ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Προσωρινή αποθήκευση και σύσταση περιεχομένου με επίγνωση της κινητικότητας των χρηστών στα άκρα του δικτύου

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

Σταυροπούλου Γεωργίας

Επιβλέπων
Συμεών Παπαβασιλείου
Καθηγητής Ε.Μ.Π.

Εργαστήριο NETwork Management & Optimal DEsign
Αθήνα, Μάρτιος 2022

Εθνικό Μετσόβιο Πολυτεχνείο
Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Επικοινωνιών Ηλεκτρονικής και Συστημάτων Πληροφορικής
Εργαστήριο NETwork Management & Optimal DEsign

Προσωρινή αποθήκευση και σύσταση περιεχομένου με επίγνωση της κινητικότητας των χρηστών στα άκρα του δικτύου

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

Σταυροπούλου Γεωργίας

Επιβλέπων
Συμεών Παπαβασιλείου
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 31η Μαρτίου 2022.

.....	.....	.....
Συμεών Παπαβασιλείου Καθηγητής Ε.Μ.Π.	Θεοδώρα Βαρβαρίγου Καθηγήτρια Ε.Μ.Π.	Βασίλειος Καρυώτης Αναπλ. Καθηγητής Ιονίου Παν/μίου

Εργαστήριο NETwork Management & Optimal DEsign
Αθήνα, Μάρτιος 2022

.....

Σταυροπούλου Γεωργία
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Copyright © Σταυροπούλου Γεωργία, 2022
Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς το συγγραφέα.

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν το συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

Περίληψη

Στη σημερινή κοινωνία, με την εμφάνιση της νέας τεχνολογίας $5G$, ο όγκος δεδομένων και πληροφορίας που μεταδίδεται, αυξάνεται με εκθετικό ρυθμό. Ταυτόχρονα, οι απαιτήσεις των χρηστών για προσπέλαση μεγάλου όγκου πληροφορίας με μικρή καθυστέρηση και για λήψη εύστοχων προτάσεων αντικειμένων από παρόχους περιεχομένου (content providers), γίνονται αυστηρότερες. Ως αποτέλεσμα, στα πλαίσια των Δικτύων Διανομής Περιεχομένου (Content Delivery Networks), παρατηρείται αυξημένη συμφόρηση και επιβάρυνση του οπισθοζευκτικού δικτύου (backhaul) προκειμένου να προσπελαστεί περιεχόμενο ενδιαφέροντος από τους χρήστες.

Η αποτελεσματική προσωρινή αποθήκευση περιεχομένου στα άκρα του δικτύου (mobile edge caching) και οι συστάσεις (recommendations) περιεχομένου με υψηλή συνάφεια ως προς τα ενδιαφέροντα των χρηστών, έχει προταθεί ως μια πολλά υποσχόμενη λύση στα προαναφερθέντα προβλήματα. Αξιοποιώντας τη συνεργασία των διαχειριστών δικτύου και των παρόχων περιεχομένου, η σύζευξη μεταξύ προσωρινής αποθήκευσης και συστάσεων αποτελεί μια νέα στρατηγική επίτευξης γρήγορης και ικανοποιητικής εξυπηρέτησης των χρηστών. Στη βιβλιογραφία, το πρόβλημα αυτό αναφέρεται ως Κοινό Πρόβλημα Αποθήκευσης και Συστάσεων (Joint Caching and Recommendations Problem).

Στην παρούσα διπλωματική εργασία, μελετάται το παραπάνω πρόβλημα λαμβάνοντας παράλληλα υπόψιν την κινητικότητα των χρηστών. Συγκεκριμένα, θεωρούμε ένα δίκτυο προσωρινής αποθήκευσης που αποτελείται από χρήστες που κινούνται στο χώρο και οι οποίοι μπορούν να αποθηκεύσουν περιεχόμενο στις συσκευές τους και να το παραδώσουν μέσω Device-to-Device (D2D) επικοινωνίας. Αρχικά, προσομοιώνουμε την κίνηση των χρηστών μέσω Τυχαίων Περιπάτων και προτείνουμε έναν τρόπο επιλογής των χρηστών, των οποίων οι συσκευές θα χρησιμοποιηθούν για προσωρινή αποθήκευση περιεχομένου. Στη συνέχεια, με στόχο τη βελτίωση της Ποιότητας Εμπειρίας (Quality Of Experience - QOE) του χρήστη, που εκφράζεται ως συνάρτηση της συνάφειας χρήστη-περιεχομένου και της αναμενόμενης καθυστέρησης παράδοσής του, αντιστοιχίζουμε τα προβλήματα της τοποθέτησης και σύστασης περιεχομένου σε γνωστά αλγοριθμικά προβλήματα, για την επίλυση των οποίων αξιοποιούμε αποδοτικούς αλγορίθμους με εγγυήσεις προσέγγισης και χρόνου εκτέλεσης.

Λέξεις κλειδιά

Προσωρινή Αποθήκευση, Συστήματα Συστάσεων, Αποσυμφόρηση Δικτύου Διανομής Περιεχομένου, Mobile Edge Caching, Επίγνωση Κινητικότητας Χρηστών σε D2D communication, Τυχαίοι Περίπατοι, Προσεγγιστικοί Αλγόριθμοι, k-Median Πρόβλημα, FPTAS, Generalized Assignment Problem, Ποιότητα Εμπειρίας Χρήστη

Abstract

In today's society, with the emergence of the new $5G$ technology, the amount of data and information being transmitted is growing exponentially. At the same time, users' requirements for accessing large volume of information with short delay and for receiving accurate recommendations from content providers are becoming more stringent. As a result, in the context of Content Delivery Networks (CDN), there is increased congestion and heavy burden on the backhaul links in order for users to access content of interest.

Efficient caching of content at the edge of the network (mobile edge caching) along with content recommendations with high relevance to the interests of users, has been proposed as a promising solution to the aforementioned problems. By leveraging the cooperation of network operators and content providers, the coupling between caching and recommendations constitutes a novel strategy which provides fast and satisfactory service to the users. In the literature, this problem is referred to as the Joint Caching and Recommendations Problem.

In this thesis, we address the above problem, while taking into account the user mobility pattern. In particular, we consider a caching network consisting of mobile users who can store content in their devices and deliver it via Device-to-Device (D2D) communication. At first, we simulate the user mobility pattern through Random Walks and we propose a scheme that selects the users whose devices will be used to cache content. Next, aiming to improve the user's Quality of Experience (QOE), which is expressed as a function of user-content relevance and its expected delivery delay, we map the content placement and recommendation problems to well-known algorithmic problems that admit efficient solution algorithms with approximation and runtime guarantees.

Keywords

Caching, Recommendation Systems, Decongestion of Content Delivery Network, Mobile Edge Caching, Mobility Awareness in D2D communication, Random Walks, Approximation Algorithms, k-Median Problem, FPTAS, Generalized Assignment Problem, User Quality of Experience

Ευχαριστίες

Αρχικά, θα ήθελα να ευχαριστήσω θερμά τον καθηγητή, κύριο Συμεών Παπαβασιλείου, για την εμπιστοσύνη που μου έδειξε με την ανάθεση της παρούσας διπλωματικής και τη στήριξή του καθ'όλη τη διάρκεια ενασχόλησής μου με αυτή.

Επίσης, θα ήθελα να ευχαριστήσω ιδιαιτέρως τη διδάκτορα Μαργαρίτα Βιτοροπούλου για την καθοδήγηση, την υπομονή και την αμέριστη βοήθειά της, αλλά και τον διδάκτορα Κωνσταντίνο Τσιτσεκλή για τη βοήθεια και το χρόνο που αφιέρωσε στα πρώτα στάδια της διπλωματικής αυτής εργασίας.

Κλείνοντας, θα ήθελα να ευχαριστήσω την οικογένειά μου αλλά και όλους τους κοντινούς μου ανθρώπους για τη συνεχή τους υποστήριξη. Ξεχωριστά, ευχαριστώ τον αδερφό μου Κώστα, για όλες τις ενδιαφέρουσες συζητήσεις μας.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

Εισαγωγή

1.1 Προσωρινή αποθήκευση και σύσταση περιεχομένου στα άκρα του δικτύου

Ο εξαιρετικά μεγάλος όγκος διακινούμενης πληροφορίας σε συνδυασμό με τις αυξανόμενες απαιτήσεις των χρηστών για λήψη συναφούς και γρήγορα προσπελάσιμου περιεχομένου, στα πλαίσια Δικτύων Διανομής Περιεχομένου, έχουν οδηγήσει σε νέες προκλήσεις ως προς τον φόρτο στις βασικές συνδέσεις δικτύου όσο και στην ποιότητα της υπηρεσίας που λαμβάνουν οι χρήστες, ως προς τη συνάφεια και το χρόνο προσπέλασης προτεινόμενου περιεχομένου. Για την αντιμετώπιση του παραπάνω προβλήματος, η μελέτη της προσωρινής αποθήκευσης περιεχομένου στα άκρα του δικτύου, μέσω της αρχιτεκτονικής Mobile Edge Caching, με σκοπό τη γρήγορη και εύκολη εξυπηρέτηση των χρηστών, σε συνδυασμό με τη σύσταση περιεχομένου με υψηλή συνάφεια ως προς τα ενδιαφέροντα και τις προτιμήσεις των χρηστών προτείνεται ως μια νέα στρατηγική επίτευξης αποδοτικής εξυπηρέτησής των χρηστών. Το παραπάνω πρόβλημα περιγράφεται ως Κοινό Πρόβλημα Αποθήκευσης και Σύστασεων και για την επίλυσή του κρίνεται επιτακτική η εποικοδομητική συνεργασία των διαχειριστών δικτύου και των παρόχων περιεχομένου ή ακόμα και η ενοποίησή τους ως πρόσφατη τάση προσέγγισης του παραπάνω προβλήματος.

1.2 Σκοπός και συνεισφορά της διπλωματικής εργασίας

Στην παρούσα διπλωματική εργασία προτείνεται ένα μοντέλο προσωρινής αποθήκευσης και σύστασης περιεχομένου στα άκρα του δικτύου, εισάγοντας ως παράμετρο στο πρόβλημα την κινητικότητα των χρηστών εντός ενός χώρου ενδιαφέροντος. Για τη μελέτη και αντιμετώπιση του προβλήματος, αξιοποιούμε αποδοτικούς αλγορίθμους με εγγύηση ως προς την προσέγγιση και το χρόνο εκτέλεσης. Η επίλυση του προβλήμα-

τος συνίσταται στην επιλογή των κατάλληλων χρηστών (Clusterheads), των οποίων οι συσκευές χρησιμοποιούνται για την προσωρινή αποθήκευση περιεχομένου και την διανομή του μέσω D2D επικοινωνίας, στην επιλογή περιεχομένου προς αποθήκευση, καθώς και στη σύσταση αντικειμένων στους χρήστες με σκοπό τη βελτίωση της εμπειρίας τους, η οποία εκφράζεται ως συνάρτηση της συνάφειας του προτεινόμενου περιεχομένου με αυτούς και του αναμενόμενου χρόνου προσπέλασής του.

Συγκεκριμένα, θεωρούμε ένα δίκτυο χρηστών, οι οποίοι κινούνται στο χώρο και μπορούν να αποθηκεύσουν προσωρινά περιεχόμενο στις συσκευές τους ώστε να το παραδώσουν σε άλλους χρήστες μέσω D2D επικοινωνίας, στα πλαίσια της αρχιτεκτονικής Mobile Edge Caching. Αρχικά, θεωρώντας ότι οι χρήστες μετακινούνται με μεγάλη πιθανότητα σε μέρη με τα οποία έχουν κάποια συνάφεια και προτιμώντας μέρη τα οποία βρίσκονται γεωγραφικά κοντά τους, προσομοιάζουμε το μοντέλο κίνησης των χρηστών μέσω Τυχαίων Περιπάτων σε γράφο. Στη συνέχεια, υπολογίζοντας τον εκτιμώμενο χρόνο που μεσολαβεί μεταξύ των συναντήσεων (expected inter-contact time) κάθε ζεύγους χρηστών, εφαρμόζουμε προσεγγιστικούς αλγορίθμους για την επιλογή των κατάλληλων χρηστών (Clusterheads) των οποίων οι συσκευές θα χρησιμοποιηθούν για την προσωρινή αποθήκευση περιεχομένου, αντιστοιχίζοντας το πρόβλημα επιλογής Clusterhead στο γνωστό k-Median πρόβλημα. Στις κινητές συσκευές αυτών, αποθηκεύουμε αντικείμενα, τα οποία εμφανίζουν υψηλή συνάφεια με τα ενδιαφέροντα των χρηστών τους οποίους κάθε Clusterhead συναντά πιθανοτικά συχνά. Το πρόβλημα της τοποθέτησης περιεχομένου στους Clusterheads, δηλαδή το ποιά αντικείμενα θα αποθηκεύσουμε και σε ποιούς Clusterheads, αντιστοιχίζεται στο γνωστό Generalized Assignment Problem. Έπειτα, σχεδιάζουμε μια τεχνική για τη σύσταση περιεχομένου σε κάθε χρήστη, στην οποία λαμβάνεται υπόψη τόσο η συνάφεια του περιεχομένου όσο και ο αναμενόμενος χρόνος προσπέλασής του από το χρήστη, ως μετρικές που επηρεάζουν την ποιότητα εμπειρίας του (Quality Of Experience - QOE), και προτείνονται σε κάθε χρήστη τα αντικείμενα αυτά, που μεγιστοποιούν το μέσο QOE του. Τέλος, για διαφορετικούς συνδυασμούς παραμέτρων από τις οποίες εξαρτάται το σύστημα, εξετάζουμε τη συμπεριφορά και την αποτελεσματικότητα του μοντέλου μας, ως προς το μέσο QOE που προσφέρει στους χρήστες.

1.3 Δομή της εργασίας

Η διπλωματική εργασία οργανώνεται σε 6 κεφάλαια ως εξής:

Στο **κεφάλαιο 2**, παραθέτουμε ένα εκτενές μαθηματικό και αλγοριθμικό υπόβαθρο σχετικά με τη Θεωρία Γράφων, τις Μαρκοβιανές Αλυσίδες και τους Τυχαίους Περιπάτους, τα προβλήματα Uncapacitated Facility Location Problem, Metric K-Median Problem, Non-Metric K-Median Problem και Generalized Assignment Problem και προσεγγιστικούς αλγορίθμους επίλυσής τους, έννοιες οι οποίες χρησιμοποιούνται στη διπλωματική αυτή εργασία και με τις οποίες είναι απαραίτητο ο αναγνώστης να έρθει σε επαφή προκειμένου να κατανοήσει το μοντέλο και τις τεχνικές που χρησιμοποιούνται για την ανάπτυξή του.

Στο **κεφάλαιο 3**, αναλύεται η αρχιτεκτονική του Mobile Edge Caching και η

τεχνική του D2D communication, η οποία χρησιμοποιείται ως τρόπος αποθήκευσης και μετάδοσης περιεχομένου μεταξύ των χρηστών στο δικό μας μοντέλο.

Στο **κεφάλαιο 4**, γίνεται αναφορά στα Συστήματα Συστάσεων και στο Κοινό Πρόβλημα Προσωρινής Αποθήκευσης και Συστάσεων, το οποίο προκύπτει από την ανάγκη εποικοδομητικής συνεργασίας των παρόχων περιεχομένου και των διαχειριστών δικτύου και την πρόσφατη τάση ενοποίησής τους, το οποίο μελετάμε κι εδώ, αφού εξετάζουμε ποιά αντικείμενα θα αποθηκεύσουμε και πού (ρόλος διαχειριστή δικτύου) και ποιά αντικείμενα θα προτείνουμε στους χρήστες (ρόλος παρόχου περιεχομένου).

Στο **κεφάλαιο 5**, αναλύεται το μοντέλο του συστήματος καθώς και ο τρόπος προσέγγισης και επίλυσης του προβλήματος, μέσω μαθηματικών εργαλείων και αλγοριθμικών τεχνικών.

Στο **κεφάλαιο 6**, εξετάζεται μέσω προσομοιώσεων, για διαφορετικούς συνδυασμούς παραμέτρων τους συστήματος, η συμπεριφορά και η αποτελεσματικότητα του μοντέλου.

Τέλος, στο **κεφάλαιο 7** παραθέτουμε μία σύνοψη και συμπεράσματα σχετικά με το μοντέλο που αναπτύξαμε και απαριθμούμε διάφορα σημεία εμβάθυνσης και ερευνητικής μελέτης για την υλοποίηση του μοντέλου μας σε ρεαλιστικές συνθήκες.

Κεφάλαιο 2

Θεωρητικό Υπόβαθρο

2.1 Θεωρία Γράφων

Οι γράφοι αποτελούν μαθηματικές δομές αναπαράστασης σχέσεων μεταξύ αντικειμένων και χρησιμοποιούνται ευρέως για την μοντελοποίηση δικτύων στον πραγματικό κόσμο όπως κοινωνικά και βιολογικά δίκτυα, τηλεπικοινωνιακά συστήματα, το διαδίκτυο. Παρακάτω παρουσιάζουμε κάποιες βασικές έννοιες και ορισμούς πάνω στους γράφους.

2.1.1 Ορισμοί

Ονομάζουμε γράφο *(graph)* $G = (V, E)$ ένα διατεταγμένο ζεύγος που αποτελείται από ένα σύνολο κορυφών *(vertices)* V και ένα σύνολο ακμών *(edges)* E που συνδέουν τις κορυφές μεταξύ τους.

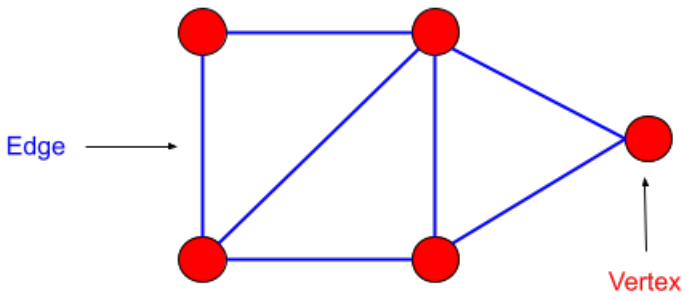


Figure 2.1: Παράδειγμα γράφου

Συνήθως οι γράφοι χωρίζονται σε δύο βασικές κατηγορίες με βάση την κατεύθυνση των ακμών τους.

- **Μη Κατευθυνόμενοι γράφοι** *(undirected graphs)*: Γράφοι των οποίων οι ακμές δεν έχουν κατεύθυνση, δηλαδή κάθε ακμή αποτελεί ένα μη διατεταγμένο ζεύγος κορυφών.
- **Κατευθυνόμενοι γράφοι** *(directed graphs)*: Γράφοι των οποίων οι ακμές έχουν κατεύθυνση, δηλαδή κάθε ακμή αποτελεί ένα διατεταγμένο ζεύγος κορυφών.

