



ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Συλλογική νοημοσύνη στο διαμοιρασμό
δεδομένων και εργασιών

ΓΕΩΡΓΙΟΣ ΑΝΤΩΝΙΟΥ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΥΠΕΥΘΥΝΟΣ

ΚΩΝΣΤΑΝΤΙΝΟΣ ΚΟΛΟΜΒΑΤΣΟΣ

Λαμία Σεπτέμβριος 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 επεξεργάστηκαν, 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:/...../20.....

Ο – Η Δηλ.

(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

Η παρούσα εργασία μελετά το πρόβλημα της διαχείρισης και του διαμοιρασμού δεδομένων μεταξύ αυτόνομων κόμβων σε ένα καταναμεμημένο περιβάλλον. Προτείνεται ένα σύστημα στο οποίο κάθε κόμβος διαχειρίζεται τοπικά τα δεδομένα του, εντοπίζει κενά διαστήματα ανάμεσα σε αυτά τα δεδομένα και αξιολογεί τα νέα εισερχόμενα δεδομένα ώστε να αποφασίσει αν θα τα διατηρήσει ή θα τα διαθέσει για μεταφορά σε άλλους κόμβους. Για την επιλογή κατάλληλων κόμβων παραληπτών χρησιμοποιήθηκε ο αλγόριθμος Artificial Bee Colony (ABC), λαμβάνοντας υπόψη την κάλυψη των κενών διαστημάτων και το κόστος επικοινωνίας. Το σύστημα αξιολογήθηκε πειραματικά μέσω τριών διαφορετικών σεναρίων: μόνο τοπική διατήρηση δεδομένων, μόνο μεταφορές δεδομένων μέσω ABC και συνδυασμός των δύο μηχανισμών. Τα αποτελέσματα έδειξαν ότι το πλήρες σύστημα μπορεί να βελτιώσει σημαντικά την τοπική κάλυψη δεδομένων καθώς και τη συνολική κάλυψη του συστήματος.

Table of Contents

ΠΕΡΙΛΗΨΗ	II
<u>ΚΕΦΑΛΑΙΟ 1 ΕΙΣΑΓΩΓΗ</u>	<u>2</u>
1.1 ΕΙΣΑΓΩΓΗ ΚΑΙ ΚΙΝΗΤΡΟ	2
1.2 ΣΤΟΧΟΣ ΚΑΙ ΑΝΤΙΚΕΙΜΕΝΟ ΤΗΣ ΕΡΓΑΣΙΑΣ	3
1.3 ΔΟΜΗ ΕΡΓΑΣΙΑΣ	3
<u>ΚΕΦΑΛΑΙΟ 2 ΚΑΤΑΝΕΜΗΜΕΝΗ ΔΙΑΧΕΙΡΙΣΗ ΔΕΔΟΜΕΝΩΝ</u>	<u>4</u>
2.1 ΒΑΣΙΚΕΣ ΕΝΝΟΙΕΣ ΚΑΙ ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ	4
2.2 ΑΠΟΘΗΚΕΥΣΗ ΚΑΙ ΚΑΤΑΝΟΜΗ ΔΕΔΟΜΕΝΩΝ	4
2.3 ΑΥΤΟΝΟΜΙΑ ΚΟΜΒΩΝ ΚΑΙ ΤΟΠΙΚΗ ΔΙΑΧΕΙΡΙΣΗ ΔΕΔΟΜΕΝΩΝ	5
2.4 ΔΙΑΜΟΙΡΑΣΜΟΣ ΚΑΙ ΜΕΤΑΦΟΡΑ ΔΕΔΟΜΕΝΩΝ ΜΕΤΑΞΥ ΚΟΜΒΩΝ	6
2.5 ΥΠΑΡΧΟΥΣΕΣ ΠΡΟΣΕΓΓΙΣΕΙΣ ΔΙΑΧΕΙΡΙΣΗΣ ΔΕΔΟΜΕΝΩΝ ΣΕ ΑΥΤΟΝΟΜΟΥΣ ΚΟΜΒΟΥΣ	7
2.6 ΣΥΝΟΨΗ	7
<u>ΚΕΦΑΛΑΙΟ 3 ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ ΚΑΙ SWARM INTELLIGENCE ΣΤΗ ΔΙΑΧΕΙΡΙΣΗ ΔΕΔΟΜΕΝΩΝ</u>	<u>8</u>
3.1 ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ ΣΕ ΚΑΤΑΝΕΜΗΜΕΝΑ ΠΕΡΙΒΑΛΛΟΝΤΑ	8
3.2 SWARM INTELLIGENCE	8
3.3 ARTIFICIAL BEE COLONY (ABC)	9
3.4 ΕΦΑΡΜΟΓΕΣ SWARM INTELLIGENCE ΣΤΗ ΔΙΑΧΕΙΡΙΣΗ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΕΡΓΑΣΙΩΝ	10
3.5 ΣΥΝΟΨΗ	11
<u>ΚΕΦΑΛΑΙΟ 4 ΤΟ ΠΡΟΤΕΙΝΟΜΕΝΟ ΣΥΣΤΗΜΑ</u>	<u>12</u>
4.1 ΕΠΙΣΚΟΠΗΣΗ ΣΥΣΤΗΜΑΤΟΣ	12
4.2 ΠΡΟΕΠΕΞΕΡΓΑΣΙΑ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΑΡΧΙΚΗ ΚΑΤΑΝΟΜΗ	13
4.3 ΣΤΑΤΙΣΤΙΚΑ ΣΤΟΙΧΕΙΑ ΚΟΜΒΟΥ ΚΑΙ ΠΑΡΑΓΩΓΗ ΕΙΣΕΡΧΟΜΕΝΩΝ ΔΕΔΟΜΕΝΩΝ ...	15
4.4 ΕΝΤΟΠΙΣΜΟΣ ΚΕΝΩΝ ΔΙΑΣΤΗΜΑΤΩΝ	17
4.5 ΣΥΣΤΑΔΟΠΟΙΗΣΗ ΕΙΣΕΡΧΟΜΕΝΩΝ ΔΕΔΟΜΕΝΩΝ	20
4.6 ΑΞΙΟΛΟΓΗΣΗ ΣΥΣΤΑΔΩΝ ΚΑΙ ΑΠΟΦΑΣΗ ΚΟΜΒΟΥ	22
4.7 ΑΝΤΑΛΛΑΓΗ ΠΛΗΡΟΦΟΡΙΑΣ ΚΕΝΩΝ ΔΙΑΣΤΗΜΑΤΩΝ	24
4.8 ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ ΜΕΤΑΦΟΡΑΣ ΔΕΔΟΜΕΝΩΝ ΜΕ ABC	25
4.8.1 ΑΝΑΠΑΡΑΣΤΑΣΗ ΛΥΣΗΣ	25
4.8.2 ΑΡΧΙΚΟΠΟΙΗΣΗ ΠΛΗΘΥΣΜΟΥ ΛΥΣΕΩΝ	26
4.8.3 ΣΥΝΑΡΤΗΣΗ ΚΑΤΑΛΛΗΛΟΤΗΤΑΣ	27
4.8.4 EMPLOYED BEES	29
4.8.5 ONLOOKER BEES	30

4.8.6 SCOUT BEES	31
4.8.7 ΕΝΗΜΕΡΩΣΗ ΚΑΙ ΕΠΙΛΟΓΗ ΚΑΛΥΤΕΡΗΣ ΛΥΣΗΣ.....	32
4.9 ΕΠΙΛΟΓΗ ΚΑΙ ΠΡΑΓΜΑΤΟΠΟΙΗΣΗ ΜΕΤΑΦΟΡΩΝ.....	33
4.10 ΕΝΣΩΜΑΤΩΣΗ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΕΝΗΜΕΡΩΣΗ ΚΟΜΒΩΝ.....	34

ΚΕΦΑΛΑΙΟ 5 ΠΕΙΡΑΜΑΤΙΚΗ ΑΞΙΟΛΟΓΗΣΗ..... 36

5.1 ΠΑΡΑΜΕΤΡΟΙ ΣΥΣΤΗΜΑΤΟΣ	36
5.2 ΜΕΤΡΙΚΕΣ ΑΞΙΟΛΟΓΗΣΗΣ.....	37
5.2.1 ΤΟΠΙΚΗ ΚΑΛΥΨΗ ΔΕΔΟΜΕΝΩΝ	37
5.2.2 ΣΥΝΟΛΙΚΗ ΚΑΛΥΨΗ ΣΥΣΤΗΜΑΤΟΣ.....	38
5.3 ΠΕΙΡΑΜΑΤΙΚΑ ΣΕΝΑΡΙΑ.....	39
5.3.1 ΜΟΝΟ ΤΟΠΙΚΗ ΔΙΑΤΗΡΗΣΗ ΔΕΔΟΜΕΝΩΝ.....	39
5.3.2 ΜΟΝΟ ΜΕΤΑΦΟΡΕΣ ΜΕΣΩ ABC.....	39
5.3.3 ΠΛΗΡΕΣ ΣΥΣΤΗΜΑ.....	39
5.4 ΑΠΟΤΕΛΕΣΜΑΤΑ.....	40
5.4.1 ΑΠΟΤΕΛΕΣΜΑΤΑ ΜΟΝΟ ΤΟΠΙΚΗΣ ΔΙΑΤΗΡΗΣΗΣ ΔΕΔΟΜΕΝΩΝ	40
5.4.2 ΑΠΟΤΕΛΕΣΜΑΤΑ ΜΟΝΟ ΜΕΤΑΦΟΡΩΝ ΜΕΣΩ ABC	45
5.4.3 ΑΠΟΤΕΛΕΣΜΑΤΑ ΠΛΗΡΟΥΣ ΣΥΣΤΗΜΑΤΟΣ	50
5.5 ΣΥΓΚΡΙΤΙΚΗ ΑΞΙΟΛΟΓΗΣΗ ΠΕΙΡΑΜΑΤΩΝ.....	55

ΚΕΦΑΛΑΙΟ 6 ΣΥΜΠΕΡΑΣΜΑΤΑ..... 56

ΒΙΒΛΙΟΓΡΑΦΙΑ..... 57

ΚΕΦΑΛΑΙΟ 1 Εισαγωγή

1.1 Εισαγωγή και κίνητρο

Η ανάπτυξη κατανεμημένων συστημάτων Edge και Fog Computing έχει οδηγήσει στη χρήση πολλών υπολογιστικών κόμβων που βρίσκονται κοντά στις πηγές παραγωγής δεδομένων. Οι κόμβοι αυτοί μπορούν να επεξεργάζονται και να αποθηκεύουν δεδομένα τοπικά, όμως συχνά διαθέτουν περιορισμένους αποθηκευτικούς και υπολογιστικούς πόρους. Για αυτόν τον λόγο, η αποτελεσματική διαχείριση και κατανομή δεδομένων μεταξύ κόμβων αποτελεί σημαντικό ζήτημα [2, 5, 6].

Παράλληλα, οι κόμβοι ενός κατανεμημένου συστήματος μπορεί να έχουν διαφορετικές ανάγκες και να διαθέτουν διαφορετικά δεδομένα. Η μεταφορά δεδομένων μεταξύ κόμβων μπορεί να βοηθήσει στην καλύτερη κάλυψη των αναγκών τους, αλλά δημιουργεί κόστος επικοινωνίας. Επομένως, είναι σημαντικό να αποφασίζεται ποια δεδομένα πρέπει να διατηρούνται τοπικά και σε ποιους κόμβους πρέπει να μεταφέρονται τα υπόλοιπα δεδομένα.

Για την αντιμετώπιση τέτοιων προβλημάτων μπορούν να χρησιμοποιηθούν τεχνικές υπολογιστικής νοημοσύνης, οι οποίες επιτρέπουν την αναζήτηση κατάλληλων λύσεων σε προβλήματα με μεγάλο αριθμό πιθανών επιλογών. Ιδιαίτερο ενδιαφέρον παρουσιάζουν οι αλγόριθμοι Swarm Intelligence, οι οποίοι βασίζονται στη συλλογική συμπεριφορά ενός συνόλου οντοτήτων και έχουν χρησιμοποιηθεί σε προβλήματα κατανομής δεδομένων και εργασιών [6, 7].

1.2 Στόχος και αντικείμενο της εργασίας

Στόχος της παρούσας εργασίας είναι η μελέτη και ανάπτυξη μίας προσέγγισης για τον διαμοιρασμό δεδομένων μεταξύ αυτόνομων κόμβων σε ένα καταναμεμημένο περιβάλλον. Κάθε κόμβος διαχειρίζεται τοπικά τα δεδομένα του και αξιολογεί τα νέα εισερχόμενα δεδομένα με βάση την υπάρχουσα κατανομή των δεδομένων του. Η εργασία εστιάζει κυρίως στη διαχείριση και κατανομή δεδομένων.

Για την υποστήριξη της διαδικασίας διαμοιρασμού χρησιμοποιήθηκε ο αλγόριθμος Artificial Bee Colony. Ο αλγόριθμος χρησιμοποιείται για την επιλογή κατάλληλων κόμβων στους οποίους μπορούν να μεταφερθούν δεδομένα που δεν διατηρούνται τοπικά, με στόχο την καλύτερη κάλυψη των αναγκών των κόμβων.

1.3 Δομή εργασίας

Η εργασία οργανώνεται σε έξι κεφάλαια. Στο πρώτο κεφάλαιο γίνεται παρουσίαση του αντικειμένου, του κινήτρου και του στόχου της εργασίας. Στο δεύτερο κεφάλαιο παρουσιάζονται βασικές έννοιες της καταναμεμημένης διαχείρισης δεδομένων, η τοπική διαχείριση στους κόμβους και υπάρχουσες προσεγγίσεις κατανομής και διαμοιρασμού δεδομένων.

Στο τρίτο κεφάλαιο παρουσιάζονται τεχνικές τεχνητής νοημοσύνης και Swarm Intelligence, με έμφαση στον αλγόριθμο Artificial Bee Colony και σε εφαρμογές του στη διαχείριση δεδομένων και εργασιών. Στο τέταρτο κεφάλαιο περιγράφεται το προτεινόμενο σύστημα και ο τρόπος με τον οποίο πραγματοποιείται η τοπική διαχείριση και ο διαμοιρασμός των δεδομένων μεταξύ των κόμβων.

Στο πέμπτο κεφάλαιο παρουσιάζεται η πειραματική αξιολόγηση του συστήματος και η σύγκριση των διαφορετικών προσεγγίσεων που εξετάστηκαν. Τέλος, στο έκτο κεφάλαιο παρουσιάζονται τα βασικά συμπεράσματα της εργασίας.

ΚΕΦΑΛΑΙΟ 2 Κατανεμημένη Διαχείριση Δεδομένων

2.1 Βασικές έννοιες και χαρακτηριστικά

Ένα κατανεμημένο σύστημα αποτελείται από ένα σύνολο διασυνδεδεμένων υπολογιστικών μονάδων. Οι μονάδες αυτές συνήθως αναφέρονται ως κόμβοι και έχουν τη δυνατότητα να λειτουργούν αυτόνομα, διαθέτοντας περιορισμένη πληροφορία σχετικά με την κατάσταση των υπόλοιπων κόμβων του συστήματος. Ωστόσο, οι κόμβοι του συστήματος μπορούν να ανταλλάσσουν μηνύματα και πληροφορίες με σκοπό την επίτευξη κάποιου κοινού στόχου [1, 2].

Στην αποκεντρωμένη διαχείριση δεδομένων έχουν προταθεί διαφορετικές προσεγγίσεις, μεταξύ των οποίων και τα Peer-to-Peer συστήματα. Σε αυτά, το σύστημα αποτελείται από αυτόνομους κόμβους, όπου κάθε κόμβος μπορεί να διατηρεί τοπικά τα δικά του δεδομένα και να συνδέεται με άλλους κόμβους. Η συνεργασία αυτή μπορεί να πραγματοποιείται χωρίς την ύπαρξη ενός ενιαίου καθολικού σχήματος δεδομένων, διατηρώντας έτσι την αυτονομία των επιμέρους κόμβων [3].

Επομένως, η λειτουργία ενός κατανεμημένου συστήματος βασίζεται στην αυτονομία των κόμβων, στην τοπική διαχείριση και στη συνεργασία μεταξύ των κόμβων.

2.2 Αποθήκευση και κατανομή δεδομένων

Σε ένα κατανεμημένο σύστημα, τα δεδομένα μπορούν να χωρίζονται σε μικρότερα τμήματα και κάθε τμήμα μπορεί να αποθηκεύεται σε διαφορετικό κόμβο. Η διαδικασία αυτή ονομάζεται κατακερματισμός (Data fragmentation). Στη βιβλιογραφία διακρίνονται τρεις βασικές μορφές κατακερματισμού: ο οριζόντιος (Horizontal Fragmentation), ο κάθετος (Vertical Fragmentation) και ο υβριδικός (Hybrid Fragmentation). Στον οριζόντιο κατακερματισμό ένα σύνολο δεδομένων χωρίζεται σε μικρότερα υποσύνολα με βάση συγκεκριμένες συνθήκες, ενώ στον κάθετο κατακερματισμό γίνεται ομαδοποίηση των χαρακτηριστικών που χρησιμοποιούνται συχνά μαζί [12].

Εφόσον γίνει κατακερματισμός των δεδομένων, πρέπει να αποφασιστεί σε ποιον κόμβο θα τοποθετηθεί κάθε τμήμα των δεδομένων. Η διαδικασία αυτή ονομάζεται κατανομή (Data allocation). Η κατανομή μπορεί επίσης να διακριθεί σε redundant και non-redundant. Στη non-redundant κατανομή κάθε τμήμα δεδομένων αποθηκεύεται σε έναν μόνο κόμβο του συστήματος, ενώ στη redundant κατανομή μπορούν να υπάρχουν αντίγραφα του ίδιου τμήματος σε περισσότερους από έναν κόμβους [12]. Η κατανομή μπορεί να είναι στατική ή δυναμική. Στη στατική κατανομή η θέση των τμημάτων δεδομένων στους κόμβους παραμένει σταθερή, ενώ στη δυναμική κατανομή μπορεί να μεταβάλλεται ανάλογα με τις αλλαγές στον τρόπο πρόσβασης στα δεδομένα. Η επιλογή των κόμβων στους οποίους θα τοποθετηθούν τα δεδομένα είναι σημαντική καθώς μπορεί να επηρεάσει την απόδοση του συστήματος [1, 12].

