί να λύσει το ίδιο πρόβλημα χρησιμοποιεί Ψηφιακό Δίδυμο (Digital Twin)[45]. Κάθε φυσικός κόμβος Edge αναπαρίσταται από ένα εικονικό αντίγραφό του στο Cloud το οποίο προσομοιώνει σε πραγματικό χρόνο τη κατάσταση του κόμβου (εύρος ζώνης, ενέργεια, φορτίο) και προβλέπει που πρέπει να γίνει η μετανάστευση. Το σύστημα καλείται να λύσει ένα πρόβλημα βελτιστοποίησης για το ποιος κόμβος

προσφέρει καλύτερη ισορροπία μεταξύ χαμηλής καθυστέρησης, χαμηλής κατανάλωσης ενέργειας κι ελάχιστου κόστους μεταφοράς.

Σε αντίθεση με on-demand στρατηγικές, προτείνεται ένας προληπτικός μηχανισμός βασισμένος σε Ενισχυτική Μάθηση (Reinforcement Learning) χρησιμοποιώντας Learning Automata (LA) [46], ένα μαθηματικό εργαλείο που επιλέγει βέλτιστες ενέργειες μαθαίνοντας από το περιβάλλον. Το LA γνωστό κι ως model learning ή grammatical inference είναι ένα μικρό υποσύνολο της θεωρητικής επιστήμης των υπολογιστών με χρήσιμες πρακτικές εφαρμογές ειδικά στους τομείς αυτόματης μοντελοποίησης και δοκιμών που βασίζονται στη μάθηση. Ένας «έξυπνος» αλγόριθμος μαθαίνει τις συνήθειες κίνησης των χρηστών και μετακινεί τα microservices (μικρά ανεξάρτητα κομμάτια μιας εφαρμογής όπως σύνδεση, πληρωμή, προφίλ κ.τ.λ.) εκεί που προβλέπει ότι ο χρήστης θα πάει, πριν αυτός φτάσει κι αφού έχει ήδη φύγει. Επιπλέον, σε περίπτωση stateful microservices, δηλαδή υπηρεσίες που διατηρούν κατάσταση (μνήμη, δεδομένα, sessions κ.α.) προτείνεται ένα μαθηματικό μοντέλο με αλγόριθμο ανθεκτικής μετεγκατάστασης και σύστημα βασισμένο σε Kubernetes [47].

Τα κύρια μοντέλα που χρησιμοποιούνται σε κατανεμημένους MEC servers χωρίζονται σε κατηγορίες [48]. Ως προς τον τρόπο μεταφοράς εργασιών, έχουμε Binary offloading όπου η εργασία εκτελείται ολόκληρη είτε τοπικά είτε στον MEC server ή Partial Offloading όπου η εργασία διαιρείται σε τμήματα (Microservices) όπως είδαμε παραπάνω. Μελετώνται συστήματα ενός χρήστη όπου το βασικό ερώτημα είναι: «πότε αξίζει να μεταφορτώσω στον server"; Επίσης συστήματα πολλών χρηστών όπου επιβάλλεται έξυπνη κατανομή πόρων. Αναλύονται κεντρικοποιημένοι αλγόριθμοι, κατανεμημένοι αλγόριθμοι βασισμένοι σε θεωρία παιγνίων (Nash Equilibrium) και συνεργατική επεξεργασία μεταξύ χρηστών. Τέλος μελετώνται ετερογενή συστήματα πολλών servers ως προς την επιλογή server, συνεργασία μεταξύ τους και το πώς μεταφέρεται η εργασία στο νέο κοντινό server. Τυπικά

σενάρια χρήσης που αναφέρονται είναι η ανάλυση βίντεο για αναγνώριση προσώπων/πινακίδων, Επαυξημένη Πραγματικότητα (AR), συνδεδεμένα οχήματα κ.τ.λ.

Όπως γίνεται αντιληπτό, η Μηχανική Μάθηση (Machine Learning - ML) χρησιμοποιείται πλέον όλο και περισσότερο για διαδικασίες όπως λήψη αποφάσεων, προβλέψεις, συσταδοποίηση και κατηγοριοποίηση κ.α. Σε συνδυασμό με μαθηματικά μοντέλα μπορεί να δημιουργήσει «έξυπνους» κόμβους Edge που προαποφασίζουν για θέματα όπως η αποθήκευση δεδομένων τοπικά ή μετανάστευση αυτών στο Cloud. Πχ μοντέλα Μηχανικής Μάθησης σε συνδυασμό με στατιστική ανάλυση χρονοσειρών [49]. Ο μηχανισμός αυτός λαμβάνει υπόψη το πλαίσιο (context) όπως χρησιμότητα δεδομένων, ηλικία δεδομένων, διαθέσιμος αποθηκευτικός χώρος, διαδικτυακές συνθήκες (συνδεσιμότητα κ.τ.λ.) και τις ανάγκες των εφαρμογών. Υπολογίζει την «αξία» κάθε συνόλου δεδομένων και ανάλογα τη τιμή, αποφασίζει να τα κρατήσει τοπικά, είτε να μεταναστεύσουν στο Cloud/for είτε να διαγραφούν αν είναι πλέον παρωχημένα ή άχρηστα. Μια διαφορετική προσέγγιση χρησιμοποιεί Μηχανική Μάθηση και συγκεκριμένα ταξινόμηση με διανύσματα στήριξης (One-Class SVM) για να μάθει εσωτερικά πρότυπα πρόσβασης στα δεδομένα και να προσδιορίζει ποια είναι σχετικά, κι ένα φίλτρο βασισμένο σε Ασαφή Λογική (Fuzzy Logic) το οποίο εκτιμά ποια δεδομένα θα απαιτηθούν στο μέλλον [50]. Με αυτό τον τρόπο η επιλογή δεδομένων δεν γίνεται τυχαία αλλά με επίγνωση του έργου (task-awareness), δηλαδή τα αιτήματα-έργα καθορίζουν τι αξίζει να κρατηθεί.

Μια άλλη παράμετρος πολύ σημαντική είναι ο χρόνος που θα γίνει η ανταλλαγή δεδομένων μεταξύ κόμβων του Edge, δηλαδή το «πότε» θα συμβεί αυτό. Προτείνεται ένας μηχανισμός ανταλλαγής συνοπτικών αναπαραστάσεων της εξαγόμενης γνώσης μεταξύ κόμβων του Edge Computing και όχι ακατέργαστων δεδομένων [51]. Η απόφαση για το «πότε» βασίζεται σε ένα στοχαστικό μοντέλο βελτιστοποίησης βασισμένο στη Θεωρία Βέλτιστης Διακοπής (Theory of Optimal Stopping). Η

θεωρία αυτή απαντά σε ερωτήματα όπως πότε είναι η καλύτερη στιγμή να λάβω μια απόφαση ώστε να μεγιστοποιήσω το κέρδος ή να ελαχιστοποιήσω το κόστος.

Σύνοψη δεδομένων μπορεί να είναι μια συμπιεσμένη αναπαράσταση ενός dataset, όπως πχ ιστόγραμμα, wavelet ή στατιστική κατανομή. Έτσι κάθε κόμβος ανακοινώνει ποια δεδομένα έχει κι όχι το σύνολό τους. Με αυτό τον τρόπο εξοικονομείται εύρος ζώνης δικτύου [52]. Η απόφαση πότε μια σύνοψη αλλάζει σημαντικά και πρέπει να ειδοποιηθούν οι γείτονες μπορεί να γίνει με χρήση μοντέλου βαθιάς μάθησης (Deep Learning) το οποίο μαθαίνει να λειτουργεί προκαταβολικά, δηλαδή εκτιμά εκ των προτέρων αν οι γείτονες θα επωφεληθούν από την ενημέρωση και δε περιμένει να ερωτηθεί πρώτα. Όμως, όπως πολλές αποφάσεις δεν είναι πάντα «άσπρο – μαύρο» αλλά κινούνται σε ένα συνεχές διάστημα [0,1]. Προτείνεται λοιπόν ένα σχήμα βασισμένο στην Ασαφή Λογική (Fuzzy Logic System) [53] το οποίο θα αποφασίζει πότε να σταλεί μια σύνοψη στους γείτονες. Η Ασαφής Λογική είναι κατάλληλη σε αυτές τις περιπτώσεις αφού η διαφορά μιας παλιάς και νέας σύνοψης μπορεί να είναι μικρή και να μη δικαιολογείται η εκ νέου αποστολή της.

Η μετακίνηση δεδομένων κι υπηρεσιών έχει και μια άλλη διάσταση που μπορεί να μελετηθεί. Αντί να μετακινούμε δεδομένα ή συνόψεις εκ των υστέρων, να τα μετακινούμε αμέσως τη στιγμή που δημιουργούνται με τέτοιο τρόπο που να δημιουργούνται datasets με ισχυρή «ακεραιότητα» [54] δηλαδή σταθερή κατανομή (πχ μέση τιμή, διάμεσος, εκτίμηση της κατανομής). Η διαδικασία αυτή δημιουργεί συστήματα ανθεκτικά σε σφάλματα. Τιμές outliers μετακινούνται απευθείας στο Cloud ενώ οι υπόλοιπες αποθηκεύονται σε πιθανοθεωρητικά επιλεγμένα datasets. Επομένως πριν αποφασίσουμε πότε και πως θα επεξεργαστούμε τα δεδομένα, αποφασίζουμε πως θα οργανωθούν αυτά τοπικά.

Σε επίπεδο υπηρεσιών, έχουμε επίσης πολλές και διαφορετικές προσεγγίσεις για το πως, πότε και ποιες εργασίες (Tasks) θα μετακινηθούν και που. Η απόφαση offloading δεν βασίζεται μόνο από άποψη πόρων (CPU, latency) αλλά και από την δημοτικότητα μιας εργασίας (πόσο δημοφιλής είναι μέσα σε ένα χρονικό παράθυρο) και τη διαθεσιμότητα των δεδομένων (ποσοστό επικάλυψης - overlapping) [55]. Κι εδώ η χρήση Ασαφούς Λογικής δύο σταδίων βοηθά στην απόφαση τοπικής εκτέλεσης, offloading σε peer κόμβο ή offloading στο Cloud. Η Ασαφής Λογική θεωρείται ιδανική εδώ γιατί τα δυο κριτήρια που προαναφέραμε (δημοτικότητα και διαθεσιμότητα), μπορεί να λειτουργούν σε αντίθετες κατευθύνσεις (π.χ. υψηλή δημοτικότητα και χαμηλή διαθεσιμότητα ή το αντίθετο) οπότε το σύστημα πρέπει να βρει μια ισορροπία μεταξύ αυτών.

Έχουμε ήδη αναφερθεί πολλάκις σε Services και Tasks. Οι υπηρεσίες (services) είναι ενότητες επεξεργασίας που μπορούν να εφαρμοστούν στα δεδομένα, για την εκτέλεση των εργασιών (Tasks) [56]. Η βασική ιδέα είναι ο κόμβος να παρακολουθεί τη ζήτηση για κάθε υπηρεσία μέσα σε ένα χρονικό παράθυρο, με στατιστικά μοντέλα εκτιμά πως θα εξελιχθεί η ζήτησή της και αποφασίζει εκ των προτέρων αν μια υπηρεσία πρέπει να διατηρηθεί τοπικά ή να μεταναστεύσει. Μέσω της στατιστικής αυτής τεχνικής, ο κόμβος εκτιμά την αναμενόμενη μελλοντική ζήτηση για κάθε υπηρεσία και ποιες είναι οι υπηρεσίες υψηλής προτεραιότητας ώστε να διατηρηθούν τοπικά. Μια διαφορετική έννοια που εισάγουν οι συγγραφείς είναι η έννοια του ερωτήματος (Query) [57]. Αντί να προβλέπεται που να εκτελεστεί μια εργασία, τώρα ένας μηχανισμός προβλέπει σε ποιον κόμβο να κατανεμηθεί ένα ερώτημα ώστε να απαντηθεί αποδοτικά. Εισάγεται η έννοια του Query Controller (QC), οντότητες που βρίσκονται στο Cloud ή το Fog και είναι υπεύθυνες να κατανέμουν τα εισερχόμενα queries στους κατάλληλους κόμβους. Με κριτήρια την υπολογιστική πολυπλοκότητα του query, τη σχετικότητά του με το dataset του κόμβου και το τρέχον φορτίο του κόμβου αποφασίζεται ποιος κόμβος του

Edge θα το εκτελέσει. Ο βασικός μηχανισμός απόφασης είναι ένα σύστημα Internal Type-2 Fuzzy Logic System (T2FLS) λόγω του ότι υπάρχουν δύο επίπεδα αβεβαιότητας (περιβάλλοντος και σχεδιασμού). Μια επιπλέον είσοδος του T2FLS προέρχεται από ένα μοντέλο διανυσμάτων στήριξης (SVM). Ο συνδυασμός Ασαφούς Λογικής και Μηχανικής Μάθησης δίνει στο σύστημα ικανότητες γενίκευσης ακόμη και σε περιπτώσεις χώρων υψηλών διαστάσεων. Ουσιαστικά η διαφορά των δύο παραπάνω μεθόδων είναι ότι η δεύτερη εισάγει πιο εξελιγμένη μορφή Fuzzy Logic ενώ η πρώτη εστιάζει σε νεοεισερχόμενα χαρακτηριστικά και πιο ελαφριά υλοποίηση.

ΚΕΦΑΛΑΙΟ 3 Προτεινόμενο Μοντέλο

3.1 Γεωμετρική αναπαράσταση μοντέλου

Συνήθως όταν αναφερόμαστε σε εργασίες και φίλτρα που σχετίζονται με τα αποθηκευμένα δεδομένα ενός κόμβου, αναφερόμαστε σε διαστήματα τιμών (από έως), και τα αντιμετωπίζουμε με αυτό το τρόπο, δηλαδή ως διαστήματα, υποσύνολα του $\mathbb{R}$.

Στη παρούσα εργασία προτείνεται μια διαφορετική προσέγγιση του προβλήματος ευφυούς διαχείρισης των φίλτρων. Αντί για απλά διαστήματα τιμών, προσεγγίζουμε εργασίες και φίλτρα μέσω ισομορφισμού, ως σημεία του επιπέδου της γεωμετρίας στο δισδιάστατο χώρο. Η μέθοδος αυτή δίνει μια πιο απτή απεικόνιση τόσο των αιτημάτων που καλούμαστε να διαχειριστούμε όσο και των φίλτρων, καθώς και των διαδικασιών που λαμβάνουν χώρα σε ένα τέτοιο περιβάλλον. Ο ισομορφισμός αυτός μας δίνει τη δυνατότητα να επεξεργαστούμε τα αιτήματα που αφικνούνται στον κόμβο αλλά και τα φίλτρα (πότε κι αν θα γίνει ανανέωση, πως θα υπολογιστεί το νέο φίλτρο κ.τ.λ.) χρησιμοποιώντας ιδιότητες της επίπεδης γεωμετρίας και των πολυμετάβλητων κατανομών. Η συγκεκριμένη μέθοδος αντιμετώπισης του προβλήματος διαχείρισης των φίλτρων μας δίνει νέες δυνατότητες στη λήψη αποφάσεων, ανάδειξη κρίσιμων πληροφοριών και βελτιστοποίηση στο διπλό σκοπό της διαχείρισης τους: ***Κάλυψη*** μελλοντικών αιτημάτων που φθάνουν στον κόμβο αλλά και σωστή ***διαχείριση*** των δεδομένων του κόμβου.

3.2 Εργασίες και φίλτρα

Υποθέτουμε ότι υπάρχουν N αυτόνομοι κόμβοι $N=\{n_1, n_2, \dots, n_N\}$ στο Edge, οι οποίοι είναι συνδεδεμένοι και δημιουργούν μία τοπολογία δικτύου. Στη παρούσα εργασία θα επικεντρωθούμε σε έναν κόμβο έστω $n_k, 1 \le k \le \mathbb{N}$. Ο κόμβος διατηρεί ένα τοπικό dataset $D_k = \{x_i\}_{i=1}^{|D_k|}$ με διανύσματα δεδομένων διάστασης d, $x = [x_1, x_2, \dots, x_d]^T \in \mathbb{R}^d$. Μια εργασία (task) $T = \{[t_1^l, t_1^h], [t_2^l, t_2^h], \dots, [t_d^l, t_d^h]\} \in \mathbb{R}^{2d}$ που ζητείται από μια εφαρμογή η

οποία επικοινωνεί με τον κόμβο n_k, απεικονίζεται ως διάνυσμα 2d διαστάσεων με d διαστήματα επιλεγμένων τιμών πάνω στα δεδομένα του dataset. Οι εκθέτες l και h, αντιπροσωπεύουν τα κάτω και άνω όρια αντίστοιχα για κάθε διάσταση των δεδομένων. Επίσης ένα φίλτρο (filter) $f = \{[f_1^l, f_1^h], [f_2^l, f_2^h], \dots, [f_d^l, f_d^h]\} \in \mathbb{R}^{2d}$ ίδιας δομής με τις εργασίες είναι αυτό που θέλουμε να παρακολουθούμε και να αποφασίζουμε ˙για το αν και πότε θα ανανεωθεί. Σύμφωνα με τις τιμές του φίλτρου, αποφασίζεται ποια δεδομένα μέσα από το dataset θα παραμείνουν τοπικά για περεταίρω επεξεργασία και ποια θα μεταναστεύσουν στο Cloud. Επίσης όσες εργασίες δεν καλύπτονται από το φίλτρο επίσης θα μεταναστεύουν σε γειτονικούς κόμβους του δικτύου ή αλλού (task offloading).