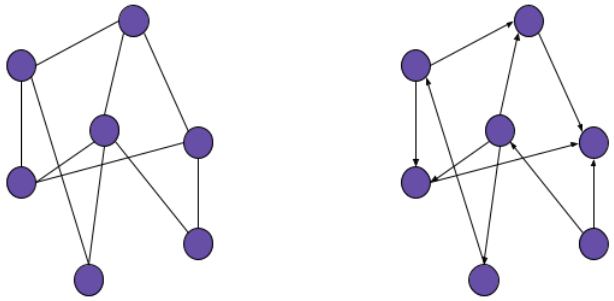


Figure 2.2: Κατευθυνόμενοι και Μη Κατευθυνόμενοι Γράφοι

Στην παρούσα διπλωματική θα μοντελοποιήσουμε τις εκάστοτε δομές μέσω μη κατευθυνόμενων γράφων.

Μία ακόμη κατηγοριοποίηση γράφων είναι σε **απλούς** και **μη απλούς**, όπου στους απλούς γράφους δεν έχουμε βρόχους (ακμή από μία κορυφή προς τον εαυτό της) και πολλαπλές ακμές (διαφορετικές ακμές οι οποίες ενώνουν τις ίδιες κορυφές). Εμείς ασχολούμαστε και με μη απλούς γράφους στους οποίους έχουμε βρόχους (μηδενικού όμως βάρους) αλλά όχι πολλαπλές ακμές.

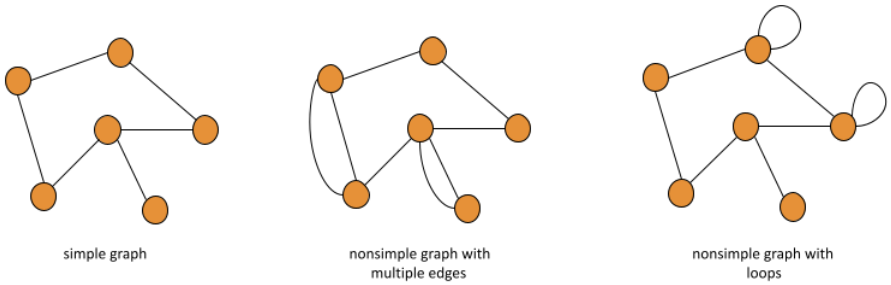


Figure 2.3: Απλοί και Μη Απλοί Γράφοι

2.1.2 Βασικά Χαρακτηριστικά

Παρακάτω παρουσιάζουμε μερικές από τις βασικές έννοιες και χαρακτηριστικά των γράφων τα οποία χρησιμοποιούμε και στην παρούσα εργασία.

- **Υπογράφος G' του G** *(subgraph)*: Γράφος του οποίου οι κορυφές και οι ακμές είναι υποσύνολο των κορυφών και των ακμών του γράφου G, δηλαδή $V(G') \subseteq V(G)$ και $E(G') \subseteq E(G)$.

- **Γειτονιά κορυφής** u *(neighborhood)*: Ορίζεται ως το σύνολο $N(u)$ των κορυφών που ενώνονται απευθείας με ακμή με την εκάστοτε κορυφή u, όπου $N(u) = \{v|(u, v) \in E\}$.

- **Βαθμός κορυφής** u *(degree)*: Το πλήθος $d(u)$ των γειτόνων μιας κορυφής u ή αλλιώς το πλήθος των στοιχείων της γειτονιάς του u, όπου $d(u) = |N(u)|$.

- **Απλό μονοπάτι** *(path)*: Ακολουθία ακμών που ενώνει πλήθος κορυφών, που είναι διακριτές μεταξύ τους (άρα διακριτές θα είναι και οι ακμές).

- **Απόσταση μεταξύ δύο κορυφών** u **και** v *(distance)*: Το πλήθος των ακμών στο συντομότερο απλό μονοπάτι που τις συνδέει. Συμβολίζουμε με $d(u, v)$.

- **Διάμετρος γραφήματος G**: Η μεγαλύτερη απόσταση μεταξύ δύο κορυφών. Συμβολίζουμε $diam(G) = max_{u,v \in V(G)} d(u, v)$.

- **Πλήρες γράφημα ή κλίκα** K *(complete graph or clique)*: Ονομάζουμε πλήρες γράφημα ή κλίκα K το μη κατευθυνόμενο γράφημα στο οποίο κάθε κορυφή είναι γείτονας με όλες τις υπόλοιπες. Για την κλίκα μεγέθους n ισχύει $|V(K)| = n$, $|E(K)| = \frac{n(n-1)}{2}$ και $d(u) = n - 1, \forall u \in V(K)$.

- **Έμβαρο γράφημα** *(weighted graph)*: Γράφημα $G = (V, E, w)$ στο οποίο κάθε ακμή αντιστοιχίζεται με έναν αριθμό, μέσω μιας συνάρτησης βάρους $w : E \to \mathbb{R}$. Συνήθως οι τιμές των βαρών εκφράζουν κόστη, μήκη ή χωρητικότητες. Στη δική μας περίπτωση θα παίζουν ρόλο κόστους, ενώ θα ασχοληθούμε μόνο με ακμές μη αρνητικού βάρους. Το βάρος μιας ακμής $e = (u, v)$ συμβολίζεται ως $w(u, v)$.

- **Συνεκτικό γράφημα** *(connected graph)*: Ένα γράφημα G ονομάζεται συνεκτικό αν κάθε ζεύγος κορυφών του $u, v \in V(G)$ ενώνεται με ένα μονοπάτι.

- **Διμερές γράφημα** *(bipartite graph)*: Διμερές γράφημα ονομάζεται αυτό του οποίου οι κορυφές μπορούν να διαχωριστούν σε δύο ξένα και ανεξάρτητα μεταξύ τους σύνολα U και V ώστε κάθε ακμή του γράφου να συνδέει μία κορυφή του U με μία κορυφή του V.

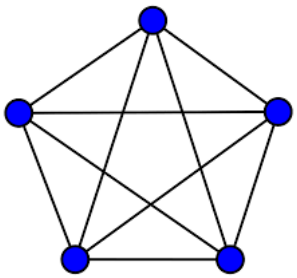


Figure 2.4: Κλίκα μεγέθους 5

2.2 Μαρκοβιανές Αλυσίδες και Τυχαίοι Περίπατοι

Στην εργασία αυτή ασχολούμαστε με στοχαστικές διαδικασίες όπως οι τυχαίοι περίπατοι, μοντέλο το οποίο χρησιμοποιείται για να περιγράψει τον τρόπο κίνησης των χρηστών του δικτύου μας. Η δυναμική πολλών συστημάτων, τα οποία συναντάμε και στον πραγματικό κόσμο, εξαρτάται μόνο από την τρέχουσα κατάσταση του συστήματος και όχι από το πώς το σύστημα βρέθηκε εκεί. Τα συστήματα αυτά ονομάζονται ***μαρκοβιανά***. Ένα χαρακτηριστικό παράδειγμα αποτελεί το σκάκι, όπου για να εκτιμήσει κανείς την πιθανότητα νίκης του κάθε παίκτη κάθε στιγμή, αρκεί μόνο να γνωρίζει την εκάστοτε τρέχουσα κατανομή που έχουν τα πιόνια πάνω στη σκακιέρα και όχι τις προηγούμενες κινήσεις των παικτών.

2.2.1 Συνοπτική περιγραφή των Μαρκοβιανών Αλυσίδων

Στην ενότητα αυτή ορίζουμε το τι είναι μια Μαρκοβιανή Αλυσίδα [1] και αναφέρουμε συνοπτικά κάποιες από τις ιδιότητες και τα χαρακτηριστικά της [2] τα οποία χρησιμοποιούμε στην εργασία αυτή.

Ορισμός 2.2.1 (Μαρκοβιανή Αλυσίδα)**.** *Μια στοχαστική διαδικασία* $X = \{X_n : n \geq 0\}$ *σε ένα αριθμήσιμο σύνολο* S *ονομάζεται* ***Μαρκοβιανή Αλυσίδα*** *(Markov Chain) αν, για οποιοδήποτε* $i, j \in S$ *και* $n \geq 0$*:*

$$P\{(X_{n+1} = j | X_0, ..., X_n\} = P\{X_{n+1} = j | X_n\}$$

Η ποσότητα $p_{ij} = P\{X_{n+1} = j | X_n = i\}$ εκφράζει την πιθανότητα η αλυσίδα να μεταβεί από την κατάσταση i στην κατάσταση j. Για αυτές τις *πιθανότητες μετάβασης* ισχύει ότι

$$\sum_{j \in S} p_{ij} = 1, \forall i \in S$$

Ο πίνακας $P = (p_{ij})$ είναι ο πίνακας μετάβασης της αλυσίδας.

Αναφέρουμε ότι εμείς εξετάζουμε Μαρκοβιανές Αλυσίδες οι οποίες αλλάζουν κατάσταση σε διακριτά χρονικά βήματα, δηλαδή **Μαρκοβιανές Αλυσίδες Διακριτού Χρόνου** *(Discrete-time Markov Chains)* [3].

Για μια μαρκοβιανή αλυσίδα, θεωρούμε το διάνυσμα της αρχικής της κατανομής π_0, όπου $\forall i \in S$, $\pi_0(i) = P\{X_0 = i\}$ εκφράζει την πιθανότητα η μαρκοβιανή αλυσίδα να ξεκινάει από την κατάσταση i. Έχουμε επίσης ότι

$$\sum_{i \in S} \pi_0(i) = 1$$

Αντίστοιχα, το διάνυσμα π_n εκφράζει την κατανομή της αλυσίδας τη χρονική στιγμή n και ισχύει $\forall i \in S$, $\pi_n(i) = P\{X_n = i\}$. Χρησιμοποιώντας το νόμο ολικής πιθανότητας και συμβολισμό πινάκων έχουμε ότι

$$\pi_{n+1} = \pi_n P \tag{2.1}$$

και από επαγωγή

$$\pi_n = \pi_0 P^n \tag{2.2}$$

Ορισμός 2.2.2 (Στάσιμη κατανομή)**.** *Μία κατανομή πιθανότητας της Μαρκοβιανής Αλυσίδας, η οποία παραμένει αμετάβλητη με την πάροδο του χρόνου, ονομάζεται* ***Στάσιμη Κατανομή*** *(Stationary Distribution). Τυπικά, αναπαρίσταται σαν διάνυσμα π του οποίου οι συντεταγμένες είναι πιθανότητες που αθροίζουν στο 1 και δεδομένου του πίνακα μετάβασης P, ικανοποιείται η παρακάτω εξίσωση*

$$\pi = \pi P \tag{2.3}$$

Με άλλα λόγια, το διάνυσμα π είναι αμετάβλητο από τον πίνακα P και ισχύει ότι $\pi_0 = \pi_1 = ... = \pi_n = \pi$.

2.2.2 Τυχαίοι Περίπατοι σε Γράφους

Στην ενότητα αυτή κάνουμε μια σύντομη εισαγωγή στους **Τυχαίους Περιπάτους** σε γράφο [4], οι οποίοι περιγράφουν τον μοντέλο κίνησης των χρηστών του δικτύου μας.

Δεδομένου ενός γράφου και ενός αρχικού σημείου από το οποίο ξεκινάμε, επιλέγουμε ένα γειτονικό του σημείο τυχαία και μετακινούμαστε σε αυτό. Έπειτα επιλέγουμε ένα γειτονικό σημείο αυτού που βρισκόμαστε και μετακινούμαστε σε αυτό κ.ο.κ. Η τυχαία ακολουθία των σημείων που επιλέξαμε με αυτό τον τρόπο είναι ένας τυχαίος περίπατος στο γράφο.

Ένας τυχαίος περίπατος είναι μία πεπερασμένη μαρκοβιανή αλυσίδα που είναι *time-reversible*, δηλαδή μία μαρκοβιανή αλυσίδα που αν ακολουθήσουμε τα βήματά της προς

τα πίσω, από το τέλος προς την αρχή, η αλληλουχία που προκύπτει είναι και αυτή μαρκοβιανή αλυσίδα. Οπότε καταλαβαίνουμε πως στην πραγματικότητα η θεωρία των μαρκοβιανών αλυσίδων και των τυχαίων περιπάτων δε διαφέρει και πολύ.

Οι τυχαίοι περίπατοι έχουν εφαρμογές στη μηχανική αλλά και σε πλήθος επιστημονικών κλάδων όπως η επιστήμη υπολογιστών, η φυσική, η χημεία, η ψυχολογία, η κοινωνιολογία, τα οικονομικά κ.λπ. Πολλά μοντέλα κίνησης στη φύση αποτελούν τυχαίο περίπατο. Μερικά χαρακτηριστικά παραδείγματα είναι η διαδρομή ενός μορίου όταν μετακινείται εντός υγρού ή αέριου σώματος ή η διαδρομή ενός ζώου όταν αναζητά τροφή [5].

Οι τυχαίοι περίπατοι βρίσκουν εφαρμογή σε πληθώρα συστημάτων, οι καταστάσεις των οποίων μπορούν να μοντελοποιηθούν ως κορυφές γράφου και απαντούν σε ποσοτικά ερωτήματα που εγείρονται γύρω από αυτά, όπως: *Πόση ώρα πρέπει να περπατήσουμε για να 1) επιστρέψουμε στο αρχικό σημείο; 2) φτάσουμε σε έναν ενδιάμεσο ή στον τελικό μας προορισμό; 3) επισκεπτούμε όλα τα δυνατά σημεία;* Και στο δικό μας μοντέλο, στο οποίο έχουμε δίκτυο χρηστών οι οποίοι κινούνται στο χώρο, θεωρούμε την κίνηση του κάθε χρήστη ως τυχαίο περίπατο σε γράφο τα σημεία του οποίου αποτελούν τα σημεία του χώρου. Δηλαδή για έναν χρήστη u του δικτύου ο οποίος ξεκινάει από ένα σημείο v_0 του δικτύου, κάθε χρονική στιγμή, εάν βρίσκεται στην κορυφή i, μπορεί να μεταβεί στην κορυφή j με πιθανότητα $p_{ij}(u)$. Οι πιθανότητες αυτές για κάθε χρήστη ορίζονται με τρόπο που θα δούμε σε επόμενο κεφάλαιο.

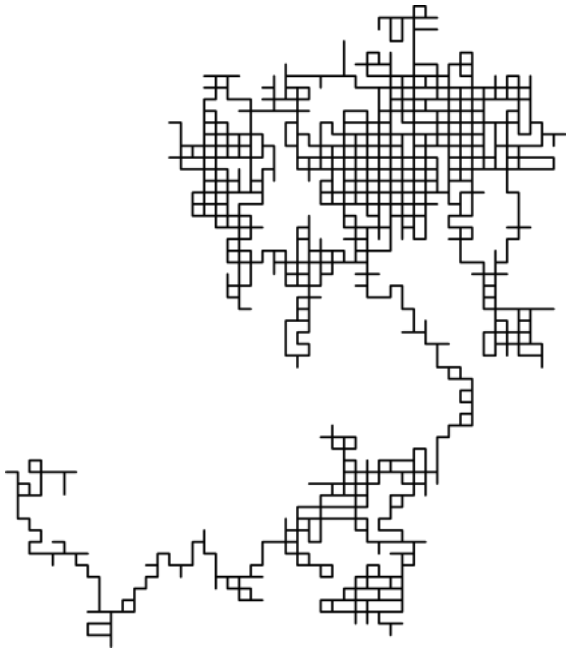


Figure 2.5: Τυχαίος περίπατος σε δισδιάστατο χώρο [5]

Θεώρημα 2.2.1 (Θεώρημα για Τυχαίους Περιπάτους)**.** *Ένας τυχαίος περίπατος σε γράφο συνεκτικό και μη διμερή έχει μοναδική στάσιμη κατανομή.*

Το παραπάνω θεώρημα 2.2.1, χρησιμοποιούμε και εμείς στην παρούσα εργασία.

2.3 Uncapacitated Facility Location Problem

Στην ενότητα αυτή παρουσιάζουμε έναν γνωστό άπληστο (greedy) αλγόριθμο για το *Uncapacitated Facility Location Problem*, ο οποίος διατυπώνεται και αναλύεται διεξοδικά στο [8]. Στο πρόβλημα αυτό ανάγεται το *Metric k-median Problem* και στην επόμενη ενότητα 2.4 παρουσιάζουμε έναν αλγόριθμο για το συγκεκριμένο πρόβλημα, ο οποίος χρησιμοποιεί λύσεις που δίνει ο αλγόριθμος για το Uncapacitated Facility Location Problem. Στο k-median πρόβλημα στηρίζουμε τον τρόπο επιλογής των Clusterheads του συστήματός μας.

2.3.1 Ορισμός του προβλήματος

Στο πρόβλημα αυτό θεωρούμε ένα σύνολο από πελάτες *(clients or demands)* D και ένα σύνολο από παροχές *(facilities)* F. Για κάθε client $j \in D$ και για κάθε facility $i \in F$, υπάρχει ένα κόστος c_{ij} εξυπηρέτησης ή σύνδεσης του client j με το facility i. Επιπλέον υπάρχει ένα κόστος f_i το οποίο σχετίζεται με κάθε facility $i \in F$. Στόχος του προβλήματος είναι να επιλέξουμε ένα υποσύνολο $F' \subseteq F$ έτσι ώστε να ελαχιστοποιηθεί το συνολικό κόστος των facilities στο F' και το συνολικό κόστος εξυπηρέτησης κάθε client $j \in D$ από το κοντινότερό του facility στο F'. Με άλλα λόγια, αναζητούμε το σύνολο F' έτσι ώστε να ελαχιστοποιήσουμε την ποσότητα $\sum_{i \in F'} f_i + \sum_{j \in D} min_{i \in F'} c_{ij}$. Στο πρώτο μέρος του κόστους θα αναφερόμαστε ως *facility cost*, ενώ στο δεύτερο ως *service cost*. Λέμε ότι *ανοίγουμε* τα facilities του συνόλου F' που προκύπτει [6].

Το uncapacitated facility location problem έχει αποδειχθεί ότι είναι NP-hard πρόβλημα, μέσω αναγωγής του στο γνωστό πρόβλημα *Set Cover* και στην γενική του μορφή μπορεί να προσεγγιστεί με ακρίβεια $O(logn)$ [7]. Γι'αυτό, όπως και για τα περισσότερα NP-hard προβλήματα, μας ενδιαφέρουν προσεγγιστικοί αλγόριθμοι επίλυσής τους. Ωστόσο, στις συνηθέστερες περιπτώσεις τους, τα facilities και οι clients αποτελούν σημεία σε ένα μετρικό χώρο *(metric space)*, με τα κόστη c_{ij} να αναπαριστούν την απόσταση του facility i από τον client j. Στην ενότητα αυτή, όταν αναφερόμαστε στο Uncapacitated Facility Location Problem θα αναφερόμαστε στη *metric* εκδοχή του και τα service costs θα επαληθεύουν την τριγωνική ανισότητα. Δηλαδή, δεδομένων clients j, l και facilities i, k, έχουμε ότι $c_{ij} \leq c_{il} + c_{kl} + c_{kj}$. Η εξέταση και ανάλυση αυτής της εκδοχής του προβλήματος πηγάζει από το γεγονός ότι υπάρχουν πολύ καλύτεροι προσεγγιστικοί αλγόριθμοι σε σχέση με τη γενικότερη *(non-metric)* εκδοχή του.