Επομένως, ο τρόπος με τον οποίο γίνεται η κατανομή των δεδομένων στους κόμβους επηρεάζει και την επικοινωνία μεταξύ τους. Η αποθήκευση των δεδομένων κοντά στο σημείο όπου χρησιμοποιούνται μπορεί να μειώσει την ανάγκη για απομακρυσμένη πρόσβαση [1]. Στην [13], το πρόβλημα της κατανομής των τμημάτων δεδομένων αντιμετωπίζεται ως πρόβλημα βελτιστοποίησης. Για την επιλογή των κόμβων στους οποίους θα τοποθετηθούν τα τμήματα δεδομένων χρησιμοποιείται ο αλγόριθμος Biogeography-Based Optimization (BBO). Στόχος της μεθόδου είναι η εύρεση μίας κατανομής που μειώνει το συνολικό κόστος μεταφοράς δεδομένων κατά την εκτέλεση ερωτημάτων [13].

2.3 Αυτονομία κόμβων και τοπική διαχείριση δεδομένων

Σε περιβάλλοντα αυτόνομων κόμβων, κάθε κόμβος μπορεί να συλλέγει και να επεξεργάζεται δεδομένα τοπικά. Επειδή οι αποθηκευτικοί και υπολογιστικοί πόροι των κόμβων μπορεί να είναι περιορισμένοι, είναι σημαντικό οι κόμβοι να διατηρούν τοπικά μόνο τα δεδομένα τα οποία θεωρούν απαραίτητα [4, 5].

Η επιλογή των δεδομένων που θα αποθηκευτούν τοπικά μπορεί να γίνει με τη χρήση μεθόδων επιλογής και φίλτρων δεδομένων. Σε τέτοιες προσεγγίσεις, κάθε κόμβος διατηρεί δεδομένα που ταιριάζουν στις τοπικές του ανάγκες ή στα δεδομένα που διαθέτει ήδη. Αν τα νέα

δεδομένα δεν ταιριάζουν στα τοπικά κριτήρια, μπορεί να γίνει κατανομή τους σε άλλους κόμβους του συστήματος [5].

Μία αποκεντρωμένη προσέγγιση για τη δυναμική διαχείριση των δεδομένων παρουσιάζεται στην [14]. Η μέθοδος παρακολουθεί τον τρόπο με τον οποίο οι κόμβοι προσπελαύνουν τα δεδομένα και χρησιμοποιεί το πρόσφατο ιστορικό πρόσβασης για να προσαρμόζει τον κατακερματισμό και την κατανομή τους. Με αυτόν τον τρόπο επιδιώκεται η αύξηση των τοπικών προσβάσεων και η μείωση της επικοινωνίας με απομακρυσμένους κόμβους [14].

2.4 Διαμοιρασμός και μεταφορά δεδομένων μεταξύ κόμβων

Σε ένα κατανεμημένο σύστημα, οι κόμβοι μπορούν να ανταλλάσσουν δεδομένα μεταξύ τους για την υποστήριξη των εργασιών του συστήματος. Όταν ένα δεδομένο γίνει αποδεκτό για τοπική αποθήκευση, μπορούν να επιλεγούν κατάλληλοι κόμβοι στους οποίους θα αντιγραφεί. Για αυτόν τον λόγο, είναι απαραίτητο να επιλέγονται κατάλληλα οι κόμβοι στους οποίους θα μεταφερθούν τα δεδομένα [4].

Η μεταφορά δεδομένων μπορεί επίσης να πραγματοποιείται δυναμικά ανάλογα με τον τρόπο πρόσβασης σε αυτά. Στην [14], ένα τμήμα δεδομένων μπορεί να μεταφερθεί σε διαφορετικό κόμβο όταν με βάση το πρόσφατο ιστορικό πρόσβασης, η μεταφορά του θα μειώσει το κόστος επικοινωνίας. Η μέθοδος μπορεί επίσης να χωρίσει ένα υπάρχον τμήμα και να μεταφέρει το νέο τμήμα σε διαφορετικό κόμβο [14].

Η επιλογή του κόμβου στον οποίο θα μεταφερθούν τα δεδομένα μπορεί να βασίζεται σε διαφορετικά κριτήρια, όπως η σχέση των δεδομένων με τα δεδομένα που διαθέτει ο κόμβος και το κόστος επικοινωνίας μεταξύ των κόμβων. Η αποφυγή μεταφορών με υψηλό κόστος μπορεί να μειώσει την επιβάρυνση του δικτύου και να κάνει τη διαδικασία κατανομής των δεδομένων πιο αποδοτική [4].

Στην [13], η επιλογή των κόμβων για την τοποθέτηση των τμημάτων δεδομένων πραγματοποιείται με τη χρήση του αλγορίθμου BBO. Ο αλγόριθμος αναζητά την κατανομή των τμημάτων στους διαθέσιμους κόμβους που μειώνει το συνολικό κόστος μεταφοράς δεδομένων κατά την εκτέλεση των ερωτημάτων [13].

2.5 Υπάρχουσες προσεγγίσεις διαχείρισης δεδομένων σε αυτόνομους κόμβους

Στη βιβλιογραφία έχουν προταθεί διαφορετικές προσεγγίσεις για τη διαχείριση δεδομένων σε κατανεμημένα περιβάλλοντα με αυτόνομους κόμβους. Ορισμένες προσεγγίσεις δίνουν έμφαση στην αυτονομία των κόμβων και την τοπική διαχείριση των δεδομένων, όπως τα Peer-to-Peer συστήματα [3] και οι μέθοδοι [4, 5].

Άλλες εργασίες επικεντρώνονται στον κατακερματισμό και στην κατανομή των δεδομένων. Στην [12] παρουσιάζονται διαφορετικές κατηγορίες κατακερματισμού και κατανομής, ενώ στην [13] η κατανομή των τμημάτων αντιμετωπίζεται ως πρόβλημα βελτιστοποίησης με στόχο τη μείωση του κόστους μεταφοράς δεδομένων.

Παράλληλα, σε δυναμικές προσεγγίσεις όπως στην [14], ο κατακερματισμός και η κατανομή μπορούν να προσαρμόζονται ανάλογα με το πρόσφατο ιστορικό πρόσβασης των κόμβων.

Συνεπώς, οι υπάρχουσες μέθοδοι διαφέρουν κυρίως ως προς το αν οι αποφάσεις είναι στατικές ή δυναμικές, κεντρικές ή αποκεντρωμένες, καθώς και ως προς τα κριτήρια που χρησιμοποιούνται για την επιλογή της κατανομής των δεδομένων.

2.6 Σύνοψη

Συνολικά, η κατανεμημένη διαχείριση δεδομένων βασίζεται στην κατανομή των δεδομένων μεταξύ διαφορετικών κόμβων και στη δυνατότητα κάθε κόμβου να διαχειρίζεται τοπικά τα δεδομένα που διαθέτει. Σε περιβάλλοντα με αυτόνομους κόμβους, η επιλογή των δεδομένων που θα αποθηκευτούν τοπικά και η απόφαση για τη μεταφορά τους σε άλλους κόμβους αποτελούν σημαντικά ζητήματα.

Οι υπάρχουσες προσεγγίσεις δείχνουν ότι οι αποφάσεις αυτές μπορούν να βασίζονται στα χαρακτηριστικά των δεδομένων, στις ανάγκες των κόμβων και στο κόστος επικοινωνίας μεταξύ τους. Για την αποτελεσματική λήψη τέτοιων αποφάσεων μπορούν να χρησιμοποιούνται τεχνικές υπολογιστικής νοημοσύνης, οι οποίες παρουσιάζονται στο επόμενο κεφάλαιο.

ΚΕΦΑΛΑΙΟ 3 Τεχνητή νοημοσύνη και Swarm Intelligence στη διαχείριση δεδομένων

3.1 Τεχνητή νοημοσύνη σε καταναμεημένα περιβάλλοντα

Σε καταναμεημένα περιβάλλοντα, όπως τα Fog και Edge συστήματα, οι κόμβοι μπορεί να είναι ετερογενείς και να διαθέτουν περιορισμένους αποθηκευτικούς και υπολογιστικούς πόρους. Για αυτόν τον λόγο, η αποτελεσματική κατανομή δεδομένων και εργασιών μεταξύ των κόμβων αποτελεί σημαντικό πρόβλημα. Για την αντιμετώπιση τέτοιων προβλημάτων έχουν προταθεί τεχνικές τεχνητής νοημοσύνης και αλγόριθμοι βελτιστοποίησης [5, 6].

Οι τεχνικές τεχνητής νοημοσύνης που έχουν χρησιμοποιηθεί σε τέτοια προβλήματα μπορούν να χωριστούν σε διαφορετικές κατηγορίες, όπως εξελικτικοί αλγόριθμοι, αλγόριθμοι μηχανικής μάθησης και συνδυαστικές προσεγγίσεις. Στην κατηγορία των εξελικτικών αλγορίθμων περιλαμβάνονται και τεχνικές Swarm Intelligence, όπως Ant Colony Optimization (ACO), Artificial Bee Colony (ABC) και Particle Swarm Optimization (PSO) [6].

3.2 Swarm Intelligence

Η συλλογική νοημοσύνη (Swarm Intelligence) βασίζεται στη συλλογική συμπεριφορά ενός συνόλου απλών οντοτήτων, οι οποίες συνεργάζονται και ανταλλάσσουν πληροφορίες μεταξύ τους. Παρόλο που κάθε οντότητα ξεχωριστά μπορεί να διαθέτει περιορισμένες δυνατότητες, η συνεργασία μεταξύ τους μπορεί να οδηγήσει σε μία πιο οργανωμένη συλλογική συμπεριφορά. Οι αλγόριθμοι Swarm Intelligence χαρακτηρίζονται από καταναμεημένη λειτουργία και μπορούν να χρησιμοποιηθούν για την επίλυση σύνθετων προβλημάτων βελτιστοποίησης. [7]

Ένα σημαντικό χαρακτηριστικό των αλγορίθμων Swarm Intelligence είναι η δυνατότητα προσαρμογής της συμπεριφοράς των οντοτήτων κατά τη διαδικασία αναζήτησης. Οι αλγόριθμοι αυτοί παρουσιάζουν χαρακτηριστικά όπως προσαρμοστικότητα, ανθεκτικότητα και δυνατότητα αναζήτησης, τα οποία τους καθιστούν κατάλληλους για σύνθετα προβλήματα βελτιστοποίησης [7].

Η λειτουργία των αλγορίθμων Swarm Intelligence είναι συχνά εμπνευσμένη από τη συμπεριφορά ομάδων οργανισμών στη φύση, όπως πουλιά, ψάρια και μέλισσες. Με βάση αυτές τις συμπεριφορές έχουν αναπτυχθεί διαφορετικοί αλγόριθμοι όπως PSO, ACO και ABC [7].

3.3 Artificial Bee Colony (ABC)

Ο αλγόριθμος Artificial Bee Colony αποτελεί έναν αλγόριθμο βελτιστοποίησης σχεδιασμένο πάνω στη συμπεριφορά ενός σμήνους μελισσών κατά την αναζήτηση τροφής. Κάθε πηγή τροφής αντιπροσωπεύει μία πιθανή λύση του προβλήματος και η ποσότητα του νέктar εκφράζει την ποιότητα ή την τιμή καταλληλότητας της συγκεκριμένης λύσης [8].

Ο πληθυσμός των μελισσών αποτελείται από τρεις κατηγορίες: employed bees, onlooker bees και scout bees. Τα employed bees εξετάζουν τις υπάρχουσες λύσεις, τα onlooker bees επιλέγουν λύσεις με βάση τις πληροφορίες που λαμβάνουν από τα employed bees και τα scout bees πραγματοποιούν τυχαία αναζήτηση για νέες πιθανές λύσεις [8].

Κατά την αρχικοποίηση του αλγορίθμου δημιουργείται ένα αρχικό σύνολο λύσεων. Σε κάθε μία από αυτές τις λύσεις αντιστοιχεί ένα employed bee το οποίο είναι υπεύθυνο για την εξερεύνηση της. Κάθε employed bee δημιουργεί μία νέα λύση και μετά την αξιολόγηση της, συγκρίνεται με την αντίστοιχη υπάρχουσα λύση. Αν η ποιότητα της νέας λύσης είναι καλύτερη, τότε αντικαθιστά την παλιά, διαφορετικά διατηρείται η υπάρχουσα λύση. Αυτός ο τρόπος επιλογής ονομάζεται greedy selection [8].

Μετά την εξερεύνηση από τα employed bees, οι πληροφορίες σχετικά με την ποιότητα των λύσεων γίνονται διαθέσιμες στα onlooker bees. Η πιθανότητα επιλογής μίας λύσης αυξάνεται ανάλογα με την ποιότητα της. Ως αποτέλεσμα, οι καλύτερες λύσεις του αλγορίθμου έχουν μεγαλύτερη πιθανότητα να επιλεγούν από τα onlooker bees. Μετά από την επιλογή μίας λύσης, ένα onlooker bee δημιουργεί μία νέα λύση, την αξιολογεί και ακολουθεί αντίστοιχη διαδικασία επιλογής με τα employed bees της προηγούμενης φάσης [8].

Όταν μία λύση αποτύχει να βελτιωθεί για έναν προκαθορισμένο αριθμό επαναλήψεων, ο αλγόριθμος την εγκαταλείπει και την αντικαθιστά με μία νέα τυχαία λύση. Αυτός είναι ο ρόλος των scout bees του αλγορίθμου [8].

Η διαδικασία αυτή επαναλαμβάνεται μέχρι να ικανοποιηθεί το κριτήριο τερματισμού. Με αυτόν τον τρόπο, τα employed και onlooker bees επικεντρώνονται στην αξιοποίηση των υπάρχουσών λύσεων, ενώ τα scout bees συμβάλλουν στην εξερεύνηση νέων λύσεων [8].

3.4 Εφαρμογές Swarm Intelligence στη διαχείριση δεδομένων και εργασιών

Στη βιβλιογραφία έχουν προταθεί αλγόριθμοι Swarm Intelligence για προβλήματα κατανομής δεδομένων σε κατανεμημένα συστήματα. Στην [9] προτείνεται μία μέθοδος βασισμένη στον αλγόριθμο Artificial Bee Colony για την κατανομή δεδομένων σε ετερογενές κατανεμημένο περιβάλλον. Η μέθοδος λαμβάνει υπόψη το κόστος επικοινωνίας, το υπολογιστικό κόστος και το κόστος του δικτύου, με στόχο τη μείωση του συνολικού κόστους του συστήματος και τη βελτίωση της κατανομής των δεδομένων [9]. Αντίστοιχα, στην [10] προτείνεται μία μέθοδος για την τοποθέτηση δεδομένων σε περιβάλλον Mobile Edge Computing, η οποία συνδυάζει Particle Swarm Optimization και Genetic Algorithm και λαμβάνει υπόψη το κόστος μετάδοσης και αποθήκευσης των δεδομένων, καθώς και περιορισμούς καθυστέρησης [10].

Στην [15] προτείνεται μία μέθοδος βασισμένη στον αλγόριθμο Ant Colony Optimization (ACO) για την κατανομή τμημάτων δεδομένων σε κατανεμημένες βάσεις δεδομένων. Η μέθοδος επιδιώκει να μειώσει τις μεταφορές δεδομένων μεταξύ των κόμβων και τον συνολικό χρόνο

απόκρισης των συναλλαγών, λαμβάνοντας υπόψη τους περιορισμούς αποθηκευτικού χώρου των κόμβων [15]. Στην [16] προτείνεται μία προσέγγιση βασισμένη στον αλγόριθμο Particle Swarm Optimization (PSO) για την κατανομή των τμημάτων δεδομένων σε διαφορετικούς κόμβους. Ο αλγόριθμος χρησιμοποιείται για την επιλογή των κατάλληλων κόμβων στους οποίους θα τοποθετηθούν τα τμήματα, με στόχο τη μείωση του συνολικού κόστους μετάδοσης δεδομένων και του κόστους εκτέλεσης των συναλλαγών [16].

Εκτός από την κατανομή δεδομένων, οι αλγόριθμοι Swarm Intelligence έχουν χρησιμοποιηθεί και για την κατανομή εργασιών. Στην [11] προτείνεται μία προσέγγιση βασισμένη στον αλγόριθμο Artificial Bee Colony για computation offloading σε περιβάλλον Edge Computing. Ο αλγόριθμος χρησιμοποιείται για τη διαχείριση του φόρτου στους Edge servers και την επιλογή κατάλληλων πόρων για την εκτέλεση των εργασιών, με στόχο τη μείωση της καθυστέρησης και του χρόνου απόκρισης [11]. Στην [17] προτείνεται μία προσέγγιση βασισμένη στον Multi-Objective Grey Wolf Optimizer (MGWO) για την κατανομή εργασιών σε περιβάλλον Cloud-Fog Computing. Ο αλγόριθμος χρησιμοποιείται για την επιλογή κατάλληλων υπολογιστικών πόρων για την εκτέλεση των εργασιών, με στόχο τη μείωση της καθυστέρησης και της κατανάλωσης ενέργειας [17].

3.5 Σύνοψη

Σε αυτό το κεφάλαιο παρουσιάστηκαν τεχνικές τεχνητής νοημοσύνης και Swarm Intelligence που μπορούν να χρησιμοποιηθούν για την αντιμετώπιση προβλημάτων κατανομής δεδομένων και εργασιών σε κατανεμημένα περιβάλλοντα. Από τη βιβλιογραφία προκύπτει ότι οι αλγόριθμοι Swarm Intelligence έχουν χρησιμοποιηθεί τόσο για την κατανομή και τοποθέτηση δεδομένων όσο και για computation offloading και κατανομή εργασιών σε Edge και κατανεμημένα συστήματα. Οι προσεγγίσεις αυτές δείχνουν ότι τέτοιες τεχνικές μπορούν να υποστηρίξουν τη λήψη αποφάσεων όταν πρέπει να επιλεγούν κατάλληλοι κόμβοι λαμβάνοντας υπόψη διαφορετικά κριτήρια και περιορισμούς. Στο επόμενο κεφάλαιο παρουσιάζεται το προτεινόμενο σύστημα και ο τρόπος με τον οποίο χρησιμοποιείται ο αλγόριθμος Artificial Bee Colony για τη διαχείριση και τον διαμοιρασμό δεδομένων μεταξύ αυτόνομων κόμβων.