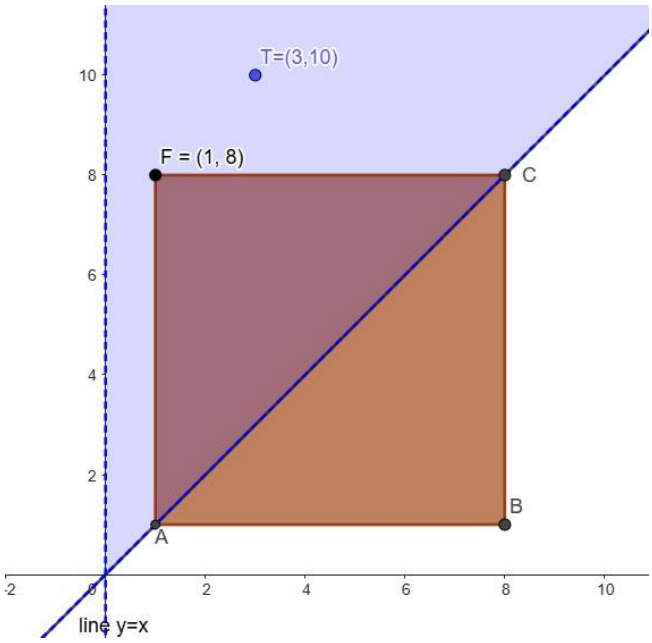
Χωρίς βλάβη της γενικότητας, θα εστιάσουμε σε μια διάσταση έστω $j\epsilon\{1,2,\dots,d\}$, δηλαδή την $j_{οστή}$ διάσταση του T ήτοι $T_j = [t_j^l, t_j^h], t_j^l \le t_j^h$ και του F ήτοι $F_j = [f_j^l, f_j^h], f_j^l \le f_j^h$. Αυτό που επιθυμούμε είναι να ανακτήσουμε όλα τα δεδομένα στον κόμβο n_k τα οποία βρίσκονται ανάμεσα σε $t_j^l \, και \, t_j^h$, δηλαδή $\{x_i \in D_k : 1 \le i \le |D_k|, t_j^l \le x_i \le t_j^h\}$. Για παράδειγμα μια εργασία με αίτημα $T_j = [3,10]$ ζητά όλες τις τιμές του dataset που βρίσκονται εντός αυτού του διαστήματος ώστε να τύχουν επεξεργασίας, π.χ. επιλογή δεδομένων για εκπαίδευση ενός μοντέλου Μηχανικής Μάθησης. Επομένως πρέπει να παρακολουθούμε τις αλλαγές που συμβαίνουν με τέτοιο τρόπο ώστε η $j_{οστή}$ διάσταση του φίλτρου, $f_j = [f_j^l, f_j^h]$ να ανανεώνεται ή να παραμένει αναλλοίωτη.

3.3 Αναπαράσταση εργασιών και φίλτρων στο μοντέλο

Από τη στιγμή που η $j_{οστή}$ διάσταση των εργασιών και του φίλτρου είναι δισδιάστατα διανύσματα $[a, b]$ με $a \le b$ ή αλλιώς διαστήματα της μορφής $[a, b]$, μπορούμε να εφαρμόσουμε έναν ισομορφισμό ώστε να αντιστοιχήσουμε αυτά τα διαστήματα σε σημεία του δισδιάστατου επιπέδου της Ευκλείδειας (Επίπεδης) Γεωμετρίας. Επομένως:

$$[a,b], a \leq b \leftrightarrow (a,b), a <= b$$

Λόγω του περιορισμού $0 \leq a \leq b$ όλα τα σημεία που δημιουργούνται, αν σχεδιαστούν σε ορθοκανονικό σύστημα αξόνων, θα βρίσκονται στο 1ο τεταρτημόριο και συγκεκριμένα όχι κάτω από την ευθεία $y = x$. Τη περιοχή αυτή θα την ονομάσουμε A_2 από και στο εξής. Για παράδειγμα, η $j_{οστή}$ συντεταγμένη της εργασίας T, $T_j = [3,10]$ θα αντιστοιχιστεί στο σημείο T (3,10) και η αντίστοιχη συντεταγμένη του φίλτρου F, $f_j = [1,8]$ θα αντιστοιχίζεται στο σημείο F(1,8).



Εικόνα 4: Αναπαράσταση εργασιών (Tasks) και φίλτρων (Filters) στο επίπεδο

Με αυτό τον τρόπο το ορθογώνιο τρίγωνο με σημεία $F(f_j^l, f_j^h), A(f_j^l, f_j^l) and\ C(f_j^h, f_j^h)$ (κόκκινη περιοχή στην εικόνα) αντιπροσωπεύει την περιοχή μέσα στην οποία αν βρεθεί σημείο που αντιστοιχεί στην εργασία T_j, αυτή καλύπτεται πλήρως από το φίλτρο F_j. Δηλαδή θα ισχύει $f_j^l \leq t_j^l \leq t_j^h \leq f_j^h$ αφού για οποιοδήποτε σημείο (a,b) εντός του ορθογωνίου τριγώνου θα ισχύει $f_j^l \leq a \leq b \leq f_j^h$. Το τρίγωνο αυτό θα το ονομάσουμε A_1 εφεξής. Επιπλέον, κάθε σημείο πάνω από την οριζόντια πλευρά του ορθογωνίου τριγώνου και κάθε σημείο αριστερά από την κατακόρυφη πλευρά του τριγώνου, θα αντιστοιχεί σε εργασίες που καλύπτονται μερικώς από το φίλτρο. Μερική κάλυψη συμβαίνει όταν τα διαστήματα T_j, F_j έχουν κοινά στοιχεία, η τομή τους δεν είναι το

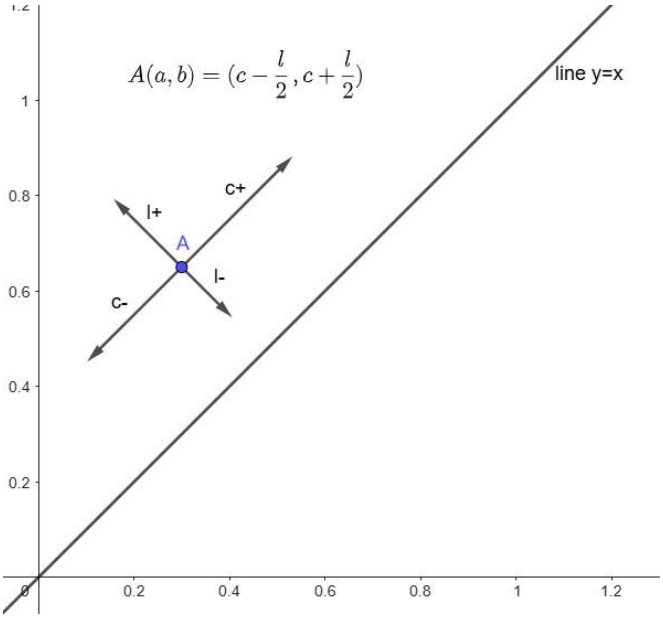
κενό σύνολο. Πχ, αν $F_j = [1,8]$ και $T_j = [3,10]$ τότε $T_j \cap F_j = [3,10] \cap [1,8] = [3,8] \neq \emptyset$. Επομένως μια εργασία που καταφθάνει στο κόμβο αντιστοιχεί σε μια από τις παρακάτω περιπτώσεις:

- *Καλύπτεται πλήρως από το φίλτρο*. Αυτό σημαίνει ότι όλα τα δεδομένα που είναι αποθηκευμένα τοπικά ικανοποιούν το ερώτημα που τίθεται από αυτήν την εργασία
- *Καλύπτεται μερικώς από το φίλτρο*. Αυτό σημαίνει ότι κάποιο ποσοστό των δεδομένων που υπάρχουν αποθηκευμένα τοπικά, ικανοποιούν αυτήν την εργασία
- *Δεν καλύπτεται καθόλου από το φίλτρο*. Σε αυτή τη περίπτωση η εργασία πρέπει να μεταναστεύσει σε γειτονικό κόμβο ο οποίος πιθανών να έχει δεδομένα σχετικά με αυτήν.(task offloading)

Είναι εύκολα αντιληπτό ότι η σωστή επιλογή φίλτρου, άρα και η διατήρηση τοπικά δεδομένων που το ικανοποιούν, είναι πολύ σημαντική τόσο για τα δεδομένα (αποφυγή data migration) όσο και για τις εργασίες που καταφθάνουν στον κόμβο (αποφυγή task offloading).

Ένα διάστημα $[a,b]$ με $a \leq b$, έχει κέντρο $c = \frac{a+b}{2}$ και ακτίνα $l = \frac{b-a}{2}$ ώστε το πλάτος του να είναι ίσο με $2 * l$. Το αντίστοιχο σημείου του επιπέδου θα είναι το $(a,b) = \left(c - \frac{l}{2}, c + \frac{l}{2}\right)$ αφού $a = c - \frac{l}{2}, b = c + \frac{l}{2}$. Αυτό σημαίνει ότι κάθε αλλαγή των άκρων ενός διαστήματος οδηγεί σε αλλαγή του κέντρου και της ακτίνας του διαστήματος κι αντιστρόφως. Κάθε μεταβολή του c, οδηγεί σε μετατόπιση του αντίστοιχου σημείου του επιπέδου, παράλληλα στην ευθεία $y = x$. Συγκεκριμένα , όταν το c αυξάνεται, το σημείο απομακρύνεται από τους άξονες $x'x, y'y$ (διάνυσμα $\overrightarrow{c^+}$ στην εικόνα) κι όταν μειώνεται, το σημείο πλησιάζει τους άξονες $x'x, y'y$ (διάνυσμα $\overrightarrow{c^-}$ στην εικόνα). Επίσης, η μεταβολή του l αντιστοιχεί σε μετακίνηση του σημείου κάθετα προς την ευθεία $y = x$ (η απόσταση του σημείου $T(a,b)$ από την ευθεία $\varepsilon: y = x$ είναι $d(T,\varepsilon) = \frac{l}{\sqrt{2}}$). Άρα όταν το l αυξάνεται το σημείο απομακρύνεται από την ευθεία $y = x$ (διάνυσμα

$\overrightarrow{l^-}$ στο σχήμα) κι όταν το l μειώνεται το σημείο πλησιάζει την ευθεία (διάνυσμα $\overrightarrow{l^+}$ στο επόμενο σχήμα).



Ένα αίτημα για την εργασία T_j σε έναν κόμβο n_k ουσιαστικά αντιπροσωπεύει τα όρια μέσα στα οποία βρίσκονται τα δεδομένα ενός υποσυνόλου του dataset D_k, τα οποία θα προσπελάσει κι επεξεργαστεί η εργασία αυτή. Δηλαδή, η εργασία T_j αιτείται να προσπελάσει όλα τα δεδομένα του κόμβου n_k που ικανοποιούν τη σχέση:

$$S_k = \{x_i \in D_k : t_j^l \leq x_i \leq t_j^h, 1 \leq i \leq |D_k|\} \subseteq D_k$$

Παράλληλα, το φίλτρο F_j αντιπροσωπεύει το εύρος τιμών που κυμαίνονται τα δεδομένα $x_i \in D_k, 1 \leq i \leq |D_k|$ που είναι αποθηκευμένα στο κόμβο. Σκοπός μας είναι λοιπόν να βρεθεί η κατάλληλη στιγμή ανανέωσής του και κατά συνέπεια τα δεδομένα που είναι εκτός ορίων του νέου φίλτρου, να μετακινηθούν σε γειτονικούς κόμβους ή στο Cloud. Στη παρούσα εργασία δε θα ασχοληθούμε με την μετανάστευση δεδομένων κι εργασιών. Ο συνδυασμός ευφυούς διαχείρισης των φίλτρων και μέθοδοι μετανάστευσης δεδομένων κι υπηρεσιών αποτελούν μελλοντική εργασία.

3.4 Μαθηματικές έννοιες που θα χρησιμοποιηθούν

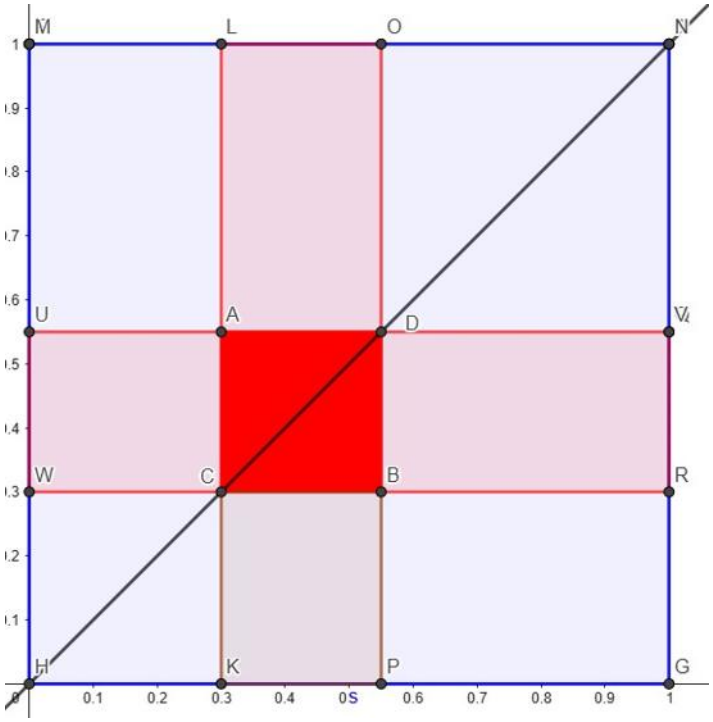
3.4.1 Εισαγωγικά θεωρήματα

Ξεκινάμε την ανάπτυξη του μαθηματικού υπόβαθρου της παρούσης εργασίας με ένα λήμμα:

Λήμμα 1: θεωρώντας ότι οι εργασίες που καταφθάνουν σε έναν κόμβο αντιστοιχούν σε σημεία του δισδιάστατου επιπέδου τα οποία ακολουθούν την Διμετάβλητη Ομοιόμορφη κατανομή (Bivariate Uniform Distribution) $T \sim U(A)$, $A = περιοχή\ δειγματοληψίας$, τότε η πιθανότητα η εργασία να καλύπτεται πλήρως από το υπάρχον φίλτρο f_j είναι ίση με $p_f = s^2$ και η πιθανότητα να καλύπτεται μερικώς είναι $p_p = 2 * s - 2 * s^2$, $όπου\ s = f_j^h - f_j^l$.

Απόδειξη: Στην Εικόνα 5, το κόκκινο τετράγωνο ACBD έχει άνω αριστερή κορυφή $A(f_j^l, f_j^h)$ και κάτω δεξιά κορυφή $B(f_j^h, f_j^l)$ οπότε οι πλευρές του τετραγώνου είναι ίσες με $s = f_j^h - f_j^l$. Όλα τα σημεία της εφικτής περιοχής ανήκουν στο τετράγωνο με κορυφές $H(0,0)$, $M(0,1)$, $G(1,0)$, $N(1,1)$ δηλαδή στο μπλε τετράγωνο του σχήματος το οποίο έχει εμβαδόν 1. Επίσης όλα τα σημεία που βρίσκονται πάνω ή κάτω από τις οριζόντιες πλευρές του τετραγώνου και αυτά που βρίσκονται αριστερά η δεξιά από τις κατακόρυφες πλευρές του, δημιουργούν δύο ορθογώνια UWRV και KPOL διαστάσεων $s \times 1$. Το εμβαδόν αυτών των δυο σχημάτων είναι αθροιστικά $2 \times 1 \times s = 2s$, Λόγω του ότι και τα δυο περιλαμβάνουν τα σημεία του τετραγώνου ACBD αφαιρώντας δυο φορές το εμβαδόν του, προκύπτει ότι το εμβαδόν των ορθογωνίων πάνω, κάτω, αριστερά και δεξιά από το τετράγωνο είναι ίσο με $2 * s - 2 * s^2 = 2s(1 - s)$. Λόγω του ότι το εμβαδόν της εφικτής περιοχής είναι ίσο με ένα και η πιθανότητα ένα τυχαίο σημείο που ακολουθεί την U(A) να βρίσκεται σε υποπεριοχή εμβαδού E είναι ίση με $p_e = \frac{|E|}{|A|} = \frac{|E|}{1} = |E|$ τελικά $p_f = s^2$ η πιθανότητα η εργασία να καλύπτεται πλήρως και $p_p = 2 * s - 2 * s^2$ η πιθανότητα να καλύπτεται μερικώς. ■

Να αναφέρουμε ότι λόγω συμμετρίας του σχήματος ως προς την ευθεία $y = x$, οι πιθανότητες παραμένουν ίδιες αν η εφικτή περιοχή είναι το άνω τρίγωνο HMN και η περιοχή πλήρους κάλυψης μιας εργασίας είναι το ορθογώνιο τρίγωνο ACD(Εικόνα 5).