2.3.2 Γραμμικά Προγράμματα και Δυϊκότητα

Παρακάτω ορίζουμε το Ακέραιο Πρόγραμμα *(Integer Program)* του προβλήματος, η αναπαράσταση του οποίου προκύπτει φυσικά από τον ορισμό του προβλήματος.

$$\text{minimize} \quad \sum_{i \in F} f_i y_i + \sum_{i \in F, j \in D} c_{ij} x_{ij} \tag{2.4}$$

$$\text{subject to} \quad \sum_{i \in F} x_{ij} = 1, \qquad \forall j \in D, \tag{2.5}$$

$$x_{ij} \leq y_i, \qquad \forall i \in F, \forall j \in D, \tag{2.6}$$

$$x_{ij} \in \{0,1\}, \qquad \forall i \in F, \forall j \in D,$$

$$y_i \in \{0,1\}, \qquad \forall i \in F.$$

Στο παραπάνω πρόγραμμα έχουμε τις μεταβλητές απόφασης $y_i \in \{0,1\}$ για κάθε facility $i \in F$, δηλαδή αν αποφασίσουμε να ανοίξουμε το facility i, τότε $y_i = 1$, διαφορετικά $y_i = 0$. Για τις μεταβλητές απόφασης $x_{ij} \in \{0,1\}$ για κάθε $i \in F$ και για κάθε $j \in D$ έχουμε ότι $x_{ij} = 1$ αν αποφασίσουμε ο client j να εξυπηρετείται από το facility i, διαφορετικά $x_{ij} = 0$. Η αντικειμενική συνάρτηση 2.4 εκφράζει την ελαχιστοποίηση του συνολικού facility και service cost. Η συνθήκη 2.5 εξασφαλίζει ότι κάθε client $j \in D$ εξυπηρετείται από ακριβώς ένα facility και η συνθήκη 2.6 εξασφαλίζει ότι οι clients εξυπηρετούνται από facilities τα οποία είναι ανοιχτά, αφού η συνθήκη επιτάσσει κάθε φορά που ισχύει ότι $x_{ij} = 1$ και ο client j εξυπηρετείται από το facility i, τότε $y_i = 1$ και το facility είναι ανοιχτό.

Παρότι η παραπάνω μορφή είναι η πιο συνηθισμένη και η μορφή αυτή που προκύπτει φυσικά από το σκοπό του προβλήματος, για την κατανόηση του αλγορίθμου επίλυσης που παραθέτουμε, σκόπιμη είναι και η παρουσίαση της παρακάτω μορφής [8].

Θεωρούμε ότι ένα *αστέρι* αποτελείται από ένα facility και κάποιους clients. Έστω S το σύνολο όλων των αστεριών. Το κόστος ενός αστεριού είναι το άθροισμα του κόστους ανοίγματος του αντίστοιχου facility και του συνολικού κόστους εξυπηρέτησης των clients του αστεριού από το facility αυτό, δηλαδή για κάθε αστέρι (i, D') όπου $i \in F$ και $D' \subseteq D$ το κόστος είναι $f_i + \sum_{j \in D'} c_{ij}$. Τότε μπορούμε να διατυπώσουμε το πρόβλημα ως επιλογή των ελάχιστου κόστους αστεριών, έτσι ώστε κάθε client να ανήκει σε τουλάχιστον ένα αστέρι. Οπότε προκύπτει το παρακάτω Integer Program στο οποίο η μεταβλητή απόφασης x_M εκφράζει το αν επιλέξουμε το αστέρι M και η μεταβλητή c_M εκφράζει το κόστος του αστεριού M.

$$\text{minimize} \quad \sum_{M \in S} c_M x_M$$

$$\text{subject to} \quad \sum_{M : j \in M} x_M \geq 1, \qquad \forall j \in D,$$

$$x_M \in \{0,1\}, \qquad \forall M \in S.$$

Η χαλάρωση του παραπάνω γραμμικού προγράμματος *(LP-relaxation)* είναι:

$$\begin{aligned} \text{minimize} \quad & \sum_{M \in S} c_M x_M & & \\ \text{subject to} \quad & \sum_{M: j \in M} x_M \geq 1, & & \forall j \in D, \\ & x_M \geq 0, & & \forall M \in S. \end{aligned}$$

Το δυϊκό *(dual)* του παραπάνω πρωταρχικού *(primal)* προγράμματος είναι:

$$\begin{aligned} \text{maximize} \quad & \sum_{j \in D} \alpha_j & & & (2.7) \\ \text{subject to} \quad & \sum_{j \in M \cap D} \alpha_j \leq c_M, & & \forall M \in S, & (2.8) \\ & \alpha_j \geq 0, & & \forall j \in D. & \end{aligned}$$

Στο παραπάνω δυϊκό πρόγραμμα, μπορούμε να ερμηνεύσουμε τις μεταβλητές α_j ως τη συνεισφορά του client j ή το μερίδιο του ως προς την κάλυψη του συνολικού κόστους.

2.3.3 Άπληστος Αλγόριθμος για το Uncapacitated Facility Location Problem

Ο αλγόριθμος που παραθέτουμε παρακάτω, έχει κάποια κοινά σημεία με μία μέθοδο που συναντάται συχνά στη διαμόρφωση προσεγγιστικών αλγορίθμων για προβλήματα βελτιστοποίησης, την *primal-dual* μέθοδο. Οι primal-dual αλγόριθμοι ξεκινούν με μία εφικτή λύση του δυϊκού προβλήματος και χρησιμοποιούν πληροφορία σχετικά με το δυϊκό πρόβλημα ώστε να παραχθεί μια λύση για το primal πρόβλημα, πιθανώς μη εφικτή. Αν η primal λύση είναι όντως μη εφικτή, τότε η dual λύση τροποποιείται έτσι ώστε να αυξηθεί η αντικειμενική συνάρτηση του δυϊκού προβλήματος. Στη γενική κλασική περίπτωση προβλημάτων, η μέθοδος αυτή συνοψίζεται στο εξής: Ξεκινάμε από μία εφικτή λύση για το dual πρόβλημα και μία συνήθως μη εφικτή λύση για το primal. Διατηρώντας εφικτή τη λύση για το dual, αυξάνουμε τις δυϊκές μεταβλητές μέχρι μία από τις συνθήκες του dual προβλήματος να γίνει tight, δηλαδή για τις συνθήκες που φράσσουν τις δυϊκές μεταβλητές με "$\leq$", οι μεταβλητές αυτές να ικανοποιούν το "$=$" στη συνθήκη. Αυτό υποδεικνύει ένα αντικείμενο που πρέπει να προσθέσουμε στην primal λύση μας. Συνεχίζουμε μέχρι η primal λύση να γίνει εφικτή. Όταν, στη συνέχεια, αναλύουμε το κόστος της primal λύσης, κάθε αντικείμενο στη λύση έχει δοθεί από μία tight συνθήκη του δυϊκού προβλήματος. Τότε μπορούμε να ξαναγράψουμε το κόστος της primal λύσης μέσω των δυϊκών μεταβλητών. Στη συνέχεια συγκρίνουμε αυτό το κόστος με το κόστος της dual αντικειμενικής συνάρτησης και αποδεικνύουμε ότι το primal κόστος φράσσεται από την dual αντικειμενική συνάρτηση επί κάποιο παράγοντα, ο οποίος συνήθως αποτελεί και τον παράγοντα προσέγγισης

του αλγορίθμου, το οποίο υποδεικνύει ότι είμαστε κοντά σε μία τιμή της βέλτιστης λύσης. Το τελευταίο συμπέρασμα οφείλεται στο γεγονός ότι μία dual λύση είναι πάντα μικρότερη ή ίση από μία λύση του αντίστοιχου primal προβλήματος. Η έννοια αυτή είναι γνωστή ως *weak duality*.

Η μέθοδος που χρησιμοποιείται για την ανάπτυξη του παρακάτω αλγορίθμου είναι γνωστή ως *dual fitting* και έχει κοινά σημεία με την primal-dual μέθοδο που περιγράψαμε συνοπτικά παραπάνω, ως προς το γεγονός ότι κι εδώ έχουμε το LP-relaxation πρόγραμμα και το δυϊκό του και επαναληπτικά κάνουμε primal και dual updates. Η διαφορά εδώ έγκειται στο γεγονός ότι η δυϊκή λύση που προκύπτει είναι, γενικά, μη εφικτή. Ωστόσο, η ακέραια λύση του primal προγράμματος καλύπτεται πλήρως από την dual που προκύπτει, δηλαδή η αντικειμενική συνάρτηση της primal λύσης, φράσσεται από αυτή της dual. Το κύριο βήμα της ανάλυσης συνίσταται στη διαίρεση της δυϊκής λύσης με έναν κατάλληλο παράγοντα γ και στην απόδειξη ότι αυτή, με αυτό τον τρόπο , γίνεται εφικτή και κάνει *fit* στη δεδομένη περίπτωση. Τότε, η "συρρικνωμένη" δυϊκή λύση, αφού είναι εφικτή, θα είναι και κάτω φράγμα στη βέλτιστη λύση (από weak duality) και ο παράγοντας γ θα αποτελεί τον παράγοντα προσέγγισης του αλγορίθμου.

Έχοντας αναφέρει επιγραμματικά τις παραπάνω έννοιες, η γνώση των οποίων είναι σημαντική για την διαισθητική κατανόηση του αλγορίθμου, μπορούμε να προχωρήσουμε στην διατύπωσή του. Στον παρακάτω ψευδοκώδικα 2.1 (τον οποίο καλούμε ως συνάρτηση με όνομα UFLP και ορίσματα F, D, c_{ij}, f_i) το σύνολο $S \subseteq D$ περιέχει τους clients που δεν έχουν ακόμη συνδεθεί με κάποιο ανοιχτό facility και το σύνολο $X \subseteq F$ περιέχει τα ανοιχτά facilities, για κάθε δεδομένη στιγμή. Επίσης ο συμβολισμός $(\alpha)_+$ ισοδυναμεί με την έκφραση $max(\alpha, 0)$ και $c(j, X) = min_{i \in X} c_{ij}$. Ο πίνακας H περιέχει για κάθε χρήστη, το ανοιχτό facility με το οποίο συνδέεται τελικά (αρχικοποιείται σε none τιμή). Ο αλγόριθμος επιστρέφει το σύνολο των ανοιχτών facilities X και τον πίνακα H με τις αναθέσεις των clients στα κοντινότερά τους ανοιχτά facilities.

Ο παρακάτω αλγόριθμος 2.1 συνοψίζεται στην εξής περιγραφή: Αυξάνουμε τη συνεισφορά α_j των clients ομοιόμορφα μέχρι κάθε client να συνδεθεί με ένα facility i του οποίου το κόστος καλύπτεται από τις συνεισφορές. Ένας client j, ο οποίος δεν είναι συνδεδεμένος με ένα facility i, συνεισφέρει τη διαφορά του α_j και του service cost για το κόστος του facility i, δηλαδή συνεισφέρει την ποσότητα $(\alpha_j - c_{ij})_+$ για το κόστος του facility i. Όταν οι συνολικές συνεισφορές για ένα facility i ισούνται με το facility cost f_i, ανοίγουμε το facility i. Επίσης, επιτρέπουμε ήδη συνδεδεμένους clients να συνεισφέρουν τη διαφορά σε service costs για το facility cost ενός πιο κοντινού σε αυτές facility. Δηλαδή, αν ο client j είναι την τρέχουσα στιγμή συνδεδεμένος με ένα facility στο σύνολο X, συνεισφέρει την ποσότητα $(c(j, X) - c_{ij})_+$ για το facility cost του i. Αν το facility i ανοιχτεί, τότε ο client j συνδέεται με το facility i, μειώνοντας το service cost του κατά $(c(j, X) - c_{ij})_+$. Όταν κάθε client έχει συνδεθεί με κάποιο ανοιχτό facility, ο αλγόριθμος τερματίζεται.

UFLP(F, D, c_{ij}, f_i)

$\alpha \leftarrow 0$
$S \leftarrow D$
$X \leftarrow \emptyset$
$H[j] \leftarrow none,\ \forall j \in D$
Όσο $S \neq \emptyset$
    Αύξηση α_j για κάθε $j \in S$ ομοιόμορφα μέχρι είτε [$\exists j \in S, i \in X$ ώστε $\alpha_j = c_{ij}$]
        είτε [$\exists i \in F - X : \sum_{j \in S}(\alpha_j - c_{ij})_+ + \sum_{j \notin S}(c(j, X) - c_{ij})_+ = f_i$]
    Αν $\exists j \in S, i \in X$ ώστε $\alpha_j = c_{ij}$ **τότε**
        $S \leftarrow S - \{j\}$
    αλλιώς
        $X \leftarrow X \cup \{i\}$
        Για όλα τα $j \in S$ για τα οποία $\alpha_j \geq c_{ij}$
            $S \leftarrow S - \{j\}$
Για κάθε $j \in D$:
    $H[j] \leftarrow argmin_{i \in X} c_{ij}$
Επιστροφή του συνόλου X και του πίνακα H

Algorithm 2.1: Άπληστος αλγόριθμος για το Uncapacitated Facility Location Problem

Αν προσπαθήσουμε να αναγνωρίσουμε στον συγκεκριμένο αλγόριθμο τις τεχνικές *primal-dual* και *dual fitting* που περιγράψαμε παραπάνω και τη σύνδεσή του με το δυϊκό πρόβλημα 2.7 που διατυπώσαμε στην προηγούμενη ενότητα, θα λέγαμε τα εξής: Η πρώτη ανισότητα 2.8 του προγράμματος μπορεί να επαναδιατυπωθεί ως $\sum_{j \in D} max(0, \alpha_j - c_{ij}) \leq f_i$ για κάθε facility i ή αλλιώς χρησιμοποιώντας τον ισοδύναμο συμβολισμό $\sum_{j \in D}(\alpha_j - c_{ij})_+ \leq f_i$. Οι δυϊκές μεταβλητές συντελούν στο να βρούμε το περισσότερο *cost-effective* αστέρι σε κάθε επανάληψη του αλγορίθμου, δηλαδή το αστέρι (i, D') που μεγιστοποιεί το λόγο $(f_i + \sum_{j \in D'} c_{ij})/|D'|$. Ξεκινώντας να αυξάνουμε ομοιόμορφα τις δυϊκές μεταβλητές a_j όλων των μη συνδεμένων clients ταυτόχρονα, το περισσότερο cost-effective αστέρι θα είναι το πρώτο για το οποίο θα ισχύει ότι $\sum_{j \in D}(\alpha_j - c_{ij})_+ = f_i$ (tight δυϊκή συνθήκη). Έτσι αν στον αλγόριθμο επιλέγαμε τα αστέρια που κάνουν tight κάποια συνθήκη του δυϊκού προβλήματος και ενώ δεν έχουμε απαραίτητα ακόμα κάποια εφικτή λύση όπως όντως συμβαίνει και στον παραπάνω αλγόριθμο (ο αλγόριθμος ξεκινά με $S = D$, δηλαδή με όλους τους clients να μην είναι συνδεδεμένοι με κανένα ανοιχτό facility και τρέχει μέχρι όλοι να συνδεθούν και η λύση να γίνει εφικτή), θα είχαμε έναν primal-dual αλγόριθμο. Αυτό όμως δεν είναι ακριβώς αυτό που συμβαίνει, αφού, όπως παρατηρούμε, η δεύτερη συνθήκη του αλγορίθμου επιτρέπει στους clients που είναι ήδη συνδεδεμένοι με κάποιο facility, να συνεισφέρουν και στο κόστος άλλων facilities την ποσότητα που θα εξοικονομούσαν αν συνδέονταν σε αυτά. Αυτό δεν μπορεί να εξασφαλίσει ότι οι ποσότητες αυτές α_j είναι εφικτές, όπως στην περίπτωση της primal-dual ανάλυσης, και πρέπει να υποστούν κλιμάκωση γ ώστε να γίνουν εφικτές, οπότε ο αλγόριθμος αυτός είναι dual fitting. Ο παράγοντας γ, όπως διατυπωνεται στο παρακάτω θεώρημα είναι ίσος με 1.61, άρα

αυτός είναι και ο παράγοντας προσέγγισης του αλγορίθμου.

Θεώρημα 2.3.1. *Ο αλγόριθμος 2.1 είναι 1.61-προσεγγιστικός, με πολυπλοκότητα $O(n^3)$, όπου n είναι το συνολικό πλήθος των clients και των facilities.*

Ο αλγόριθμος 2.1 διατυπώνεται και αναλύεται στο [8] όπως και η απόδειξη του θεώρηματος 2.3.1.

Σημείωση: *Έχει αποδειχτεί ότι ο καλύτερος δυνατός παράγοντας προσέγγισης για το Uncapacitated Facility Location Probelm είναι 1.463 [9], εκτός κι αν κάθε πρόβλημα στο NP έχει κάποιο αλγόριθμο πολυπλοκότητας $O(n^{O(loglogn)})$ και ο τρέχων καλύτερος αλγόριθμος είναι 1.488-προσεγγιστικός [10]. Βλέπουμε λοιπόν ότι ο αλγόριθμος που χρησιμοποιούμε είναι πολύ κοντά στις προσεγγίσεις αυτές.*

2.3.4 Χρήσιμες Προσθήκες

Εδώ, θα αναφέρουμε κάποια επιπλέον χαρακτηριστικά και ιδιότητες του Uncapacitated Facility Location Problem για να καταλήξουμε σε ένα θεώρημα που χρησιμοποιούμε για την ανάλυση του αλγορίθμου για το k-median πρόβλημα που περιγράφουμε στην επόμενη ενότητα.

Προσθέτουμε, αρχικά, το LP-χαλαρωμένο πρόγραμμα του Uncapacitated Facility Location Problem για την πρώτη μορφή Ακεραίου Προγράμματος που παραθέσαμε (2.4):

$$\begin{aligned} \text{minimize} \quad & \sum_{i\in F} f_i y_i + \sum_{i\in F, j\in D} c_{ij} x_{ij} && && (2.9)\\ \text{subject to} \quad & \sum_{i\in F} x_{ij} = 1, && \forall j \in D, \\ & x_{ij} \le y_i, && \forall i \in F, \forall j \in D, \\ & x_{ij} \ge 0, && \forall i \in F, \forall j \in D, \\ & y_i \ge 0, && \forall i \in F. \end{aligned}$$

Κατόπιν, προσθέτουμε το αντίστοιχο δυϊκό πρόγραμμα:

$$\begin{aligned} \text{maximize} \quad & \sum_{j\in D} v_j && && (2.10)\\ \text{subject to} \quad & \sum_{j\in D} w_{ij} \le f_i, && \forall i \in F, \\ & v_j - w_{ij} \le c_{ij}, && \forall i \in F, \forall j \in D, \\ & w_{ij} \ge 0, && \forall i \in F, \forall j \in D. \end{aligned}$$

Από τα παραπάνω προκύπτει το εξής θεώρημα, από τις [8] [12].