ΚΕΦΑΛΑΙΟ 4 Το Προτεινόμενο Σύστημα

4.1 Επισκόπηση Συστήματος

Το προτεινόμενο σύστημα αποτελείται από δέκα αυτόνομους κόμβους οι οποίοι αποθηκεύουν, επεξεργάζονται και ανταλλάσσουν δεδομένα μεταξύ τους έχοντας ελάχιστη επικοινωνία. Κάθε κόμβος αποθηκεύει δεδομένα για τέσσερα διαφορετικά αριθμητικά χαρακτηριστικά, τα οποία επεξεργάζεται ανεξάρτητα.

Κάθε κόμβος υπολογίζει και αποθηκεύει πληροφορίες και στατιστικά σχετικά με την κατάσταση των αποθηκευμένων δεδομένων του καθώς και τα κενά διαστήματα για τα οποία δεν έχει αποθηκευμένα δεδομένα. Ταυτόχρονα, χρησιμοποιεί αυτές τις πληροφορίες για την παραγωγή νέων συνθετικών δεδομένων με σκοπό την προσομοίωση εισερχόμενων ροών δεδομένων (Data Streams). Τα παραγόμενα δεδομένα οργανώνονται σε συστάδες (Clusters) και ο κόμβος τα αξιολογεί με βάση το κατά πόσο ταιριάζουν στα ήδη αποθηκευμένα δεδομένα του.

Οι συστάδες που εμφανίζουν υψηλό ποσοστό ταιριάσματος με έναν κόμβο διατηρούνται σε μία λίστα με σκοπό να ενσωματωθούν στα τοπικά δεδομένα του. Αντιθέτως, οι συστάδες που δεν ταιριάζουν σε έναν κόμβο διατηρούνται σε διαφορετική λίστα ως υποψήφιες προς αποστολή σε άλλους κόμβους του συστήματος. Κάθε κόμβος χρησιμοποιεί έναν αλγόριθμο ABC (Artificial Bee Colony) για να εντοπίσει ποιες συστάδες μπορούν να αποσταλούν σε ποιους κόμβους, με στόχο τη μεγαλύτερη κάλυψη κενών διαστημάτων λαμβάνοντας υπόψη το κόστος μεταφοράς μίας συστάδας από έναν κόμβο προς έναν άλλο.

Μετά την ολοκλήρωση των μεταφορών καθένας από τους κόμβους ενσωματώνει τα δεδομένα που αποφάσισε να διατηρήσει και τα δεδομένα που έλαβε από τους υπόλοιπους κόμβους. Αφού γίνει η ενσωμάτωση δεδομένων ο κόμβος ενημερώνει τις στατιστικές πληροφορίες του καθώς και τα κενά διαστήματα του. Η διαδικασία γίνεται επαναληπτικά, επιτρέποντας στους κόμβους να γεμίζουν τα κενά διαστήματά τους.

4.2 Προεπεξεργασία δεδομένων και αρχική κατανομή

Για την αρχικοποίηση του συστήματος χρησιμοποιήθηκε το Iris Species σύνολο δεδομένων. Το αρχικό σύνολο δεδομένων αποτελείται από πέντε χαρακτηριστικά, τέσσερα από αυτά είναι αριθμητικά και ένα αλφαριθμητικό το οποίο αναπαριστά το είδος του φυτού, για καθένα από αυτά τα χαρακτηριστικά, το αρχικό σύνολο δεδομένων περιέχει 150 εγγραφές. Στη συγκεκριμένη υλοποίηση γίνεται χρήση μόνο των αριθμητικών χαρακτηριστικών. Όλα τα αριθμητικά χαρακτηριστικά περιέχουν τιμές με μέγιστο ένα δεκαδικό ψηφίο και το καθένα έχει διαφορετικές ελάχιστες και μέγιστες τιμές. Σε όλα τα στάδια της υλοποίησης καθένα από αυτά τα χαρακτηριστικά διαχειρίζεται ανεξάρτητα από τα υπόλοιπα.

Πριν γίνει η αρχική κατανομή των δεδομένων στους κόμβους πραγματοποιείται προεπεξεργασία. Όλες οι τιμές του συνόλου πολλαπλασιάζονται επί εκατό και στρογγυλοποιούνται σε ακέραιες.

$$x' = \text{round}(100x)$$

Μετά την κλιμάκωση των δεδομένων οι δεκαδικές τιμές έχουν μετατραπεί σε ακέραιες τιμές. Για τη συνέχεια της υλοποίησης οι κόμβοι επεξεργάζονται και διαχειρίζονται δεδομένα ακέραιων τιμών. Παράλληλα, το σύστημα υπολογίζει για κάθε χαρακτηριστικό τη συνολική ελάχιστη και μέγιστη τιμή από το κλιμακωμένο σύνολο δεδομένων. Οι τιμές αυτές αποθηκεύονται σε πίνακες και είναι κοινές για όλους τους κόμβους.

$$\min_{global} = [\min_1^{global}, \min_2^{global}, \min_3^{global}, \min_4^{global}]$$

$$\max_{global} = [\max_1^{global}, \max_2^{global}, \max_3^{global}, \max_4^{global}]$$

Τελικά για να γίνει η κατανομή στους κόμβους πρέπει να χωριστεί το κλιμακωμένο σύνολο δεδομένων, το οποίο αποτελείται από M εγγραφές σε K υποσύνολα, όπου K είναι ο αριθμός των κόμβων του συστήματος. Στο τέλος της διαδικασίας κάθε κόμβος θα έχει αποθηκεύσει N εγγραφές για κάθε χαρακτηριστικό. Στη συγκεκριμένη υλοποίηση το σύνολο δεδομένων έχει 150 εγγραφές και το σύστημα περιέχει 10 κόμβους.

$$N = \frac{M}{K} = \frac{150}{10} = 15$$

Τα δεδομένα κατανέμονται στους κόμβους διαδοχικά, επομένως σε κάθε κόμβο θα αποθηκευτεί διαφορετικό υποσύνολο των δεδομένων. Ο τρόπος που τα δεδομένα αποθηκεύονται μέσα σε έναν κόμβο δεν είναι με τη μορφή ενός δισδιάστατου πίνακα. Αντίθετα ο κόμβος δημιουργεί μία λίστα με τέσσερις εσωτερικές λίστες :

$$"node_data" = [L_1, L_2, L_3, L_4]$$

όπου κάθε μία από τις εσωτερικές λίστες χρησιμοποιείται για την αποθήκευση των δεδομένων κάθε χαρακτηριστικού,

$$L_f = [x_1, x_2, \dots, x_n]$$

με n το πλήθος των δεδομένων για το χαρακτηριστικό f

4.3 Στατιστικά στοιχεία κόμβου και παραγωγή εισερχόμενων δεδομένων

Μετά την αρχική κατανομή των δεδομένων στους κόμβους, κάθε κόμβος πρέπει να υπολογίσει κάποια βασικά στατιστικά στοιχεία από τα δεδομένα που αποθήκευσε τοπικά πριν μπορέσει να προχωρήσει στην παραγωγή δεδομένων. Οι υπολογισμοί γίνονται ανεξάρτητα για καθένα από τα χαρακτηριστικά. Κάθε κόμβος υπολογίζει για κάθε χαρακτηριστικό τη μέση τιμή, τυπική απόκλιση, ελάχιστη τιμή και μέγιστη τιμή. Ο τρόπος που ο κόμβος αποθηκεύει τα παραπάνω στατιστικά στοιχεία είναι με χρήση διαφορετικού μονοδιάστατου πίνακα για το καθένα από αυτά. Κάθε πίνακας περιέχει τέσσερις τιμές όπου η κάθε μία αντιστοιχεί σε ένα χαρακτηριστικό των δεδομένων.

$$\mu_{node} = [\mu_1, \mu_2, \mu_3, \mu_4]$$

$$\sigma_{node} = [\sigma_1, \sigma_2, \sigma_3, \sigma_4]$$

$$\min_{node} = [\min_1, \min_2, \min_3, \min_4]$$

$$\max_{node} = [\max_1, \max_2, \max_3, \max_4]$$

Μετά τον υπολογισμό των στατιστικών στοιχείων ο κόμβος μπορεί να προχωρήσει στην παραγωγή νέων συνθετικών δεδομένων, τα οποία θα χρησιμοποιήσει για την προσομοίωση των εισερχόμενων δεδομένων προς τον κόμβο. Η παραγωγή νέων δεδομένων γίνεται ανεξάρτητα για κάθε χαρακτηριστικό του κόμβου με τη χρήση κανονικής κατανομής η οποία βασίζεται στη μέση τιμή και τυπική απόκλιση. Στη συγκεκριμένη υλοποίηση ο κόμβος παράγει 20 νέα δεδομένα για κάθε χαρακτηριστικό.

$$X_f \sim N(\mu_f, \sigma_f^2)$$

όπου :

- X_f : νέα τιμή για το χαρακτηριστικό f
- μ_f : τοπική μέση τιμή
- σ_f : τοπική τυπική απόκλιση

Ένας κόμβος δεν πρέπει να παράγει τιμές μικρότερες από την ελάχιστη ή μεγαλύτερες από τη μέγιστη τοπικά αποθηκευμένη τιμή του

αντίστοιχου χαρακτηριστικού. Για αυτόν τον λόγο, μετά την παραγωγή, τα νέα δεδομένα που βρίσκονται εκτός ορίων περιορίζονται στην αντίστοιχη ελάχιστη ή μέγιστη τιμή του κόμβου.

$$X'_f = \min(\max_f, \max(\min_f, X_f))$$

όπου :

- X_f : αρχική παραγόμενη τιμή
- $\min_f$: ελάχιστη τοπικά αποθηκευμένη τιμή του f
- $\max_f$: μέγιστη τοπικά αποθηκευμένη τιμή του f
- X'_f : τελική τιμή μετά τον περιορισμό

πρακτικά :

- αν $X_f < \min_f$, τότε $X'_f = \min_f$
- αν $X_f > \max_f$, τότε $X'_f = \max_f$
- αλλιώς $X'_f = X_f$

Τέλος, οι παραγόμενες τιμές στρογγυλοποιούνται σε ακέραιες, ώστε να είναι συνεπείς με τα υπόλοιπα δεδομένα.

$$X''_f = \text{round}(X'_f)$$

όπου X''_f η τελική τιμή που θα αποθηκευτεί στη λίστα παραγόμενων δεδομένων.

4.4 Εντοπισμός κενών διαστημάτων

Κάθε κόμβος εντοπίζει τα κενά διαστήματα στα δεδομένα του, δηλαδή τις περιοχές για τις οποίες δεν διαθέτει αποθηκευμένα δεδομένα. Τα κενά διαστήματα αποτελούν την πληροφορία που ανταλλάσσουν οι κόμβοι μεταξύ τους και είναι βασικό χαρακτηριστικό του αλγορίθμου βελτιστοποίησης. Για τον εντοπισμό των κενών διαστημάτων χρησιμοποιούνται τα τοπικά αποθηκευμένα δεδομένα και οι καθολικές ελάχιστες και μέγιστες τιμές κάθε χαρακτηριστικού.

Η διαδικασία υπολογισμού των κενών διαστημάτων γίνεται ξεχωριστά για κάθε χαρακτηριστικό. Κάθε χαρακτηριστικό έχει διαφορετική καθολική ελάχιστη και μέγιστη τιμή, με αποτέλεσμα να διαφέρει το εύρος των τιμών και τα όρια των κενών διαστημάτων. Για αυτόν τον λόγο ο εντοπισμός των κενών διαστημάτων γίνεται ξεχωριστά για το καθένα.

Κατά την εκκίνηση της διαδικασίας, η καθολική ελάχιστη και μέγιστη τιμή προστίθενται προσωρινά στα αποθηκευμένα δεδομένα ενός κόμβου για κάθε χαρακτηριστικό αντίστοιχα.

$$L'_f = \text{sort}([min_f^{global}, L_f, max_f^{global}])$$

Στη συνέχεια γίνεται ταξινόμηση σε αύξουσα σειρά. Ο τρόπος που γίνεται ο εντοπισμός των κενών διαστημάτων είναι με διαδοχική σύγκριση των πλέον ταξινομημένων αποθηκευμένων τιμών. Όπως αναφέρθηκε παραπάνω, οι κόμβοι επεξεργάζονται και διαχειρίζονται μόνο ακέραια δεδομένα, οπότε θεωρείται ότι υπάρχει κενό διάστημα ανάμεσα σε δύο τιμές όταν η διαφορά τους είναι μεγαλύτερη του ένα.

$$x_{i+1} - x_i > 1$$

Για παράδειγμα :

- 430 και 431 $\rightarrow$ διαφορά 1 $\rightarrow$ δεν υπάρχει κενό
- 430 και 432 $\rightarrow$ διαφορά 2 $\rightarrow$ υπάρχει κενό

Στην υλοποίηση, για τον εντοπισμό κενών χρησιμοποιείται κατώφλι (threshold) ίσο με 1.5. Επειδή οι τιμές είναι ακέραιες, η συνθήκη διαφοράς μεγαλύτερης του 1.5 είναι ισοδύναμη με διαφορά τουλάχιστον 2, οπότε λείπει τουλάχιστον μία τιμή ανάμεσα σε x_i, x_{i+1} .

Κάθε φορά που εντοπίζεται ένα κενό πρέπει να καθοριστούν τα όρια του. Οι τιμές x_i, x_{i+1} υπάρχουν στα τοπικά δεδομένα του κόμβου, οπότε δεν πρέπει να εμφανίζονται στα όρια των κενών διαστημάτων. Για τον σωστό καθορισμό, πρέπει στα κενά διαστήματα να εμφανίζονται μόνο τιμές οι οποίες λείπουν από τα τοπικά δεδομένα ενός κόμβου. Για αυτόν τον λόγο, όταν εντοπιστεί κενό ανάμεσα στις τιμές x_i, x_{i+1} πρέπει να προστεθεί μία μονάδα στο κάτω όριο και να αφαιρεθεί μία μονάδα από το άνω όριο. Έτσι εντοπίζεται η τελική μορφή ενός κενού διαστήματος G .

$$G_i = [x_i + 1, x_{i+1} - 1]$$

Για παράδειγμα :

$$\text{για } x_i = 430 \text{ και } x_{i+1} = 440$$

προκύπτει το κενό διάστημα :

$$G = [431, 439]$$

Στην περίπτωση που η καθολικά ελάχιστη τιμή ενός χαρακτηριστικού f δεν περιλαμβάνεται στα τοπικά δεδομένα ενός κόμβου, τότε πρέπει να χρησιμοποιηθεί ως κάτω όριο του πρώτου κενού διαστήματος. Έτσι προκύπτει το κενό :

$$G = [\min_f^{global}, x_{i+1} - 1]$$

Αντίστοιχα στην περίπτωση που δεν περιλαμβάνεται η καθολικά μέγιστη τιμή στα τοπικά δεδομένα του κόμβου, τότε χρησιμοποιείται ως άνω όριο του τελευταίου κενού διαστήματος :

$$G = [x_i + 1, \max_f^{global}]$$

Τελικά ο τρόπος που αποθηκεύει ο κόμβος τα κενά διαστήματα είναι παρόμοιος με τον τρόπο που αποθηκεύει τοπικά δεδομένα. Η τελική δομή αποτελείται από μία λίστα με τέσσερις εσωτερικές λίστες, όπου κάθε εσωτερική λίστα περιέχει τα κενά διαστήματα για το αντίστοιχο χαρακτηριστικό.

$$\text{"empty_regions"} = [E_1, E_2, E_3, E_4]$$

όπου για κάθε χαρακτηριστικό f :

$$E_f = [G_1, G_2, \dots, G_n]$$

και κάθε G_i είναι ένα κενό διάστημα :

$$G_i = [start_i, end_i]$$

4.5 Συσταδοποίηση εισερχόμενων δεδομένων

Πριν ληφθεί η απόφαση για το αν ένας κόμβος θα αποθηκεύσει τα εισερχόμενα δεδομένα τοπικά ή θα προσπαθήσει να τα στείλει προς άλλους κόμβους, τα εισερχόμενα δεδομένα οργανώνονται σε συστάδες (Clusters). Η διαδικασία συσταδοποίησης γίνεται ανεξάρτητα για κάθε χαρακτηριστικό. Από εδώ και πέρα οι συστάδες αναφέρονται ως clusters.

Αρχικά γίνεται ταξινόμηση των εισερχόμενων δεδομένων σε αύξουσα σειρά. Η πρώτη τιμή χρησιμοποιείται ως αρχή του πρώτου cluster και στη συνέχεια κάθε νέα τιμή συγκρίνεται με την αμέσως προηγούμενη. Αν η διαφορά των δύο τιμών είναι μικρότερη ή ίση από ένα προκαθορισμένο κατώφλι, η νέα τιμή προστίθεται στο ίδιο cluster. Αντίθετα, αν η διαφορά μεταξύ τιμών ξεπερνάει το κατώφλι, τότε το τρέχον cluster “κλείνει” και δημιουργείται νέο με πρώτη τιμή την τρέχουσα εξεταζόμενη.