Εικόνα 5: Περιοχές πλήρους κάλυψης (κόκκινο) και μερικής κάλυψης (ροζ) εργασιών από το φίλτρο

3.4.2 Διμετάβλητη Ομοιόμορφη Κατανομή

Υποθέτουμε ότι έχουμε μια συνεκτική και συνεχή περιοχή A του $\mathbb{R}^2$ της οποίας τα σημεία ακολουθούν την Διμετάβλητη Ομοιόμορφη Κατανομή U(A). Η μέση τιμή και διακυμάνσεις θα είναι

$\boldsymbol{\mu} = [mx, my] = \left[\frac{\iint_A x dA}{\iint_A dA}, \frac{\iint_A y dA}{\iint_A dA}\right]$ το οποίο είναι το κεντροειδές της περιοχής A, και $\sigma_x^2 = \frac{\iint_A (x-mx)^2 dA}{\iint_A dA}$, $\sigma_y^2 = \frac{\iint_A (y-my)^2 dA}{\iint_A dA}$. Συγκεκριμένα αν η περιοχή A είναι τρίγωνο με κορυφές $A(x_A, y_A), B(x_B, y_B), C(x_C, y_C)$ τότε το κεντροειδές είναι το κέντρο βάρους του τριγώνου $[mx, my] = \left[\frac{x_A+x_B+x_C}{3}, \frac{y_A+y_B+y_C}{3}\right]$. Επομένως το ορθογώνιο τρίγωνο ACD (περιοχή A_1 από και στο εξής) της εικόνας 5, με $A(f_j^l, f_j^h)$ ή $A(l, h)$ για απλοποίηση, θα έχει υπόλοιπες κορυφές $C(l, l), D(h, h)$ οπότε:

- Το εμβαδόν της περιοχής θα είναι $A_1 = \frac{(h-l)^2}{2}$
- Κεντροειδές $\mu(mx, my)$ ή $\mu\left(\frac{2l+h}{3}, \frac{l+2h}{3}\right)$
- Διακυμάνσεις ίσες μεταξύ τους $\sigma_x^2 = \sigma_y^2 = \frac{(h-l)^2}{18}$
- Συνδιακύμανση $cov(x,y) = \frac{(h-l)^2}{36}$ οπότε ο πίνακας συνδιακύμανσης είναι:
- $\Sigma = \frac{(h-l)^2}{36}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$

Για την περιοχή $A_c = A_2 \backslash A_1$ που περιλαμβάνει τα σημεία εντός της εφικτής περιοχής (τρίγωνο HMN στην Εικόνα 5) που βρίσκονται εκτός του τριγώνου ACD (περιοχή A_1), επειδή δεν είναι εύκολος ο υπολογισμός των τύπων λόγω σχήματος (ορθογώνιο τρίγωνο που έχει αφαιρεθεί μικρότερο ορθογώνιο τρίγωνο), θα χρησιμοποιήσουμε εναλλακτικά:

$\mu_C = \frac{A_2\mu_2 - A_1\mu_1}{A_2 - A_1}$ και $\Sigma_c = \frac{A_2(\Sigma_2 + \mu_2\mu_2^T) - A_1(\Sigma_1 + \mu_1\mu_1^T)}{A_2 - A_1} - \mu_C\mu_C^T$

όπου $\left(\mu_2 = \left(\frac{1}{3}, \frac{2}{3}\right), \Sigma_2 = \frac{1}{36}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}\right)$.

3.4.3 Διμετάβλητη Κανονική κατανομή

Ένα δισδιάστατο διάνυσμα $X = (X_1, X_2)^T$ ακολουθεί την Δισδιάστατη Κανονική Κατανομή αν κάθε συντεταγμένη X_1, X_2 κι οποιοσδήποτε γραμμικός συνδυασμός $\alpha X_1 + \beta X_2$ ακολουθούν Κανονική κατανομή [58], [59]. Το διάνυσμα του μέσου είναι $\boldsymbol{\mu} = (\mu_1, \mu_2)^T$, ο πίνακας συνδιακύμανσης είναι $\Sigma = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}$ και η ΣΠΠ: $f(x_1, x_2) = \frac{1}{2\pi|\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}$. Από τη στιγμή που γνωρίζουμε τις τιμές των $\boldsymbol{\mu}, \Sigma$ μπορούμε να υπολογίσουμε την απόσταση Mahalanobis [60] κάθε σημείου του επιπέδου σύμφωνα με τον τύπο $d = \sqrt{(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}$. Η απόσταση αυτή είναι η πολυμετάβλητη γενίκευση του τετραγώνου της τυποποιημένης κανονικής κατανομής $z = \frac{x-\mu}{\sigma}$, δηλαδή μετράει πόσες

τυπικές αποκλίσεις «μακριά» βρίσκεται το σημείο από τον μέσο $\boldsymbol{\mu} = (m_x, m_y)$. Για συγκεκριμένες τιμές του d, προκύπτει μια εξίσωση έλλειψης (πχ, για $d = 1\sigma$ παίρνουμε την 1σ-έλλειψη η οποία θα έχει μικρό και μεγάλο άξονα πάνω στις διευθύνσεις των ιδιοδιανυσμάτων του πίνακα Σ. Ο μικρός άξονας αντιστοιχεί στο ιδιοδιάνυσμα με τη μικρότερη ιδιοτιμή κι ο μεγάλος άξονας σε αυτό με την μεγαλύτερη ιδιοτιμή. Αν οι ιδιοτιμές είναι ίσες (μια ιδιοτιμή με αλγεβρική πολλαπλότητα 2) τότε η έλλειψη εκφυλίζεται σε κύκλο ,με κέντρο το σημείο **μ**.

3.4.4 Sequential Probability Ratio Test – SPRT

Το SPRT είναι ένα ακολουθιακό τεστ υποθέσεων, μέθοδος που χρησιμοποιείται για να αποφασίσουμε μεταξύ δυο υποθέσεων καθώς λαμβάνουμε συνεχώς (σε πραγματικό χρόνο) δεδομένα, παρά όταν γνωρίζουμε τα δεδομένα εκ των προτέρων [62], [63]. Συγκρίνει το λόγο πιθανοφάνειας των δεδομένων που καταφθάνουν μεταξύ δυο υποθέσεων $\Lambda_n = \frac{Likelihood\ under\ H_1}{Likelihood\ under\ H_0}$ και σταματάει όταν ο λόγος αυτός ξεπεράσει κάποια όρια που έχουμε ορίσει. Είναι χρήσιμη διαδικασία όταν έχουμε δυο πιθανές κατανομές που μπορεί να ανήκουν τα δεδομένα που θα λάβουμε, δεν ξέρουμε σε ποια από τις δυο ανήκουν και θέλουμε να απαντήσουμε όσο το δυνατόν νωρίτερα. Όταν λοιπόν ο λόγος $\Lambda_n \geq A$ δεχόμαστε την H_1 ενώ όταν $\Lambda_n \leq B$ δεχόμαστε την H_0. Τα Α,Β εξαρτώνται από τα σφάλματα τύπου Ι και ΙΙ και συγκεκριμένα

- **Σφάλμα τύπου** Ι: $\alpha = P(reject\ H_0 | H_0)$
- **Σφάλμα τύπου ΙΙ**: $\beta = P(accept\ H_0 | H_1)$

Οπότε $A = \frac{1-\beta}{\alpha}$, $B = \frac{\beta}{1-\alpha}$. Τα α, β πρέπει να αποφασισθούν πριν την έναρξη της διαδικασίας ώστε να ορισθούν τα όρια Α και Β αντίστοιχα. Για ευκολία θα χρησιμοποιούμε τον λογάριθμο του λόγου πιθανοφάνειας, $l_n = log\Lambda_n = \sum_{i=1}^{n} log\frac{f_1(Xi)}{f_2(X_i)}$ οπότε θα συνεχίζουμε να λαμβάνουμε δεδομένα όσο ισχύει $b = \log(B) < l_n < a = \log(A)$.

3.5 Μεθοδολογία ανανέωσης φίλτρων

Υποθέτουμε ότι σε έναν αυτόνομο κόμβο υπάρχουν αποθηκευμένα δεδομένα και ένα φίλτρο όπως περιγράφεται παραπάνω, ως ένα διάστημα τιμών μέσα στο οποίο βρίσκονται όλα τα δεδομένα του κόμβου. Εργασίες καταφθάνουν στον κόμβο κι αιτούνται επεξεργασία των δεδομένων. Μη γνωρίζοντας εκ των προτέρων τι κατανομή ακολουθούν αυτές οι εργασίες, θεωρούμε δυο Διμετάβλητες Κανονικές κατανομές, μια για τα σημεία της περιοχής A_1 και μια της περιοχής $A_c = A_2 \backslash A_1$ όπως αυτές ορίσθηκαν νωρίτερα. Σκοπός μας είναι να αποφασίσουμε σε ποια από τις δυο κατανομές ανήκουν οι εργασίες που θα καταφθάσουν ώστε να αποφασίσουμε αν και πότε θα ανανεώσουμε το φίλτρο.

Θεωρώντας ότι ήδη υπάρχει ενεργό κάποιο φίλτρο, ορίζονται οι περιοχές A_1, A_2, A_c (A_1: τρίγωνο ACD στην Εικόνα 5 και A_2:τρίγωνο που περιέχει όλες τις εργασίες που κατέφθασαν πρόσφατα). Υπολογίζονται οι μέσες τιμές και οι διακυμάνσεις των περιοχών A_1, A_c μέσω της Διμετάβλητης Ομοιόμορφης κατανομής. Για κάθε νέα εργασία που καταφθάνει, εξετάζουμε αν είναι outlier και για τις δυο κατανομές. Στην περίπτωση αυτή, θεωρούμε ότι η εργασία μετακινείται σε άλλο γειτονικό κόμβο και δε λαμβάνεται υπόψιν στην διαδικασία. Για τις υπόλοιπες εργασίες, με χρήση του SPRT, υπολογίζουμε το λογάριθμο του λόγου πιθανοφάνειας κι αποφασίζουμε με βάση αυτόν.

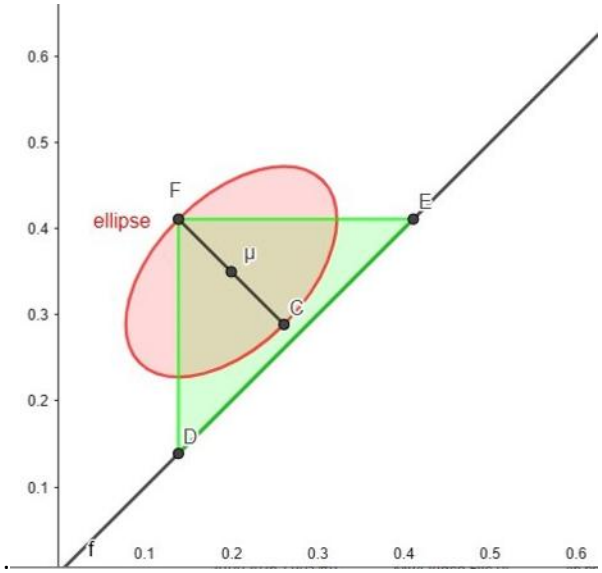
Αν $l_n < b$ αποδεχόμαστε την υπόθεση H_1, ότι δηλαδή οι εργασίες που καταφθάνουν ανήκουν στην Διμετάβλητη Κανονική κατανομή της περιοχής A_c οπότε πρέπει να ανανεωθεί το φίλτρο διότι δεν καλύπτει τις εργασίες αυτές. Υπολογίζουμε τη μέση τιμή και την τυπική απόκλιση των εργασιών που έχουν ήδη καταφθάσει χρησιμοποιώντας μια παράμετρο φθοράς γ. Οπότε η μέση τιμή των εργασιών που έχουν φθάσει είναι $\boldsymbol{\mu} = \frac{\sum_{i=1}^{100} \gamma^{100-i} * \boldsymbol{T_i}}{\sum_{i=1}^{100} \boldsymbol{\gamma}^{\mathbf{100-i}}}$ όπου $T_i = [y^l, y^h]_i$ Ο παράγοντας γ δείχνει πόσο μεγάλη σημασία δίνουμε στις παλαιότερες εργασίες σε σχέση με

τις νεοαφιχθείσες. Για τιμές του γ κοντά στο μηδέν, δίνουμε έμφαση περισσότερο στις νεότερες εργασίες (είναι ένας αλγόριθμος «μυωπικός» όπως λέμε) ενώ για τιμές κοντά στο ένα, δίνουμε σχεδόν την ίδια αξία σε όλες τις εργασίες που έχουν καταφθάσει (είναι ένας «διορατικός» παράγοντας).

Αν $l_n > a$ τότε αποδεχόμαστε την υπόθεση H_0, ότι δηλαδή οι εργασίες που κατέφθασαν ανήκουν σε Δισδιάστατη κανονική κατανομή εντός της περιοχής A_1 που δημιουργήθηκε από το ήδη υπάρχον φίλτρο. Για να καλύψουμε πιθανές μικρές διαφοροποιήσεις στις εργασίες που ήρθαν πρόσφατα κάνουμε εκ νέου υπολογισμό του φίλτρου χρησιμοποιώντας όμως μόνο τις εργασίες που μείωσαν τον λόγο SPRT. Δηλαδή χρησιμοποιούμε μόνο εργασίες που ενίσχυσαν την υπόθεση H_0 . Υπολογίζουμε τη μέση τιμή και τυπική απόκλιση αυτών των σημείων χωρίς τη χρήση του παράγοντα φθοράς, (μπορούμε ισοδύναμα να υποθέσουμε ότι γ=1).

Αν $a \leq l_n \leq b$ μετά από κάποιο πλήθος εργασιών, αποφασίζουμε ότι δεν μπορεί να ληφθεί απόφαση από τον SPRT οπότε διατηρούμε το φίλτρο και ξεκινάει η διαδικασία από την αρχή Σε όλες τις εκτελέσεις του πειράματος, θεωρήθηκε μέγιστο παράθυρο εργασιών ίσο με 100.

Οποτεδήποτε αποφασίσουμε να ανανεώσουμε το φίλτρο, υπολογίζουμε μέση τιμή $\boldsymbol{\mu} = \begin{bmatrix} m_x \\ m_y \end{bmatrix}$ και τυπική απόκλιση $\boldsymbol{s} = \begin{bmatrix} s_x \\ \boldsymbol{s_y} \end{bmatrix}$ των επιλεγμένων εργασιών ανάλογα με τη περίπτωση κι όπως έχει εξηγηθεί παραπάνω. Για να υπολογίσουμε το νέο φίλτρο, έχοντας υποθέσει ότι οι εργασίες που έχουν καταφθάσει αλλά κι οι επόμενες που θα έρθουν, ακολουθούν Διμετάβλητη Κανονική κατανομή $N_2(\boldsymbol{\mu}, \boldsymbol{s^2})$ **.** Το φίλτρο τοποθετείται στο άκρο του μικρού άξονα της σ-έλλειψης που βρίσκεται πιο μακριά από την ευθεία$y = x$ (Εικόνα 6). Η παράμετρος σ είναι μια θετική παράμετρος και οι σ-έλλειψη περιέχει όλα τα σημεία που έχουν απόσταση Mahalanobis μικρότερη ή ίση με σ από το σημείο **μ**



Εικόνα 6: Φίλτρο F, τρίγωνο πλήρους κάλυψης και σ-έλλειψη

Σκοπός μας είναι το ανανεωμένο φίλτρο να καλύψει πλήρως ή μερικώς τις νέες εργασίες που πρόκειται να καταφθάσουν στον κόμβο, κατ' ελάχιστον αυτές οι οποίες ως σημεία θα βρίσκονται εντός της σ-έλλειψης.