Θεώρημα 2.3.2. *Έστω S το σύνολο των facilities που ανοίγει ο αλγόριθμος 2.1 για το Uncapacitated Facility Location Problem. Τότε ισχύει ότι*
$c(S) \leq 1.61(\sum_{j \in D} v_j - \sum_{i \in S} f_i)$, *όπου* $c(S) = \sum_{j \in D} min_{i \in S} c_{ij}$

2.4 Metric k-median Problem

Στο σημείο αυτό μπορούμε να ορίσουμε το πρόβλημα που χρησιμοποιούμε άμεσα στην εργασία αυτή και το οποίο είναι παραλλαγή του Uncapacitated Facility Location Problem που περιγράψαμε προηγουμένως.

2.4.1 Ορισμός του προβλήματος

Όπως και στο Uncapacitated Facility Location Problem, έτσι κι εδώ έχουμε ένα σύνολο από clients D και ένα σύνολο από facilities F με κόστη εξυπηρέτησης c_{ij}, για κάθε $i \in F$ και $j \in D$. Όμως εδώ δεν έχουμε facility costs, κόστη δηλαδή ανοίγματος των facilities, αλλά έχουμε ως παράμετρο, έναν θετικό ακέραιο k, ο οποίος είναι ένα άνω φράγμα στο πλήθος των facilities που μπορούν να ανοιχτούν. Σκοπός του προβλήματος είναι να επιλέξουμε ένα υποσύνολο από facilities, πλήθους το πολύ k και μία σύνδεση των clients στα ανοιχτά facilities, έτσι ώστε να ελαχιστοποιηθεί το συνολικό κόστος εξυπηρέτησης.

Και αυτό το πρόβλημα έχει αποδειχτεί ότι είναι NP-hard και, όπως και το Uncapacitated Facility Location Problem, στη γενική του non-metric μορφή, δεν μπορεί να προσεγγιστεί με μεγαλύτερη ακρίβεια από $O(logn)$ [11]. Θεωρούμε κι εδώ τη metric εκδοχή του προβλήματος, όπου οι clients και τα facilities είναι σημεία στο μετρικό χώρο και τα service costs c_{ij} είναι η απόσταση μεταξύ του client j και του facility i, οπότε ισχύει κι εδώ η τριγωνική ανισότητα.

2.4.2 Γραμμικά Προγράμματα και η τεχνική Langrangian Relaxation

Μπορούμε να μοντελοποιήσουμε το k-median problem με το παρακάτω ακέραιο πρόγραμμα, με μορφή αρκετά παρόμοια με το αντίστοιχο πρόγραμμα του Uncapacitated Facility Location Problem.

$$\text{minimize} \quad \sum_{i\in F, j\in D} c_{ij}x_{ij} \tag{2.11}$$

$$\begin{aligned}
\text{subject to} \quad & \sum_{i\in F} x_{ij} = 1, && \forall j \in D, \\
& x_{ij} \le y_i, && \forall i \in F, \forall j \in D, \\
& \sum_{i\in F} y_i \le k, && \\
& x_{ij} \in \{0,1\}, && \forall i \in F, \forall j \in D, \\
& y_i \in \{0,1\}, && \forall i \in F.
\end{aligned} \tag{2.12}$$

Οι μόνες διαφορές του ακέραιου αυτού προγράμματος σε σχέση με το αντίστοιχο του Uncapacitated Facility Location Problem, είναι στην αντικειμενική συνάρτηση 2.11, στην οποία δεν υπάρχει ο όρος για τα facility costs και στη συνθήκη 2.12, η οποία φράσσει το πλήθος των ανοιχτών facilities από το k. Οι υπόλοιπες συνθήκες και οι μεταβλητές απόφασης εκφράζουν ό,τι και στο Uncapacitated Facility Location Problem.

Η χαλάρωση του παραπάνω γραμμικού προγράμματος *(LP-relaxation)* είναι:

$$\text{minimize} \quad \sum_{i\in F, j\in D} c_{ij}x_{ij} \tag{2.13}$$

$$\begin{aligned}
\text{subject to} \quad & \sum_{i\in F} x_{ij} = 1, && \forall j \in D, \\
& x_{ij} \le y_i, && \forall i \in F, \forall j \in D, \\
& \sum_{i\in F} y_i \le k, && \\
& x_{ij} \ge 0, && \forall i \in F, \forall j \in D, \\
& y_i \ge 0, && \forall i \in F.
\end{aligned}$$

Χρησιμοποιούμε, στο σημείο αυτό, την ιδέα του *Langrangian Relaxation*. Με την τεχνική αυτή εξαλείφουμε τους περίπλοκους περιορισμούς αλλά προσθέτουμε ποινές για την παραβίασή τους στην αντικειμενική συνάρτηση. Μπορούμε, επομένως, να πλησιάσουμε ακόμα περισσότερο την μορφή του γραμμικού προγράμματος με το οποίο αναπαρίσταται το Uncapacitated Facility Location Problem, καταργώντας τη συνθήκη $\sum_{i\in F} y_i \le k$, αλλά προσθέτοντας ως ποινή την ποσότητα $\lambda(\sum_{i\in F} y_i - k)$ στην αντικειμενική συνάρτηση, για κάποια σταθερά $\lambda \ge 0$. Αυτή η ποινή ευνοεί λύσεις που υπακούουν στη συνθήκη που καταργήσαμε. Οπότε το πρόγραμμα που έχουμε τώρα για το k-median πρόβλημα είναι:

$$
\begin{aligned}
\text{minimize} \quad & \sum_{i\in F, j\in D} c_{ij}x_{ij} + \sum_{i\in F} \lambda y_i - \lambda k && && (2.14)\\
\text{subject to} \quad & \sum_{i\in F} x_{ij} = 1, && \forall j \in D,\\
& x_{ij} \le y_i, && \forall i \in F, \forall j \in D,\\
& x_{ij} \ge 0, && \forall i \in F, \forall j \in D,\\
& y_i \ge 0, && \forall i \in F.
\end{aligned}
$$

Αρχικά παρατηρούμε ότι κάθε εφικτή λύση για το LP-relaxed πρόγραμμα του k-median προβλήματος 2.13, είναι εφικτή και για το γραμμικό πρόγραμμα 2.14. Επίσης, για κάθε $\lambda \ge 0$, κάθε εφικτή λύση του LP-relaxed προγράμματος του k-median προβλήματος 2.13, έχει τιμή αντικειμενικής συνάρτησης στο πρόγραμμα 2.14, που αποτελεί κάτω φράγμα της αντίστοιχης τιμής της στο 2.13. Άρα το πρόγραμμα 2.14 είναι και κάτω φράγμα του κόστους μιας βέλτιστης λύσης OPT_k για το k-median πρόβλημα. Επίσης, αν εξαιρέσουμε τον όρο $-\lambda k$, το γραμμικό πρόγραμμα 2.14 είναι ακριβώς ίδιο με το LP-relaxed πρόγραμμα 2.9 για το Uncapacitated Facility Location Problem, στο οποίο για τα facility costs ισχύει $f_i = \lambda$, $\forall i \in F$.

Το αντίστοιχο δυϊκό πρόγραμμα του προγράμματος 2.14 είναι:

$$
\begin{aligned}
\text{maximize} \quad & \sum_{j\in D} v_j - \lambda k && && (2.15)\\
\text{subject to} \quad & \sum_{j\in D} w_{ij} \le \lambda, && \forall i \in F,\\
& v_j - w_{ij} \le c_{ij}, && \forall i \in F, \forall j \in D,\\
& w_{ij} \ge 0, && \forall i \in F, \forall j \in D.
\end{aligned}
$$

Και εδώ έχουμε ότι το πρόγραμμα 2.15 είναι ίδιο με το δυϊκό του LP-relaxed προγράμματος για το Uncapacitated Facility Location Problem 2.10, με διαφορά ότι για τα facility costs ισχύει ότι $f_i = \lambda$, $\forall i \in F$ και στην αντικειμενική συνάρτηση υπάρχει η ποινή $-\lambda k$.

2.4.3 Αλγόριθμος για το Metric k-median

Φαίνεται, λοιπόν, φυσικό το να θέλουμε να τροποποιήσουμε τον άπληστο αλγορίθμο 2.1 για το Uncapacitated Facility Location Problem, με όλα τα facility costs f_i να είναι ίσα με λ για κάποιο $\lambda \ge 0$, ώστε να τον χρησιμοποιήσουμε στο πρόβλημα αυτό. Αν και μπορούμε, με έναν τέτοιο τρόπο, να ανοίξουμε κάποια facilities, πώς μπορούμε να εξασφαλίσουμε ότι θα ανοίξουμε το πολύ k και πώς μπορεί να προκύψει κάποια

εγγύηση για την προσέγγιση;

Από το θεώρημα 2.3.2 έχουμε ότι:

$$c(S) \leq 1.61(\sum_{j \in D} v_j - \sum_{i \in S} f_i)$$

Αντικαθιστώντας όλα τα κόστη f_i με λ, προκύπτει ότι:

$$c(S) \leq 1.61(\sum_{j \in D} v_j - \lambda |S|)$$

Παρατηρούμε, λοιπόν, ότι αν ο αλγόριθμος 2.1 άνοιγε ένα σύνολο από facilities S, έτσι ώστε $|S| = k$, τότε θα είχαμε ότι:

$$c(S) \leq 1.61(\sum_{j \in D} v_j - \lambda k) \leq 1.61 \cdot OPT_k$$

Αυτό προκύπτει από το γεγονός ότι η ποσότητα $\sum_{j \in D} v_j - \lambda k$ είναι η αντικειμενική συνάρτηση του δυϊκού προγράμματος 2.15, η οποία όπως αναφέραμε προηγουμένως, είναι κάτω φράγμα και σε μια βέλτιστη λύση για το k-median πρόβλημα.

Μια φυσική ιδέα είναι να προσπαθήσουμε να βρούμε κάποια τιμή για το λ έτσι ώστε ο αλγόριθμος 2.1 για το Uncapacitated Facility Location Problem να ανοίγει ένα σύνολο από facilities S για το οποίο να ισχύει $|S| = k$. Αυτό μπορεί να συμβεί μέσω δυαδικής ανάζήτησης. Για να ξεκινήσουμε την αναζήτηση, χρειαζόμαστε δύο αρχικές τιμές για το λ, μία η οποία να ανοίγει τουλάχιστον k facilities και μία που να ανοίγει το πολύ k facilities. Μπορούμε να θεωρήσουμε ότι για $\lambda = 0$, ο αλγόριθμος 2.1 ανοίγει τουλάχιστον k, διαφορετικά προσθέτουμε τα υπόλοιπα $k - |S|$ χωρίς επιπλέον κόστος. Επίσης, είναι εύκολο να δούμε ότι για $\lambda = \sum_{j \in D} \sum_{i \in F} c_{ij}$, ο αλγόριθμος 2.1 ανοίγει ένα facility.

Οπότε μπορούμε να τρέξουμε τη δυαδική αναζήτηση ως εξής: Θέτουμε αρχικά $\lambda_1 = 0$ και $\lambda_2 = \sum_{j \in D} \sum_{i \in F} c_{ij}$, για τις οποίες τιμές ο αλγόριθμος επιστρέφει λύσεις S_1 και S_2 αντίστοιχα με $|S_1| > k$ και $|S_2| < k$. Τρέχουμε, έπειτα, τον αλγόριθμο για $\lambda = \frac{1}{2}(\lambda_1 + \lambda_2)$. Αν ο αλγόριθμος επιστρέψει λύση S με $|S| = k$, λόγω των όσων αναφέραμε παραπάνω, έχουμε τελειώσει και έχουμε μια λύση κόστους το πολύ $1.61OPT_k$. Αν $|S| > k$, τότε θέτουμε $\lambda_1 = \lambda$ και $S_1 = S$, διαφορετικά αν ισχύει ότι $|S| < k$, θέτουμε $\lambda_2 = \lambda$ και $S_2 = S$. Επαναλαμβάνουμε με τον ίδιο τρόπο, μέχρι είτε ο αλγόριθμος να βρει λύση με ακριβώς k facilities, είτε το διάστημα $\lambda_2 - \lambda_1$ να γίνει αρκετά μικρό, που στην περίπτωση αυτή μπορούμε να συνθέσουμε μια λύση S που συνδυάζει τις τελευταίες λύσεις S_1 και S_2 που έδωσε ο αλγόριθμος, για την οποία θα ισχύει ότι $|S| = k$. Η μικρότερη τιμή που επιτρέπουμε να πάρει το διάστημα $\lambda_2 - \lambda_1$ είναι ίση με $\varepsilon c_{min}/|F|$, όπου c_{min} είναι το μικρότερο από τα service costs. Οπότε η δυαδική αναζήτηση κάνει $O(log\frac{|F|\sum c_{ij}}{\varepsilon c_{min}})$ κλήσεις του αλγορίθμου 2.1, άρα τρέχει σε πολυωνιμικό χρόνο.

Αν δεν έχουμε λοιπόν τερματίσει με ακριβώς k facilities, τότε ο αλγόριθμος τερματίζει με λύσεις S_1 και S_2 για τις οποίες ισχύει $|S_1| > k > |S_2|$ και τις οποίες μπορούμε να χρησιμοποιήσουμε για να συνθέσουμε λύση S με $|S| = k$ για την οποία ισχύει ότι $c(S) \leq 2(1.61 + \varepsilon)OPT_k$. Έστω α_1 και α_2 ώστε $\alpha_1|S_1| + \alpha_2|S_2| = k$, $\alpha_1 + \alpha_2 = 1$ και $\alpha_1, \alpha_2 \geq 0$, τότε έχουμε ότι

$$\alpha_1 = \frac{k - |S_2|}{|S_1| - |S_2|} \quad \kappa\alpha\iota \quad \alpha_2 = \frac{|S_1| - k}{|S_1| - |S_2|}$$

. Ο αλγόριθμος που παρουσιάζουμε διακρίνει τότε δύο περιπτώσεις:

- Αν $\alpha_2 \geq \frac{1}{2}$, τότε παίρνουμε τη λύση S_2 ως λύση για το k-median.
- Αν $\alpha_2 < \frac{1}{2}$, τότε για κάθε facility $i \in S_2$, ανοίγουμε το κοντινότερο του facility $h \in S_1$. Αν με αυτό τον τρόπο δεν ανοιχτούν $|S_2|$ facilities στο S_1, επειδή ενδεχομένως κάποια facilities στο S_2 έχουν ως κοντινότερο το ίδιο facility στο S_1, ανοίγουμε τυχαία κάποια ακόμα facilites στο S_1 έτσι ώστε να έχουν ανοιχτεί συνολικά ακριβώς $|S_2|$ facilities στο S_1. Έπειτα επιλέγουμε ένα τυχαίο υποσύνολο πλήθους $k - |S_2|$ από τα $|S_1| - |S_2|$ facilities του S_1 που απομένουν και ανοίγουμε και αυτά.

Αναφέρουμε ότι στη δική μας εργασία μας ενδιαφέρει να έχουμε ακριβώς k facilities οπότε αρκεί να προσθέσουμε στον παραπάνω αλγόριθμο που περιγράψαμε ότι, στην περίπτωση $\alpha_2 \geq \frac{1}{2}$, που είναι και η μοναδική περίπτωση που μας δίνει λιγότερα από k facilities, εκτός από τα facilities της λύσης S_2 επιλέγουμε και κάποια τυχαία facilities από το S_1 ώστε να έχουμε συνολικά k. Προφανώς με την προσθήκη περισσότερων facilities το κόστος δεν αυξάνεται, άρα η προσέγγιση του αλγορίθμου διατηρείται.

Παρακάτω παραθέτουμε τον αλγόριθμο αυτό 2.2 με την παραπάνω προσθήκη, σε ψευδοκώδικα ως συνάρτηση με όνομα Metric_k_Median και με ορίσματα F, D, c_{ij}, k. Ο αλγόριθμος επιστρέφει το σύνολο των επιλεγμένων facilities καθώς και τον πίνακα H που περιέχει τις αναθέσεις των clients στα κοντινότερά τους επιλεγμένα facilities. (Αναφέρουμε ότι δε χρειαζόμαστε σε αυτό τον αλγόριθμο τον πίνακα των αναθέσεων που επιστρέφει το Uncapacitated Facility Location Problem για αυτό και όταν καλούμε τον αλγόριθμο UFLP το δεύτερο στοιχείο που επιστρέφει το αγνοούμε και στη θέση του βάζουμε "_"):

Metric_k_Median (F, D, c_{ij}, k)

$S_1 \leftarrow \emptyset$
$S_2 \leftarrow \emptyset$
$S \leftarrow \emptyset$
$\lambda_1 \leftarrow 0$
$\lambda_2 \leftarrow \sum_{j \in D} \sum_{i \in F} c_{ij}$
$H[j] \leftarrow none, \ \forall j \in D$
$S_1, \_ \leftarrow UFLP(F, D, c_{ij}, \lambda_1)$
$S_2, \_ \leftarrow UFLP(F, D, c_{ij}, \lambda_2)$
Αν $|S_1| = k$ **τότε**
    Για κάθε $j \in D$:
        $H[j] \leftarrow argmin_{i \in S_1} c_{ij}$
    Επιστροφή του συνόλου S_1 και του πίνακα H
αλλιώς αν $|S_2| = k$
    Για κάθε $j \in D$:
        $H[j] \leftarrow argmin_{i \in S_2} c_{ij}$
    Επιστροφή του συνόλου S_2 και του πίνακα H
αλλιώς
    Όσο $\lambda_1 - \lambda_2 > \varepsilon c_{min}/|F|$
        $\lambda \leftarrow \frac{\lambda_1 + \lambda_2}{2}$
        $S, \_ \leftarrow UFLP(F, D, c_{ij}, \lambda)$
        Αν $|S| = k$ **τότε**
            Για κάθε $j \in D$:
                $H[j] \leftarrow argmin_{i \in S} c_{ij}$
            Επιστροφή του συνόλου S και του πίνακα H
        αλλιώς αν $|S| > k$
            $\lambda_1 = \lambda$
            $S_1 = S$
        αλλιώς
            $\lambda_2 = \lambda$
            $S_2 = S$
    $\alpha_1 \leftarrow \frac{k - |S_2|}{|S_1| - |S_2|}$
    $\alpha_2 \leftarrow \frac{|S_1| - k}{|S_1| - |S_2|}$
    Αν $\alpha_2 \geq \frac{1}{2}$ **τότε**
        $S \leftarrow S_2$
        Όσο $|S| \neq k$
            Τυχαία Επιλογή $h \in S_1$
            $S \leftarrow S \cup \{h\}$
            $S_1 \leftarrow S_1 - \{h\}$
        Για κάθε $j \in D$:
            $H[j] \leftarrow argmin_{i \in S} c_{ij}$
        Επιστροφή του συνόλου S και του πίνακα H

αλλιώς
$S \leftarrow \emptyset$
Για κάθε $i \in S_2$:
$S \leftarrow S \cup \{argmin_{h \in S_1} c_{ih}\}$
$S_1 \leftarrow S_1 - \{argmin_{h \in S_1} c_{ih}\}$
Όσο $|S| \neq k$
Τυχαία Επιλογή $h \in S_1$
$S \leftarrow S \cup \{h\}$
$S_1 \leftarrow S_1 - \{h\}$
Για κάθε $j \in D$:
$H[j] \leftarrow argmin_{i \in S} c_{ij}$
Επιστροφή του συνόλου S και του πίνακα H

Algorithm 2.2: Αλγόριθμος για το Metric k-Median Problem

Θεώρημα 2.4.1. *Ο αλγόριθμος 2.2 είναι* $2(1.61+\varepsilon)$*-προσεγγιστικός και έχει πολυπλοκότητα* $O(n^3 \cdot log\frac{|F|\sum c_{ij}}{\varepsilon c_{min}})$

Η διατύπωση και η ανάλυση του αλγορίθμου περιγράφονται εδώ [12]. Η απόδειξη του παραπάνω θεωρήματος 2.4.1 οφείλεται σε ένα συνδυασμό των αποτελεσμάτων των [8], [12] .