Στην υλοποίηση του συστήματος χρησιμοποιήθηκε τιμή κατωφλίου ίση με 5.

$$THRESHOLD = 5$$

πρακτικά :

- αν $x_i - x_{i-1} \leq 5 \rightarrow$ προσθήκη στο ίδιο cluster
- αν $x_i - x_{i-1} > 5 \rightarrow$ εκκίνηση νέου cluster

Για παράδειγμα, έστω οι τιμές : [430, 432, 434, 441, 443]

- Το πρώτο cluster ξεκινάει από το 430
- $432 - 430 = 2 \rightarrow$ ίδιο cluster
- $434 - 432 = 2 \rightarrow$ ίδιο cluster
- $441 - 434 = 7 \rightarrow$ εκκίνηση νέου cluster
- $443 - 441 = 2 \rightarrow$ ίδιο cluster

Έτσι από το παράδειγμα θα προκύψουν δύο clusters :

$$c_1 = [430, 432, 434] \text{ και } c_2 = [441, 443]$$

Η αποθήκευση ακολουθεί την ίδια διαδικασία με τα τοπικά δεδομένα και τα κενά διαστήματα :

$$\text{"generated_data_clusters"} = [C_1, C_2, C_3, C_4]$$

όπου για κάθε χαρακτηριστικό f :

$$C_f = [c_1, c_2, \dots, c_n]$$

και κάθε cluster :

$$c_i = [x_1, x_2, \dots, x_n]$$

4.6 Αξιολόγηση συστάδων και απόφαση κόμβου

Μετά τη διαδικασία συσταδοποίησης κάθε κόμβος πρέπει να αξιολογήσει τα clusters που δημιούργησε προκειμένου να αποφασίσει αν θα ενσωματώσει τις τιμές τους στα δικά του τοπικά δεδομένα ή θα προσπαθήσει να τα στείλει σε άλλους κόμβους για να γεμίσει τα κενά διαστήματα τους. Η αξιολόγηση πραγματοποιείται ξεχωριστά για κάθε cluster και βασίζεται στο κατά πόσο οι τιμές του ταιριάζουν με τα δεδομένα που είναι ήδη αποθηκευμένα στον κόμβο.

Για κάθε χαρακτηριστικό, γίνεται χρήση της μέσης τιμής και της τυπικής απόκλισης, οι οποίες υπολογίστηκαν στα αρχικά στάδια της υλοποίησης. Ο κόμβος χρησιμοποιεί αυτά τα στατιστικά στοιχεία για τη δημιουργία ενός διαστήματος γύρω από τη μέση τιμή. Όταν μία τιμή ενός cluster βρίσκεται εντός αυτού του διαστήματος θεωρείται ότι ταιριάζει με τα δεδομένα του κόμβου. Συγκεκριμένα, το διάστημα εκτείνεται από μία τυπική απόκλιση κάτω από τη μέση τιμή έως μία τυπική απόκλιση πάνω από αυτή :

$$[\mu_f - \sigma_f, \mu_f + \sigma_f]$$

Για κάθε cluster ο κόμβος υπολογίζει το ποσοστό των τιμών του που βρίσκονται εντός του διαστήματος ταιριάσματος. Κάθε τιμή ενός cluster ελέγχεται ως προς το αν βρίσκεται εντός του διαστήματος και στη συνέχεια υπολογίζεται το πλήθος των τιμών που ικανοποιούν τη συγκεκριμένη συνθήκη.

$$Match(C) = \frac{n_{match}}{|C|} * 100$$

όπου :

- $Match(C)$: ποσοστό ταιριάσματος του cluster
- C : εξεταζόμενο cluster
- n_{match} : πλήθος τιμών του cluster εντός του διαστήματος
- $|C|$: συνολικό πλήθος τιμών του cluster

Στη συνέχεια ο κόμβος αξιολογεί κάθε cluster προκειμένου να αποφασίσει αν θα το διατηρήσει ή θα γίνει υποψήφιο προς αποστολή. Η αξιολόγηση γίνεται με βάση το ποσοστό ταιριάσματος του cluster. Αν αυτό το ποσοστό είναι τουλάχιστον 80%, δηλαδή 80% των τιμών του cluster βρίσκονται εντός του διαστήματος ταιριάσματος, το cluster θεωρείται ότι ταιριάζει με τα αποθηκευμένα δεδομένα του κόμβου και θα διατηρηθεί από τον κόμβο.

$$Match(C) \geq 80\% \rightarrow \text{Διατήρηση}$$

Σε περίπτωση που το ποσοστό ταιριάσματος είναι μικρότερο από 80% τότε εξετάζεται το μέγεθος του cluster. Στη συγκεκριμένη υλοποίηση, αν το πλήθος των τιμών του cluster είναι μεγαλύτερο ή ίσο του 15, τότε πάλι ο κόμβος αποφασίζει να το διατηρήσει.

$$Match(C) < 80\% \text{ AND } |C| \geq 15 \rightarrow \text{Διατήρηση}$$

Σε κάθε άλλη περίπτωση, το cluster θα γίνει υποψήφιο προς αποστολή.

$$Match(C) < 80\% \text{ AND } |C| < 15 \rightarrow \text{Αποστολή}$$

4.7 Ανταλλαγή πληροφορίας κενών διαστημάτων

Μετά τον εντοπισμό των κενών διαστημάτων, οι κόμβοι ανταλλάσσουν μεταξύ τους πληροφορίες σχετικά με τις περιοχές για τις οποίες δεν έχουν δεδομένα. Η επικοινωνία αυτή είναι απαραίτητη ώστε κάθε κόμβος να γνωρίζει σε ποιους από τους υπόλοιπους κόμβους μπορεί να στείλει ένα υποψήφιο cluster για να γεμίσει ένα κενό διάστημα.

Για την αποθήκευση των κενών διαστημάτων των υπόλοιπων κόμβων, κάθε κόμβος χρησιμοποιεί ένα λεξικό. Για κάθε απομακρυσμένο κόμβο χρησιμοποιείται το αναγνωριστικό του (ID) ως κλειδί και ως αντίστοιχη τιμή αποθηκεύεται η λίστα κενών διαστημάτων του. Τα κενά διαστήματα του ίδιου κόμβου δεν αποθηκεύονται στο λεξικό.

"known_empty_regions" = {node_id: empty_regions}

για έναν κόμβο i :

$$\text{known_empty_regions}_i = \{j : E_j \mid j \neq i\}$$

όπου :

- i : τρέχων κόμβος
- j : ένας άλλος κόμβος
- E_j : κενά διαστήματα του κόμβου j

4.8 Βελτιστοποίηση μεταφοράς δεδομένων με ABC

Ο αλγόριθμος ABC που υιοθετεί κάθε κόμβος προσπαθεί να βρει τους βέλτιστους συνδυασμούς μεταξύ cluster, κόμβων παραληπτών και κενών διαστημάτων με στόχο οι τιμές των αποστελλόμενων cluster να καλύψουν όσο το δυνατόν καλύτερα τα κενά διαστήματα των κόμβων παραληπτών.

4.8.1 Αναπαράσταση λύσης

Κάθε υποψήφια μεταφορά δεδομένων αναπαρίσταται από έναν συνδυασμό 4 στοιχείων : το χαρακτηριστικό στο οποίο ανήκει το cluster προς αποστολή, τον δείκτη του συγκεκριμένου cluster, το αναγνωριστικό του κόμβου παραλήπτη και τον δείκτη ενός κενού διαστήματος του παραλήπτη για το αντίστοιχο χαρακτηριστικό, το οποίο θα προσπαθήσει να καλύψει το cluster.

$$T = (f, cluster_idx, receiver_idx, empty_idx)$$

όπου :

- f : χαρακτηριστικό
- $cluster_idx$: δείκτης του cluster προς αποστολή
- $receiver_idx$: αναγνωριστικό του κόμβου παραλήπτη
- $empty_idx$: δείκτης κενού διαστήματος

Κάθε λύση αποτελείται από μία λίστα μεταφορών :

$$S = [T_1, T_2, \dots, T_m]$$

όπου m το μέγιστο πλήθος μεταφορών μίας λύσης.

4.8.2 Αρχικοποίηση πληθυσμού λύσεων

Αρχικά ο αλγόριθμος δημιουργεί έναν πληθυσμό είκοσι τυχαίων λύσεων. Κάθε λύση παράγεται ανεξάρτητα και αποτελείται από έως τριάντα μεταφορές. Για καθεμία από αυτές τις λύσεις αρχικοποιείται ένας μετρητής, ο οποίος καταγράφει τον αριθμό αποτυχημένων προσπαθειών βελτίωσης της λύσης.

Για καθένα από τα τέσσερα χαρακτηριστικά, γίνεται έλεγχος αν υπάρχουν διαθέσιμα clusters προς αποστολή. Για κάθε μεταφορά που θα αποτελεί μέρος μίας λύσης γίνεται τυχαία επιλογή ανάμεσα σε αυτά τα χαρακτηριστικά. Το επόμενο βήμα είναι η τυχαία επιλογή ενός cluster από τη λίστα με τα υποψήφια για το επιλεγμένο χαρακτηριστικό. Στη συνέχεια γίνεται επιλογή ενός αναγνωριστικού κόμβου από τα κλειδιά του λεξικού και εφόσον ο κόμβος διαθέτει κενά διαστήματα για το επιλεγμένο χαρακτηριστικό, επιλέγεται ένα από αυτά.

Αυτή η διαδικασία γίνεται επαναληπτικά μέχρι το πλήθος των μεταφορών να φτάσει το μέγιστο ή μέχρι να συμπληρωθεί ο προκαθορισμένος αριθμός προσπαθειών. Μέσα σε μία λύση δεν μπορούν να υπάρχουν δύο όμοιες μεταφορές. Πριν γίνει προσθήκη μίας νέας μεταφοράς στη λύση, γίνεται έλεγχος για όμοια μεταφορά ήδη μέσα στη λύση και αν υπάρχει τότε δεν προστίθεται.

Στη συγκεκριμένη υλοποίηση χρησιμοποιήθηκαν οι παράμετροι :

`"SOLUTION_COUNT" = 20`

`"SOLUTION_SIZE" = 30`

`"MAX_ATTEMPTS" = 100`

οπότε προκύπτει η λίστα με τις αρχικοποιημένες λύσεις :

`solutions = [S1, S2, ... , S20]`

και ο πίνακας με τον αντίστοιχο μετρητή για κάθε λύση :

`trials = [0,0, ... ,0]`

4.8.3 Συνάρτηση καταλληλότητας

Αμέσως μετά την αρχικοποίηση των λύσεων γίνεται χρήση της συνάρτησης καταλληλότητας (Fitness Function) για την αξιολόγηση κάθε λύσης. Η τιμή που επιστρέφει η συνάρτηση εκφράζει το πόσο αποτελεσματικά μία λύση καλύπτει τα κενά διαστήματα των κόμβων παραληπτών, λαμβάνοντας υπόψη το κόστος μεταφοράς των δεδομένων από τα clusters του κόμβου αποστολέα προς τους κόμβους παραλήπτες. Όσο μεγαλύτερη η τιμή του fitness τόσο καλύτερη θεωρείται μία λύση.

Για κάθε μεταφορά της λύσης υπολογίζεται η επικάλυψη (Overlap) μεταξύ του cluster προς αποστολή και του επιλεγμένου κενού διαστήματος. Η επικάλυψη υπολογίζεται μετρώντας το πλήθος των τιμών του cluster που βρίσκονται εντός των ορίων του κενού διαστήματος. Αν καμία τιμή δεν βρίσκεται μέσα σε αυτό το διάστημα τότε η επικάλυψη είναι ίση με μηδέν.

πρακτικά :

$$Overlap(C, G) = |\{x \in C : a \leq x \leq b\}|$$

όπου :

- C : cluster
- G : κενό διάστημα

Για παράδειγμα :

$$C = [430, 435, 440, 445]$$

$$G = [430, 439]$$

$$Overlap(C, G) = 2$$

Στην περίπτωση που περισσότερα από ένα clusters στοχεύουν να καλύψουν το ίδιο κενό διάστημα, οι επικαλύψεις των αντίστοιχων μεταφορών δεν μετράνε αθροιστικά. Για το συγκεκριμένο κενό διάστημα, η μεταφορά με τη μεγαλύτερη επικάλυψη είναι αυτή που θα μετρήσει στον υπολογισμό του τελικού fitness :

$$"best_overlaps(G)" = \max\{Overlap(C_1, G), Overlap(C_2, G), \dots\}$$

Μετά από την εξέταση όλων των μεταφορών, οι μεγαλύτερες επικάλυψεις για κάθε κενό διάστημα αθροίζονται και προκύπτει η συνολική επικάλυψη της λύσης :

$$"total_overlap" = \sum_G best_overlaps(G)$$

Στη συνέχεια υπολογίζεται το πλήθος των μεταφορών που έχουν επικάλυψη μεγαλύτερη από 0. Σε περίπτωση που περισσότερες από μία προσπαθούν να καλύψουν το ίδιο κενό διάστημα τότε στο πλήθος μετράει μόνο αυτή με τη μεγαλύτερη επικάλυψη. Στη συγκεκριμένη υλοποίηση το κόστος μεταφοράς είναι πέντε μονάδες για κάθε έγκυρη μεταφορά.

$$"TRANSMIT_COST" = 5$$

Το συνολικό κόστος της λύσης προκύπτει από :

$$"total_cost" = N_{valid} * 5$$

όπου N_{valid} το πλήθος των έγκυρων μεταφορών.

Τελικά, το fitness μίας λύσης υπολογίζεται ως η συνολική επικάλυψη της λύσης μείον το συνολικό κόστος μεταφορών :

$$fitness = "total_overlap" - "total_cost"$$

4.8.4 Employed Bees

Στη φάση των Employed bees του αλγορίθμου εξετάζονται και τροποποιούνται ελάχιστα οι υπάρχουσες λύσεις. Για κάθε λύση δημιουργείται μία νέα λύση, μέσω μικρής τροποποίησης μίας από τις μεταφορές της. Στη συνέχεια υπολογίζεται το fitness της νέας λύσης και συγκρίνεται με το fitness της υπάρχουσας. Αν η νέα έχει μεγαλύτερο fitness, αντικαθιστά την υπάρχουσα και ο μετρητής αποτυχημένων προσπαθειών μηδενίζεται. Αντίθετα, αν η νέα λύση έχει μικρότερο ή ίσο fitness από αυτή της υπάρχουσας, δεν γίνεται αντικατάσταση και ο μετρητής αυξάνεται κατά μία μονάδα.

$$\text{An } \text{Fitness}(S_{\text{new}}) > \text{Fitness}(S_j) \rightarrow S_j = S_{\text{new}}, \text{trials}[j] = 0$$

διαφορετικά :

$$\text{An } \text{Fitness}(S_{\text{new}}) \leq \text{Fitness}(S_j) \rightarrow \text{trials}[j] = \text{trials}[j] + 1$$

Για τη δημιουργία νέας λύσης, επιλέγεται τυχαία μία μεταφορά από την υπάρχουσα λύση και γίνεται μία τροποποίηση. Η τροποποίηση αυτή έχει 50% πιθανότητα να είναι αλλαγή του cluster προς αποστολή για το ίδιο χαρακτηριστικό, ενώ ο κόμβος παραλήπτης και το κενό διάστημα παραμένουν ίδια. Στη δεύτερη περίπτωση τροποποίησης, το cluster προς αποστολή παραμένει ίδιο και αλλάζουν ο κόμβος παραλήπτης και το κενό διάστημα του.

$$P(\text{αλλαγή cluster}) = 0.5$$

$$P(\text{αλλαγή παραλήπτη/κενού}) = 0.5$$

4.8.5 Onlooker bees

Στη φάση των Onlooker bees του αλγορίθμου οι λύσεις έχουν διαφορετική πιθανότητα να εξεταστούν και να τροποποιηθούν ανάλογα με το fitness τους. Όσο μεγαλύτερο το fitness μίας λύσης τόσο μεγαλύτερη πιθανότητα έχει η αντίστοιχη λύση να επιλεγεί για τροποποίηση. Με αυτόν τον τρόπο η διαδικασία επικεντρώνεται στις λύσεις που παρουσιάζουν την καλύτερη απόδοση.

Η πιθανότητα επιλογής μίας λύσης j :

$$P_j = \frac{Fitness_j}{\sum_{k=1}^N Fitness_k}$$

όπου N το πλήθος των λύσεων

Λόγω του κόστους μεταφοράς, η συνάρτηση καταλληλότητας είναι πιθανό να επιστρέψει αρνητικές τιμές. Σε αυτή την περίπτωση, πριν γίνει υπολογισμός των πιθανοτήτων, γίνεται μετατόπιση των τιμών fitness σε θετικές τιμές. Αρχικά γίνεται εντοπισμός της μικρότερης τιμής fitness και γίνεται αφαίρεση της από όλες τις τιμές. Στη συνέχεια προστίθεται μία μικρή σταθερά σε όλες τις τιμές fitness, με σκοπό ακόμα και η μικρότερη τιμή να έχει μη μηδενική πιθανότητα επιλογής.

Για παράδειγμα :

$$Fitness = [-10, -5, 5]$$

η μικρότερη τιμή είναι :

$$\min(Fitness) = -10$$

μετά τη μετατροπή :

$$Fitness' = [0.000001, 5.000001, 15.000001]$$

Σε αυτή την περίπτωση η πιθανότητα επιλογής :

$$P_j = \frac{Fitness'_j}{\sum_{k=1}^N Fitness'_k}$$

Σε περίπτωση που όλες οι λύσεις έχουν μηδενικό fitness, όλες οι λύσεις θα έχουν την ίδια πιθανότητα επιλογής :

$$P_j = \frac{1}{N}$$

Μετά τον υπολογισμό πιθανοτήτων, γίνονται είκοσι επιλογές λύσεων με βάση τις αντίστοιχες πιθανότητες. Για κάθε επιλεγμένη λύση δημιουργείται μία νέα τροποποιημένη ακολουθώντας την ίδια διαδικασία τροποποίησης με την προηγούμενη φάση του αλγορίθμου. Τέλος, υπολογίζεται το fitness της νέας λύσης και αν είναι μεγαλύτερο από το fitness της αντίστοιχης υπάρχουσας λύσης γίνεται αντικατάσταση και μηδενισμός του αντίστοιχου μετρητή, διαφορετικά ο μετρητής αυξάνεται κατά μία μονάδα.