Μετά την ανανέωση του φίλτρου, επαναφέρουμε τον λόγο SPRT, επαναϋπολογίζουμε τις περιοχές A_1, A_2, A_c κι επαναλαμβάνεται η όλη διαδικασία. Η περιοχή A_2 επαναπροσδιορίζεται με τέτοιο τρόπο ώστε να καλύπτει όλες τις εργασίες που έφτασαν μεταξύ δυο διαδοχικών ανανεώσεων του φίλτρου, περίοδο που θα την ονομάσουμε από δω και μπρος εποχή. Για να συμβεί αυτό θεωρούμε το ορθογώνιο τρίγωνο με κορυφή της ορθής γωνίας του το σημείο $A(\alpha, \beta)$ όπου $a = \min\{t_j^l\}_i, i = 1,2,..,r$ και $b = \min\{t_j^h\}_i, i = 1,2,..,r$ και r είναι το πλήθος των εργασιών εντός της τελευταίας εποχής.

ΚΕΦΑΛΑΙΟ 4 Πειραματική Αποτίμηση

4.1 Περιγραφή του Πειράματος

4.1.1 Δημιουργία dataset

Στα πειράματα που πραγματοποιήθηκαν, χρησιμοποιήθηκαν συνθετικές εργασίες και δύο dataset, ένα συνθετικό D1 κι ένα με πραγματικές τιμές D2. Οι εργασίες αποτελούνταν από διανύσματα του $\mathbb{R}^2$ και συγκεκριμένα $T = [t_l.t_h] \in [0,1] \times [0,1]$ με $t_l < t_h$. Για την κατασκευή τους χρησιμοποιήθηκαν τρεις διαφορετικές μεθόδους που περιγράφονται παρακάτω, κάθε μια προσομοιώνει μια διαφορετική κατάσταση. Επίσης τα συνθετικά δεδομένα $D1 = \{x_i \sim U(0,1), i = 1,2,3,\dots,|D1|\}$ δημιουργήθηκαν ακολουθώντας την Ομοιόμορφη κατανομή U(0,1). Το πραγματικό σύνολο δεδομένων D2 είναι το dataset ποιότητας αέρα [62] που έχει 9358 δεδομένα πολλών διαστάσεων που έχουν καταγραφεί ανά ώρα. Τα δεδομένα αυτά έχουν ληφθεί από συσκευές πολλαπλών αισθητήρων χημικής ανάλυσης αέρα σε μια μολυσμένη περιοχή, σε δρόμο Ιταλικής πόλης. Τα δεδομένα κανονικοποιήθηκαν σε διάστημα [0,1].

4.1.2 Δημιουργία συνθετικών εργασιών

4.1.2.1 Uniform Task Generator (UTG)

Μια εργασία κατανέμεται ομοιόμορφα στον δειγματικό χώρο. Ειδικά το κέντρο του διαστήματος $T = [t_l.t_h]$ θα ακολουθεί την U(0,1), δηλαδή $c \sim U(0,1)$ και το μήκος του διαστήματος θα ακολουθεί την U(0.1,0.3), δηλαδή $l \sim U(0.1,0.3)$. Τα όρια για τα t_l, t_h θα είναι σε αυτή τη περίπτωση $t_l = \max\left\{0, c - \frac{l}{2}\right\}$ και $t_h = \min\{1, c + \frac{l}{2}\}$. Η μέθοδος UTG δημιουργεί εργασίες που καλύπτουν όλη την εφικτή περιοχή , αποφεύγοντας τον αποκλεισμό συγκεκριμένων υποπεριοχών τιμών.

4.1.2.2 Data-driven Task Generator (DTG)

Μια εργασία δημιουργείται ακολουθώντας την κατανομή των δεδομένων που είναι αποθηκευμένα στον κόμβο. Το κέντρο του διαστήματος $T = [t_l . t_h]$ ορίζεται χρησιμοποιώντας μια μη παραμετρική εκτίμηση ΣΠΠ με χρήση πυρήνων (Incremental Kernel Density Estimator – KDE). Το μήκος του διαστήματος $l \sim U(0.5\sigma, 2\sigma)$ όπου σ είναι η τυπική απόκλιση των δεδομένων. Τα άκρα t_l, t_h υπολογίζονται όπως στην προηγούμενη περίπτωση.

4.1.2.3 Adaptive Task Generator (ATG)

Μια εργασία T_i , δημιουργείται χρησιμοποιώντας τις τιμές του κέντρου και του πλάτους της προηγούμενης εργασίας T_{i-1} . Συγκεκριμένα $c_i = c_{i-1} + u, όπου\ u \sim U(-0.2, 0.2)$ και $l_i = i_{i-1} + v\ όπου\ v \sim U(-0.1, 0.1)$. Αυτό σημαίνει ότι κάθε εργασία δεν είναι τυχαία αλλά έχει κάποια σχέση με την προηγούμενή της. Η φιλοσοφία αυτής της μεθόδου είναι ότι τις περισσότερες φορές, διαδοχικές εργασίες ζητούν δεδομένα που βρίσκονται σε κοινά διαστήματα τιμών (η τομή των δύο διαστημάτων δεν είναι το κενό σύνολο). Και πάλι τα άκρα του διαστήματος $T = [t_l . t_h]$ υπολογίζονται με τον ίδιο τρόπο όπως στην UTG.

Σε όλες τις παραπάνω περιπτώσεις κρατάμε τις τελευταίες 100 εργασίες, αποθηκευμένες σε κυκλική λίστα (circular buffer) ώστε να χρησιμοποιηθούν για τον υπολογισμό κι ανανέωση του φίλτρου σύμφωνα με τις μεθόδους που έχουν εξηγηθεί παραπάνω.

4.1.3 Εκτέλεση πειράματος

Έγιναν δοκιμές προσομοίωσης της διαδικασίας με κώδικα σε Python, δοκιμάζοντας διάφορες τιμές για παραμέτρους του προβλήματος. Συγκεκριμένα για την παράμετρο σ της σ-έλλειψης, έγιναν δοκιμές με sigma: $s \in \{1,2,3\}$, για την παράμετρο φθοράς γ έγιναν δοκιμές με gamma: $\gamma \in \{0.7, 0.9, 0.99\}$ και τέλος η τιμή του z-score για τις εργασίες που ήταν outliers πήρε τιμές threshold: $thr \in \{2,3,5\}$. Ο κώδικας έτρεξε και για τους 27 συνδυασμούς τιμών αυτών των παραμέτρων ώστε να μελετηθεί η

επίδραση του καθενός στις μετρικές που υιοθετήθηκαν για την αποτίμηση του αποτελέσματος.

Μικρότερες τιμές της παραμέτρου σ, οδηγούν σε μικρότερο διάστημα τιμών (σε πλάτος) για το φίλτρο, που σημαίνει ότι γινόμαστε πιο «αυστηροί» στον υπολογισμό του φίλτρου με ότι αυτό συνεπάγεται για τα δεδομένα που θα διατηρηθούν στον κόμβο. Μεγαλύτερες τιμές του σ οδηγούν σε μεγαλύτερο διάστημα σε πλάτος για το φίλτρο άρα και εύρος τιμών στα δεδομένα. Έχουμε δηλαδή μια πιο «χαλαρή» πολιτική στην επιλογή του φίλτρου. Η παράμετρος γ όπως έχει ήδη αναφερθεί, μεταβάλλει το ποσοστό συμμετοχής παλαιότερων εργασιών στον υπολογισμό του νέου φίλτρου. Ειδικά τιμές πολύ κοντά στη μονάδα δηλώνουν σχεδόν πλήρη συμμετοχή όλων των εργασιών στους υπολογισμούς. Τέλος η τιμή για το threshold καθορίζει ποιες εργασίες θα θεωρούνται outliers για τις δυο Δισδιάστατες κανονικές κατανομές που συμμετέχουν στον SPRT. Όσο μεγαλύτερη είναι η τιμή του threshold, τόσο μικρότερη είναι η πιθανότητα μια εργασία να θεωρηθεί outlier και να απορριφθεί από τη διαδικασία.

Για κάθε έναν από τους 27 παραπάνω συνδυασμούς τιμών των παραμέτρων και για κάθε μέθοδο δημιουργίας εργασιών (UTD, DTG, ATG) ο κώδικας έτρεξε μέχρι να συμπληρωθούν 100 ανανεώσεις φίλτρων, δηλαδή 100 εποχές όπως τις ονομάσαμε νωρίτερα. Καταγράφηκε για κάθε εποχή το παράθυρο εργασιών δηλαδή το πλήθος αιτημάτων που επεξεργάστηκε η διαδικασία.

4.2 Μετρικές πειράματος

Σκοπός του προσδιορισμού του χρόνου που θα γίνει η ανανέωση ενός φίλτρου αλλά και του τρόπου υπολογισμού του νέου φίλτρου, ήταν να μεγιστοποιηθεί το ποσοστό μελλοντικών εργασιών που θα καταφθάσουν στο κόμβο και τα οποία θα καλύπτονται πλήρως ή μερικώς από αυτό (είναι μια task – awareness διαδικασία). Αυτό συνεπάγεται ότι υπάρχουν δεδομένα αποθηκευμένα στο κόμβο που ικανοποιούν τα φίλτρα αυτά σε μεγάλο ποσοστό.

Δεδομένου ενός σημείου $F(f^l, f^h)$, και του αντίστοιχου ορθογωνίου τριγώνου με κορυφές $F(f^l, f^h), A(f^l, f^l)$ και $B(f^h, f^h)$, η περιοχή του 1ου τεταρτημόριου πάνω από την ευθεία $y = x$, που «καλύπτεται» πλήρως ή μερικώς είναι $E = \frac{s^2 - 2s}{2}$ όπου $s = f^h - f^l$, το πλάτος του διαστήματος του φίλτρου F. Η εφικτή περιοχή μέσα στην οποία μπορούν να βρίσκονται εργασίες και φίλτρα είναι το τριγωνικό κομμάτι πάνω από την ευθεία $y = x$ των σημείων $[a, b], 0 \leq a < b \leq 1$. Η περιοχή αυτή έχει εμβαδόν $E_t = \frac{1}{2}$. Έχει αποδειχθεί ότι για ένα σημείο, τυχαία επιλεγμένο στην περιοχή E_t , η πιθανότητα να βρίσκεται μέσα στη περιοχή E, είναι $p = \frac{E}{E_t} = d^2 - 2d = p_f + p_p$. Υποθέτουμε λοιπόν ότι η τυχαία επιλογή ενός σημείου της εφικτής περιοχής $A(a, b), 0 \leq a < b \leq 1$ με $a \sim U(0,1)$ και $b \sim U(a, 1)$ να δίνει σημείο του E, είναι δοκιμή Bernoulli με πιθανότητα επιτυχίας ίση με p. Θα συγκρίνουμε αυτή την πιθανότητα με το ποσοστό των k επόμενων εργασιών που καλύπτονται πλήρως ή μερικώς από το υπάρχον φίλτρο. Για όλα τα παραπάνω εισάγονται οι επόμενες μετρικές.

4.2.1 Information Efficiency – IE

Όπως έχει αναφερθεί, μια νέα εργασία που καταφθάνει στο κόμβο, θεωρείται ότι καλύπτεται πλήρως ή μερικώς αν η τομή των δύο συνόλων, εργασίας και φίλτρου, είναι διάστημα διάφορο του κενού. Στο δισδιάστατο επίπεδο, αυτό σημαίνει ότι το σημείο T που αντιστοιχεί στην εργασία βρίσκεται εντός του ορθογωνίου τριγώνου (πλήρης κάλυψη) ή σε μια από τις δυο περιοχές πάνω από την οριζόντια πλευρά του τριγώνου και την αριστερή κατακόρυφη πλευρά του τριγώνου (μερική κάλυψη). Επίσης, αποδείξαμε ότι η πιθανότητα ένα τυχαία επιλεγμένο σημείο να αντιστοιχεί σε εργασία που καλύπτεται πλήρως ή μερικώς είναι $p_f = s^2$, $p_p = s^2 - 2 * s$ με συνολική πιθανότητα κάλυψης (πλήρους ή μερικής) $p = p_f + p_p = 2 * s^2 - 2 * s$. Ορίζουμε την μετρική Information Efficiency – IE :

$$IE = \frac{\textit{ποσοστό μελλοντικών εργασιών που καλύπτονται από το φίλτρο}}{2 * s^2 - 2 * s}$$

η οποία μετράει πόσο καλύτερα η μέθοδος που μελετάμε αποδίδει σε σχέση με τυχαία επιλογή φίλτρων που ακολουθεί τη Διμετάβλητη Ομοιόμορφη κατανομή.

- $IE \approx 1 \rightarrow \textit{βασική κάλυψη}$
- $IE > 1 \rightarrow task\_awareness$
- $IE \gg 1 \rightarrow \textit{υψηλό } task - awareness$

Η IE είναι μια σημαντική μετρική διότι μετράει την αποδοτικότητα ανά μονάδα για το επιλεγμένο φίλτρο κι όχι απλά το ποσοστό κάλυψης. Ένα φίλτρο με πλάτος διαστήματος κοντά στο εύρος τιμών που κινούνται εργασίες και δεδομένα (πχ όταν $f^h - f^l \cong 1$) καλύπτει πλήρως σχεδόν όλες τις εργασίες και διατηρεί όλα τα δεδομένα στο κόμβο. Ένα τέτοιο φίλτρο όμως δεν είναι αποδοτικό αφού δεν αφήνει τίποτα εκτός. Στον πίνακα Table IIa, παρατίθενται τα ολικά ποσοστά κάλυψης (πλήρους και μερικής) για τις τρεις μεθόδους παραγωγής εργασιών και για όλους τους συνδυασμούς παραμέτρων sigma, gamma, threshold. Τα ποσοστά αυτά είναι οι αριθμητές της IE. Παρατηρούμε ότι η UTG έχει συνήθως τα χαμηλότερα ποσοστά λόγω της τυχαιότητάς της. Η μέθοδος DTG (KDE) έχει τα καλύτερα αποτελέσματα λόγω του ότι είναι μέθοδος οδηγούμενη από τα δεδομένα. Ο καλύτερος μέσος όρος και των τριών μεθόδων μαζί ήταν 89,33% κάλυψη μελλοντικών εργασιών για $\sigma = 3,\ \gamma = 0.9,\ thr = 2.$

Table IIa – Future tasks covered (%) by filter

σ	γ	Thr	Uniform	KDE	Adaptive	Best
1	0,7	2	63.4%	91.7%	86.1%	KDE
1	0,7	3	50.4%	90.4%	**93.7%**	Adaptive
1	0,7	5	35.6%	69.2%	92.0%	Adaptive
1	0,9	2	76.2%	94.5%	75.8%	KDE
1	0,9	3	50.0%	98.0%	79.7%	KDE
1	0,9	5	39.3%	72.5%	80.3%	Adaptive
1	0,99	2	70.7%	99.0%	82.8%	KDE
1	0,99	3	45.2%	95.9%	68.5%	KDE

1	0,99	5	41.1%	93.6%	64.6%	KDE
2	0,7	2	78.2%	94.5%	83.8%	KDE
2	0,7	3	42.7%	84.4%	90.1%	Adaptive
2	0,7	5	44.6%	78.2%	91.1%	Adaptive
2	0,9	2	84.7%	96.9%	78.8%	KDE
2	0,9	3	52.3%	98.9%	89.7%	KDE
2	0,9	5	50.4%	88.4%	85.0%	KDE
2	0,99	2	85.2%	94.5%	64.0%	KDE
2	0,99	3	59.7%	97.1%	68.9%	KDE
2	0,99	5	49.4%	90.7%	60.6%	KDE
3	0,7	2	82.7%	96.4%	82.9%	Adaptive
3	0,7	3	51.0%	97.3%	91.4%	Adaptive
3	0,7	5	45.9%	79.6%	91.1%	KDE
3	0,9	2	**88.6%**	96.7%	82.7%	KDE
3	0,9	3	65.4%	**99.5%**	89.8%	Adaptive
3	0,9	5	58.3%	93.9%	91.3%	KDE
3	0,99	2	88.2%	98.4%	76.0%	KDE
3	0,99	3	70.3%	**99.5%**	75.4%	KDE
3	0,99	5	61.9%	95.8%	75.3%	KDE
Overall best (avg across methods): sigma=3, gamma=0.9, thr=2 : avg 89.33%						

Η (κατά μέσο όρο) ελάχιστη, μέγιστη και μέση τιμή της IE για τους 27 συνδυασμούς παραμέτρων φαίνονται στον πίνακα Table IIb.