Σημείωσεις: *1)Έχει αποδειχθεί ότι δεν μπορεί να υπάρξει α-προσεγγιστικός αλγόριθμος για το k-median πρόβλημα με σταθερά* $\alpha < 1 + \frac{2}{e} \simeq 1.736$*, εκτός και αν κάθε πρόβλημα στο NP έχει κάποιο αλγόριθμο πολυπλοκότητας* $O(n^{O(loglogn)})$ *[8] και ο τρέχων καλύτερος αλγόριθμος είναι* $(3+\varepsilon)$*-προσεγγιστικός [13]. Άρα ο αλγόριθμος που χρησιμοποιούμε για το πρόβλημα αυτό, βλέπουμε ότι είναι πολύ κοντά σε αυτές τις προσεγγίσεις.*
2) Η επεξήγηση και ανάλυση των αλγορίθμων για το Uncapacitated Facility Location Problem και το Metric k Median βασίστηκαν στο σπουδαίο και ιδιαίτερα κατατοπιστικό βιβλίο για προσεγγιστικούς αλγορίθμους [6].

2.5 Non-metric k-median Problem

Όπως αναφέραμε και στην προηγούμενη ενότητα, η πλαισίωση του προβλήματος k-median σε μετρικό *(metric)* χώρο, δηλαδή σε χώρο στον οποίο τα service costs επαληθεύουν την τριγωνική ανισότητα, δίνει αρκετά καλύτερους προσεγγιστικούς αλγορίθμους, από την εξέταση του προβλήματος σε μη μετρικό *(non-metric)* χώρο. Ωστόσο, δεν μπορούμε να εξασφαλίσουμε ότι σε όλα τα πραγματικά μοντέλα, τα οποία εμπεριέχουν ως στόχο το άνοιγμα κάποιων facilities, τα κόστη εκφράζουν αποστάσεις σε μετρικό χώρο και ικανοποιούν την τριγωνική ανισότητα. Αυτό δε μας εμποδίζει, απαραίτητα, να χρησιμοποιήσουμε το k-median πρόβλημα στις περιπτώσεις αυτές, αλλά μας οδηγεί στο να στραφούμε στη μη μετρική εκδοχή του.

2.5.1 Αλγόριθμος για το Non-metric k-median Problem

Ο αλγόριθμος που παραθέτουμε [14] χρησιμοποιεί με έναν ευρύ τρόπο την τεχνική του *Randomized Rounding*[15]. Πολύ αφαιρετικά αναφέρουμε ότι αυτή η τεχνική συνίσταται στο εξής: Έχοντας το Ακέραιο Πρόγραμμα που αναπαριστά το εκάστοτε πρόβλημα, παίρνουμε το αντίστοιχο LP-χαλαρωμένο και βρίσκουμε τη βέλτιστη λύση του x^*, με έναν από τους γνωστούς αλγορίθμους επίλυσης γραμμικών προβλημάτων, κάποιοι από τους οποίους απαριθμίζονται εδώ [16]. Θέλουμε να στρογγυλοποιήσουμε τις κλασματικές τιμές *(fractional)* της λύσης x^* είτε στο 0 είτε στο 1 με τέτοιο τρόπο ώστε να πάρουμε μια εφικτή λύση για το ακέραιο πρόγραμμα χωρίς να αυξήσουμε κατά πολύ το κόστος. Η κύρια ιδέα είναι ότι ερμηνεύουμε κάθε κλασματική λύση x_i^* ως την πιθανότητα η ακέραια μεταβλητή x_i να πάρει την τιμή 1.

Οπότε θεωρούμε εμείς το ακέραιο (2.11) και το γραμμικό πρόγραμμα (2.13) του k-median, όπως τα διατυπώσαμε στην προηγούμενη ενότητα 2.4, χωρίς όμως η τριγωνική ανισότητα να ισχύει απαραίτητα και θεωρούμε μια βέλτιστη fractional λύση (x^*, y^*) για το γραμμικό πρόβλημα. Έστω ότι η λύση αυτή είναι κόστους, δηλαδή έχει τιμή αντικειμενικής συνάρτησης, d και έστω σταθερά $\varepsilon > 0$.

Τότε ο αλγόριθμος Non_metric_k_median που χρησιμοποιεί την fractional αυτή λύση, φαίνεται στην επόμενη σελίδα (αλγόριθμος 2.3) και παίρνει ως ορίσμα το σύνολο των facilities F, το σύνολο των clients D, τα service costs c_{ij}, το κόστος d της fractional λύσης του LP-relaxed προγράμματος 2.13 και την σταθερά $\varepsilon > 0$.

Στον παρακάτω ψευδοκώδικα 2.3, το σύνολο X αναπαριστά το σύνολο των facilities που ανοίγει και επιστρέφει ο αλγόριθμος, αφού πρώτα συνδέσει τον κάθε client με το κοντινότερό του facility στο σύνολο αυτό μέσω του πίνακα H. Η συνάρτηση $\varphi(j, i)$ εκφράζει το λόγο του service cost ενός client $j \in D$ από ένα facility $i \in F$ προς το συνολικό fractional κόστος d που υπολόγισε το LP-relaxed του προβλήματος επί έναν παράγοντα $1 + \varepsilon$. Η συνάρτηση $f(j)$ εκφράζει το τρέχον facility το οποίο θεωρούμε ότι εξυπηρετεί τον client j και αρχικοποιείται σε none τιμή. Αρχικοποιούμε επίσης την $\varphi(j, none)$ στην τιμή 1, για κάθε $j \in D$. Όσο υπάρχει client ο οποίος δεν έχει συνδεθεί με κάποιο facility (δηλαδή έχει τιμή none ως προς τη συνάρτηση f) ή το συνολικό service cost των clients από τα facilities με τα οποία έχουν συνδεθεί είναι μεγάλο ($cost > d(1 + \varepsilon)$), ο αλγόριθμος βρίσκει για κάθε facility i, τους clients για τους οποίους η τιμή της φ ως προς το current facility που τους αναλογεί είναι μεγαλύτερη από την αντίστοιχη ως προς το facility i. Διαισθητικά βρίσκουμε, δηλαδή, τους clients που το facility i θα μπορούσε να εξυπηρετήσει καλύτερα από το facility με το οποίο είναι μέχρι τώρα συνδεδεμένοι, δηλαδή που με τη σύνδεσή τους στο facility i θα μπορούσαν να πετύχουν μια βελτίωση (μείωση) $\varphi(j, f(j)) - \varphi(j, i)$ σε σχέση με την τωρινή τους σύνδεση, και τους προσθέτουμε σε ένα σύνολο C_i.

Non_metric_k_median $(F, D, c_{ij}, d, \varepsilon)$

$X \leftarrow \emptyset$
$cost \leftarrow 0$
$H[j] \leftarrow none, \ \forall j \in D$
$\varphi(j,i) \leftarrow c_{ij}/d(1+\varepsilon), \quad \forall i \in F, \forall j \in D$
$f(j) \leftarrow none, \quad \forall j \in D$
$\varphi(j, none) \leftarrow 1, \quad \forall j \in D$
Όσο $[\exists j \in D : f(j) = none]$ ή $cost > d(1+\varepsilon)$
    Για κάθε $i \in F$
        $C_i = \{j : \varphi(j, f(j)) > \varphi(j,i)\}$
    Επιλογή $h \in F$ ώστε $h = argmax_{i \in F} \sum_{j \in C_i} \varphi(j, f(j)) - \varphi(j,i)$
    Για κάθε $j \in C_h$
        $f(j) \leftarrow h$
    $cost \leftarrow 0$
    Για κάθε $j \in D$ έτσι ώστε $f(j) \neq none$
        $cost \leftarrow cost + c_{f(j)j}$
Για κάθε $j \in D$
    $X \leftarrow X \cup f(j)$
Για κάθε $j \in D$:
    $H[j] \leftarrow argmin_{i \in X} c_{ij}$
Επιστροφή του συνόλου X και του πίνακα H

Algorithm 2.3: Αλγόριθμος για το Non-metric k-median Problem

Επιλέγουμε το facility $h \in F$ που μεγιστοποιεί, για τους clients που προσθέσαμε στο αντίστοιχο σύνολό του C_h, το άθροισμα των βελτιώσεων που μπορεί να προκαλέσει το h στη σύνδεσή τους, αν συνδεθούν στο facility αυτό. Μόλις επιλέξουμε αυτό το facility h, θέτουμε για κάθε $j \in C_h$ ως τρέχον facility εξυπηρέτησης το facility h $(f(j) \leftarrow h)$. Ανανεώνουμε στη συνέχεια το συνολικό κόστος της λύσης μας (cost) ως το άθροισμα των service costs κάθε client $j \in D$ από το αντίστοιχό του facility $f(j) \in F$. Επαναλαμβάνουμε μέχρι όλοι οι clients να συνδεθούν με κάποιο facility και για το συνολικό κόστος να ισχύει ότι $cost \leq d(1+\varepsilon)$ και προσθέτουμε στο σύνολο X, τα facilities $f(j)$, $\forall j \in D$. Τέλος συνδέουμε καθε client με το κοντινότερό του facility στο X, μέσω του πίνακα H και επιστρέφουμε το σύνολο X με τα facilities που ανοίγουμε.

Παρατηρούμε ότι ναι μεν η παραπάνω ανάλυση δεν ακολουθεί με την αυστηρή έννοια την τεχνική του Randomized Rounding, αλλά η fractional λύση d που υπολογίζουμε, χρησιμοποιοείται στη συνάρτηση φ η οποία καθορίζει, σε κάθε βήμα, ποιο facility θα επιλέξουμε ως αυτό που μεγιστοποιεί τις βελτιώσεις στη σύνδεση των clients, αν αυτοί συνδεθούν με το facility αυτό. Το facility αυτό, έστω i, θα επιλεγεί ως facility εξυπηρέτησης για τους clients αυτούς, τουλάχιστον προσωρινά, οπότε θεωρούμε ότι η μεταβλητή απόφασης y_i είναι προσωρινά 1, με το fractional κόστος d να έχει παίξει ρόλο στην επιλογή αυτή.

Θεώρημα 2.5.1. *Έστω $0 < \varepsilon < 1$ και έστω d το κόστος μιας fractional βέλτιστης λύσης του LP-relaxed προγράμματος για το Non-metric k-median πρόβλημα. Τότε ο αλγόριθμος 2.3 βρίσκει λύση κόστους το πολύ $d(1+\varepsilon)$ και επιστρέφει πλήθος facilities το πολύ $k \cdot ln(n + \frac{n}{\varepsilon})$. Ο αλγόριθμος κάνει $O(k \cdot ln(\frac{n}{\varepsilon}))$ επαναλήψεις.*

Βλέπουμε, δηλαδή, από το παραπάνω θεώρημα ότι ο αλγόριθμος δεν επιστρέφει το πολύ k facilities, αλλά το πολύ $k \cdot ln(n + \frac{n}{\varepsilon})$, έχοντας όμως προσέγγιση κόστους πολύ καλή και ίση με $1 + \varepsilon$. Η απόδειξη του θεωρήματος και η ανάλυση του αλγορίθμου βρίσκονται εδώ [14].

2.6 Generalized Assignment Problem

Ο τελευταίος προσεγγιστικός αλγόριθμος που παρουσιάζουμε και χρησιμοποιούμε σε αυτή τη διπλωματική εργασία, είναι ένας άπληστος αλγόριθμος για το Generalized Assignment Problem (GAP). Η τεχνική που χρησιμοποιείται μεταφράζει έναν οποιονδήποτε αλγόριθμο για το ευρέως γνωστό πρόβλημα συνδυαστικής βελτιστοποίησης *Knapsack Problem*, σε έναν προσεγγιστικό αλγόριθμο για το GAP. Παρακάτω θα δώσουμε τους ορισμούς για τα δύο αυτά προβλήματα, έναν προσεγγιστικό αλγόριθμο για το knapsack καθώς και τον άπληστο αλγόριθμο για το GAP, ο οποίος χρησιμοποιεί αυτόν του knapsack.

2.6.1 Ορισμοί Προβλημάτων

Παρακάτω δίνουμε τους ορισμούς για το NP-complete [17] Knapsack Problem και το NP-hard [18] Generalized Assignment Problem.

Knapsack Problem

Έστω σύνολο $N = \{a_1, ..., a_n\}$ n αντικειμένων, όπου κάθε αντικείμενο a_i έχει μια αξία v_i και ένα μέγεθος s_i. Θεωρούμε επίσης ότι έχουμε ένα σακίδιο χωρητικότητας B. Ο σκοπός του προβλήματος είναι να βρούμε ένα υποσύνολο των αντικειμένων $S \subseteq N$ που θα τοποθετήσουμε στο σακίδιο, το οποίο να μεγιστοποιεί τη συνολική αξία τους $\sum_{i \in S} v_i$ και το συνολικό τους μέγεθος να μην ξεπερνάει τη χωρητικότητα B, δηλαδή $\sum_{i \in S} s_i \leq B$. Αναφέρουμε ότι εξετάζουμε *το 0 1 Knapsack Problem*, δηλαδή θεωρούμε ότι μπορούμε να πάρουμε είτε 0 είτε 1 αντίγραφο κάθε αντικειμένου στο σακίδιο, για χάρη ευκολίας και απλότητας. Παρακάτω βλέπουμε το πρόβλημα εκφρασμένο σε Ακέραιο Πρόγραμμα:

$$\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{n} v_i x_i \\
\text{subject to} \quad & \sum_{i=1}^{n} s_i x_i \leq B \\
& x_i \in \{0,1\},\ i \in \{1,...,n\}.
\end{aligned}$$

Η μεταβλητή απόφασης x_i παίρνει την τιμή 1 αν προσθέσουμε το αντικείμενο a_i στο σακίδιο, διαφορετικά παίρνει την τιμή 0.

Generalized Assignment Problem

Το GAP είναι γενίκευση του Knapsack Problem και διατυπώνεται ως εξής: Έστω $N = \{a_1, .., a_n\}$ ένα σύνολο n αντικειμένων και $M = \{C_1, ..., C_m\}$ ένα σύνολο m σακιδίων. Κάθε σακίδιο $C_j \in M$ έχει χωρητικότητα c_j. Για κάθε αντικείμενο $a_i \in N$ και κάθε σακίδιο $C_j \in M$ έχουμε μια αξία v_{ij} και ένα μέγεθος s_{ij}. Σκοπός του προβλήματος είναι να τοποθετήσουμε τα αντικείμενα στα σακίδια, έτσι ώστε να μεγιστοποιηθεί η συνολική αξία χωρίς όμως τα αντικείμενα που βάζουμε σε κάθε σακίδιο να ξεπερνάνε τη χωρητικότητά του. Παρακάτω βλέπουμε το Ακέραιο Πρόγραμμα του προβλήματος (κι εδώ θεωρούμε ότι μπορούμε να πάρουμε είτε 0 είτε 1 αντίγραφο κάθε αντικειμένου στη λύση μας):

$$\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{n}\sum_{j=1}^{m} v_{ij} x_{ij} \\
\text{subject to} \quad & \sum_{i=1}^{n} s_{ij} x_{ij} \leq c_j, \qquad j \in \{1,...,m\} && (2.16) \\
& \sum_{j=1}^{m} x_{ij} \leq 1, \qquad i \in \{1,...,n\} && (2.17) \\
& x_i \in \{0,1\},\ i \in \{1,...,n\},\ j \in \{1,...,m\}.
\end{aligned}$$

Η μεταβλητή απόφασης x_{ij} παίρνει την τιμή 1 αν στη λύση μας τοποθετούμε το αντικείμενο a_i στο σακίδιο C_j, διαφορετικά παίρνει την τιμή 0. Η συνθήκη 2.16 εξασφαλίζει ότι για κάθε σακίδιο $C_j \in M$, το μέγεθος των αντικειμένων που τοποθετούμε σε αυτό δεν ξεπερνά τη χωρητικότητά του c_j και η συνθήκη 2.17 εξασφαλίζει ότι κάθε αντικείμενο $a_i \in N$ τοποθετείται το πολύ σε ένα σακίδιο.

2.6.2 FPTAS για το Knapsack Problem

Ξεκινάμε την ανάλυση παραθέτοντας τρεις σημαντικούς ορισμούς [19]:

Ορισμός 2.6.1 (Approximation Scheme)**.** *Έστω* Π *ένα NP-hard πρόβλημα με αντικειμενική συνάρτηση* f_Π*. Ο αλγόριθμος A είναι ένα* ***Approximation Scheme*** *για το* Π*, αν για ορίσματα προβλήματος* (I, ε)*, όπου* I *είναι ένα στιγμιότυπο του* Π *και* $\varepsilon > 0$ *μία παράμετρος λάθους, παράγει λύση* s *τέτοια ώστε:*

- $f_\Pi(I, s) \leq (1 + \varepsilon) \cdot OPT$ *αν το* Π *είναι πρόβλημα ελαχιστοποίησης*
- $f_\Pi(I, s) \geq (1 - \varepsilon) \cdot OPT$ *αν το* Π *είναι πρόβλημα μεγιστοποίησης*

Ορισμός 2.6.2 (PTAS)**.** *Το approximation scheme A λέγεται* ***Polynomial Time Approximation Scheme (PTAS)*** *αν για ένα σταθερό* $\varepsilon > 0$*, ο χρόνος εκτέλεσής του φράσσεται από ένα πολυώνυμο του μεγέθους στιγμιοτύπου* I*. Αυτό, όμως, σημαίνει ότι ο χρόνος μπορεί να είναι εκθετικός ως προς το* $1/\varepsilon$*, όπου σε αυτή την περίπτωση το να πλησιάσουμε τη βέλτιστη λύση είναι πολύ δύσκολο.*

Ορισμός 2.6.3 (FPTAS)**.** *Το approximation scheme A λέγεται* ***Fully Polynomial Time Approximation Scheme (FPTAS)*** *αν για ένα σταθερό* $\varepsilon > 0$*, ο χρόνος εκτέλεσής του φράσσεται από ένα πολυώνυμο του μεγέθους στιγμιοτύπου* I *και από το* $1/\varepsilon$*.*

Ένα FPTAS είναι το καλύτερο που μπορούμε να έχουμε για ένα NP-hard πρόβλημα βελτιστοποίησης, αν υποθέσουμε φυσικά ότι $P \neq NP$. Για το Knapsack Πρόβλημα υπάρχει FPTAS.