4.8.6 Scout Bees

Στη φάση των Scout bees εξετάζεται ο μετρητής αποτυχημένων προσπαθειών κάθε λύσης. Αν ο αντίστοιχος μετρητής φτάσει ή ξεπεράσει το προκαθορισμένο όριο, ο αλγόριθμος εγκαταλείπει τη συγκεκριμένη λύση και μία νέα τυχαία λύση παίρνει τη θέση της. Στη συνέχεια υπολογίζεται το fitness της νέας λύσης και ο αντίστοιχος μετρητής αποτυχημένων προσπαθειών μηδενίζεται. Στη συγκεκριμένη υλοποίηση το όριο αποτυχημένων προσπαθειών είναι εικοσιπέντε.

$$"TRIAL_LIMIT" = 25$$

Στην περίπτωση που ο μετρητής της τρέχουσας καλύτερης λύσης φτάσει το όριο, γίνεται εξαίρεση και η λύση δεν εγκαταλείπεται. Επίσης γίνεται επανεκκίνηση του μετρητή της στο μηδέν.

πρακτικά :

$$\text{Αν } trials[j] \geq 25 \rightarrow \text{Αντικατάσταση του } S_j, trials[j] = 0$$

εκτός αν :

$$S_j = S_{best} \rightarrow trials[j] = 0$$

4.8.7 Ενημέρωση και επιλογή καλύτερης λύσης

Μετά την ολοκλήρωση της τελευταίας φάσης του αλγορίθμου εντοπίζεται η λύση με το μεγαλύτερο fitness για την τρέχουσα επανάληψη. Η τιμή συγκρίνεται με το καλύτερο fitness που έχει καταγραφεί έως τώρα για όλες τις προηγούμενες επαναλήψεις. Αν είναι μεγαλύτερη, η συγκεκριμένη λύση γίνεται η νέα καλύτερη λύση του αλγορίθμου.

Για την εύρεση της καλύτερης λύσης :

$$\text{"current_best_index"} = \text{argmax}(\text{Fitnesses})$$

$$\text{An } \text{Fitnesses}[\text{"current_best_index"}] > \text{Fitness_best}$$

και την ενημέρωση της :

$$S_{best} = \text{Solutions}[\text{current_best_index}]$$

$$\text{Fitness}_{best} = \text{Fitnesses}[\text{current_best_index}]$$

Η παραπάνω διαδικασία επαναλαμβάνεται για έναν προκαθορισμένο αριθμό επαναλήψεων του αλγορίθμου. Μετά την ολοκλήρωσή τους, ο αλγόριθμος επιστρέφει την καλύτερη λύση που εντόπισε. Στην παρούσα υλοποίηση ο αριθμός επαναλήψεων είναι :

$$\text{"ITERATION_COUNT"} = 100$$

4.9 Επιλογή και πραγματοποίηση μεταφορών

Αφού κάθε κόμβος εκτελέσει τον αλγόριθμο ABC και εντοπίσει την καλύτερη λύση, το επόμενο βήμα είναι να πραγματοποιήσει τις μεταφορές. Από το σύνολο μεταφορών της καλύτερης λύσης, με την ίδια λογική που χρησιμοποιήθηκε κατά τον υπολογισμό του fitness, γίνεται επιλογή των μεταφορών με επικάλυψη μεγαλύτερη του μηδέν. Εφόσον βρεθούν οι έγκυρες μεταφορές, αποθηκεύονται σε μία λίστα και ο κόμβος είναι έτοιμος να τις πραγματοποιήσει.

$$"valid_transactions" = [T_1, T_2, \dots, T_n]$$

όπου n το πλήθος έγκυρων μεταφορών.

Στη συνέχεια κάθε κόμβος πραγματοποιεί τις μεταφορές. Για κάθε έγκυρη μεταφορά ο κόμβος στέλνει το επιλεγμένο cluster προς το αντίστοιχο κενό διάστημα του επιλεγμένου κόμβου παραλήπτη.

Όταν ένας κόμβος παραλαμβάνει δεδομένα, τα τοποθετεί προσωρινά στο "incoming_buffer" του αντίστοιχου χαρακτηριστικού, πριν τα ενσωματώσει στα τοπικά αποθηκευμένα δεδομένα του.

Το "incoming_buffer" κάθε κόμβου είναι ένα λεξικό, όπου το κλειδί είναι το χαρακτηριστικό f και η τιμή είναι μία λίστα με τις εισερχόμενες τιμές που παρέλαβε ο κόμβος.

$$"incoming_buffer" = \{f: [x_1, x_2, \dots, x_n]\}$$

Ένας κόμβος δεν προχωράει στην ενσωμάτωση των δεδομένων μέχρι όλοι οι κόμβοι του συστήματος να πραγματοποιήσουν τις μεταφορές τους.

4.10 Ενσωμάτωση δεδομένων και ενημέρωση κόμβων

Όταν όλοι οι κόμβοι ολοκληρώσουν τις μεταφορές της καλύτερης λύσης που βρήκαν, προχωρούν στην ενσωμάτωση των δεδομένων που αποφάσισαν να κρατήσουν και των δεδομένων που έλαβαν από τους υπόλοιπους κόμβους.

Αρχικά, για κάθε χαρακτηριστικό συλλέγονται όλες οι τιμές από τα clusters που αποφάσισε ότι θα κρατήσει ο κόμβος και προστίθενται στα τοπικά δεδομένα του κόμβου.

Για κάθε χαρακτηριστικό f :

$$K_f = [\text{τιμές των cluster προς διατήρηση}]$$

και

$$L_f^{(1)} = L_f^{old} \mid K_f$$

όπου :

- L_f^{old} : λίστα με τα τοπικά δεδομένα για το χαρακτηριστικό f
- K_f : λίστα τιμών που θα ενσωματωθούν τοπικά

Στη συνέχεια εξετάζεται το "incoming_buffer" του κόμβου. Για κάθε χαρακτηριστικό συλλέγονται οι τιμές που λήφθηκαν από τους υπόλοιπους κόμβους και ενσωματώνονται και αυτές στην αντίστοιχη τοπική λίστα.

Για κάθε χαρακτηριστικό f :

$$B_f = \text{incoming_buffer}[f]$$

και

$$L_f^{new} = L_f^{(1)} \mid B_f$$

Μόλις τελειώσει η διαδικασία ενσωμάτωσης των εισερχόμενων δεδομένων το "incoming_buffer" κάθε κόμβου αδειάζει ως προετοιμασία για την επόμενη επανάληψη του συστήματος.

$$\text{"incoming_buffer"} = \{ \}$$

Τελικά, μετά την ενσωμάτωση, τα δεδομένα του ενός κόμβου έχουν πλέον αλλάξει. Κάθε κόμβος επαναυπολογίζει τα τοπικά στατιστικά στοιχεία του και με βάση αυτά παράγει νέα δεδομένα που θα προσομοιώσουν τα εισερχόμενα δεδομένα της επόμενης επανάληψης του συστήματος. Στη συνέχεια υπολογίζονται τα κενά διαστήματα, γίνεται συσταδοποίηση των δεδομένων και αξιολόγηση των παραγόμενων clusters. Με την επικοινωνία των νέων κενών διαστημάτων τελειώνει μία επανάληψη του συστήματος.

ΚΕΦΑΛΑΙΟ 5 Πειραματική αξιολόγηση

5.1 Παράμετροι συστήματος

Για την πειραματική αξιολόγηση χρησιμοποιήθηκε η υλοποίηση του συστήματος που παρουσιάστηκε στο Κεφάλαιο 4.

Παράμετρος	Ονομασία στον κώδικα	Τιμή
Αριθμός κόμβων	NODE_COUNT	10
Επαναλήψεις συστήματος	ITERATION_COUNT (main.py)	100
Συνθετικές τιμές ανά χαρακτηριστικό	size στη μέθοδο generate_new_data()	20
Κατώφλι εντοπισμού κενών	THRESHOLD στη μέθοδο get_empty_regions()	1.5
Κατώφλι συσταδοποίησης	THRESHOLD στη μέθοδο get_clusters()	5
Ελάχιστο ποσοστό ταιριάσματος για διατήρηση		80%
Ελάχιστο μέγεθος συστάδας για διατήρηση		15
Πλήθος λύσεων ABC	SOLUTION_COUNT	20
Επαναλήψεις ABC	ITERATION_COUNT στη μέθοδο abc_swarm()	100
Όριο αποτυχημένων προσπαθειών	TRIAL_LIMIT	25
Μέγιστο μέγεθος λύσης	SOLUTION_SIZE	30
Μέγιστες προσπάθειες δημιουργίας λύσης	MAX_ATTEMPTS	100
Κόστος μεταφοράς	TRANSMIT_COST	5

Table 1: Παράμετροι του συστήματος που χρησιμοποιήθηκαν στην πειραματική αξιολόγηση.

5.2 Μετρικές αξιολόγησης

Για την αξιολόγηση του προτεινόμενου συστήματος εξετάζεται η εξέλιξη της κάλυψης δεδομένων κατά τη διάρκεια των επαναλήψεων. Στόχος είναι να φανεί αν οι κόμβοι αποκτούν σταδιακά νέες τιμές από το συνολικό εύρος τιμών κάθε χαρακτηριστικού. Επίσης εξετάζεται αν η βελτίωση προέρχεται κυρίως από τα δεδομένα που διατηρούν τοπικά οι κόμβοι ή από τα δεδομένα που λαμβάνουν μέσω μεταφορών.

5.2.1 Τοπική κάλυψη δεδομένων

Η πρώτη μετρική που χρησιμοποιείται είναι η τοπική κάλυψη δεδομένων. Η συγκεκριμένη μετρική αναπαριστά πόσες διαφορετικές τιμές διαθέτει ένας κόμβος για κάθε χαρακτηριστικό, σε σχέση με το συνολικό εύρος τιμών του αντίστοιχου χαρακτηριστικού.

Η τοπική κάλυψη υπολογίζεται ως :

$$Coverage_{i,f} = \frac{|U_{i,f}|}{max_f^{global} - min_f^{global} + 1} * 100$$

όπου :

- $|U_{i,f}|$: το σύνολο των διαφορετικών τιμών του κόμβου i για το χαρακτηριστικό f

Η μέτρηση πραγματοποιείται αμέσως μετά την αρχικοποίηση των κόμβων και μετά το τέλος κάθε επανάληψης. Με αυτόν τον τρόπο είναι εμφανής η εξέλιξη της κάλυψης μετά από κάθε επανάληψη του συστήματος.

5.2.2 Συνολική κάλυψη συστήματος

Η δεύτερη μετρική που χρησιμοποιείται αξιολογεί τη συνολική κάλυψη όλου του συστήματος. Για τον υπολογισμό της λαμβάνονται υπόψη οι διαφορετικές τιμές που υπάρχουν σε όλους τους κόμβους για κάθε χαρακτηριστικό.

Η συνολική κάλυψη υπολογίζεται ως :

$$SystemCoverage_f = \frac{|U_f^{system}|}{max_f^{global} - min_f^{global} + 1} * 100$$

όπου :

- $|U_f^{system}|$: το σύνολο των διαφορετικών τιμών για το χαρακτηριστικό f που υπάρχουν σε όλους τους κόμβους.

Η μετρική χρησιμοποιείται για την παρακολούθηση της συνολικής κάλυψης του εύρους τιμών ενός χαρακτηριστικού κατά τη διάρκεια των επαναλήψεων. Αν μία τιμή υπάρχει σε τουλάχιστον έναν κόμβο του συστήματος και σε επόμενη επανάληψη προστεθεί στα δεδομένα ενός άλλου κόμβου, η τοπική κάλυψη του κόμβου θα αυξηθεί, ενώ η συνολική κάλυψη του συστήματος θα παραμείνει σταθερή.

5.3 Πειραματικά σενάρια

Για την εξέταση της συνεισφοράς των δύο μηχανισμών του συστήματος στην κάλυψη των κενών πραγματοποιούνται διαφορετικά πειράματα. Στόχος είναι να συγκριθεί αν η βελτίωση της κάλυψης προκύπτει από τη διατήρηση δεδομένων τοπικά στους κόμβους ή από τις μεταφορές δεδομένων μεταξύ κόμβων.

5.3.1 Μόνο τοπική διατήρηση δεδομένων

Στο πρώτο σενάριο, οι κόμβοι ενσωματώνουν μόνο τα δεδομένα που αποφάσισαν να διατηρήσουν τοπικά. Σε αυτό το σενάριο εκτέλεσης ο αλγόριθμος ABC δεν εκτελείται. Με αυτόν τον τρόπο, οποιαδήποτε βελτίωση προκύψει θα οφείλεται αποκλειστικά στα δεδομένα που οι κόμβοι αποφάσισαν να διατηρήσουν.

5.3.2 Μόνο μεταφορές μέσω ABC

Στο δεύτερο σενάριο, οι κόμβοι δεν ενσωματώνουν τα δεδομένα που αποφάσισαν να διατηρήσουν. Ο αλγόριθμος ABC εκτελείται κανονικά και πραγματοποιούνται μόνο οι μεταφορές δεδομένων μεταξύ των κόμβων. Με αυτόν τον τρόπο, οποιαδήποτε βελτίωση προκύψει θα οφείλεται αποκλειστικά στα δεδομένα που λαμβάνουν οι κόμβοι μέσω των μεταφορών.

5.3.3 Πλήρες σύστημα

Στο τρίτο και τελευταίο σενάριο, οι κόμβοι ενσωματώνουν τα δεδομένα που αποφάσισαν να κρατήσουν τοπικά και τα δεδομένα που έλαβαν από τους υπόλοιπους κόμβους μέσω μεταφορών. Με αυτόν τον τρόπο αξιολογείται η βελτίωση της κάλυψης όταν και οι δύο μηχανισμοί λειτουργούν ταυτόχρονα.

5.4 Αποτελέσματα

Σε αυτή την ενότητα παρουσιάζονται τα αποτελέσματα των πειραματικών σεναρίων που περιγράφηκαν στην προηγούμενη ενότητα. Για κάθε σενάριο εξετάζονται η τοπική κάλυψη κάθε κόμβου καθώς και η συνολική κάλυψη του συστήματος κατά τη διάρκεια των επαναλήψεων.

5.4.1 Αποτελέσματα μόνο τοπικής διατήρησης δεδομένων

Στο πρώτο πειραματικό σενάριο, εξετάζεται η εξέλιξη της τοπικής κάλυψης κάθε κόμβου και της συνολικής κάλυψης του συστήματος όταν οι κόμβοι ενσωματώνουν αποκλειστικά τα δεδομένα που αποφάσισαν να διατηρήσουν.

Το πείραμα εκτελέστηκε είκοσι φορές, χρησιμοποιώντας διαφορετικό seed για κάθε εκτέλεση. Για κάθε κόμβο και κάθε χαρακτηριστικό υπολογίστηκε η κάλυψη σε κάθε επανάληψη και στη συνέχεια υπολογίστηκε ο μέσος όρος των εκτελέσεων. Με τον ίδιο τρόπο υπολογίστηκε και η μέση συνολική κάλυψη του συστήματος.

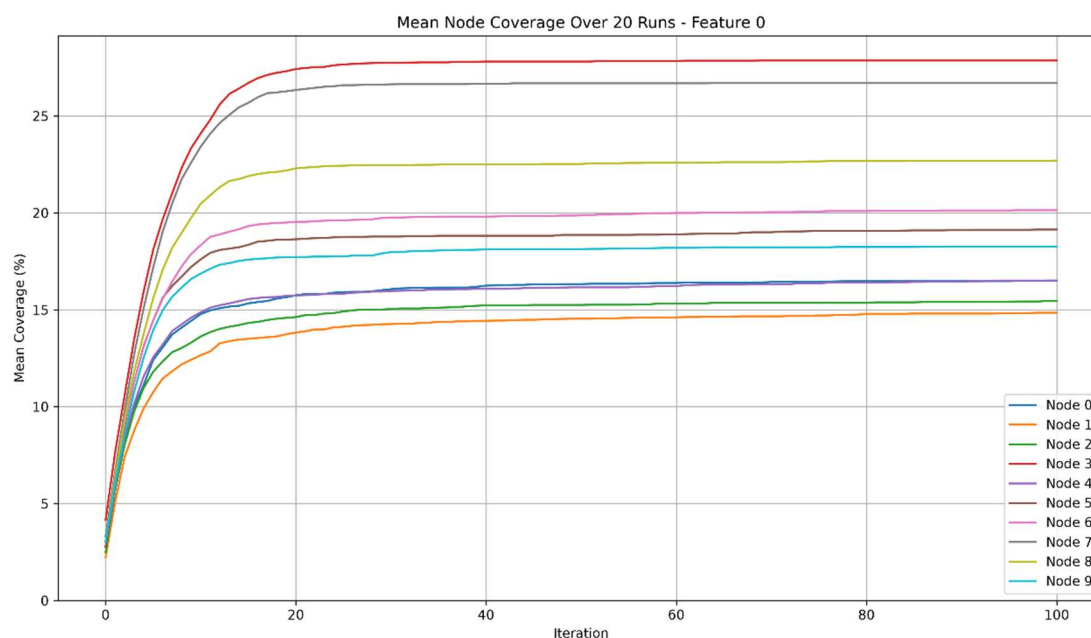


Figure 1: Μέση τοπική κάλυψη των κόμβων για το Feature 0 σε 20 εκτελέσεις. (μόνο διατήρηση)