Table IIb – Statistics of Mean IE			
Method	**Min**	**Average**	**Max**
Uniform	0.89	1.52	3.71
KDE	1.31	2.46	3.56
Adaptive	0.81	1.37	1.80

Παρατηρούμε ότι η μέση τιμή της IE σε όλες τις μεθόδους είναι πάνω από τη μονάδα, που σημαίνει ότι έχουμε υψηλή επίγνωση μελλοντικών εργασιών, ειδικά στην DTG. Η ανανέωση φίλτρων με τη χρήση SPRT δείχνει να έχει πολύ καλή συμπεριφορά σχετικά με μελλοντικές εργασίες.

Λόγω του ότι η μετρική αυτή δεν ασχολείται με το είδος των δεδομένων αλλά μόνο με τις εργασίες που καταφθάνουν, οι τιμές για UTG,ATG είναι παραπλήσιες στα σύνολα δεδομένων D1 και D2. Για την

μέθοδο DTG τρέξαμε τον κώδικα σε κάθε μια από τις παραμέτρους του dataset D2 και προέκυψαν οι τιμές του πίνακα Table IIc.

Table IIc – Statistics of Mean IE for D2 using DTG			
Parameter	**Min**	**Average**	**Max**
PT08.S1(CO)	0.55	5.27	12.35
C6H6(GT)	0.40	6.92	16.92
PT08.S2(NMHC)	0.73	5.73	12.89
PT08.S3(NOx)	0.71	7.45	17.49
PT08.S4(NO2)	0.38	5.61	13.29
PT08.S5(O3)	0.64	5.07	12.33
T	0.36	4.29	11.40
RH	0.29	3.52	8.18
AH	0.33	4.33	11.71

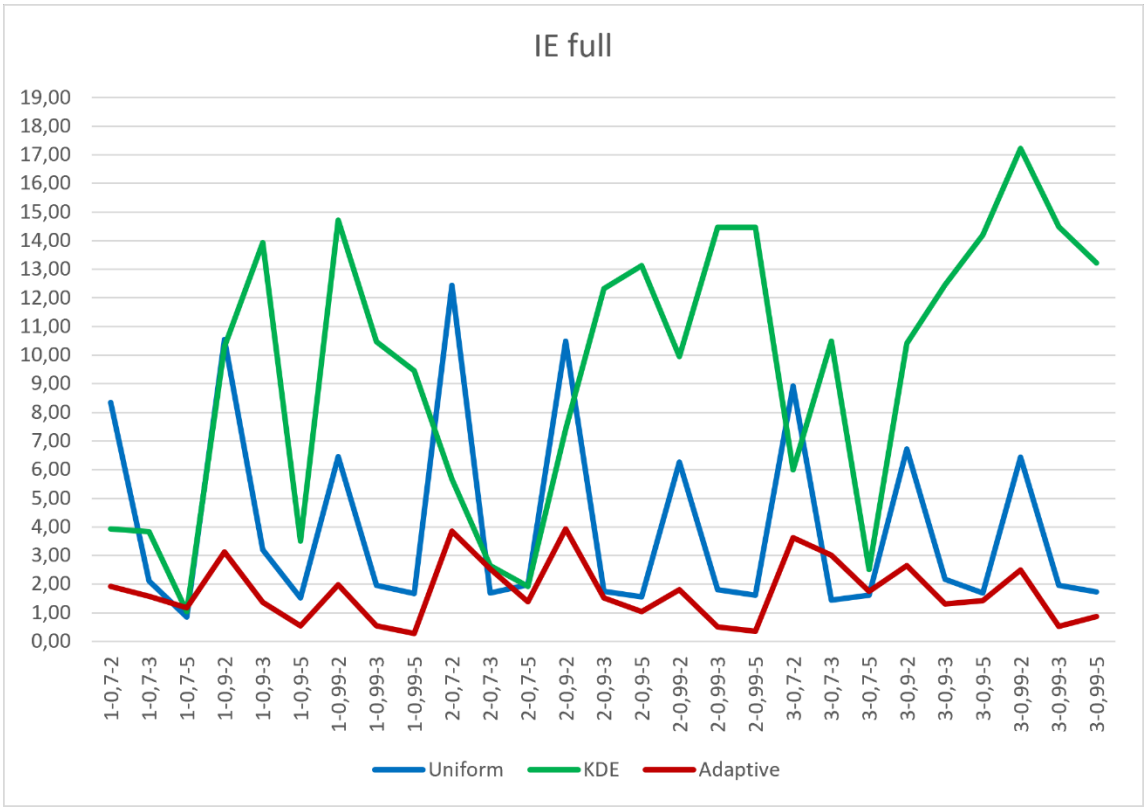
Συμπεραίνουμε ότι σε πραγματικά δεδομένα που έχουν σχέση μεταξύ τους και κάποια εξάρτηση (όπως χρονοσειρές) η μέθοδος που προτείνεται δίνει πολύ καλή προβλεψιμότητα. Μια τιμή του IE, πχ $IE = 8$ δηλώνει ότι το τρέχον φίλτρο που δημιουργείται με την προτεινόμενη μέθοδο κάλυψε 8 φορές περισσότερες εργασίες από την αναμενόμενη τιμή (η αναμενόμενη τιμή μιας Bernoulli διαδικασίας $B(n,p)$ είναι ίση με p).

Αν διαχωρίσουμε τις εργασίες που καλύπτονται πλήρως με αυτές που καλύπτονται μερικώς μπορούμε να δημιουργήσουμε δύο (υπο)μετρικές IE_f και IE_p αντιστοίχως. Συγκεκριμένα θα έχουμε

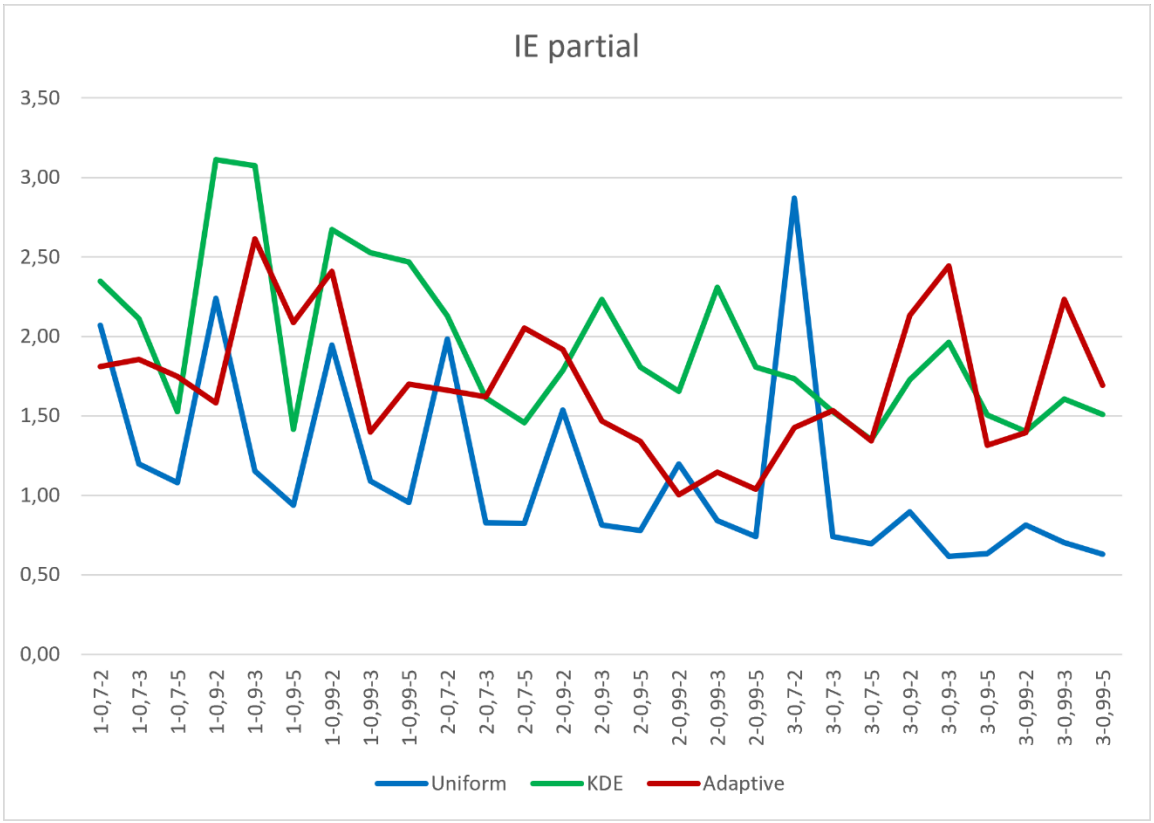
$$IE_f = \frac{\textit{ποσοστό μελλοντικών εργασιών που καλύπτονται πλήρως}}{s^2}$$

$$IE_p = \frac{\textit{ποσοστό μελλοντικών εργασιών που καλύπτονται μερικώς}}{2s - 2s^2}$$

Οι παρανομαστές αντιστοιχούν στις πιθανότητες που έχουμε αποδείξει για τις περιοχές πλήρους και μερικής κάλυψης. Οι Εικόνες 7α – 7β παρουσιάζουν τις μέσες τιμές των δυο δεικτών για όλους τους συνδυασμούς των παραμέτρων sigma, gamma, thr.



Εικόνα 7α: IE full



Εικόνα 7β: IE partial

Οι μέσες τιμές για όλους τους συνδυασμούς παραμέτρων του δείκτη IE_f ήταν 4,03 (UTG), 9,41 (DTG) και 1,74 (ATG), ενώ οι αντίστοιχες τιμές για τον δείκτη IE_p ήταν 1,14 (UTG), 1,94 (DTG) και 1,70 (ATG).

4.2.2 Data-Filter Equilibrium Error – DFEE

Καθώς θα γίνει η ανανέωση του φίλτρου, δεδομένα αποθηκευμένα στο κόμβο που δεν καλύπτονται από τα όρια που θέτει αυτό, πρέπει να μεταναστεύσουν. Θέλοντας να γνωρίζουμε το ποσοστό δεδομένων που θα μεταναστεύσει εισάγουμε την μετρική DFEE. $DFEE = 1 - \frac{\text{κάλυψη δεδομένων κατά την ανανέωση φίλτρου}}{|D_k|}$, όπου D_k είναι το σύνολο των αποθηκευμένων δεδομένων στο κόμβο. Η DFEE μετράει ποσοστό δεδομένων που δεν καλύπτονται από το νέο φίλτρο.

- $DFEE \to 0$ σημαίνει τέλεια ευθυγράμμιση δεδομένων και φίλτρου
- $DFEE \to 1$ το νέο φίλτρο απορρίπτει τα περισσότερα δεδομένα

Είναι το πλησιέστερο ανάλογο προς τους λόγους αποδοτικότητας κάλυψης που μελετήθηκαν στην επιλεκτικότητα δεδομένων με επίγνωση εργασιών στο Edge [49], [50].

Table III – DFEE

σ	γ	threshold	Uniform	KDE	Adaptive
1	0,7	2	0,5429	0,3738	0,3847
1	0,7	3	0,3883	0,3034	0,3764
1	0,7	5	0,5053	0,4754	0,3854
1	0,9	2	0,3424	0,2368	0,3437
1	0,9	3	0,1081	0,1308	0,2956
1	0,9	5	0,0804	0,2608	0,2275
1	0,99	2	0,2183	0,1266	0,2585
1	0,99	3	0,0355	0,1067	0,0970
1	0,99	5	0,0249	0,0993	0,0764
2	0,7	2	0,4571	0,3370	0,3576
2	0,7	3	0,4082	0,3481	0,3457
2	0,7	5	0,3595	0,4001	0,3967
2	0,9	2	0,2655	0,2191	0,3683

2	0,9	3	0,0778	0,1001	0,2846
2	0,9	5	0,0647	0,1375	0,2763
2	0,99	2	0,0955	0,1941	0,2233
2	0,99	3	0,0159	0,0917	0,0786
2	0,99	5	0,0129	0,1071	0,0913
3	0,7	2	0,3574	0,2711	0,3885
3	0,7	3	0,3053	0,1556	0,3252
3	0,7	5	0,3286	0,3433	0,3759
3	0,9	2	0,2651	0,1516	0,3175
3	0,9	3	0,0359	0,0774	0,2910
3	0,9	5	0,0382	0,0944	0,2742
3	0,99	2	0,1101	0,0831	0,1999
3	0,99	3	0,0158	0,0841	0,1047
3	0,99	5	0,0260	0,0824	0,1555

Η μέση τιμή της DFEE για όλους τους συνδυασμούς παραμέτρων ήταν 21% για Uniform(UTG), 20% για την Incremental KDE (DTG) και 27% για την Adaptive (ATG). Αυτό σημαίνει ότι μόλις το 20% περίπου των δεδομένων χρειάστηκε να μεταναστεύσει σε γειτονικούς κόμβους ή στο Cloud.

Στα πραγματικά δεδομένα του D2, η εικόνα είναι διαφορετική. Ενδεικτικά σε μια από τις διαστάσεις των δεδομένων, αυτή του αισθητήρα PT08.S1(CO), προέκυψαν τιμές της DFEE: 25% (UTG), 3% (DTG) και 58% (ATG). Σε όλες τις υπόλοιπες διαστάσεις τα αποτελέσματα ήταν ανάλογα. Επομένως φίλτρα προσαρμοσμένα στα δεδομένα (DTG) δίνουν πολύ καλές τιμές DFEE, δηλαδή η μετανάστευση δεδομένων γίνεται σε πολύ μικρό ποσοστό.

4.2.3 Evidence Resolution Speed – ERS

Η μετρική ERS είναι ίση με το παράθυρο μιας εποχής, δηλαδή το πλήθος εργασιών μεταξύ δυο διαδοχικών ανανεώσεων του φίλτρου. Βασίζεται άμεσα στο αρχικό θεώρημα βελτιστοποίησης του Wald [61], [62], το οποίο αποδεικνύει ότι το SPRT ελαχιστοποιεί το αναμενόμενο μέγεθος δείγματος που χρειαζόμαστε για να καταλήξουμε σε ένα συμπέρασμα.

$$ERS = \textit{πλήθος επεξεργασμένων εργασιών μέχρι την απόφαση του SPRT}$$

Μετράει πόσο γρήγορα δημιουργείται επαρκής γνώση ώστε να ανανεωθεί το φίλτρο. Από την προσομοίωση της προτεινόμενης μεθόδου για όλους τους συνδυασμούς παραμέτρων πήραμε τις (κατά μέσο όρο) ελάχιστες, μέγιστες και μέσες τιμές του ERS που φαίνονται στον πίνακα table Iva.

Table IVα – Statistics of Mean ERS			
Method	**Min**	**Average**	**Max**
Uniform	2.32	10.93	24.39
KDE	6.28	35.91	70.13
Adaptive	5.85	14.13	25.06

Οι μέθοδοι UTG και ATG έδωσαν παραπλήσιο αριθμό εισερχομένων εργασιών μέχρι να αποφασίσει ο SPRT. Η DTG χρειάστηκε πολύ περισσότερες κατά μέσο όρο οπότε δημιουργήθηκαν μεγαλύτερα παράθυρα, και μάλιστα πολλές φορές (13 στις 27) το παράθυρο άγγιξε το 100, που ήταν το όριο που είχαμε θέσει για τη διακοπή του SPRT και την απόφαση μη ανανέωσης του φίλτρου.

Στο D2, υπάρχει παρόμοια εικόνα με το D1. Ενδεικτικά για τις μετρήσεις του αισθητήρα PT08.S1(CO), κατά μέσο όρο στις 27 επαναλήψεις του πειράματος είχαμε τα αποτελέσματα του πίνακα Table IVb.

Table IVb – Statistics of Mean ERS for D2			
Method	**Min**	**Average**	**Max**
Uniform	1.00	9.73	45.18
KDE	1.00	35.12	98.00
Adaptive	1.07	13.56	72.07

Παρόμοιες τιμές παρουσιάζει ο δείκτης ERS και στις υπόλοιπες διαστάσεις των δεδομένων στο D2.

4.2.4 Filter Stability Index – FSI

Μετράμε τη διαφορά πλάτους των διαστημάτων μεταξύ δυο διαδοχικών φίλτρων, πόσο δηλαδή αλλάζει το εύρος τιμών που δέχεται το φίλτρο από εποχή σε εποχή.

$$FSI_t = \left|\left(f_t^h - f_t^l\right) - \left(f_{t-1}^h - f_{t-1}^l\right)\right|$$

Η ανανέωση του φίλτρου εξαρτάται από τις εργασίες που έχουν φτάσει στο κόμβο. Η χαμηλή τιμή του δείκτη FSI, αντιστοιχεί σε μικρές αλλαγές στο πλάτος του φίλτρου ενώ μεγαλύτερες τιμές υποδεικνύουν ταλάντωση τιμών. Στον πίνακα Table V φαίνονται οι τιμές (κατά μέσο όρο) ελάχιστης, μέγιστης και μέσης τιμής του FSI για τις τρεις μεθόδους και τις διάφορες τιμές των sigma, gamma, thr.