Περιγράφουμε, εδώ, τον pseudo-polynomial time αλγόριθμο, που στηρίζεται σε δυναμικό προγραμματισμό για να βρει τη βέλτιστη λύση, τον οποίο χρησιμοποιούμε ώστε να δημιουργήσουμε ένα FPTAS για το Knapsack: Έστω V η αξία του αντικειμένου που έχει τη μεγαλύτερη αξία σε σχέση με όλα τα υπόλοιπα, δηλαδή $V = max_{i \in \{1,...,n\}} v_i$. Μπορούμε να φράξουμε τη συνολική αξία που μπορεί να επιτευχθεί, από την ποσότητα nV για τα n αντικείμενα. Για κάθε $i \in \{1, ..., n\}$ και $v \in \{1, ..., nV\}$, έστω $S_{i,v}$ υποσύνολο του συνόλου $\{a_1, ...a_i\}$, το οποίο έχει συνολική αξία v και καταλαμβάνει το λιγότερο δυνατό χώρο του σακιδίου. Έστω $A(i, v)$ το μέγεθος του συνόλου $S_{i,v}$, θεωρώντας άπειρη αξία αν αυτό το σύνολο δεν υπάρχει. Για το $A(i, v)$ θεωρούμε τη βασική περίπτωση $A(1, v)$, όπου $A(1, v_1)$ είναι ίσο με s_1 και όλες οι άλλες τιμές είναι ∞. Χρησιμοποιούμε την ακόλουθη αναδρομή για να υπολογίσουμε όλες τις τιμές $A(i, v)$:

$$A(i+1,v) = \begin{cases} min\{A(i,v),\ s_{i+1} + A(i, v - v_{i+1})\}, & \text{αν } v_{i+1} \leq v \\ A(i,v), & \text{διαφορετικά} \end{cases}$$

Figure 2.6: Αλγόριθμος Δυναμικού Προγραμματισμού για το Knapsack Πρόβλημα

Το βέλτιστο υποσύνολο, αντιστοιχεί στο σύνολο $S_{n,v}$, για το οποίο το v μεγιστοποιείται και $A(n,v) \leq B$. Χρειάζονται n επαναλήψεις ώστε να υπολογιστεί κάθε $A(i,v)$, οπότε ο συνολικός χρόνος εκτέλεσης είναι $O(n^2V)$ και ο παραπάνω αλγόριθμος είναι pseudo-polynomial αφού θεωρούμε instance μεγέθους n.

Από τα παραπάνω, παρατηρούμε ότι αν οι αξίες των αντικειμένων ήταν όλες μικρές, δηλαδή πολυωνυμικά φραγμένες από το n, τότε θα είχαμε πολυωνυμικό αλγόριθμο. Χρησιμοποιούμε αυτή την παρατήρηση για να δημιουργήσουμε FPTAS αλγόριθμο. Συγκεκριμένα, μειώνουμε τις αξίες όλων των αντικειμένων αρκετά, έτσι ώστε όλες οι αξίες να είναι πολυωνυμικά φραγμένες από το n και το $1/\varepsilon$, χρησιμοποιούμε δυναμικό προγραμματισμό στο νέο στιγμιότυπο και παίρνουμε μια λύση αξίας τουλάχιστον $(1-\varepsilon) \cdot OPT$ σε πολυωνυμικό χρόνο ως προς το n και το $1/\varepsilon$.

Ο παρακάτω αλγόριθμος, FPTAS παίρνει ως ορίσματα το σύνολο N των αντικειμένων, τις αξίες τους v_i και τα μεγέθη τους s_i, τη χωρητικότητα B του σακιδίου και την παράμετρο λάθους ε. Επιστρέφει το σύνολο S' που υπολογίζει το σχήμα δυναμικού προγραμματισμού 2.6 για τις νέες αξίες των αντικειμένων.

FPTAS $(N, v_i, s_i, B, \varepsilon)$
1. Δεδομένου $\varepsilon > 0$, έστω $K = \frac{\varepsilon V}{N}$
2. Για κάθε αντκείμενο a_i, ορίζουμε $v'_i = \lfloor \frac{v_i}{K} \rfloor$
3. Με αυτές τις αξίες για τα αντικείμενα, χρησιμοποιώντας τον Αλγόριθμο Δυναμικού Προγραμματισμού 2.6, βρίσκουμε το βέλτιστο υποσύνολο αντικειμένων S'
4. Επιστροφή του συνόλου S'

Algorithm 2.4: FPTAS για το Knapsack Problem

Θεώρημα 2.6.1. *Ο παραπάνω αλγόριθμος 2.4 είναι FPTAS για το Knapsack με χρόνο εκτέλεσης* $O(n^2 \lfloor \frac{n}{\varepsilon} \rfloor)$.

Η απόδειξη του παραπάνω θεωρήματος 2.6.1 καθώς και η διατύπωση των παραπάνω αλγορίθμων βρίσκονται εδώ [19].

2.6.3 Άπληστος Αλγόριθμος για το Generalized Assignment Problem

Έχοντας αναφέρει τα παραπάνω, μπορούμε τώρα να περάσουμε στην ανάλυση του άπληστου αλγορίθμου για το Generalized Assignment Problem. Όπως αναφέραμε ο αλγόριθμος αυτός στηρίζεται στη μετατροπή ενός οποιουδήποτε αλγορίθμου για το Knapsack πρόβλημα σε προσεγγιστικό αλγόριθμο για το GAP. Συγκεκριμένα, αν ο λόγος προσέγγισης του αλγορίθμου που χρησιμοποιούμε για το Knapsack είναι a και ο χρόνος εκτέλεσης $O(f(n))$, τότε ο αλγόριθμος για το GAP θα έχει λόγο προσέγγισης $\frac{a}{a+1} - \varepsilon$ και χρόνο εκτέλεσης $O(m \cdot f(n) + m \cdot n)$ όπου n είναι το πλήθος των αντικειμένων και m είναι το πλήθος των σακιδίων.

Παραθέτουμε παρακάτω τον εν λόγω αλγόριθμο GAP, ο οποίος παίρνει ως ορίσματα το σύνολο N των αντικειμένων, το σύνολο M των σακιδίων, τις αξίες των αντικειμένων v_{ij}, τα μεγέθη τους s_{ij}, τις χωρητικότητες c_j των σακιδίων και την παράμετρο λάθους ε. Ο πίνακας T περιέχει την προσωρινή τοποθέτηση των αντικειμένων στα σακίδια, δηλαδή αν η τιμή $T[i]$ είναι j, τότε το αντικείμενο a_i είναι προσωρινά τοποθετημένο στο σακίδιο C_j. Αρχικοποιούμε για όλα τα αντικείμενα τις τιμές του πινακα T στο -1, δηλαδή αρχικά κανένα αντικείμενο δεν είναι τοποθετημένο σε κανένα σακίδιο. Κατόπιν για κάθε σακίδιο $C_j \in M$, δημιουργούμε τις αξίες των αντικειμένων με τις οποίες θα καλέσουμε τον FPTAS αλγόριθμο του Knapsack για αυτό το σακίδιο. Συγκεκριμένα οι νέες αυτές αξίες P_{ij} διαμορφώνονται ως εξής: Για κάθε αντικείμενο $a_i \in N$, αν το αντικείμενο δεν έχει τοποθετηθεί ακόμα σε κανένα σακίδιο, τότε η αξία του P_{ij} ως προς το σακίδιο C_j είναι όσο και η αρχική v_{ij}. Αν, όμως, έχει ήδη τοποθετηθεί προσωρινά σε ένα σακίδιο, έστω C_k, τότε η αξία P_{ij} είναι $v_{ij} - v_{ik}$. Έχοντας ορίσει τις νέες αξίες, καλούμε τον FPTAS αλγόριθμο 2.4 για το σακίδιο C_j, με ορίσματα το N ως σύνολο αντικειμένων, τις νέες αξίες P_{ij} ως αξίες των αντικειμένων ως προς το σακίδιο C_j, τα s_{ij} ως μεγέθη των αντικειμένων ως προς το σακίδιο C_j , την χωρητικότητα c_j του σακιδίου C_j και την παράμετρο λάθους ε. Ο αλγόριθμος FPTAS επιστρέφει το σύνολο S_j των αντικειμένων που υπολόγισε ότι θα τοποθετηθούν στο σακίδιο C_j και τα αντικείμενα του συνόλου αυτού θεωρούμε ότι τοποθετούνται προσωρινά στο σακίδιο C_j μέσω του βήματος $T[i] \leftarrow j, \forall a_i \in S_j$. Επαναλαμβάνουμε τη διαδικασία αυτή για όλα τα σακίδια $C_j \in M$ και επιστρέφουμε τον πίνακα T ο οποίος στο τέλος περιέχει τις τελικές τοποθετήσεις των αντικειμένων στα σακίδια, που έκανε ο αλγόριθμος. Για κάθε $a_i \in N$, αν $T[i] = -1$, τότε το αντικείμενο αυτό δεν τοποθετήθηκε σε κανένα σακίδιο, διαφορετικά θεωρούμε ότι τοποθετήθηκε στο σακίδιο $C_{T[i]}$.

GAP $(N, M, v_{ij}, s_{ij}, c_j, \varepsilon)$
$T[i] \leftarrow -1, \quad \forall i \in \{1, ..., n\}$
Για κάθε $j \in \{1, ..., m\}$
Για κάθε $i \in \{1, ..., n\}$
$P_{ij} \leftarrow \begin{cases} v_{ij}, & \text{αν T[i] = -1} \\ v_{ij} - v_{ik}, & \text{αν T[i] = k} \end{cases}$
$S_j = FPTAS(N, P_{ij}, s_{ij}, c_j, \varepsilon)$
$T[i] \leftarrow j, \quad \forall a_i \in S_j$
Επιστροφή του πίνακα T

Algorithm 2.5: Άπληστος Αλγόριθμος για το Generalized Assignment Problem

Θεώρημα 2.6.2. *Ο αλγόριθμος 2.5 είναι* $(\frac{1-\varepsilon}{2-\varepsilon} - \varepsilon)$*-προσεγγιστικός και έχει πολυπλοκότητα* $O(m \cdot n^2 \lfloor \frac{n}{\varepsilon} \rfloor)$*, όπου* m *το πλήθος των σακιδίων και* n *το πλήθος των αντικειμένων.*

Η διατύπωση και η ανάλυση του αλγορίθμου παρουσιάζονται εδώ [20]. Η απόδειξη του παραπάνω θεωρήματος 2.6.2 οφείλεται σε ένα συνδυασμό των όσων αναφέρονται στο [20] και στο γεγονός ότι ο αλγόριθμος που χρησιμοποιούμε για το Knapsack είναι ο FPTAS (2.4).

Σημείωση: Ο τρέχων καλύτερος αλγόριθμος για το GAP έχει λόγο προσέγγισης $1 - \frac{1}{e}$ και παρουσιάζεται στο [21]. Άρα ο αλγόριθμος που χρησιμοποιούμε βλέπουμε ότι είναι αρκετά κοντά σε αυτή την προσέγγιση, ενώ παράλληλα έχει καλύτερο χρόνο εκτέλεσης.

Κεφάλαιο 3

Προσωρινή αποθήκευση περιεχομένου στα άκρα του δικτύου (Mobile Edge Caching)

3.1 Mobile Edge Computing και Mobile Edge Caching

Η εξέλιξη των δικτύων κινητής τηλεφωνίας γνώρισε τέσσερις γενιές τις τελευταίες δεκαετίες με την πρόοδο στην τεχνολογία των τηλεπικοινωνιών και της πληροφορικής. Με την ευρεία χρήση διαφόρων εφαρμογών στα κινητά τηλέφωνα, ο όγκος της κίνησης στα ασύρματα δίκτυα αυξάνεται με εκθετικό ρυθμό, ενώ οι απαιτήσεις των χρηστών για υψηλή ρυθμαπόδοση και εξαιρετικά χαμηλή καθυστέρηση [22], γίνονται όλο και πιο αυστηρές. Ενώ διανύουμε την πέμπτη γενιά δικτύων κινητής τηλεφωνίας (5G), η εμφάνιση νέων ειδών έξυπνων συσκευών και νέων εφαρμογών, όπως η εικονική πραγματικότητα (virtual reality) και το διαδίκτυο των πραγμάτων (Internet of Things - IoT), καθώς και η αυξανόμενη ζήτηση για μαζικές υπηρεσίες πολυμέσων (video, μουσική) μέσω του κινητού κυψελοειδούς δικτύου, θέτουν μεγάλες προκλήσεις όσον αφορά τη χωρητικότητα και την επιβάρυνση του οπισθοζευκτικού δικτύου (backhaul) [23] με την παραδοσιακή αρχιτεκτονική κεντρικού δικτύου να κρινεται ανεπαρκής στο να ανταποκριθεί γρήγορα και ικανοποιητικά στα ζητήματα αυτά.

Ένας τρόπος αντιμετώπισης των παραπάνω προβλημάτων δίνεται από την εκφόρτωση κάποιων υπολογιστικών διαδικασιών και δεδομένων στο cloud (**Mobile Cloud Computing - MCC**). Ωστόσο, προκύπτουν αρκετές προκλήσεις, όπως η μεγάλη καθυστέρηση και η υψηλή κατανάλωση εύρους ζώνης backhaul, καθιστώντας τον ακατάλληλο για real-time εφαρμογές. Προκειμένου να λυθεί το πρόβλημα αυτό, η αρχιτεκτονική **Mobile Edge Computing (MEC)** προτείνεται ώστε οι υπολο-

γιστικές διαδικασίες και τα περιεχόμενα να μετακινηθούν προς τα άκρα του δικτύου και κατά συνέπεια κοντινότερα στους χρήστες.

Στο Mobile Edge Computing, servers τοποθετούνται κοντινότερα στους χρήστες και η εκφόρτωση υπολογιστικής ισχύος γίνεται σε αυτούς τους servers στα άκρα του δικτύου και όχι σε απομακρυσμένο cloud. Με αυτό τον τρόπο, ο χρόνος απόκρισης της υπηρεσίας μπορεί να μειωθεί σημαντικά και ως εκ τούτου να βελτιωθεί η εμπειρία του χρήστη, καθώς και να αποσυμφορηθεί το οπισθοζευκτικό δίκτυο. Επιπλέον, οι servers αυτοί μπορούν να χρησιμοποιηθούν για την αποθήκευση δημοφιλούς περιεχομένου που ζητάται από τους χρήστες του δικτύου. Η προσωρινή αποθήκευση δεδομένων στους mobile edge servers αναφέρεται ως **Mobile Edge Caching**.

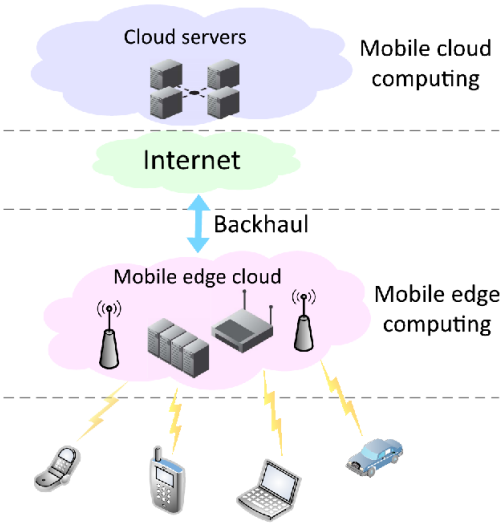


Figure 3.1: Αρχιτεκτονικές MCC και MEC [22]

Το Mobile Edge Caching, χρησιμοποιεί τους αποθηκευτικούς χώρους (caches) που προσφέρονται από τους mobile edge servers, που μπορεί να είναι ανεξάρτητοι servers προσαρτημένοι σε έναν mobile edge κόμβο ή η μνήμη του edge κόμβου. Χωρίς το mobile edge caching, η ζήτηση περιεχομένου από τους χρήστες ικανοποιείται από απομακρυσμένους servers περιεχομένου στο διαδίκτυο (Web Servers). Όταν οι χρήστες προσπελάζουν το ίδιο δημοφιλές περιεχόμενο από τους απομακρυσμένους αυτούς servers, αυτοί πρέπει να στείλουν τα ίδια αρχεία επανειλημμένα, γεγονός που μπορεί να οδηγήσει σε συμφόρηση του δικτύου, καθυστέρηση και σπατάλη δικτυακών πόρων. Ωστόσο, με την προσωρινή αποθήκευση δημοφιλούς περιεχομένου πιο κοντά στους χρήστες (στα άκρα του δικτύου), η καθυστέρηση μπορεί να μειωθεί σημαντικά και η πολλαπλή μετάδοση των ίδιων αρχείων από τους servers περιεχομένου να αποφευχθεί. Επιπλέον, καθώς το κόστος των αποθηκευτικών χώρων μειώνεται, η αξιοποίηση των caches στα άκρα του δικτύου γίνεται λιγότερο δαπανηρή. Στο

Mobile Edge Caching, τα αιτήματα περιεχομένου, τα οποία γίνονται μέσω των εξοπλισμών των χρηστών (User Equipment - UE), εξυπηρετούνται από τον κοντινότερο κόμβο, ο οποίος περιέχει το ζητούμενο περιεχόμενο. Επιγραμματικά, τα πλεονεκτήματα του Mobile Edge Caching είναι μείωση καθυστέρησης, μείωση εύρους ζώνης στις backhaul συνδέσεις, υψηλή απόδοση ενέργειας, προσφορά υπηρεσιών εγγύτητας και αξιοποίηση πληροφοριών περιβάλλοντος.