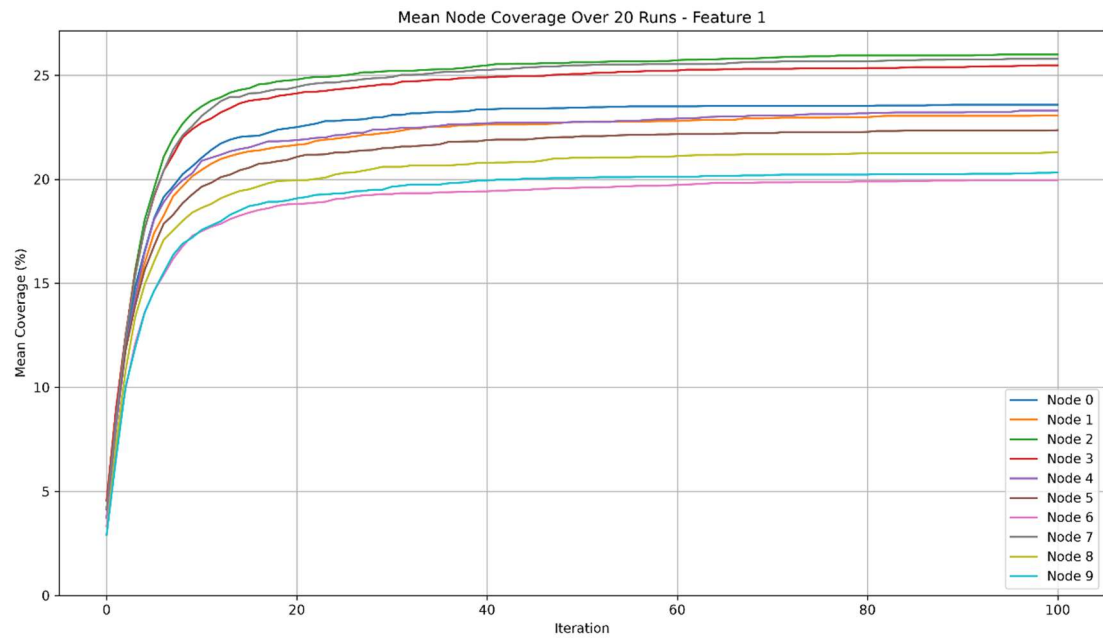


Figure 2: Μέση τοπική κάλυψη των κόμβων για το Feature 1 σε 20 εκτελέσεις. (μόνο διατήρηση)

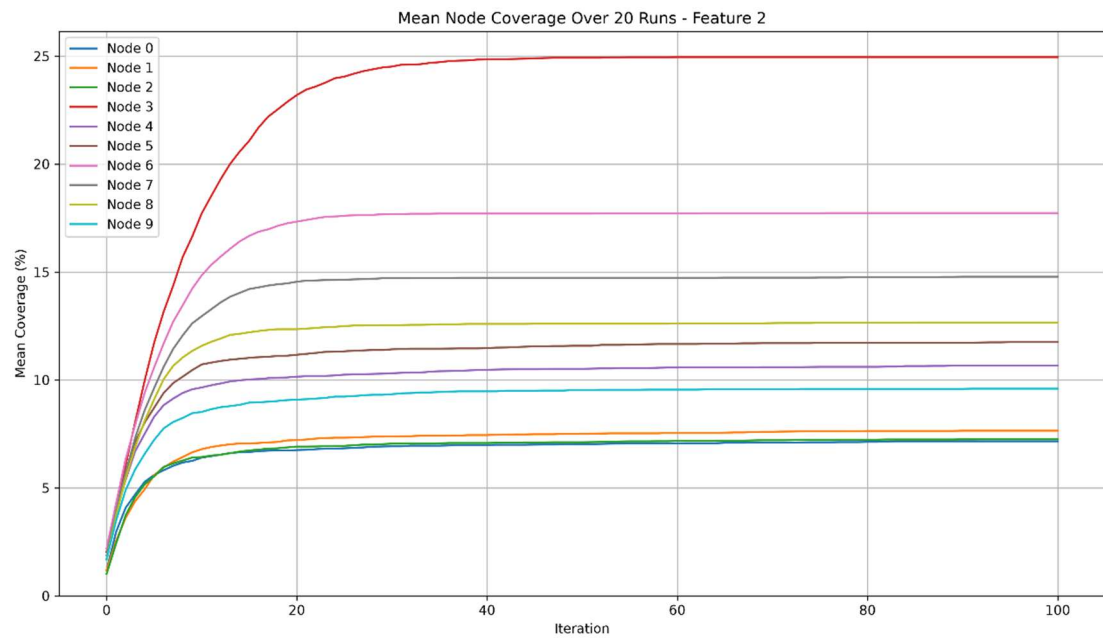


Figure 3: Μέση τοπική κάλυψη των κόμβων για το Feature 2 σε 20 εκτελέσεις. (μόνο διατήρηση)

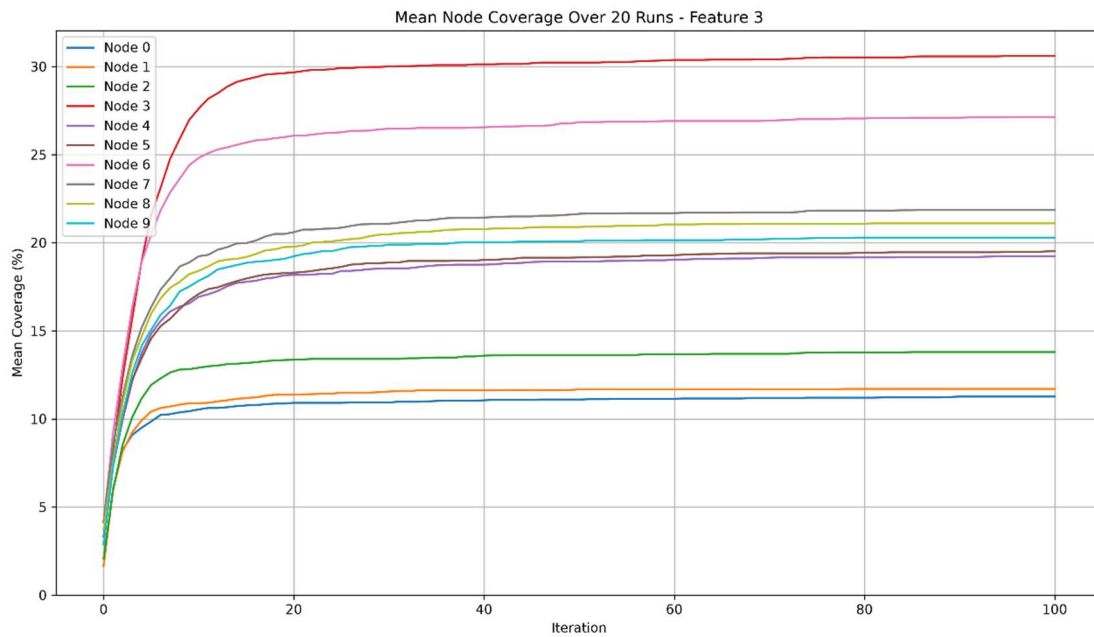


Figure 4: Μέση τοπική κάλυψη των κόμβων για το Feature 3 σε 20 εκτελέσεις. (μόνο διατήρηση)

Στα διαγράμματα της τοπικής κάλυψης παρουσιάζεται η εξέλιξη της κάλυψης κάθε κόμβου για κάθε χαρακτηριστικό αντίστοιχα. Στους περισσότερους κόμβους η κάλυψη αυξάνεται κυρίως στις πρώτες επαναλήψεις και στη συνέχεια ο ρυθμός βελτίωσης μειώνεται, με αποτέλεσμα την κάλυψη να σταθεροποιείται.

Παράλληλα εμφανίζονται διαφορές μεταξύ κόμβων, οι οποίες οφείλονται στα διαφορετικά αρχικά δεδομένα κάθε κόμβου και κατά συνέπεια στα στατιστικά στοιχεία του. Η τελική κάλυψη κάθε κόμβου κυμαίνεται από 7% έως 30% του εύρους τιμών ανάλογα με τον κόμβο και το χαρακτηριστικό.

Η συμπεριφορά του πειράματος δείχνει ότι η τοπική παραγωγή και διατήρηση μπορεί να αυξήσει την κάλυψη αλλά μετά από ένα σημείο η βελτίωση σταματά και η κάλυψη σταθεροποιείται. Αυτό οφείλεται στον τρόπο με τον οποίο ο κόμβος παράγει δεδομένα. Αρχικά υπάρχει μεγαλύτερη πιθανότητα να παραχθούν τιμές που δεν υπάρχουν ήδη στα τοπικά δεδομένα του κόμβου. Καθώς προστίθενται νέες τιμές στα δεδομένα του αυξάνεται η πιθανότητα για την παραγωγή των ίδιων τιμών. Επομένως μειώνεται ο αριθμός των νέων μοναδικών τιμών που αυξάνουν την κάλυψη.

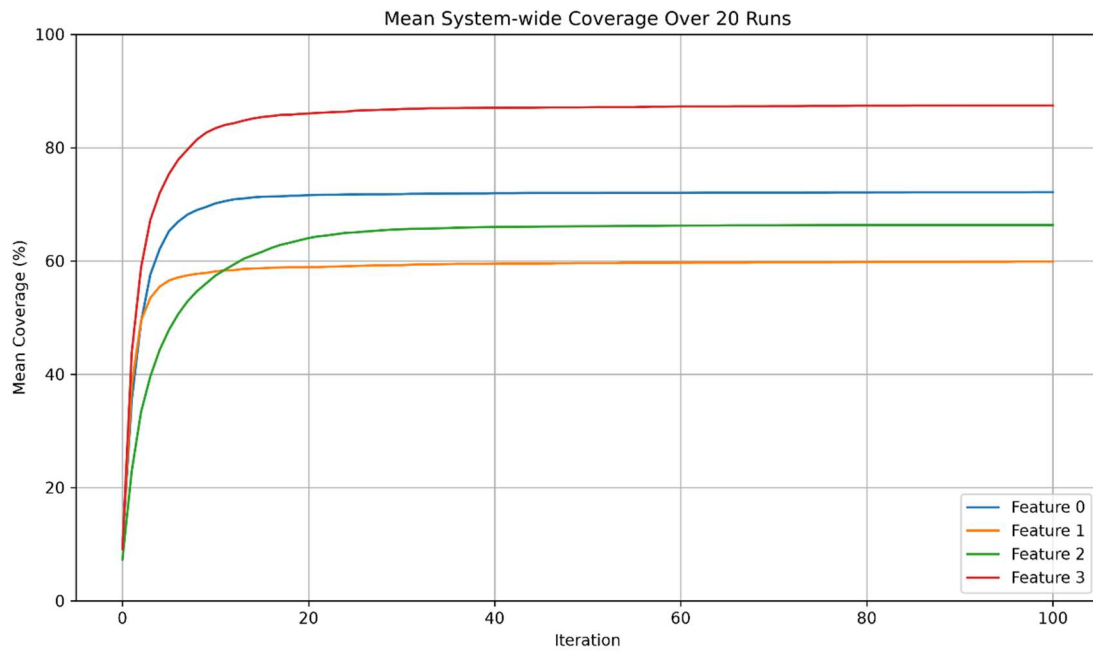


Figure 5: Μέση συνολική κάλυψη του συστήματος ανά χαρακτηριστικό σε 20 εκτελέσεις. (μόνο διατήρηση)

Στο διάγραμμα της συνολικής κάλυψης του συστήματος παρατηρείται παρόμοια συμπεριφορά με την τοπική κάλυψη. Η κάλυψη αυξάνεται κυρίως στις πρώτες επαναλήψεις και στη συνέχεια ο ρυθμός βελτίωσης μειώνεται μέχρι η κάλυψη να σταθεροποιηθεί. Η συνολική κάλυψη είναι σημαντικά υψηλότερη καθώς για τον υπολογισμό της λαμβάνονται υπόψη όλες οι μοναδικές τιμές από κάθε κόμβο του συστήματος.

Συγκεκριμένα η κάλυψη κάθε χαρακτηριστικού πριν και μετά την εκτέλεση του πειράματος είναι :

Χαρακτηριστικό	Αρχική	Τελική	Αύξηση
0	9.70%	72.13%	62.43 ποσοστιαίες μονάδες
1	9.54%	59.90%	50.36 ποσοστιαίες μονάδες
2	7.28%	66.33%	59.05 ποσοστιαίες μονάδες
3	9.13%	87.43%	78.30 ποσοστιαίες μονάδες

Από τα αποτελέσματα προκύπτει ότι και στα τέσσερα χαρακτηριστικά υπάρχει σημαντική αύξηση. Η μεγαλύτερη αύξηση σημειώνεται στο χαρακτηριστικό 3 με 78.30 ποσοστιαίες μονάδες, ενώ η μικρότερη στο χαρακτηριστικό 1 με 50.36 ποσοστιαίες μονάδες.

Συνολικά, το πρώτο πειραματικό σενάριο δείχνει ότι η τοπική διατήρηση δεδομένων μπορεί να βελτιώσει σημαντικά την τοπική κάλυψη των κόμβων. Ωστόσο, μετά από έναν αριθμό επαναλήψεων η βελτίωση περιορίζεται και η κάλυψη παραμένει σταθερή.

5.4.2 Αποτελέσματα μόνο μεταφορών μέσω ABC

Στο δεύτερο πειραματικό σενάριο, εξετάζεται η εξέλιξη της τοπικής κάλυψης κάθε κόμβου και της συνολικής κάλυψης του συστήματος όταν οι κόμβοι ενσωματώνουν αποκλειστικά τα δεδομένα που έλαβαν μέσω των μεταφορών που πραγματοποιήθηκαν μετά την εκτέλεση του αλγορίθμου ABC.

Όπως και το πρώτο σενάριο, το πείραμα εκτελέστηκε είκοσι φορές, χρησιμοποιώντας διαφορετικό seed για κάθε εκτέλεση. Για κάθε κόμβο και κάθε χαρακτηριστικό υπολογίστηκε η κάλυψη σε κάθε επανάληψη και στη συνέχεια υπολογίστηκε ο μέσος όρος των εκτελέσεων. Με τον ίδιο τρόπο υπολογίστηκε και η μέση συνολική κάλυψη του συστήματος.

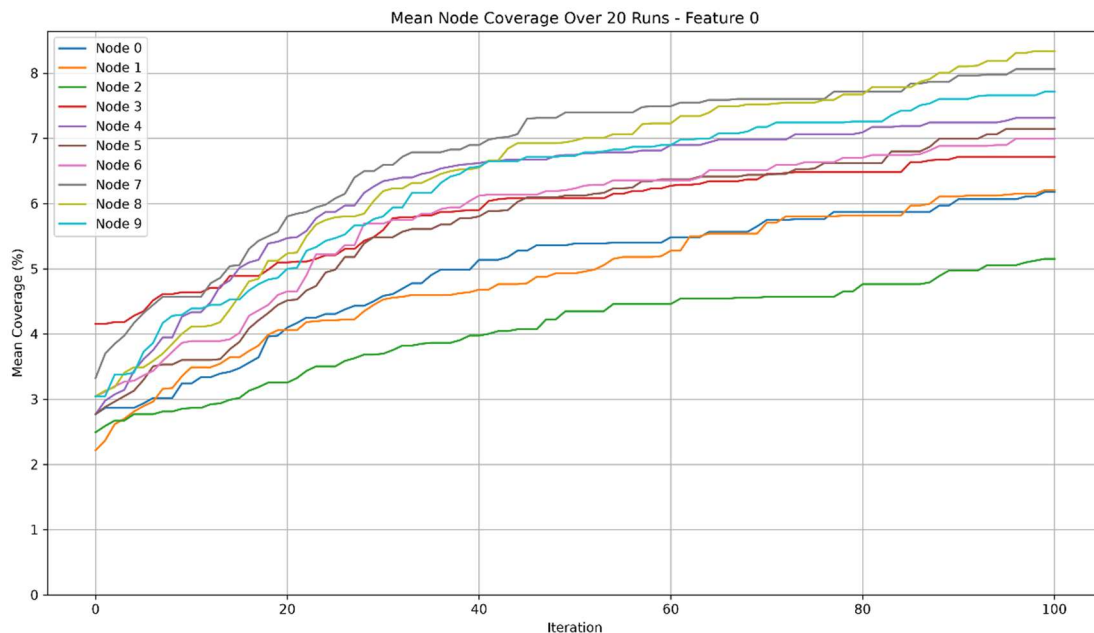


Figure 6: Μέση τοπική κάλυψη των κόμβων για το Feature 0 σε 20 εκτελέσεις. (μόνο μεταφορές)

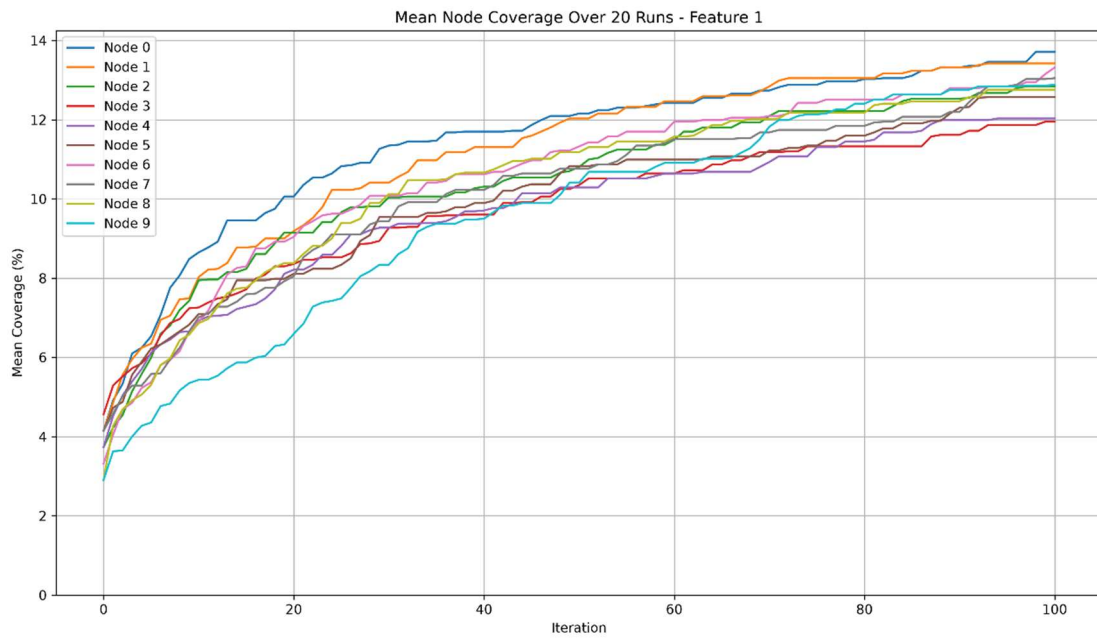


Figure 7: Μέση τοπική κάλυψη των κόμβων για το Feature 1 σε 20 εκτελέσεις. (μόνο μεταφορές)