Table V – Statistics of Mean FSI

Method	**Min**	**Average**	**Max**
Uniform	0.0128	0.0618	0.1318
KDE	0.0197	0.0453	0.0868
Adaptive	0.0206	0.0795	0.1493

Για τα δεδομένα του πραγματικού dataset D2, οι τιμές είναι παραπλήσιες. Ενδεικτικά για τον αισθητήρα PT08.S1(CO) η μέση τιμή των μέσων όρων (mean of averages) για όλες τις εκτελέσεις του πειράματος ήταν: 0,0592 (UTG), 0.0340 (DTG) και 0.0790 (ATG). Και για τα δυο datasets, παρατηρούμε ότι τη μεγαλύτερη ταλάντωση την παρουσιάζει η Adaptive μέθοδος και τη μικρότερη η προσαρμοσμένη στα δεδομένα DTG μέθοδος.

4.2.5 Regeneration Pressure – RP

Η ανανέωση του φίλτρου αλλά και η μετανάστευση δεδομένων που δεν το ικανοποιούν είναι δυο εργασίες που κοστίζουν λόγω των περιορισμένων δυνατοτήτων του κόμβου. Εισάγουμε λοιπόν μια μετρική που συνδυάζει τις δυο αυτές διαδικασίες και αποτυπώνει το «κέρδος» της μεθόδου που ακολουθούμε. Συγκεκριμένα, ορίζουμε:

$$RP = (1 - DFEE) * ERS$$

Η ποσότητα $1 - DFEE$ δίνει το ποσοστό των δεδομένων που παραμένουν στο κόμβο αποθηκευμένα μετά την ανανέωση του φίλτρου και το ERS το πλήθος εργασιών που αφίχθηκαν μέχρι την απόφαση ανανέωσης του φίλτρου. Μεγαλύτερες τιμές του RP δηλώνουν μικρότερη επιβάρυνση στο κόμβο. Αυτό συμβαίνει διότι η αύξηση του RP προκύπτει από την αύξηση ποσοστού δεδομένων που δεν χρειάζονται μετακίνηση και του πλήθους εργασιών που εξυπηρετούνται χωρίς να χρειάζεται ανανέωση του φίλτρου. Ο πίνακας Table VIa, δίνει κατά μέσο όρο τις ελάχιστες, μέγιστες και μέσες τιμές για κάθε μέθοδο κατασκευής εργασιών. Θεωρητική μέγιστη τιμή είναι 100 για $DFEE = 0, ERS = 100$, οπότε μπορεί να θεωρηθεί και ως ποσοστό του μέγιστου κέρδους που μπορούμε να αποκομίσουμε.

Table VIa – Statistics of Mean RP			
Method	**Min**	**Average**	**Max**
Uniform	0.1936	9.2558	53.9286
KDE	0.6136	31.316	86.9400
Adaptive	0.0513	9.8095	73.3214

Για τα πραγματικά δεδομένα D2, και συγκεκριμένα για μια διάσταση προέκυψαν οι τιμές που φαίνονται στον πίνακα Table VIb

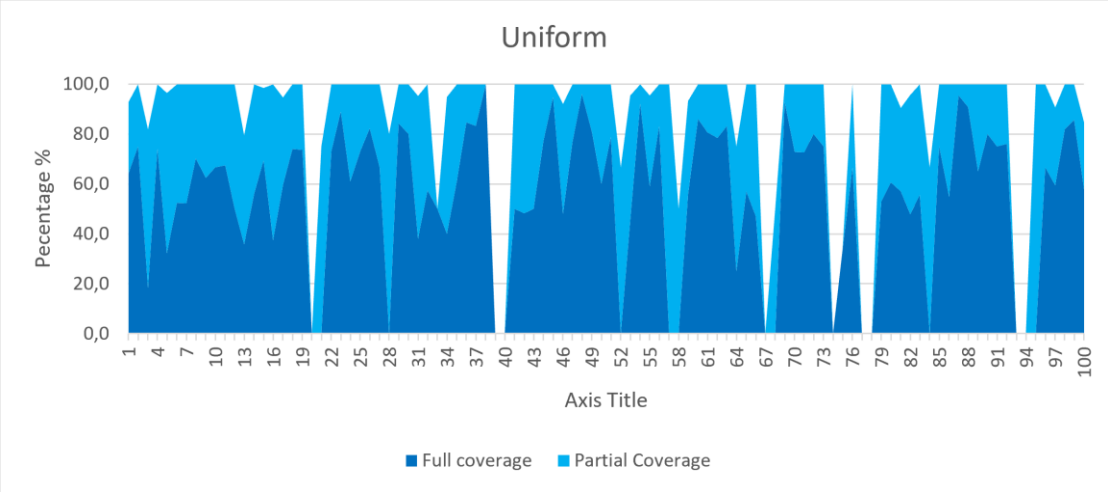
Table VIb – Statistics of Mean RP for D2			
Method	**Min**	**Average**	**Max**
Uniform	0.1403	7.1620	44.3137
KDE	1.8739	29.3082	93.6052
Adaptive	0.0188	7.1978	71.911

Συγκρίνοντας τους δυο πίνακες βλέπουμε μικρές διαφορές μεταξύ τους. Όμως η μέθοδος κατασκευής εργασιών DTG δίνει σαφώς μεγαλύτερα «κέρδη» σε σχέση με τις άλλες δυο μεθόδους. Ακολουθεί η ATG όπου λόγω φιλοσοφίας δημιουργίας εργασιών όπου κάθε μια επηρεάζεται από την προηγούμενή της, δίνει καλύτερη συμπεριφορά του δείκτη RP σε σχέση με την μέθοδο ATG που είναι μια αμιγώς τυχαία διαδικασία.

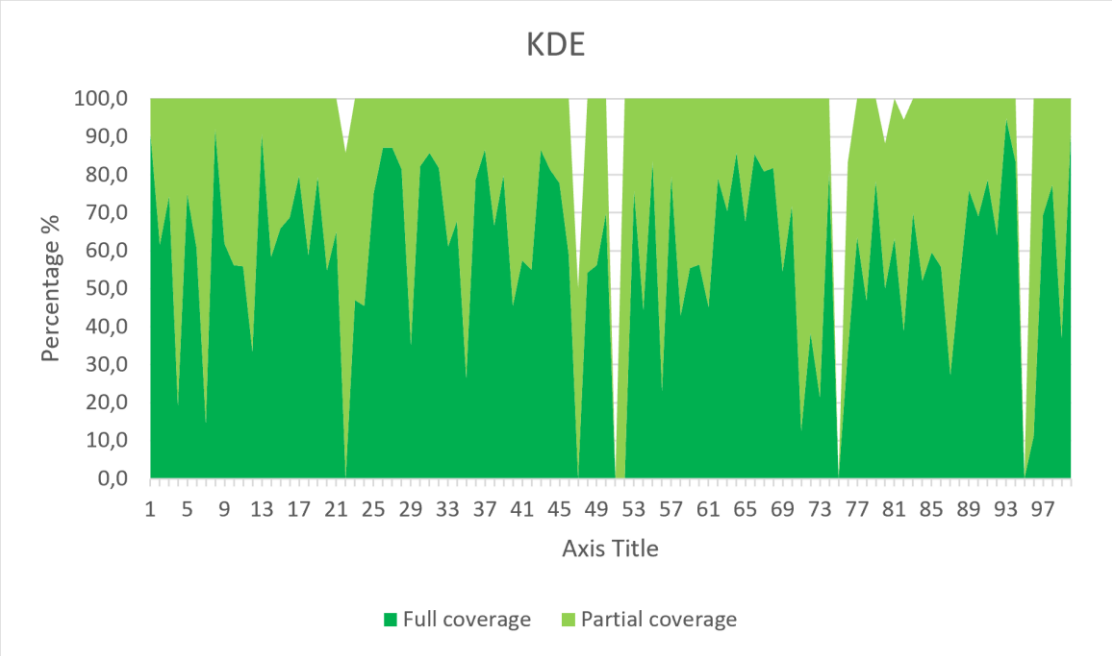
4.3 Πειραματική αποτίμηση

Η χρήση όλων των παραπάνω μετρικών αυτόνομα και συνδυαστικά δίνει χρήσιμα συμπεράσματα για την προτεινόμενη μέθοδο διαχείρισης φίλτρων.

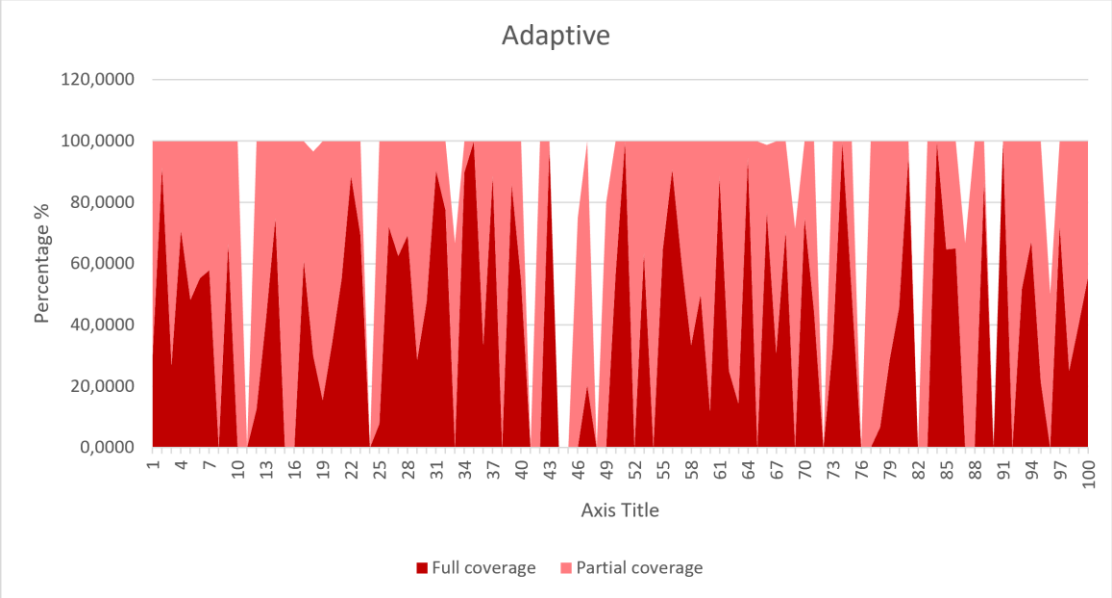
Οι δύο βασικοί στόχοι της μεθόδου, η κάλυψη μελλοντικών εργασιών αλλά και η διαχείριση των δεδομένων που είναι αποθηκευμένα στον κόμβο, επιτεύχθηκαν σε ικανοποιητικό βαθμό όπως φαίνεται στα παρακάτω γραφήματα. Συγκεκριμένα, οι εικόνες 8, 9 και 10, δείχνουν την κάλυψη (ολική, μερική και αθροιστικά) για τις τρεις μεθόδους κατασκευής εργασιών και για συγκεκριμένο setup: σ=3, γ=0.9 και thr=2



Εικόνα 8: Ποσοστά μελλοντικών εργασιών που καλύπτονται πλήρως (μπλε) και μερικώς (γαλάζιο) στη μέθοδο UTG



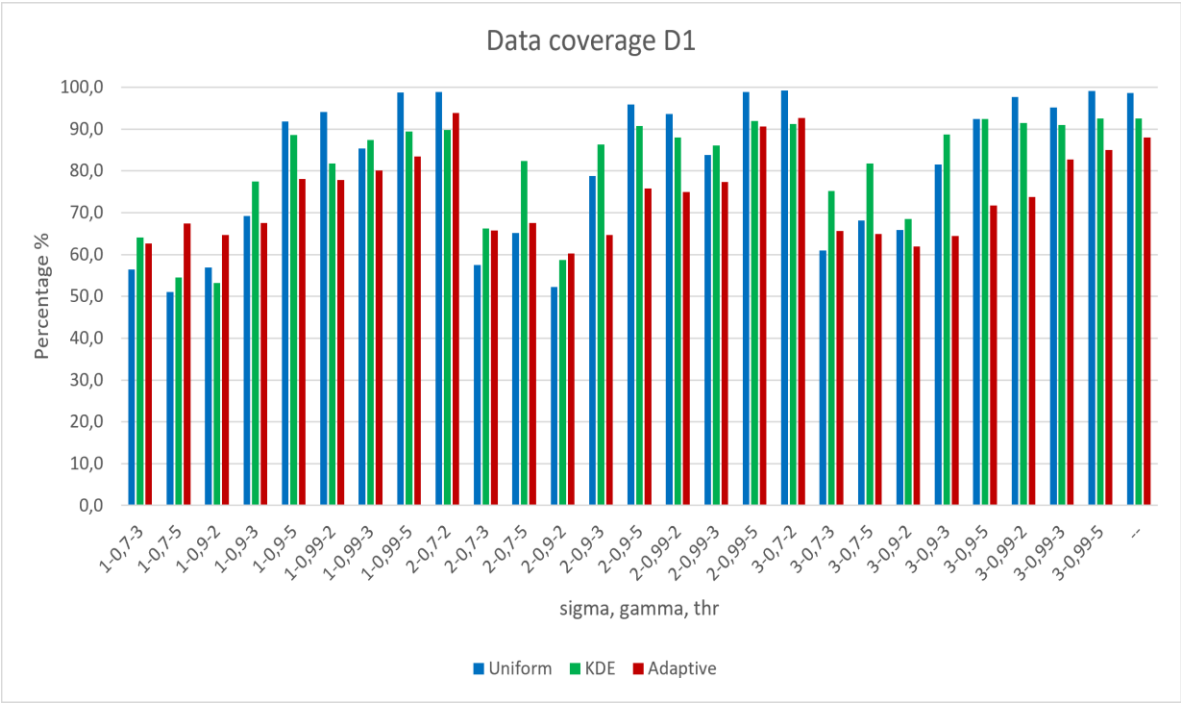
Εικόνα 9: Ποσοστά μελλοντικών εργασιών που καλύπτονται πλήρως (πράσινο σκούρο) και μερικώς (πράσινο ανοικτό) στη μέθοδο DTG



Εικόνα 10: Ποσοστά μελλοντικών εργασιών που καλύπτονται πλήρως (κόκκινο) και μερικώς (ροζ) στη μέθοδο ATG

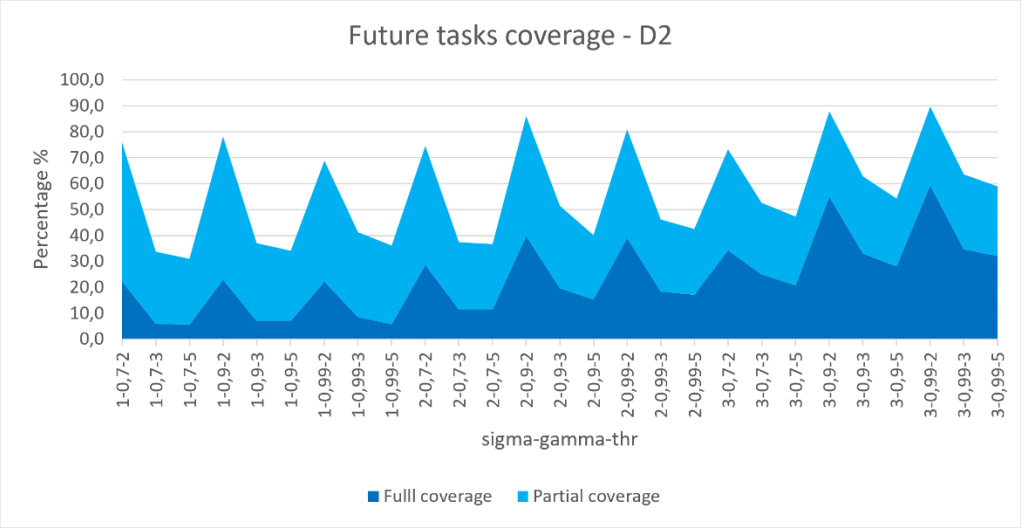
Και στις τρεις μεθόδους, το συνολικό ποσοστό κάλυψης ήταν 100% τις περισσότερες φορές. Η πλήρης κάλυψη ήταν (ειδικά στην DTG μέθοδο) πάντα υψηλή. Αυτό είναι πολύ σημαντικό διότι σε αυτή τη περίπτωση όλα τα δεδομένα που είναι αποθηκευμένα στο κόμβο ικανοποιούν τα κριτήρια που τίθενται από τις αφικνούμενες εργασίες. Επομένως δεν χρειάζεται μετακίνηση δεδομένων, κάνοντας εξοικονόμηση πόρων στο κόμβο.