3.2 Τοποθεσίες Προσωρινής Αποθήκευσης

Τα προβλήματα σχετικά με το "πού, πώς και τί" να αποθηκεύσουμε αποτελούν τα βασικά ερευνητικά ζητήματα γύρω από το Mobile Edge Caching. Το "πού" να αποθηκεύσουμε αναφέρεται στην επιλογή των θέσεων προσωρινής αποθήκευσης. Οι βασικές τοποθεσίες αποθήκευσης, μεταξύ άλλων, είναι οι παρακάτω [22] [23]:

- **Αποθήκευση περιεχομένου σε Base Stations**: Το περιεχόμενο ενδιαφέροντος μπορεί να αποθηκευτεί προσωρινά σε σταθμούς βάσης (Base Stations), οι οποίοι βρίσκονται σε διάφορα σημεία και γεωγραφικά κοντά στους χρήστες, κάνοντας τα παραδοσιακά κέντρα δεδομένων και τις διαδικτυακές εφαρμογές άμεσα προσβάσιμα στους χρήστες μέσω ενός κατανεμημένου δικτύου υπολογιστών. Ωστόσο, αυτή η τεχνική έρχεται αντιμέτωπη με διάφορα προβλήματα, όπως περιορισμένη κάλυψη, αβεβαιότητα των ασύρματων συνδέσεων και παρεμβολές μεταξύ κυψελών.

- **Αποθήκευση περιεχομένου στις Συσκευές Χρηστών**: Ένας διαφορετικός τρόπος αποθήκευσης είναι μέσω των κινητών συσκευών των χρηστών του δικτύου. Τα σύγχρονα smartphones εξελίσσονται διαρκώς ως προς τις υπολογιστικές και αποθηκευτικές τους δυνατότητες. Επομένως, οι συσκευές των χρηστών του δικτύου μπορούν να χρησιμεύσουν ως caches για την προσωρινή αποθήκευση περιεχομένου τοπικά και την ανταλλαγή του μέσω Device-to-Device επικοινωνίας (**D2D communication**), χρησιμοποιώντας τεχνολογία Bluetooth, WiFi direct, κ.ο.κ., μειώνοντας την επιβάρυνση στις συνδέσεις του κεντρικού δικτύου και αποτρέποντας την κατασπατάληση πόρων του δικτύου. Στην περίπτωση αυτή, οι σταθμοί βάσης παρακολουθούν το caching status (διαθεσιμότητα του περιεχομένου και διαθέσιμος αποθηκευτικός χώρος) κάθε συσκευής και κατευθύνουν αιτήματα από μία συσκευή στις κατάλληλες κοντινές συσκευές που διαθέτουν το περιεχόμενο αυτό. Αν καμία από τις κοντινές συσκευές δε διαθέτει το περιεχόμενο, οι σταθμοί βάσης παρέχουν το ζητούμενο αντικείμενο μέσω επικοινωνίας με το κεντρικό δίκτυο. Η αποθήκευση σε UE χρησιμεύει ως τεχνική βελτίωσης της ποιότητας εμπειρίας του χρήστη (Quality of Experience). Ένα σημαντικό πλεονέκτημα που εμφανίζει η τεχνική αυτή είναι ότι το περιεχόμενο που αποθηκεύεται σε καθέναν από αυτούς τους χρήστες μπορεί να είναι περισσότερο συναφές με τις προτιμήσεις των χρηστών αυτών καθώς και των χρηστών που είναι πιθανότερο να εξυπηρετήσουν. Οι χρήστες που

προσφέρουν μέρος από την μνήμη της συσκευής τους για την προσωρινή αποθήκευση περιεχομένου αναφέρονται ως Clusterheads. Μερικές από τις προκλήσεις που αντιμετωπίζει η D2D επικοινωνία είναι η σχετικά μικρή χωρητικότητα που παρουσιάζουν οι κινητές συσκευές έναντι του κεντρικού δικτύου και των σταθμών βάσης, η περιορισμένη διάρκεια ζωής της μπαταρίας των συσκευών και η παρεμβολή στη μετάδοση από άλλες συσκευές. Στην παρούσα εργασία, δεν εξετάζουμε αυτούς τους περιορισμούς αλλά εστιάζουμε στα πλεονεκτήματα της τεχνικής αυτής όσον αφορά την αποθήκευση και το διαμοιρασμό περιεχομένου όταν λαμβάνουμε υπόψιν την κινητικότητα των χρηστών του δικτύου.

3.3 D2D επικοινωνία με επίγνωση της κινητικότητας των χρηστών

Γενικά, εξαιτίας των χαρακτηριστικών της D2D επικοινωνίας, οι κοινωνικές σχέσεις έχουν πολύ μεγάλη επίδραση στο πρόβλημα της D2D αποθήκευσης και αναγνωρίζονται διάφορες ιδιότητες σχετικά με τις κοινωνικές συμπεριφορές των χρηστών [22]. Δύο βασικές εξ' αυτών είναι ότι η συνάφεια στα ενδιαφέροντα των χρηστών δημιουργεί ομάδες χρηστών που μεταδίδουν και μοιράζονται περιεχόμενο με μεγάλη συχνότητα και ότι οι χρήστες, οι οποίοι είναι γεωγραφικά κοντά, έχουν υψηλότερες τάσεις ανταλλαγής πληροφοριών. Τις δύο αυτές ιδιότητες ικανοποιεί και το μοντέλο που παρουσιάζουμε στο Κεφάλαιο 5. Στο μοντέλο μας, οι χρήστες που έχουν παρόμοια ενδιαφέροντα, μέσω της κίνησής τους καταλήγουν γεωγραφικά κοντά με μεγαλύτερη πιθανότητα (βρίσκονται με μεγάλη πιθανότητα συχνά στο ίδιο μέρος), δημιουργώντας ομάδες που ανταλάσσουν περιεχόμενο με μεγαλύτερη πιθανότητα. Έτσι η ανάλυση αυτή, λαμβάνει υπόψιν και την κίνητικότητα (αν και θεωρητική) η οποία συμβαίνει ρεαλιστικά, αφού οι χρήστες δεν είναι στατικοί και δε θα έπρεπε να αγνοείται κατά τη μελέτη προβλημάτων προσωρινής αποθήκευσης στα άκρα του δικτύου. Η κινητικότητα των χρηστών στο δίκτυο επιβάλλει σοβαρούς περιορισμούς όσον αφορά τον αποτελεσματικό διαμοιρασμό δεδομένων και η αποδοτική εκφόρτωση δεδομένων στις συσκευές των χρηστών από το κεντρικό δίκτυο στηρίζεται στην προσεκτική εξέταση των ιδιοτήτων της κινητικότητας των χρηστών. Εστιάζοντας σε διάφορες πτυχές της D2D επικοινωνίας, πολλές ερευνητικές εργασίες έχουν μελετήσει διάφορους παράγοντες και μοντέλα κινητικότητας για την αξιολόγηση της απόδοσης των επικοινωνιών D2D σε σενάρια πραγματικής ζωής, όπως αναφέρεται εδώ [24]. Ωστόσο, παρά την πρόοδο και τις συνεισφορές που έχουν γίνει, ο σημαντικός τομέας της έρευνας για την κινητικότητα των χρηστών του δικτύου βρίσκεται ακόμη σε εξέλιξη. Γνωστά μοντέλα τα οποία προτείνονται για την προσομοίωση και την πρόβλεψη της κινητικότητας των χρηστών είναι το Heterogeneous Human Walk (HHW) [25], στο οποίο λαμβάνονται υπόψη για την πρόβλεψη της κινητικότητας των χρηστών, τα κοινωνικά χαρακτηριστικά τους και το Self-similar Least Action Walk (SLAW) [26], το οποίο λαμβάνει υπόψη στατιστικά μοτίβα που διέπουν την κίνηση των χρηστών. Η έννοια των τυχαίων περιπάτων εξετάζεται κυρίως ως μοντέλο εκτίμησης της εντελώς απρόβλεπτης κίνησης

των χρηστών, αλλά κάτι τέτοιο είναι περιορισμένης χρηστικότητας, καθώς δε λαμβάνει υπόψη τα ενδιαφέροντα των χρηστών και την απόσταση των μερών ενδιαφέροντος που λειτουργούν ως κίνητρο ώστε οι χρήστες να μετακινηθούν από το ένα μέρος στο άλλο.

3.4 Τρόπος οργάνωσης αποθηκευτικών χώρων για D2D επικοινωνία

Προκειμένου να αποφασίσουμε τί να αποθηκεύσουμε και πώς θα οργανώσουμε τις μνήμες προσωρινής αποθήκευσης των χρηστών, πρέπει να λάβουμε υπόψιν τη δημοφιλία του περιεχομένου. Κατά κύριο λόγο, οι χρήστες ελκύονται περισσότερο και ζητούν με μεγάλη πιθανότητα δημοφιλή περιεχόμενα, όπως τα "viral" βίντεο. Διακρίνουμε τους εξής τρόπους οργάνωσης της μνήμης [23]:

- **Δυναμικός:** Το περιεχόμενο που τοποθετείται στις μνήμες ανανεώνεται συχνά, ώστε νέα δημοφιλή αντικείμενα να είναι άμεσα διαθέσιμα στους χρήστες κάθε χρονική στιγμή και η προσπέλασή τους, που παραδοσιακά θα γινόταν από το κεντρικό δίκτυο, να γίνεται άμεσα, χωρίς επιβάρυνση στις backhaul συνδέσεις του δικτύου.

- **Στατικός:** Το περιεχόμενο που τοποθετείται στις μνήμες δεν αλλάζει συχνά και η μέθοδος αποθήκευσης επικεντρώνεται στην μακροπρόθεσμη ζήτηση δημοφιλούς περιεχομένου. Έτσι καινούριο "viral" περιεχόμενο δεν είναι διαθέσιμο άμεσα στους χρήστες με αποτέλεσμα να μπορούν να το προσπελάσουν με τις καθυστερήσεις που επιβάλλει το core network. Εμείς, στην παρούσα εργασία, εξετάζουμε αυτό τον τρόπο οργάνωσης, τοποθετώντας στατικά περιεχόμενο στους Clusterheads και προτείνοντας μία τομή αυτού και του διαθέσιμου περιεχομένου στους χρήστες. Μπορούμε, ωστόσο, να θεωρήσουμε ότι εξετάζουμε κβαντισμένες χρονικές περιόδους, όπου η δημοφιλία των περιεχομένων θεωρείται στατική και μπορούμε να εφαρμόσουμε εκ νέου το μοντέλο, δηλαδή να αποθηκεύσουμε νέο περιεχόμενο στις συσκευές των Clusterheads, όταν η δημοφιλία του τρέχοντος περιεχομένου εξασθενίσει.

Κεφάλαιο 4

Συστήματα Συστάσεων

Τα Συστήματα Συστάσεων (Recommendation Systems) έχουν γίνει αναπόσπαστο κομμάτι των εφαρμογών παροχής περιεχομένου. Σκοπός τους είναι να παρέχουν εξατομικευμένες προτάσεις για ταινίες, video clips, μουσική ή άλλα αντικείμενα τα οποία να ταιριάζουν όσο το δυνατόν περισσότερο με τα ενδιαφέροντα και τις προτιμήσεις του κάθε χρήστη, ανακουφίζοντάς τον από την υπερφόρτωσή του με μη συναφές υλικό [27]. Αυτό συμβάλει στην ικανοποίηση του χρήστη και κατ'επέκταση την δέσμευσή του στην εφαρμογή καθώς και στην διαμόρφωση των προτιμήσεών του ως προς το ποιο περιεχόμενο θα επιλέξει να προσπελάσει [28]. Για παράδειγμα, το σύστημα συστάσεων που χρησιμοποιεί η εφαρμογή Netflix είναι υπεύθυνο για το 80% [29] των ωρών ροής του, ενώ το αντίστοιχο σύστημα του Youtube για το 30% [30].

4.1 Μέθοδοι Συστάσεων

Η χρήση αποδοτικών και εύστοχων τεχνικών συστάσεων περιεχομένου είναι πολύ σημαντική για ένα σύστημα το οποίο επιθυμεί να παρέχει καλές και χρήσιμες συστάσεις σε κάθε χρήστη ατομικά. Επομένως, η σημασία της κατανόησης των χαρακτηριστικών και των δυνατοτήτων των διαφορετικών τεχνικών σύστασης είναι απαραίτητη. Διακρίνουμε τις παρακάτω κατηγορίες τεχνικών σύστασης [31]:

- **Content Based:** Η τεχνική αυτή δίνει έμφαση στην ανάλυση των χαρακτηριστικών των αντικειμένων προκειμένου να παραχθούν συστάσεις. Αυτές γίνονται σύμφωνα με τα προφίλ των χρηστών, μέσω εξαγωγής χαρακτηριστικών από το περιεχόμενο των αντικειμένων το οποίο έχουν αξιολογήσει οι χρήστες στο παρελθόν. Αντικείμενα τα οποία είναι συναφή με αυτά που οι χρήστες αξιολόγησαν θετικά, προτείνονται στους χρήστες. Ένα σημαντικό μειονέκτημα της τεχνικής αυτής είναι η ενδεχόμενη έλλειψη διαθεσιμότητας των χαρακτηριστικών των αντικειμένων. Η τεχνική αυτή είναι ιδιαίτερα δημοφιλής όταν προτείνονται αρχεία όπως ιστοσελίδες, δημοσιεύσεις ή ειδήσεις.

- **Collaborative Filtering:** Στην περίπτωση του Collaborative Filtering, δημιουργείται μία βάση δεδομένων (user-item matrix) που περιλαμβάνει πληροφορίες για τις προτιμήσεις των χρηστών ως προς τα αντικείμενα. Σύμφωνα με αυτή, υπολογίζεται η ομοιότητα μεταξύ των χρηστών και δημιουργούνται ομάδες χρηστών με συναφή ενδιαφέροντα και προτιμήσεις. Ένας χρήστης λαμβάνει συστάσεις για αυτά τα αντικείμενα τα οποία δεν έχει αξιολογήσει στο παρελθόν, αλλά έχουν ήδη αξιολογηθεί θετικά από χρήστες οι οποίοι έχουν βρεθεί όμοιοι με αυτόν από το σύστημα. Ένα σημαντικό μειονέκτημα της τεχνικής αυτής είναι η ενδεχόμενη αδυναμία σύνδεσης ενός χρήστη με περισσότερες από μία ομάδες. Η τεχνική αυτή χρησιμοποιείται κατά κύριο λόγο για προτάσεις ταινιών και μουσικής.

- **Hybrid Filtering:** Η τεχνική αυτή αναφέρεται στο συνδυασμό διαφορετικών τεχνικών προτάσεων, προκειμένου να βελτιστοποιηθούν οι συστάσεις και να αντιμετωπιστούν περιορισμοί και προβλήματα χρήσης μεμονωμένων τεχνικών. Ο συνδυασμός διαφορετικών τεχνικών συντελεί στην αντιμετώπιση των αδυναμιών της μίας τεχνικής από τα πλεονεκτήματα της άλλης.

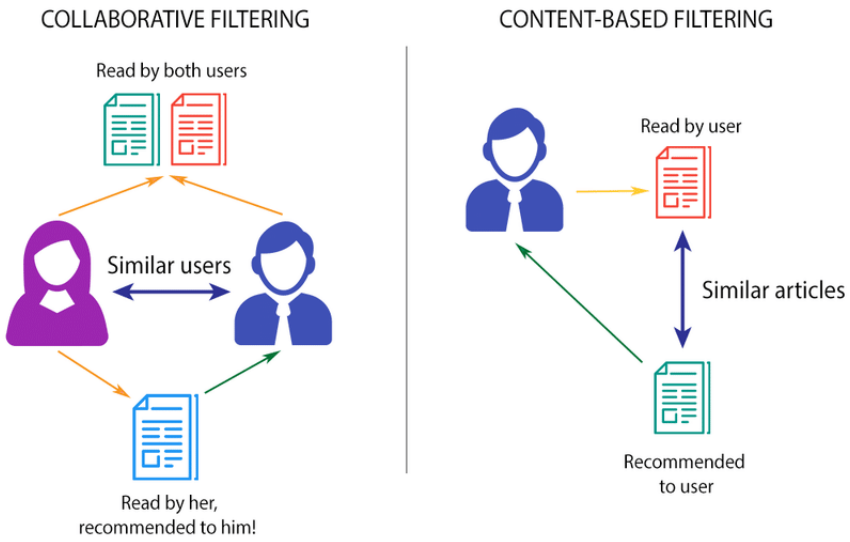


Figure 4.1: Οι τεχνικές Collaborative Filtering και Content Based [32]

4.2 Κοινό Πρόβλημα Προσωρινής Αποθήκευσης και Συστάσεων

Τα Συστήματα Συστάσεων ελέγχονται από τους παρόχους περιεχομένου μέσω των εφαρμογών με τις οποίες αλληλεπιδρούν οι χρήστες, ενώ οι υποδομές αποθήκευσης ανήκουν και ελέγχονται κατά κύριο λόγο από τους διαχειριστές δικτύων (wireless network operators) [28].

Οι πάροχοι περιεχομένου δεν προβαίνουν πάντα σε προτάσεις οι οποίες είναι αποκλειστικά προς όφελος των χρηστών, αλλά και σε προτάσεις περιεχομένου το οποίο έχει γίνει ήδη cached, ώστε τα αιτήματα των χρηστών να ικανοποιούνται σε μικρό χρόνο. Δηλαδή, οι προτάσεις έχουν ασφαλώς κάποια συνάφεια με τα ενδιαφέροντα των χρηστών (όχι τη μέγιστη δυνατή) και αφορούν αντικείμενα τα οποία οι χρήστες, μέσω της αρχιτεκτονικής δικτύου που χρησιμοποιείται (Mobile Edge Caching στην δική μας περίπτωση), να μπορούν να προσπελασουν εύκολα και γρήγορα. Άλλωστε μέσω των προτάσεων που γίνονται προς τους χρήστες κατευθύνονται οι επιλογές τους ως ένα βαθμό, οπότε η ζήτηση συγκεντρώνεται ολοένα και περισσότερο στα cached αντικείμενα, τα οποία ο πάροχος έχει συμφέρον να προτείνει. Οι διαφημίσεις που υπάρχουν στις εφαρμογές αυτές, είναι ένα ακόμη στοιχείο που δηλώνει το οικονομικό κίνητρο πίσω από τις προτάσεις που γίνονται από τον πάροχο. Ωστόσο στο δικό μας σύστημα, θεωρούμε ότι οι προτάσεις προς τους χρήστες είναι επικεντρωμένες στα ενδιαφέροντα τους και στην ικανοποίηση όσο το δυνατόν περισσότερων χρηστών γρήγορα και αποδοτικά.

Από την άλλη οι διαχειριστές, που είναι υπεύθυνοι για την κίνηση στα δίκτυα, την αποσυμφόρησή τους και για το περιεχόμενο που θα γίνει cached στις εκάστοτε δομές αποθήκευσης, επιζητούν την αποδοτικότερη και βέλτιστη αξιοποιήση των υποδομών του δικτύου προς όφελός τους, ευνοϊκότερη πρόσβαση στις οποίες επιζητούν, παράλληλα, και οι πάροχοι για να ικανοποιήσουν τα δικά τους συμφέροντα.