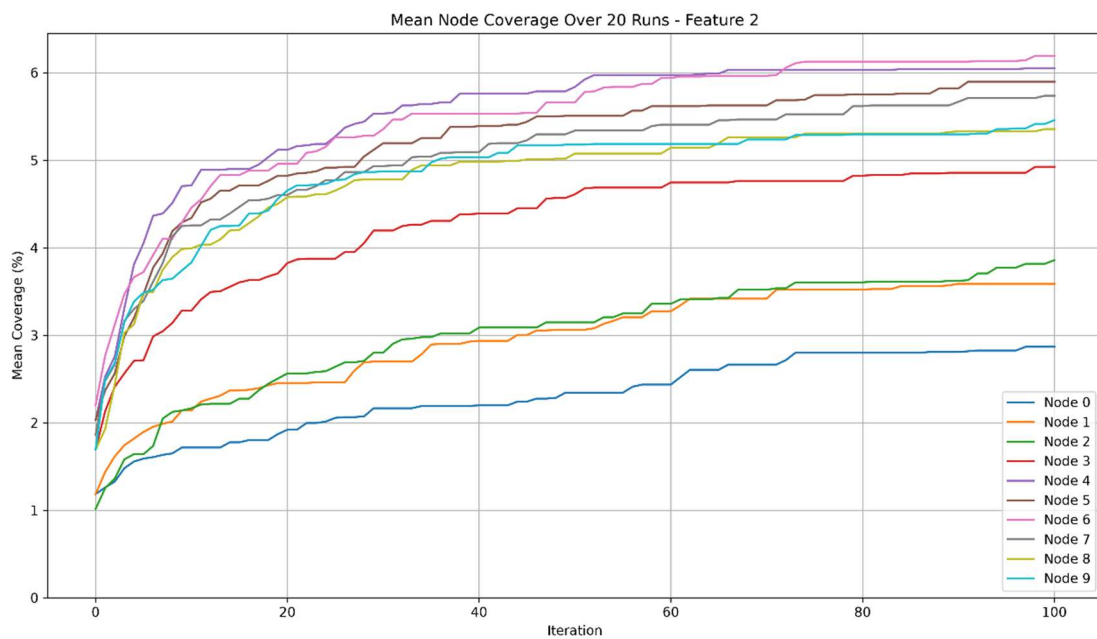


Figure 8: Μέση τοπική κάλυψη των κόμβων για το Feature 2 σε 20 εκτελέσεις. (μόνο μεταφορές)

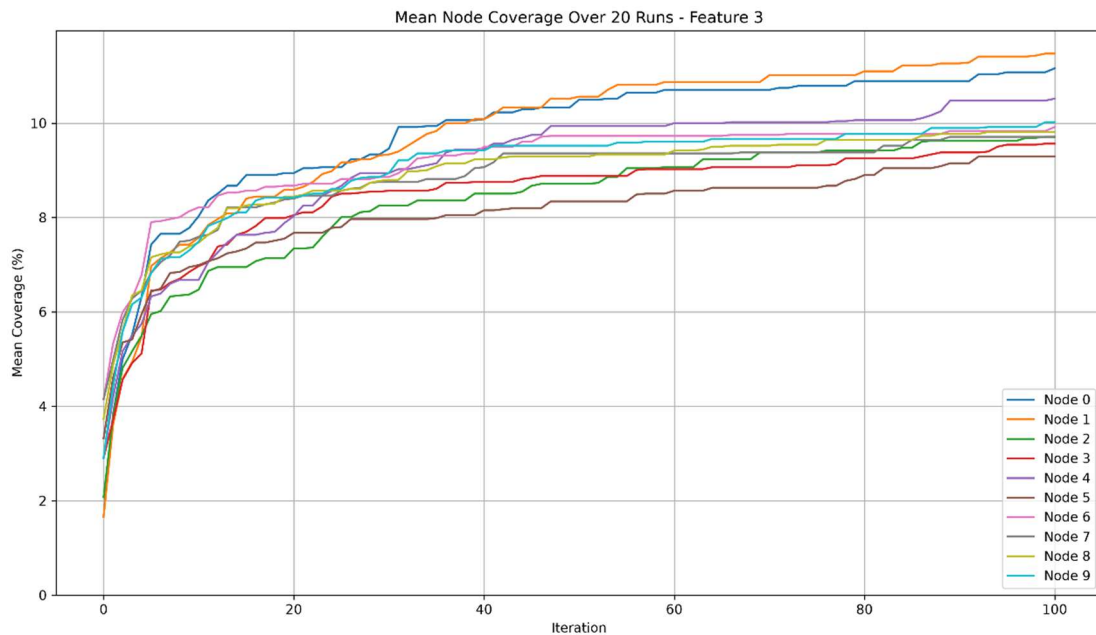


Figure 9: Μέση τοπική κάλυψη των κόμβων για το Feature 3 σε 20 εκτελέσεις. (μόνο μεταφορές)

Στα διαγράμματα της τοπικής κάλυψης παρατηρείται σταδιακή αύξηση της κάλυψης των κόμβων κατά τη διάρκεια των επαναλήψεων. Σε αντίθεση με το προηγούμενο πείραμα, η αύξηση συνεχίζεται για το μεγαλύτερο μέρος της εκτέλεσης.

Η τελική κάλυψη κάθε κόμβου κυμαίνεται από 3% έως 14% του συνολικού εύρους τιμών ανάλογα με τον κόμβο και το χαρακτηριστικό. Οι τιμές είναι χαμηλότερες από εκείνες του πρώτου πειράματος επειδή οι κόμβοι ενσωματώνουν μόνο τα δεδομένα που έλαβαν μέσω των μεταφορών.

Η συμπεριφορά του πειράματος δείχνει ότι οι μεταφορές μέσω του αλγορίθμου ABC μπορούν να αυξήσουν σταδιακά την τοπική κάλυψη των κόμβων. Σε κάθε επανάληψη ο αλγόριθμος εκτελεί μόνο έγκυρες μεταφορές δεδομένων και οι παραλήπτες ενσωματώνουν μόνο τα δεδομένα που έλαβαν από αυτές τις μεταφορές. Καθώς οι κόμβοι ενσωματώνουν τα λαμβανόμενα δεδομένα, τα κενά διαστήματα τους αλλάζουν επηρεάζοντας τις επόμενες επαναλήψεις. Για αυτόν τον λόγο η αύξηση είναι πιο σταδιακή και συνεχίζεται σε μεγαλύτερο μέρος της εκτέλεσης.

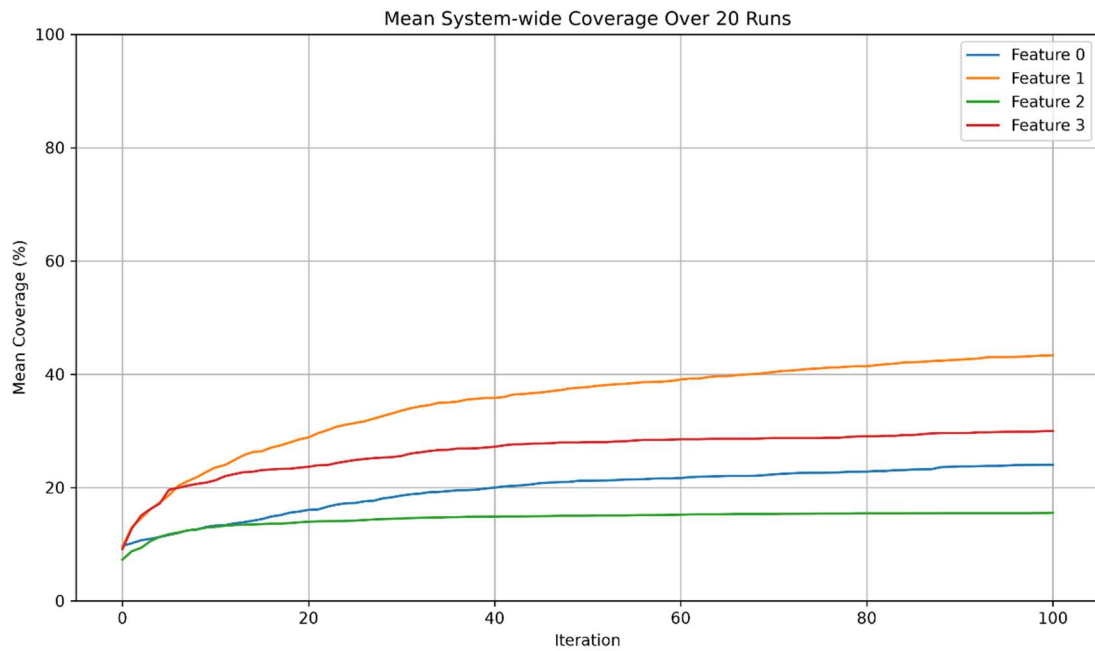


Figure 10: Μέση συνολική κάλυψη του συστήματος ανά χαρακτηριστικό σε 20 εκτελέσεις. (μόνο μεταφορές)

Στο διάγραμμα της συνολικής κάλυψης του συστήματος παρατηρείται σταδιακή αύξηση και στα τέσσερα χαρακτηριστικά κατά τη διάρκεια των επαναλήψεων. Ωστόσο η τελική συνολική κάλυψη είναι χαμηλότερη από το πρώτο πείραμα για όλα τα χαρακτηριστικά. Αυτό δείχνει ότι οι μεταφορές μέσω ABC συνεισφέρουν στην αύξηση της κάλυψης αλλά όχι όσο η τοπική διατήρηση δεδομένων για τον ίδιο αριθμό επαναλήψεων.

Συγκεκριμένα η κάλυψη κάθε χαρακτηριστικού πριν και μετά την εκτέλεση του πειράματος είναι :

Χαρακτηριστικό	Αρχική	Τελική	Αύξηση
0	9.70%	24.02%	14.32 ποσοστιαίες μονάδες
1	9.54%	43.34%	33.80 ποσοστιαίες μονάδες
2	7.28%	15.55%	8.27 ποσοστιαίες μονάδες
3	9.13%	29.98%	20.85 ποσοστιαίες μονάδες

Από τα αποτελέσματα προκύπτει ότι και στα τέσσερα χαρακτηριστικά υπάρχει αύξηση της κάλυψης. Η μεγαλύτερη αύξηση προκύπτει στο χαρακτηριστικό 1 με 33.80 ποσοστιαίες μονάδες, ενώ η μικρότερη στο χαρακτηριστικό 2 με 8.27 ποσοστιαίες μονάδες.

Συνολικά, το δεύτερο πειραματικό σενάριο δείχνει ότι μέσω των μεταφορών του αλγορίθμου ABC μπορεί να βελτιωθεί τόσο η κάλυψη των κόμβων όσο και η συνολική κάλυψη του συστήματος. Ωστόσο, για τον ίδιο αριθμό επαναλήψεων, η βελτίωση είναι μικρότερη από εκείνη που επιτυγχάνεται μόνο μέσω τοπικής διατήρησης δεδομένων.

5.4.3 Αποτελέσματα πλήρους συστήματος

Στο τρίτο πειραματικό σενάριο, εξετάζεται η εξέλιξη της τοπικής κάλυψης κάθε κόμβου και της συνολικής κάλυψης του συστήματος όταν λειτουργούν και οι δύο μηχανισμοί του συστήματος. Οι κόμβοι ενσωματώνουν τα δεδομένα που αποφάσισαν να διατηρήσουν και τα δεδομένα που έλαβαν από τις μεταφορές που πραγματοποιήθηκαν.

Όπως και τα προηγούμενα σενάρια, το πείραμα εκτελέστηκε είκοσι φορές, χρησιμοποιώντας διαφορετικό seed για κάθε εκτέλεση. Για κάθε κόμβο και κάθε χαρακτηριστικό υπολογίστηκε η κάλυψη σε κάθε επανάληψη και στη συνέχεια υπολογίστηκε ο μέσος όρος των εκτελέσεων. Με τον ίδιο τρόπο υπολογίστηκε και η μέση συνολική κάλυψη του συστήματος.

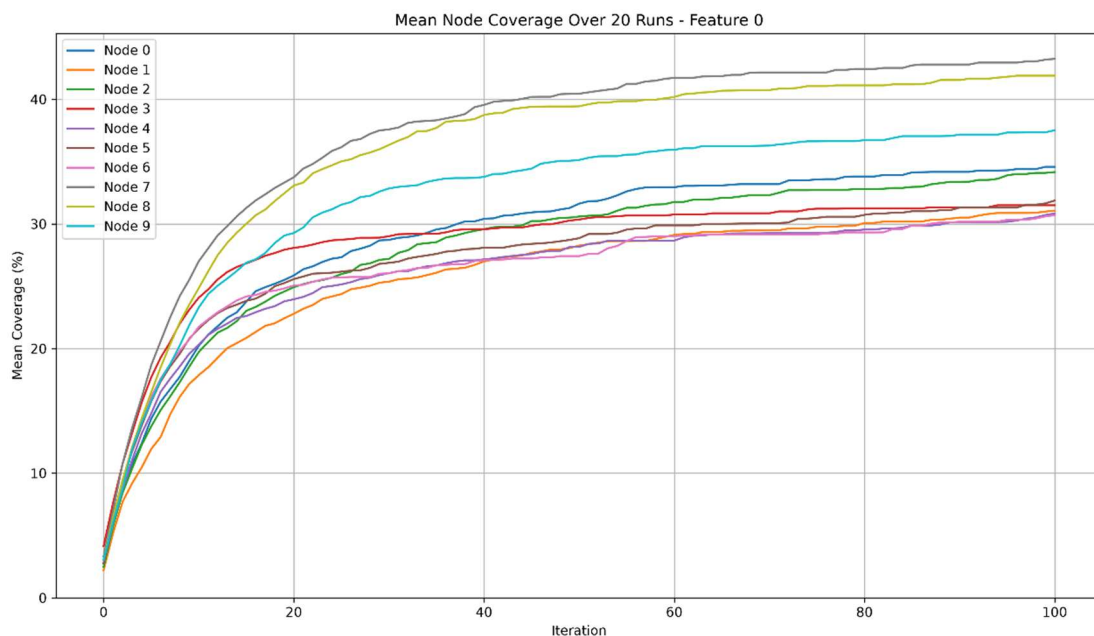


Figure 11: Μέση τοπική κάλυψη των κόμβων για το Feature 0 σε 20 εκτελέσεις. (Πλήρες σύστημα)

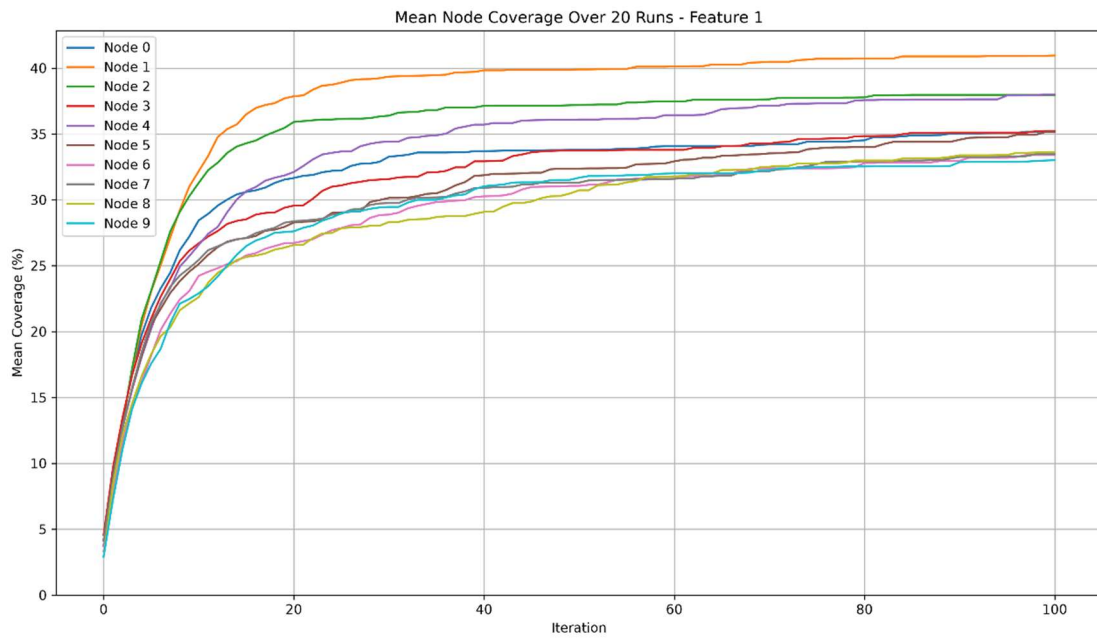


Figure 12: Μέση τοπική κάλυψη των κόμβων για το Feature 1 σε 20 εκτελέσεις. (Πλήρες σύστημα)

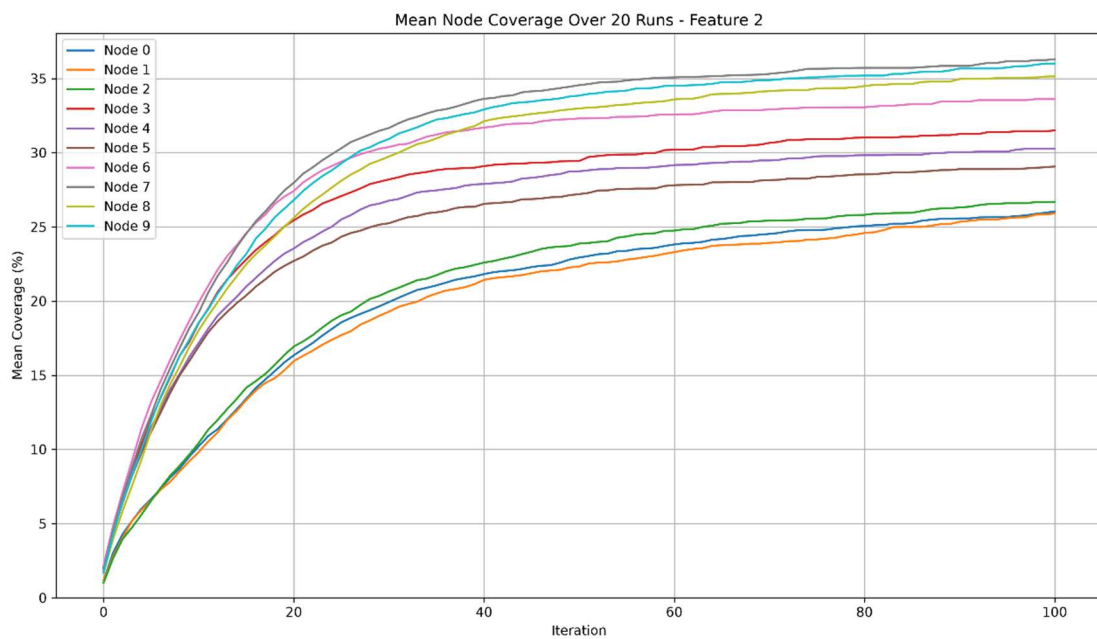


Figure 13: Μέση τοπική κάλυψη των κόμβων για το Feature 2 σε 20 εκτελέσεις. (Πλήρες σύστημα)