Αντίστοιχα, στις περιπτώσεις που χρειάζεται αντικατάσταση των υπαρχόντων δεδομένων με νέα που να ανταποκρίνονται στις τρέχουσες εργασίες, συνήθως το ποσοστό μετακίνησης είναι μικρό, παρά την τυχαιότητα που μπορεί να ακολουθούν αυτές. Στην Εικόνα 11 παρουσιάζονται τα ποσοστά δεδομένων που παραμένουν αποθηκευμένα στο κόμβο (κατά μέσο όρο) για κάθε συνδυασμό των παραμέτρων του πειράματος.

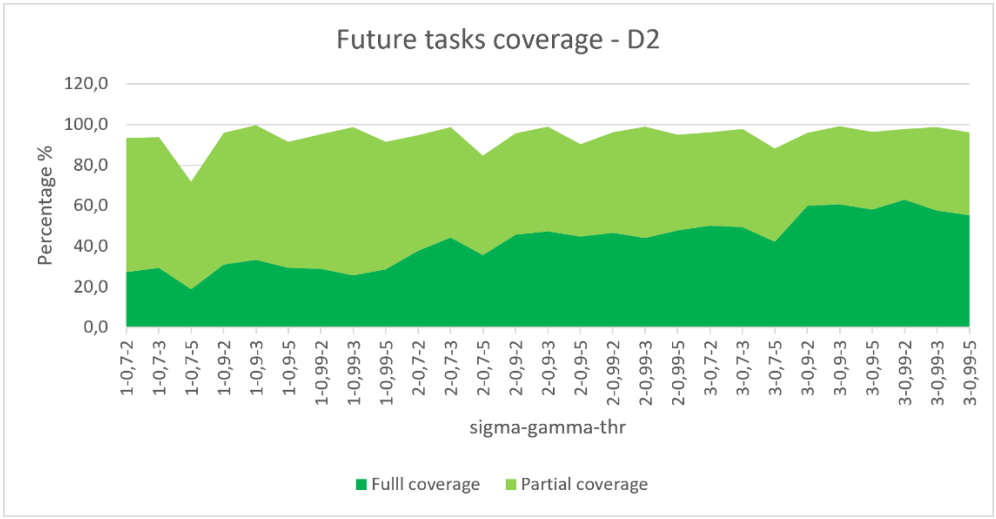


Εικόνα 11: Ποσοστά κάλυψης των δεδομένων από το φίλτρο για διάφορους συνδυασμούς παραμέτρων του πειράματος

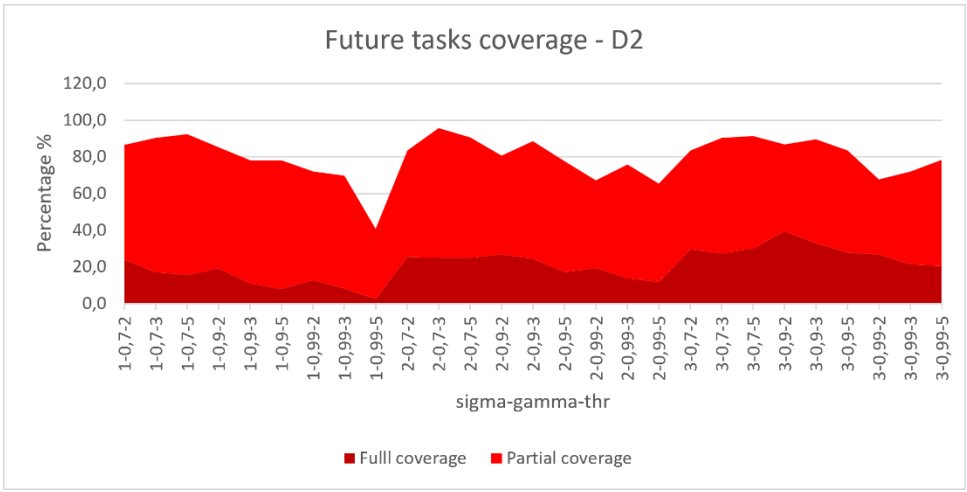
Εξίσου ενθαρρυντικά είναι και τα αντίστοιχα αποτελέσματα στο dataset D2, με πραγματικά δεδομένα. Στις εικόνες 12, 13 και 14 απεικονίζονται τα ποσοστά πλήρους και μερικής κάλυψης των εισερχόμενων εργασιών για κάθε μέθοδο και στην εικόνα 15 παρουσιάζονται τα ποσοστά κάλυψης των δεδομένων από το εκάστοτε φίλτρο τη στιγμή που αυτό ανανεώνεται. Η μέθοδος DTG δίνει τα καλύτερα αποτελέσματα, ακολουθεί η Adaptive μέθοδος και χειρότερη απόδοση έχει η κατανομή Uniform λόγω της τυχαιότητας των εργασιών.



Εικόνα 12: Ποσοστά μελλοντικών εργασιών που καλύπτονται πλήρως (μπλε) και μερικώς (γαλάζιο) στη μέθοδο UTG και για το σύνολο δεδομένων D2

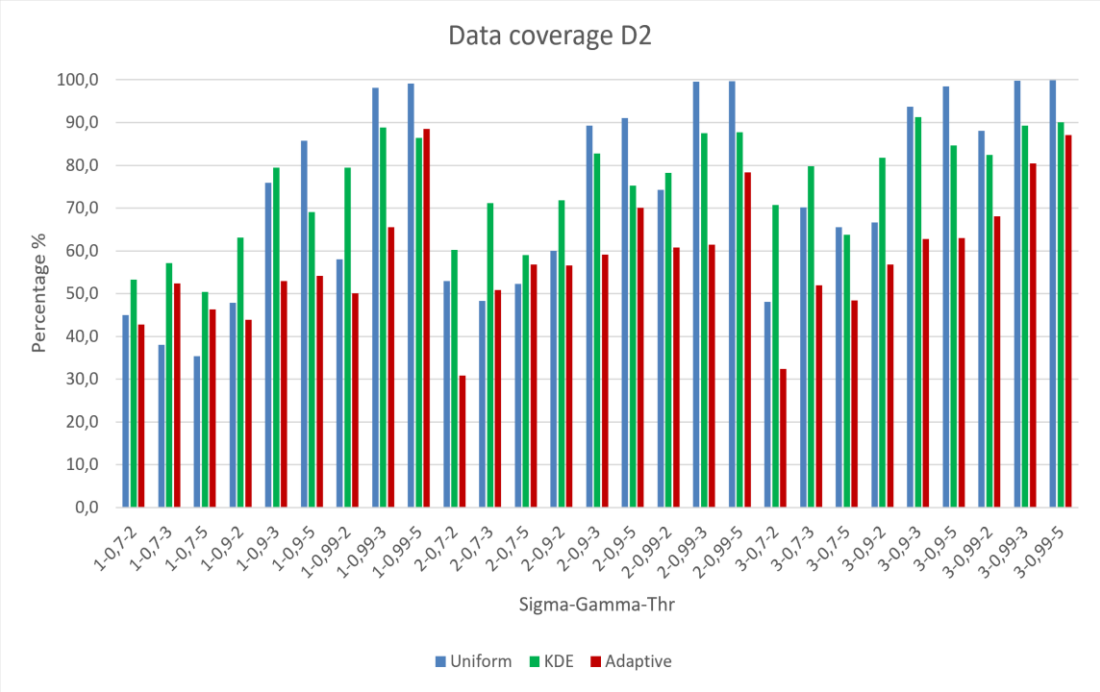


Εικόνα 13: Ποσοστά μελλοντικών εργασιών που καλύπτονται πλήρως (σκούρο πράσινο) και μερικώς (ανοικτό πράσινο) στη μέθοδο DTG και για το σύνολο δεδομένων D2



Εικόνα 14: Ποσοστά μελλοντικών εργασιών που καλύπτονται πλήρως (σκούρο κόκκινο) και μερικώς (ανοικτό κόκκινο) στη μέθοδο ATG και για το σύνολο δεδομένων D2

Ένα άλλο συμπέρασμα που προκύπτει από τις μετρικές που χρησιμοποιήθηκαν είναι ότι η επιλογή των σωστών παραμέτρων για το sigma, gamma και threshold, είναι μείζονος σημασίας. Τιμές του σ κοντά στο 2, γ κοντά στο 0.9 και threshold στο 2-3 στις περισσότερες φορές έδιναν τα μέγιστα αποτελέσματα. Θα μπορούσε να μελετηθεί η σχέση των παραμέτρων αυτών με τα υφιστάμενα δεδομένα, μελέτη που μπορεί να γίνει στο μέλλον.



Εικόνα 15: Ποσοστά κάλυψης των δεδομένων από το φίλτρο για διάφορους συνδυασμούς παραμέτρων του πειράματος και το σύνολο δεδομένων D2

ΚΕΦΑΛΑΙΟ 5 Συμπεράσματα και Μελλοντικές Προεκτάσεις

5.1 Συμπεράσματα

Σκοπός της παρούσας εργασίας ήταν η ευφυής διαχείριση φίλτρων σε αυτόνομους κόμβους. Σε περιβάλλοντα constrained συσκευών ή κόμβων Edge, οι πόροι όπως υπολογιστική δύναμη, ισχύς, αποθηκευτικός χώρος κ.α. είναι μειωμένοι. Είναι σημαντικό λοιπόν οι αυτόνομοι κόμβοι του Edge να μπορούν να διαχειριστούν έξυπνα τόσο τις εργασίες που καταφθάνουν κι αιτούνται επεξεργασία των δεδομένων που είναι αποθηκευμένα στο κόμβο, όσο και τα ίδια τα δεδομένα, δηλαδή να αποφασίζουν κατά καιρούς ποια δεδομένα να κρατήσουν τοπικά και ποια να μεταναστεύσουν σε γειτονικούς – συνεργαζόμενους κόμβους ή ακόμη και στο Cloud. Η χρήση φίλτρων που θα πετυχαίνουν να βελτιστοποιήσουν και τα δύο, εργασίες που αφικνούνται κι αποθηκευμένα δεδομένα, είναι μια από τις πολλές διαφορετικές προσεγγίσεις που αναφέρθηκαν στην εργασία αυτή. Η σωστή διαχείρισή τους περιέχει στοιχεία που υπάρχουν και σε άλλες εργασίες πάνω στο Edge Computing όπως task – awareness, task offloading, data management. Επομένως θα μπορούσε άνετα να συνδυαστεί η παρούσα προτεινόμενη μέθοδος με κάποιες από τις τεχνικές και μηχανισμούς που αναπτύχθηκαν στο κεφάλαιο 2.

Οι μετρικές που χρησιμοποιήθηκαν στην αποτίμηση της πειραματικής διαδικασίας, είναι παραπλήσιες σε μετρικές που χρησιμοποιούνται συχνά σε διαδικασίες που εμπεριέχουν Θεωρία Πληροφορίας, χρόνους εξόδου από μια εργασία [63], [64], σταθερότητα φίλτρων [65], κέρδος πληροφορίας ανά δράση [66], [67], ποιότητα εφαρμογής στατιστικού μοντέλου [68] κ.α.

Δυνατά σημεία της προτεινόμενης εργασίας είναι οι σχετικά απλοί μαθηματικοί υπολογισμοί που χρειάζονται για την διαδικασία

ανανέωσης των φίλτρων και τα αποτελέσματα που προέκυψαν από την πειραματική εκτέλεση κώδικα προσομοίωσης της διαδικασίας. Τόσο στα συνθετικά όσο και στα πραγματικά δεδομένα, το μοντέλο έδωσε υψηλά ποσοστά προβλεψιμότητας ειδικά στο τομέα της κάλυψης των μελλοντικών εργασιών από το τρέχων φίλτρο κάθε φορά. Αυτό συνεπάγεται την έγκαιρη επιλογή δεδομένων που θα παραμείνουν αποθηκευμένα στον τοπικό κόμβο μειώνοντας το κόστος μετακίνησής τους.

Η δοκιμή διαφόρων παραμέτρων στα ίδια datasets ώστε να υπάρχει μέτρο σύγκρισης, έδειξε ότι κάθε μια από τις τρεις μεθόδους δημιουργίας εργασιών (UTG, DTG, ATG) επωφελείται από διαφορετικούς συνδυασμούς τιμών, αλλά υπάρχουν κάποιες τιμές που γενικά δίνουν συνολικά πολύ καλή συμπεριφορά . Η UTG (Uniform) μέθοδος ήταν η πιο απρόβλεπτη από τις τρεις, κάτι αναμενόμενο λόγω της τυχαιότητας, ενώ η πιο καλή σε προβλέψεις ήταν η DTG (Kernel Density Estimator). Με ποσοστό επιτυχούς μερικής ή ολικής κάλυψης ακόμη και του 90% μελλοντικών εργασιών (κατά μέσο όρο ανά μέθοδο) τα αποτελέσματα της μεθοδολογίας αυτής δίνουν ενθαρρυντικά μηνύματα πως η προτεινόμενη διαδικασία ανανέωσης φίλτρων βρίσκεται σε καλή πορεία. Η απλότητά της αλλά ταυτοχρόνως η δυναμική της, δείχνουν ότι μπορεί να συνδυαστεί με μεθόδους τόσο μηχανικής μάθησης όσο και με άλλα μαθηματικά μοντέλα με σκοπό να γίνει ακόμη πιο καλή στην προβλεπτική (proactive).

5.2 Μελλοντικές προεκτάσεις

Μελλοντικά μπορεί να ερευνηθεί η συμπεριφορά όταν τα δεδομένα είναι πολυδιάστατα. Με ποιο τρόπο επιλέγουμε σε εκείνη την περίπτωση την ανανέωση των φίλτρων; Μπορεί να δουλέψει μια μορφή «πολυδιάστατου» SPRT και πως; Ποιοι άλλοι μηχανισμοί θα μπορούσαν να συνδυαστούν με τη μέθοδο αυτή; Ειδικά μηχανισμοί που αναφέρθηκαν στο κεφάλαιο 2 και οι οποίοι δεν ήταν μέσα στα πλαίσια της παρούσας εργασίας, όπως ενδεικτικά, η επιλογή κόμβων που θα

μετακινηθούν τα δεδομένα, ο τρόπος επικοινωνίας μεταξύ κόμβων για ανταλλαγή εργασιών, αιτημάτων, ερωτημάτων και δεδομένων και γενικά θέματα συνεργασίας κόμβων του Edge.

Η μέθοδος DTG (Data-driven Task Generator) σχεδόν σε όλο το φάσμα μετρικών που χρησιμοποιήθηκαν έδωσε τα καλύτερα σκορ. Θα ήταν ενδιαφέρουσα μια μελλοντική μελέτη συνδυασμού αποθήκευσης δεδομένων συνεργατικά ή μη με τον διαμοιρασμό εργασιών σύμφωνα με τα δεδομένα και τη κατανομή τους στους συνεργαζόμενους κόμβους (στο κεφάλαιο 2 αναφέρθηκαν μερικοί τρόποι) και τη μέθοδο που αναπτύχθηκε στη παρούσα εργασία ώστε να δημιουργηθούν μοντέλα με υψηλή προβλεψιμότητα μελλοντικών εργασιών και ευφυή διαχείριση δεδομένων και φίλτρων.