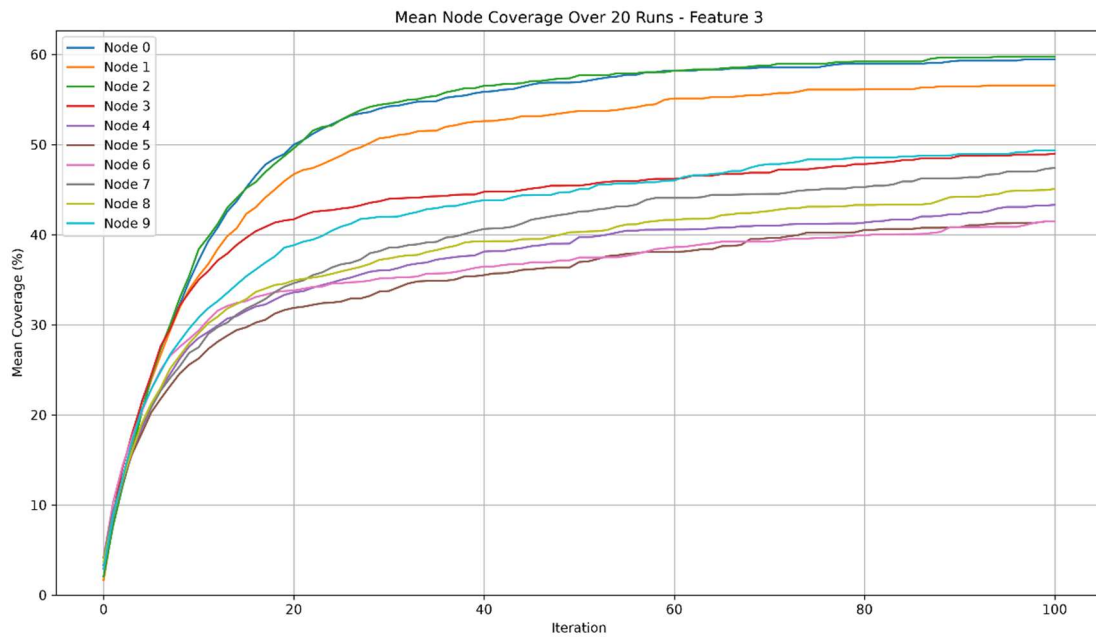


Figure 14: Μέση τοπική κάλυψη των κόμβων για το Feature 3 σε 20 εκτελέσεις. (Πλήρες σύστημα)

Στα διαγράμματα της τοπικής κάλυψης παρατηρείται σημαντική αύξηση της κάλυψης των κόμβων για όλα τα χαρακτηριστικά. Η αύξηση είναι έντονη στις πρώτες επαναλήψεις και στη συνέχεια συνεχίζεται με μικρότερο ρυθμό.

Η τελική κάλυψη κάθε κόμβου κυμαίνεται περίπου από 26% έως 60% του συνολικού εύρους τιμών ανάλογα με τον κόμβο και το χαρακτηριστικό. Οι τιμές είναι υψηλότερες από εκείνες των δύο προηγούμενων πειραμάτων καθώς στο πλήρες σύστημα οι κόμβοι ενσωματώνουν τόσο τα δεδομένα που αποφάσισαν να διατηρήσουν όσο και τα δεδομένα που έλαβαν από τους υπόλοιπους κόμβους.

Η συμπεριφορά αυτή φαίνεται να προκύπτει από συνδυασμό των μηχανισμών που εξετάστηκαν στα δύο προηγούμενα πειράματα. Η τοπική διατήρηση επιτρέπει την αύξηση της κάλυψης κυρίως στις πρώτες επαναλήψεις, ενώ οι μεταφορές μέσω του αλγορίθμου ABC συμβάλλουν στη σταθερή βελτίωση σε όλη τη διάρκεια της εκτέλεσης.

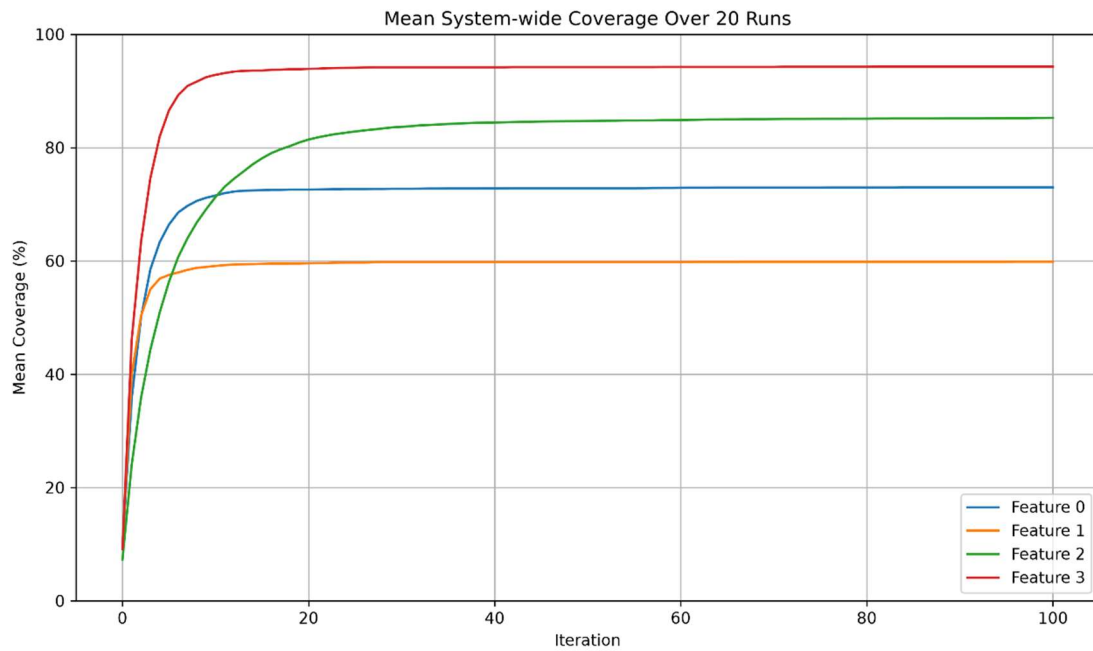


Figure 15: Μέση συνολική κάλυψη του συστήματος ανά χαρακτηριστικό σε 20 εκτελέσεις. (Πλήρες σύστημα)

Στο διάγραμμα της συνολικής κάλυψης του συστήματος παρατηρείται σημαντική αύξηση και στα τέσσερα χαρακτηριστικά. Η αύξηση είναι έντονη στις πρώτες επαναλήψεις και στη συνέχεια συνεχίζεται με μικρότερο ρυθμό. Σε σύγκριση με τα δύο προηγούμενα πειράματα, το πλήρες σύστημα επιτυγχάνει υψηλότερες τελικές τιμές κάλυψης στα περισσότερα χαρακτηριστικά. Συνεπώς ο συνδυασμός των δύο μηχανισμών φαίνεται να είναι πιο αποτελεσματικός ως προς την κάλυψη των κενών.

Συγκεκριμένα η κάλυψη κάθε χαρακτηριστικού πριν και μετά την εκτέλεση του πειράματος είναι :

Χαρακτηριστικό	Αρχική	Τελική	Αύξηση
0	9.70%	72.99%	63.29 ποσοστιαίες μονάδες
1	9.54%	59.88%	50.34 ποσοστιαίες μονάδες
2	7.28%	85.25%	77.97 ποσοστιαίες μονάδες
3	9.13%	94.29%	85.16 ποσοστιαίες μονάδες

Από τα αποτελέσματα προκύπτει ότι και στα τέσσερα χαρακτηριστικά υπάρχει σημαντική αύξηση της κάλυψης. Η μεγαλύτερη αύξηση προκύπτει στο χαρακτηριστικό 3 με 85.16 ποσοστιαίες μονάδες, ενώ η μικρότερη στο χαρακτηριστικό 1 με 50.34 ποσοστιαίες μονάδες.

Συνολικά το τρίτο πειραματικό σενάριο δείχνει ότι η ταυτόχρονη λειτουργία της τοπικής διατήρησης και των μεταφορών μέσω ABC οδηγεί σε σημαντική αύξηση της κάλυψης. Η τοπική διατήρηση συμβάλλει στη γρήγορη αύξηση στις πρώτες επαναλήψεις, ενώ οι μεταφορές συμβάλλουν στη σταδιακή αύξηση κατά τη διάρκεια όλης της εκτέλεσης. Ο συνδυασμός τους επιτυγχάνει μεγαλύτερη κάλυψη για τα περισσότερα χαρακτηριστικά σε σχέση με τη χρήση του κάθε μηχανισμού ξεχωριστά.

5.5 Συγκριτική αξιολόγηση πειραμάτων

Μετά την αξιολόγηση των τριών πειραμάτων ξεχωριστά, πραγματοποιείται σύγκριση των τελικών τιμών της συνολικής κάλυψης με στόχο την εξέταση της συνεισφοράς κάθε μηχανισμού στη συνολική κάλυψη.

Χαρακτηριστικό	Μόνο διατήρηση	Μόνο ABC	Πλήρες σύστημα
0	72.13%	24.02%	72.99%
1	59.90%	43.34%	59.88%
2	66.33%	15.55%	85.25%
3	87.43%	29.98%	94.29%

Από τη σύγκριση των αποτελεσμάτων προκύπτει ότι η τοπική διατήρηση συνεισφέρει περισσότερο στην αύξηση της κάλυψης σε σχέση με τις μεταφορές μέσω ABC όταν οι δύο μηχανισμοί εξετάζονται ξεχωριστά.

Ο συνδυασμός και των δύο μηχανισμών παρουσιάζει τις υψηλότερες τιμές κάλυψης για τρία από τα τέσσερα χαρακτηριστικά. Στο χαρακτηριστικό 1, η μεγαλύτερη τιμή κάλυψης παρατηρείται στο σενάριο μόνο τοπικής διατήρησης με διαφορά μόλις 0.02 ποσοστιαίων μονάδων από το πλήρες σύστημα. Τα υπόλοιπα χαρακτηριστικά παρουσιάζουν μεγαλύτερη κάλυψη στο σενάριο του πλήρους συστήματος.

Ωστόσο, η συνολική κάλυψη δεν αποτυπώνει πλήρως τη συνεισφορά των μεταφορών στην κάλυψη των κόμβων. Για τον υπολογισμό της συνολικής κάλυψης μία τιμή προσμετράται εφόσον υπάρχει σε τουλάχιστον έναν κόμβο. Επομένως, όταν η ίδια τιμή μεταφέρεται σε έναν δεύτερο κόμβο, η τοπική του κάλυψη μπορεί να αυξηθεί χωρίς να αλλάξει η συνολική κάλυψη.

Όπως φαίνεται από τα διαγράμματα, στο πλήρες σύστημα οι κόμβοι παρουσιάζουν σημαντικά μεγαλύτερη τοπική κάλυψη σε σχέση με το σενάριο μόνο τοπικής διατήρησης. Επομένως, πέρα από τη βελτίωση της συνολικής κάλυψης που παρατηρείται στα περισσότερα χαρακτηριστικά, ο συνδυασμός των δύο μηχανισμών επιτρέπει στους κόμβους να καλύψουν μεγαλύτερο μέρος του συνολικού εύρους τιμών.

ΚΕΦΑΛΑΙΟ 6 Συμπεράσματα

Σε αυτή την εργασία μελετήθηκε το πρόβλημα του διαμοιρασμού και της κατανομής δεδομένων μεταξύ αυτόνομων κόμβων σε ένα κατανεμημένο περιβάλλον. Στο προτεινόμενο σύστημα κάθε κόμβος διαχειρίζεται τοπικά τα δεδομένα του, εντοπίζει κενά διαστήματα ανάμεσα στα δεδομένα του και αξιολογεί νέα εισερχόμενα δεδομένα με βάση τα ήδη αποθηκευμένα δεδομένα του. Παράλληλα, χρησιμοποιήθηκε ο αλγόριθμος Artificial Bee Colony για την επιλογή κατάλληλων κόμβων στους οποίους μπορούν να μεταφερθούν τα δεδομένα που δεν ταιριάζουν σε έναν κόμβο, λαμβάνοντας υπόψη την κάλυψη των κενών και το κόστος επικοινωνίας.

Για την αξιολόγηση του συστήματος εξετάστηκαν τρεις διαφορετικές προσεγγίσεις: η τοπική διαχείριση δεδομένων χωρίς διαμοιρασμό, η χρήση μόνο του αλγορίθμου ABC για τη μεταφορά δεδομένων και το πλήρες προτεινόμενο σύστημα, το οποίο συνδυάζει την τοπική διαχείριση με τον διαμοιρασμό δεδομένων. Τα αποτελέσματα έδειξαν ότι ο συνδυασμός των δύο μηχανισμών επιτρέπει στους κόμβους να βελτιώνουν σημαντικά την τοπική κάλυψη των δεδομένων τους, ενώ παράλληλα διατηρείται υψηλή κάλυψη στο συνολικό σύστημα.

Συνολικά, τα αποτελέσματα δείχνουν ότι η χρήση τεχνικών συλλογικής νοημοσύνης μπορεί να υποστηρίξει αποτελεσματικά τη λήψη αποφάσεων για τον διαμοιρασμό δεδομένων μεταξύ αυτόνομων κόμβων. Ο αλγόριθμος ABC επιτρέπει την αναζήτηση κατάλληλων προορισμών για τα δεδομένα, ενώ ο συνδυασμός του με την τοπική επιλογή δεδομένων οδηγεί σε πιο ολοκληρωμένη διαχείριση των δεδομένων.

BIBΛΙΟΓΡΑΦΙΑ

- [1] Özsu, M. T., & Valduriez, P. (2020). *Principles of distributed database systems* (4th ed.). Springer.
- [2] Van Steen, M., & Tanenbaum, A. S. (2016). A brief introduction to distributed systems. *Computing*, 98(10), 967-1009.
- [3] Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., & Rosati, R. (2006). Data management in peer-to-peer data integration systems. In R. Baldoni, G. Cortese, F. Davide, & A. Melpignano (Eds.), *Global data management* (Vol. 8, pp. 177–201). IOS Press.
- [4] Kolomvatsos, K., Oikonomou, P., Koziri, M., & Loukopoulos, T. (2018, October). A distributed data allocation scheme for autonomous nodes. In *2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)* (pp. 1651-1658). IEEE.
- [5] Kolomvatsos, K., & Anagnostopoulos, C. (2024). Autonomous proactive data management in support of pervasive edge applications. *Future Generation Computer Systems*, 155, 108-120.
- [6] Nayeri, Z. M., Ghafarian, T., & Javadi, B. (2021). Application placement in Fog computing with AI approach: Taxonomy and a state of the art survey. *Journal of Network and Computer Applications*, 185, 103078.
- [7] Sun, W., Tang, M., Zhang, L., Huo, Z., & Shu, L. (2020). A survey of using swarm intelligence algorithms in IoT. *Sensors*, 20(5), 1420.
- [8] Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of global optimization*, 39(3), 459-471.
- [9] Tarun, S., Dubey, M. K., Batth, R. S., & Kaur, S. (2022). An optimized cost-based data allocation model for heterogeneous distributed computing systems. *International Journal of Electrical and Computer Engineering*, 12(6), 6373–6386.
- [10] Chen, Z., Hu, J., Min, G., & Chen, X. (2021). Effective data placement for scientific workflows in mobile edge computing using genetic particle swarm optimization. *Concurrency and Computation: Practice and Experience*, 33(8), e5413.
- [11] Babar, M., Khan, M. S., Din, A., Ali, F., Habib, U., & Kwak, K. S. (2021). Intelligent computation offloading for IoT applications in scalable edge computing using artificial bee colony optimization. *Complexity*, 2021(1), 5563531.

- [12] Nashat, D., & Amer, A. A. (2018). A comprehensive taxonomy of fragmentation and allocation techniques in distributed database design. *ACM Computing Surveys (CSUR)*, 51(1), 1-25.
- [13] Singh, A., Kahlon, K. S., & Virk, R. S. (2014). Nonreplicated Static Data Allocation in Distributed Databases Using Biogeography-Based Optimization. *Chinese Journal of Engineering*, 2014(1), 785321.
- [14] Hauglid, J. O., Ryeng, N. H., & Nørvåg, K. (2010). DYFRAM: dynamic fragmentation and replica management in distributed database systems. *Distributed and Parallel Databases*, 28(2), 157-185.
- [15] Karimi Adl, R., & Rouhani Rankoohi, S. M. T. (2009). A new ant colony optimization based algorithm for data allocation problem in distributed databases. *Knowledge and Information Systems*, 20(3), 349-373.
- [16] Mahi, M., Baykan, O. K., & Kodaz, H. (2018). A new approach based on particle swarm optimization algorithm for solving data allocation problem. *Applied Soft Computing*, 62, 571-578.
- [17] Saif, F. A., Latip, R., Hanapi, Z. M., & Shafinah, K. (2023). Multi-objective grey wolf optimizer algorithm for task scheduling in cloud-fog computing. *IEEE Access*, 11, 20635-20646.