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. Atzori, Luigi & Iera, Antonio & Morabito, Giacomo. (2010). The Internet of Things: A Survey. Computer Networks. 2787-2805. 10.1016/j.comnet.2010.05.010.
2. Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, Marimuthu Palaniswami, Internet of Things (IoT): A vision, architectural elements, and future directions, Future Generation Computer Systems, Volume 29, Issue 7, 2013, Pages 1645-1660
3. Daniele Miorandi, Sabrina Sicari, Francesco De Pellegrini, Imrich Chlamtac, Internet of things: Vision, applications and research challenges, Ad Hoc Networks, Volume 10, Issue 7, 2012, Pages 1497-1516, ISSN 1570-8705, https://doi.org/10.1016/j.adhoc.2012.02.016.
4. Want, Roy & Schilit, Bill & Jenson, Scott. (2015). Enabling the Internet of Things. Computer. 48. 28-35. 10.1109/MC.2015.12.
5. K. Finkenzeller, *RFID Handbook*, Wiley, 2010.
6. S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," IETF RFC 2460, 1998.
7. K. Ashton, "That 'Internet of Things' Thing," *RFID Journal*, 2009.
8. H. Sundmaeker et al., *Vision and Challenges for Realising the IoT*, EU Publications, 2010.
9. M. Swan, "Sensor mania! The Internet of Things, wearable computing, objective metrics, and the quantified self," *Journal of Sensor and Actuator Networks*, 2012.
10. Kadiyan, Vivek & Singh, Mandeep & Kumar, Adarsh & Saini, Ashu & Singh, Himanshu. (2023). SMART HOME AUTOMATION SYSTEM BASED ON IoT. 7. 1-8. 10.55041/IJSREM21794.
11. Ruiz-Garcia, L., Lunadei, L., Barreiro, P., & Robla, I. (2009). A Review of Wireless Sensor Technologies and Applications in Agriculture and Food Industry: State of the Art and Current Trends. *Sensors*, *9*(6), 4728-4750. https://doi.org/10.3390/s90604728
12. Lee, I. and Lee, K. (2015) The Internet of Things (IoT): Applications, Investments, and Challenges for Enterprises. Business Horizons, 58, 431-440. https://doi.org/10.1016/j.bushor.2015.03.008
13. Hermann, Mario & Pentek, Tobias & Otto, Boris. (2015). Design Principles for Industrie 4.0 Scenarios: A Literature Review.10.13140/RG.2.2.29269.22248.
14. Zanella, Andrea & Bui, Nicola & Castellani, Angelo & Vangelista, Lorenzo & Zorzi, Michele. (2012). Internet of Things for Smart Cities. Internet of Things Journal, IEEE. 1. 10.1109/JIOT.2014.2306328.
15. Chen, M., Mao, S. and Liu, Y. (2014) Big Data: A Survey. Mobile Networks and Applications, 19, 171-209. https://doi.org/10.1007/s11036-013-0489-0
16. Sicari, S., Rizzardi, A., Grieco, L.A. and Coen-Porisini, A. (2015) Security, Privacy and Trust in Internet of Things: The Road Ahead. *Computer Networks*, 76, 146-164.https://doi.org/10.1016/j.comnet.2014.11.008

17. Lin, Jie & Yu, Wei & Zhang, Nan & Yang, Xinyu & Zhang, Hanlin & Zhao, Wei. (2017). A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications. IEEE Internet of Things Journal. PP. 1-1. 10.1109/JIOT.2017.2683200.
18. W. Shi, J. Cao, Q. Zhang, Y. Li and L. Xu, "Edge Computing: Vision and Challenges," in *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637-646, Oct. 2016, doi: 10.1109/JIOT.2016.2579198.
19. M. Wang, "Dependent task offloading of cross-applications and service caching for end–Edge–Cloud computing in consumer electronics," *IEEE Trans. Consum. Electron.*, vol. 71, no. 4, pp. 10383–10394, Nov. 2025.
20. A. Rehman, M. U. Hassan, S. Ali, K. Mahmood, N. A. Wani και M. S. Anwar, "Adaptive Edge Intelligence Framework for Resource-Constrained IoT in Consumer Electronics", στο IEEE Transactions on Consumer Electronics, τόμος 72, αρ. 1, σ. 2624-2631, February 2026, doi: 10.1109/TCE.2025.3648062.
21. X. Zhang, D. Hou, Z. Xiong, Y. Liu, S. Wang, and Y. Li, "EALLR: Energy-aware low-latency routing data driven model in mobile Edge computing," *IEEE Trans. Consum. Electron.*, vol. 71, no. 2, pp. 6612–6626, May 2025.
22. Shen Z, Liu B, Dou W. An effective data placement strategy for IoT applications. Concurrency Computat Pract Exper. 2024; 36(10):e7977. doi: 10.1002/cpe.7977
23. X. Du, S. Tang, Z. Lu, J. Wet, K. Gai and P. C. K. Hung, "A Novel Data Placement Strategy for Data-Sharing Scientific Workflows in Heterogeneous Edge-Cloud Computing Environments," 2020 IEEE International Conference on Web Services (ICWS), Beijing, China, 2020, pp. 498-507, doi: 10.1109/ICWS49710.2020.00073
24. Chanhyuk Lee, Jisoo Kim, Heedong Ko, Byounghyun Yoo, Addressing IoT storage constraints: A hybrid architecture for decentralized data storage and centralized management, Internet of Things, Volume 25, 2024, 101014, ISSN 2542-6605, https://doi.org/10.1016/j.iot.2023.101014
25. Al-Asadi, S.A., Al-Mamory, S.O.: A survey on Edge and fog nodes' placement methods, techniques, parameters, and constraints. IET Netw. 12(5), 197–228 (2023). https://doi.org/10.1049/ntw2.12087
26. L. Yuan *et al.*, "CSEdge: Enabling Collaborative Edge Storage for Multi-Access Edge Computing Based on Blockchain," in *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 8, pp. 1873-1887, 1 Aug. 2022, doi: 10.1109/TPDS.2021.3131680Samie, Farzad. (2018). Resource Management for Edge Computing in Internet of Things (IoT). 10.5445/IR/1000081031.
27. Xia, X., Chen, F., He, Q., Grundy, J.C., Abdelrazek, M., & Jin, H. (2021). Online Collaborative Data Caching in Edge Computing. *IEEE Transactions on Parallel and Distributed Systems, 32*, 281-294.
28. X. Gao, W. Bao, X. Zhu, G. Wu and L. Liu, "An Edge Storage Acceleration Service for Collaborative Mobile Devices," in IEEE Transactions on Services Computing, vol. 15, no. 4, pp. 1993-2006, 1 July-Aug. 2022, doi: 10.1109/TSC.2020.3029136
29. Kim CK, Kim T, Lee S, Lee S, Cho A, et al. (2022) Delay-aware distributed program caching for IoT-Edge networks. PLOS ONE 17(7): e0270183. https://doi.org/10.1371/journal.pone.0270183

30. A. Aral and T. Ovatman, "A Decentralized Replica Placement Algorithm for Edge Computing," in *IEEE Transactions on Network and Service Management*, vol. 15, no. 2, pp. 516-529, June 2018, doi: 10.1109/TNSM.2017.2788945.
31. Shao, ZL., Huang, C. & Li, H. Replica selection and placement techniques on the IoT and Edge computing: a deep study. Wireless Netw 27, 5039–5055 (2021). https://doi.org/10.1007/s11276-021-02793-x
32. Yin Y, Deng L. A dynamic decentralized strategy of replica placement on Edge computing. International Journal of Distributed Sensor Networks. 2022;18(8). doi:10.1177/15501329221115064
33. Yanling Shao, Chunlin Li, Hengliang Tang, A data replica placement strategy for IoT workflows in collaborative Edge and Cloud environments, Computer Networks, Volume 148, 2019, Pages 46-59, ISSN 1389-1286, https://doi.org/10.1016/j.comnet.2018.10.017.
34. Anglano, C., Canonico, M., Gaeta, R. et al. Popularity-based data placement in erasure-coded Edge Storage Systems. Cluster Comput 28, 1047 (2025). https://doi.org/10.1007/s10586-025-05701-6
35. Y. Hassanzadeh-Nazarabadi, A. Küpçü and O. Ozkasap, "Decentralized Utility- and Locality-Aware Replication for Heterogeneous DHT-Based P2P Cloud Storage Systems," in IEEE Transactions on Parallel and Distributed Systems, vol. 31, no. 5, pp. 1183-1193, 1 May 2020, doi: 10.1109/TPDS.2019.2960018
36. Z. Shamsa and M. Dehghan, "Placement of replicas in distributed systems using particle swarm optimization algorithm and its fuzzy generalization," 2013 13th Iranian Conference on Fuzzy Systems (IFSC), Qazvin, Iran, 2013, pp. 1-6, doi: 10.1109/IFSC.2013.6675641.
37. F. Samie, V. Tsoutsouras, D. Masouros, L. Bauer, D. Soudris and J. Henkel, "Fast Operation Mode Selection for Highly Efficient IoT Edge Devices," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 3, pp. 572-584, March 2020, doi: 10.1109/TCAD.2019.2897633
38. Joshi, R., Somesula, R.S., Katkoori, S. (2024). Empowering Resource-Constrained IoT Edge Devices: A Hybrid Approach for Edge Data Analysis. In: Puthal, D., Mohanty, S., Choi, BY. (eds) Internet of Things. Advances in Information and Communication Technology. IFIPIoT 2023. IFIP Advances in Information and Communication Technology, vol 683. Springer, Cham. https://doi.org/10.1007/978-3-031-45878-1_12
39. Mughal, F.R., He, J., Das, B. et al. Adaptive federated learning for resource-constrained IoT devices through Edge intelligence and multi-Edge clustering. Sci Rep 14, 28746 (2024). https://doi.org/10.1038/s41598-024-78239-z
40. Farooq, O., Singh, P., Hedabou, M., Boulila, W., & Benjdira, B. (2023). Machine Learning Analytic-Based Two-Staged Data Management Framework for Internet of Things. Sensors, 23(5), 2427. https://doi.org/10.3390/s23052427
41. Majjari Sudhakar, Koteswara Rao Anne, optimizing data processing for Edge-enabled IoT devices using deep learning based heterogeneous data clustering approach, Measurement: Sensors, Volume 31, 2024, 101013, ISSN 2665-9174, https://doi.org/10.1016/j.measen.2023.101013. (https://www.sciencedirect.com/science/article/pii/S2665917423003495)
42. Q. Long, C. Anagnostopoulos and K. Kolomvatsos, "Enhancing KnowlEdge Reusability: A Distributed Multitask Machine Learning Approach," in IEEE

Transactions on Emerging Topics in Computing, vol. 13, no. 1, pp. 207-221, Jan.-March 2025, doi: 10.1109/TETC.2024.3390811
43. Oikonomou, P., Karanika, A., Anagnostopoulos, C., & Kolomvatsos, K. (2020). On the Road from Edge Computing to the Edge Mesh. ArXiv, abs/2010.05187.
44. Okwuibe, Jude & Haavisto, Juuso & Harjula, Erkki & Ahmad, Ijaz & Ylianttila, Mika. (2019). Orchestrating Service Migration for Low Power MEC-Enabled IoT Devices. 10.48550/arXiv.1905.12959.
45. E. Bozkaya-Aras, "Optimizing Service Migration in IoT Edge Networks: Digital Twin-Based Computation and Energy-Efficient Approach," 2025 IEEE Wireless Communications and Networking Conference (WCNC), Milan, Italy, 2025, pp. 1-6, doi: 10.1109/WCNC61545.2025.10978782.
46. K. Ray, A. Banerjee and N. C. Narendra, "Learning-Based Microservice Placement and Migration for Multi-Access Edge Computing," in *IEEE Transactions on Network and Service Management*, vol. 21, no. 2, pp. 1969-1982, April 2024, doi: 10.1109/TNSM.2023.3344192
47. A. Calagna, Y. Yu, P. Giaccone and C. F. Chiasserini, "Design, Modeling, and Implementation of Robust Migration of Stateful Edge Microservices," in IEEE Transactions on Network and Service Management, vol. 21, no. 2, pp. 1877-1893, April 2024, doi: 10.1109/TNSM.2023.3331750
48. Y. Mao, C. You, J. Zhang, K. Huang and K. B. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective," in *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322-2358, Fourthquarter 2017, doi: 10.1109/COMST.2017.2745201
49. Kostas Kolomvatsos, Christos Anagnostopoulos, Autonomous proactive data management in support of pervasive Edge applications, Future Generation Computer Systems, Volume 155, 2024, Pages 108-120, ISSN 0167-739X, https://doi.org/10.1016/j.future.2024.02.003.
50. A. Koukosias, C. Anagnostopoulos and K. Kolomvatsos, "Task-Aware Data Selectivity in Pervasive Edge Computing Environments," in IEEE Transactions on KnowlEdge and Data Engineering, vol. 37, no. 1, pp. 513-525, Jan. 2025, doi: 10.1109/TKDE.2024.3485531.
51. K. Kolomvatsos, C. Anagnostopoulos, M. Koziri and T. Loukopoulos, "Proactive & Time-Optimized Data Synopsis Management at the Edge," in IEEE Transactions on KnowlEdge and Data Engineering, vol. 34, no. 7, pp. 3478-3490, 1 July 2022, doi: 10.1109/TKDE.2020.3021377.
52. Fountas, P., Kolomvatsos, K., Anagnostopoulos, C. (2021). A Deep Learning Model for Data Synopses Management in Pervasive Computing Applications. In: Arai, K. (eds) Intelligent Computing. Lecture Notes in Networks and Systems, vol 284. Springer, Cham. https://doi.org/10.1007/978-3-030-80126-7_44
53. Kolomvatsos, K. (2020). An Intelligent Scheme for Uncertainty Management of Data Synopses Management in Pervasive Computing Applications. ArXiv, abs/2007.12648.
54. Kolomvatsos, K. (2020). A Probabilistic Approach for Data Management in Pervasive Computing Applications. ArXiv, abs/2009.04739.
55. Anagnostopoulos, C., Aladwani, T., Alghamdi, I., & Kolomvatsos, K. (2022). Data-Driven Analytics Task Management Reasoning Mechanism in Edge

Computing. Smart Cities, 5(2), 562-582. https://doi.org/10.3390/smartcities5020030
56. K. Kolomvatsos and C. Anagnostopoulos, "A Proactive Statistical Model Supporting Services and Tasks Management in Pervasive Applications," in IEEE Transactions on Network and Service Management, vol. 19, no. 3, pp. 3020-3031, Sept. 2022, doi: 10.1109/TNSM.2022.3161663
57. Kostas Kolomvatsos, Christos Anagnostopoulos, Proactive, uncertainty-driven queries management at the Edge, Future Generation Computer Systems, Volume 118, 2021, Pages 75-93, ISSN 0167-739X, https://doi.org/10.1016/j.future.2020.12.028. (https://www.sciencedirect.com/science/article/pii/S0167739X21000029)
58. Kotz, S., Balakrishnan, N., & Johnson, N. L. (2000). Continuous multivariate distributions, Vol. 1: Models and applications (2nd ed.). Wiley.
59. Morrison, D.F. (2014). Bivariate Normal Distribution. In Wiley StatsRef: Statistics Reference Online (eds N. Balakrishnan, T. Colton, B. Everitt, W. Piegorsch, F. Ruggeri and J.L. Teugels). https://doi.org/10.1002/9781118445112.stat05828
60. Mahalanobis, P.C. (1936) On the Generalized Distance in Statistics. *Proceedings of the National Academy of Sciences of India*, 2, 49-55.
61. Wald, A. (1945). Sequential tests of statistical hypotheses. *Annals of Mathematical Statistics*, 16(2), 117–186
62. Wald, A., & Wolfowitz, J. (1948). Optimum character of the sequential probability ratio test. Annals of Mathematical Statistics, 19(3), 326–339.
63. E. Massera, M. Piga, L. Martinotto, G. Di Francia, 'On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario', Sensors and Actuators B: Chemical, vol. 129(2),2008, pp. 750–757.
64. Liu, S., Fengler, A., Frank, M.J., & Harrison, M.T. (2025). Efficient Inference in First Passage Time Models. *Statistics and computing, 36*.
65. Roger Ratcliff, Gail McKoon; The Diffusion Decision Model: Theory and Data for Two-Choice Decision Tasks. *Neural Comput* 2008; 20 (4): 873–922. doi: https://doi.org/10.1162/neco.2008.12-06-420
66. T. Y. Al-Naffouri, A. Zerguine and M. Bettayeb, "Convergence analysis of the LMS algorithm with a general error nonlinearity and an IID input," *Conference Record of Thirty-Second Asilomar Conference on Signals, Systems and Computers (Cat. No.98CH36284)*, Pacific Grove, CA, USA, 1998, pp. 556-559 vol.1, doi: 10.1109/ACSSC.1998.750925.
67. R. Vanamadi, A. Kar, S. Burra, A. Anand and B. Majhi, "Convergence Performance Evaluation of MSF-Based LMS Adaptive Algorithm," *2019 16th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, Pattaya, Thailand, 2019, pp. 597-600, doi: 10.1109/ECTI-CON47248.2019.8955136.
68. LESNE A. Shannon entropy: a rigorous notion at the crossroads between probability, information theory, dynamical systems and statistical physics. *Mathematical Structures in Computer Science*. 2014;24(3):e240311. doi:10.1017/S0960129512000783
69. Nguyen, K. P., Josić, K., & Kilpatrick, Z. P. (2019). Optimizing sequential decisions in the drift-diffusion model. *Journal of mathematical psychology*, *88*, 32–47. https://doi.org/10.1016/j.jmp.2018.11.001