Προκειμένου να βελτιωθεί η ποιότητα εμπειρίας (Quality Of Experience) των χρηστών, είναι απαραίτητη η συνεργασία μεταξύ των παρόχων υπηρεσιών και των διαχειριστών δικτύου που πολλές φορές είναι δύσκολη, αφού οι σκοποί και οι ρόλοι των οντοτήτων αυτών είναι διαφορετικοί ή και αντικρουόμενοι και για αυτό έχει παρατηρηθεί η πρόσφατη τάση ενοποίησής τους, μέσω της ταυτόχρονης δράσης των παρόχων περιεχομένου ως διαχειριστές CDN (Content Delivey Network) [33]. Οι αποφάσεις του δικτύου σχετικά με την αποθήκευση περιεχομένου γίνονται πολλές φορές χωρίς να ληφθούν υπόψιν οι προτάσεις αντικειμένων στις οποίες θα προβεί ο πάροχος και αντίστοιχα οι αποφάσεις σχετικά με το ποια αντικείμενα θα προταθούν στους χρήστες γίνονται πολλές φορές, χωρίς να εκτιμηθεί ο χρόνος μετάδοσής τους από το δίκτυο. Το πρόβλημα που εγείρεται, λόγω των παραπάνω, αντιστοιχεί στο Κοινό Πρόβλημα Προσωρινής Αποθήκευσης και Συστάσεων (Joint Caching and Recommendations Problem).

Διάφορες ερευνητικές εργασίες έχουν επικεντρωθεί στην επίλυσή του προβλήματος αυτού. Στην [28], προσεγγίζεται η επίλυση του προβλήματος μέσω προτάσεων αντικειμένων στους χρήστες που δεν είναι πρώτα στις προτιμήσεις τους (είναι όμως εν-

τός ενός παραθύρου επιτρεπτής απόκλισης από αυτές) τα οποία είναι υψηλής ζήτησης από πολλούς άλλους χρήστες. Στην [33] προτείνεται προσεγγιστικός αλγόριθμος για το πρόβλημα αυτό, το οποίο προσεγγίζεται ως πρόβλημα μεγιστοποίησης του Quality Of Experience (QOE) των χρηστών, το οποίο ορίζεται ως γραμμικός συνδυασμός της ποιότητας συστάσεων (Quality Of Recommendations - QOR) (πόσο αρέσουν στους χρήστες τα προτεινόμενα αντικείμενα) καθώς και της ποιότητας εξυπηρέτησης (Quality Of Service - QOS) (με ποια συχνότητα εξυπηρέτησης μπορούν να τα προσπελάσουν). Οι μελέτες αυτές θεωρούν ότι υπάρχουν κάποιες σταθερές caches από τις οποίες μπορούν οι χρήστες να προσπελάζουν τα προτεινόμενα και τα ζητούμενα αντικείμενα και δε στηρίζονται στη D2D μέθοδο επικοινωνίας. Στον αλγόριθμο που προτείνεται στο [34], ωστόσο, χρησιμοποιείται η τεχνική D2D communication, επιλέγοντας τους υπεύθυνους χρήστες για διαμοιρασμό περιεχομένου (Clusterheads), μέσω εντοπισμού και διαμόρφωσης κοινοτήτων. Στις παραπάνω εργασίες, οι προτάσεις που γίνονται στους χρήστες επηρεάζουν τα αιτήματα των χρηστών με διαφορετικούς τρόπους σε κάθε περίπτωση. Στη δική μας μελέτη περιοριζόμαστε μόνο στις προτάσεις αντικειμένων προς τους χρήστες και στην προσπέλαση αυτών των αντικειμένων και δεν εξετάζουμε αιτήματα από τους χρήστες. Ωστόσο, στην παρούσα εργασία εξετάσουμε και την κινητικότητα των χρηστών, η έννοια της οποίας δεν εμπεριέχεται στα παραπάνω, και διαμορφώνουμε μοντέλο αποθήκευσης και πρότασης περιεχομένου, μέσω αποδοτικών προσεγγιστικών αλγορίθμων παρέχοντας έτσι εγγύηση προσέγγισης και χρόνου εκτέλεσης για το σύστημά μας, το οποίο περιγράφουμε και αναλύουμε στο ακόλουθο κεφάλαιο.

Κεφάλαιο 5

Μοντέλο Συστήματος Προσωρινής Αποθήκευσης και Σύστασης Περιεχομένου

5.1 Περιγραφή Μοντέλου

Στην παρούσα διπλωματική εργασία, μελετάμε το πρόβλημα της προσωρινής αποθήκευσης και σύστασης περιεχομένου στα άκρα του δικτύου, λαμβάνοντας υπόψιν την κινητικότητα των χρηστών. Συγκεκριμένα, ασχολούμαστε με μία θεωρητική μοντελοποίηση ενός συστήματος στο οποίο έχουμε χρήστες δικτύου *(users)* οι οποίοι κινούνται σε πεπερασμένο χώρο και στους οποίους θέλουμε να προτείνουμε αντικείμενα *(items)* τα οποία να βρίσκονται υψηλά ως προς τις προτιμήσεις τους και τα οποία να μπορούν να προσπελάσουν σε μικρό χρόνο. Τα αντικείμενα αυτά θεωρούμε ότι μπορούν να αποθηκευτούν στις κινητές συσκευές κάποιων χρηστών και διαμοιράζονται στους υπόλοιπους χρήστες μέσω D2D επικοινωνίας.

Θεωρούμε ότι οι χρήστες του συστήματος εκτελούν τυχαίους περιπάτους σε γράφο, οι κορυφές του οποίου αποτελούν τα μέρη *(places)* τα οποία επισκέπτονται οι χρήστες. Η πιθανότητα με την οποία κινείται κάθε χρήστης από το ένα μέρος στο άλλο κάθε χρονική στιγμή, θεωρούμε ότι εξαρτάται από το πόσο του αρέσει το μέρος αυτό και από την απόσταση που αυτό έχει από το μέρος στο οποίο βρισκόταν ο χρήστης την προηγούμενη χρονική στιγμή. Κατόπιν, υπολογίζουμε τον εκτιμώμενο χρόνο που μεσολαβεί μεταξύ των διαδοχικών συναντήσεων κάθε χρήστη με κάθε άλλο *(expected inter-contact time)* διαμορφώνοντας το κριτήριο επιλογής των Clusterheads, δηλαδή των χρηστών στους οποίους θα αποθηκεύσουμε τα αντικείμενα και οι οποίοι είναι υπεύθυνοι για να τα διαμοιράσουν στους υπόλοιπους. Επιλέγουμε ως Clusterheads τους χρήστες αυτούς, για τους οποίους το άθροισμα των εκτιμώμενων χρόνων συνάντησής τους με τους υπόλοιπους χρήστες ελαχιστοποιείται. Στο άθροισμα αυτό προσμετράμε για κάθε χρήστη, το μικρότερο expected inter-contact time του χρήστη ως προς

κάποιον από τους Clusterheads που επιλέγουμε και θεωρούμε ότι σε αυτόν τον Clusterhead ανατίθεται ο χρήστης.

Στη συνέχεια αποθηκεύουμε σε κάθε Clusterhead αντικείμενα που είναι συναφή με τους χρήστες που έχουν ανατεθεί σε αυτόν. Εκφράζοντας για κάθε χρήστη το *Quality Of Experience (QOE)* ως συνάρτηση της συνάφειας ενός αντικειμένου και του χρήστη, και του πόσο γρήγορα μπορεί να το προσπελάσει ανάλογα με το που είναι αποθηκευμένο, κάνουμε προτάσεις *(recommendations)* σε κάθε χρήστη, τα αντικείμενα αυτά που μεγιστοποιούν το QOE του. Θεωρούμε ότι ένας χρήστης μπορεί να λάβει περιεχόμενο από οποιονδήποτε από τους επιλεγμένους Clusterheads, αλλά προφανώς συναντά πιθανοτικά συχνότερα τον Clusterhead στον οποίο έχει ανατεθεί, οπότε εξασφαλίζουμε ότι σε κάθε Clusterhead αποθηκεύονται αντικείμενα τα οποία έχουν υψηλή συνάφεια με όλους τους χρήστες που έχουν ανατεθεί σε αυτόν. Στην περίπτωση που για ένα χρήστη, ούτε ο δικός του Clusterhead, ούτε οποιοσδήποτε άλλος διαθέτει ένα αντικείμενο το οποίο θέλουμε να προτείνουμε στο χρήστη αυτόν, θεωρούμε ότι το λαμβάνει από το core network αλλά με αρκετά μεγαλύτερη καθυστέρηση.

Τέλος, αξιολογούμε την επίδοση του προτεινόμενου συστήματος μέσω προσομοιώσεων σε συνθετικά δεδομένα. Εξετάζουμε την επίδραση διαφόρων παραμέτρων, συγκεκριμένα, του πλήθους των χρηστών και των αντικειμένων, του πλήθους των Clusterheads, του πλήθους των προτάσεων που κάνουμε σε κάθε χρήστη, του πλήθους των θεματικών κατηγοριών στις οποίες θεωρούμε ότι υπάγονται τα αντικείμενα του συστήματός μας, καθώς και του μεγέθους της χωρητικότητας του αποθηκευτικού χώρου των συσκευών των χρηστών, μελετώντας τη μεταβολή του μέσου QOE των χρηστών και παρατηρώντας τη συμπεριφορά του αλγορίθμου μας, για διαφορετικούς συνδυασμούς τιμών. Τα χαρακτηριστικά του συστήματος και τα βήματα του αλγορίθμου παρουσιάζονται αναλυτικά στις επόμενες ενότητες.

5.2 Χαρακτηριστικά συστήματος

5.2.1 Χρόνος

Θεωρούμε ότι κάθε χρονική στιγμή αντιστοιχεί σε μία διακριτή τιμή χρόνου i. Η εξέταση του μοντέλου που αναλύουμε παρακάτω γίνεται εντός διαστήματος t χρονικών βημάτων, δηλαδή $i \in [0, t] \in \mathbb{N}$.

5.2.2 Κατηγορίες

Θεωρούμε l θεματικές κατηγορίες, οι οποίες σχετίζονται με το είδος των αντικειμένων που διαχειρίζεται το εκάστοτε σύστημα συστάσεων που μπορεί να υλοποιηθεί μέσω αυτού του μοντέλου. Για παράδειγμα για ένα σύστημα συστάσεων ταινιών, οι l κατηγορίες μπορεί να αφορούν τα είδη των ταινιών όπως Comedy, Drama, Action,

κ.ο.κ., ενώ σε συστήματα συστάσεων μουσικής, οι κατηγορίες μπορεί να αφορούν είδη μουσικής όπως Pop, Rock, Jazz, κ.ο.κ. Χάριν απλότητας και ευελιξίας δεν διευκρινίζουμε τις κατηγορίες στις οποίες διακρίνονται τα αντικείμενα ούτε και το είδος του Recommendation System, καθώς θεωρούμε ότι μια πληθώρα από διαφορετικά συστήματα συστάσεων θα μπορούσαν να υλοποιηθούν μέσω του μοντέλου που αναλύουμε στη συνέχεια.

5.2.3 Χρήστες

Θεωρούμε ότι έχουμε ένα σύνολο χρηστών $U = \{u_1, ..., u_n\}$ οι οποίοι μετακινούνται σε ένα χώρο ενδιαφέροντος και στους οποίους θέλουμε να προτείνουμε περιεχόμενο. Ορίζουμε διάνυσμα χαρακτηριστικών *(feature vector)* f_u για κάθε $u \in U$ μεγέθους l, όπου το κάθε στοιχείο $f_u(j)$ αντιστοιχεί στο πόση συνάφεια *(relevance)* έχει ο χρήστης u με την κατηγορία $j \in \{1, ..., l\}$. Τα διανύσματα αυτά είναι κανονικοποιημένα και ισχύει ότι $\sum_{j=1}^{l} f_u(j) = 1, \forall u \in U$. Θεωρούμε, δηλαδή, ότι κάθε χρήστης έχει μη μηδενική συνάφεια τουλάχιστον με μία (έως και με όλες) από τις θεματικές κατηγορίες αλλά με διαφορετική βαρύτητα ως προς την κάθε μία.

Επίσης υποθέτουμε πως όλοι οι χρήστες είναι υποψήφιοι Clusterheads, δηλαδή διατεθειμένοι να παραχωρήσουν χώρο από τη συσκευή τους για την αποθήκευση περιεχομένου. Κάθε χρήστης διαθέτει σταθερό αποθηκευτικό χώρο χωρητικότητας C_u.

5.2.4 Αντικείμενα

Θεωρούμε ένα σύνολο αντικειμένων $I = \{i_1, ..., i_c\}$ κάποια εκ των οποίων θέλουμε να αποθηκεύσουμε αρχικά σε Clusterheads και στη συνέχεια να τα προτείνουμε και να τα διαμοιράσουμε στους χρήστες του δικτύου. Κάθε αντικείμενο έχει ένα πεπερασμένο μέγεθος s_i το οποίο παίρνει τιμές στο διάστημα $[1, 10]$ μέσω της συνάρτησης ομοιόμορφης κατανομής. Ορίζουμε κι εδώ ένα διάνυσμα χαρακτηριστικών f_i για κάθε $i \in I$ μεγέθους l, όπου το κάθε στοιχείο $f_i(j)$ αντιστοιχεί στο πόση συνάφεια έχει το αντικείμενο i με την κατηγορία $j \in \{1, ..., l\}$. Τα διανύσματα αυτά είναι κανονικοποιημένα και ισχύει ότι $\sum_{j=1}^{l} f_i(j) = 1, \forall i \in I$. Θεωρούμε κι εδώ ότι κάθε αντικείμενο μπορεί να έχει συνάφεια με περισσότερες από μία κατηγορίες, όπως και με τους χρήστες. Χάριν απλότητας έχουμε θεωρήσει ότι έχουμε μόνο ένα αντίγραφο του κάθε αντικειμένου.

5.2.5 Μέρη ενδιαφέροντος

Ορίζουμε ένα σύνολο από μέρη που επισκέπτονται οι χρήστες του δικτύου εντός μιας περιοχής ενδιαφέροντος, στην οποία κινούνται. Τα μέρη αυτά μπορούν να αντιστοιχιστούν σε γεωγραφικές περιοχές, αλλά για τη μαθηματική και θεωρητική εξέταση του μοντέλου, τα θεωρούμε χάριν απλότητας, σημεία. Η εξεταζόμενη περιοχή είναι το μοναδιαίο τετράγωνο $[0, 1]^2$ οπότε έχουμε σύνολο από μέρη $O = \{o_1, ..., o_p\}$, όπου

$o_i \in [0,1]^2,\ \forall i \in \{1,...,p\}$. Κι εδώ ορίζουμε διάνυσμα χαρακτηριστικών f_o για κάθε $o \in O$ μεγέθους l, όπου το κάθε στοιχείο $f_o(j)$ αντιστοιχεί στο πόση συνάφεια έχει το μέρος o με την κατηγορία $j \in \{1,...,l\}$. Τα διανύσματα αυτά είναι κανονικοποιημένα και ισχύει ότι $\sum_{j=1}^{l} f_o(j) = 1, \forall o \in O$.

5.2.6 Συνάφεια χρήστη-αντικειμένου

Ορίζουμε τη συνάφεια ενός χρήστη $u \in U$ με ένα αντικείμενο $i \in I$, μέσω του cosine similarity των διανυσμάτων f_u και f_i (με αυτόν τον τρόπο ορίζεται η συνάφεια και εδώ [28]):

$$sim(u,i) = \frac{\sum_{j=1}^{l} \mathbf{f_u}(j) \cdot \mathbf{f_i}(j)}{\sqrt{\sum_{j=1}^{l} \mathbf{f_u}^2(j)}\sqrt{\sum_{j=1}^{l} \mathbf{f_i}^2(j)}}. \tag{5.1}$$

5.2.7 Συνάφεια χρήστη-μέρους

Ορίζουμε τη συνάφεια ενός χρήστη $u \in U$ με ένα μέρος $o \in O$, μέσω του cosine similarity των διανυσμάτων f_u και f_o:

$$sim(u,o) = \frac{\sum_{j=1}^{l} \mathbf{f_u}(j) \cdot \mathbf{f_o}(j)}{\sqrt{\sum_{j=1}^{l} \mathbf{f_u}^2(j)}\sqrt{\sum_{j=1}^{l} \mathbf{f_o}^2(j)}}. \tag{5.2}$$

5.2.8 Μοντέλο κίνησης χρηστών

Έστω $G_o = (O, E_o, D)$ το έμβαρο πλήρες γράφημα των μερών, όπου D είναι ο $p \times p$ πίνακας των ευκλείδειων αποστάσεων μεταξύ των μερών. Το βάρος της ακμής που ορίζεται μεταξύ των μερών $o_i = (x_i, y_i) \in [0,1]^2$ και $o_j = (x_j, y_j) \in [0,1]^2$, όπου $i,j \in \{1,...,p\}$, ισούται με το στοιχείο (i,j) του πίνακα D, δηλαδή την ευκλείδια απόστασή τους $d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$.

Θεωρούμε, στη συνέχεια, την κανονικοποιημένη μορφή των αποστάσεων μεταξύ των χρηστών ως εξής:

$$d'_{ij} = \frac{d_{ij} - d_{min}}{d_{max} - d_{min}}$$

όπου $d_{min} = min_{i,j \in \{1,..,p\}} d_{ij}$ και $d_{max} = max_{i,j \in \{1,..,p\}} d_{ij}$

Η κινητικότητα των χρηστών του μοντέλου εκφράζεται μέσω τυχαίων περιπάτων πάνω στο γράφημα G_o. Η πιθανότητα μετάβασης από ένα μέρος $o_i \in O$ σε ένα άλλο $o_j \in O$ για κάθε χρήστη, εξαρτάται από την (κανονικοποιημένη) απόσταση d'_{ij} των δύο μερών και τη συνάφεια του χρήστη με το μέρος o_j. Οπότε ορίζουμε τον πίνακα μετάβασης P_u του τυχαίου περιπάτου κάθε χρήστη $u \in U$ μέσω των στοιχείων του, ως γραμμική συνάρτηση g των δύο παραπάνω ποσοτήτων. Ασφαλώς τα στοιχεία του πίνακα θα

είναι κανονικοποιημένα όπως φαίνεται και παρακάτω

$$p_{ij}(u) = \frac{g(sim(u, o_j), 1 - d'_{ij})}{\sum_{j=1}^{p} g(sim(u, o_j), 1 - d'_{ij})} \tag{5.3}$$

όπου $p_{ij}(u)$ η πιθανότητα μετάβασης του χρήστη $u \in U$ από το μέρος $o_i \in O$ στο μέρος $o_j \in O$. Η παραπάνω έκφραση μας επιτρέπει να μπορούμε να θεωρήσουμε πληθώρα συστημάτων στα οποία άλλοτε για τους χρήστες να παίζει σημαντικότερο ρόλο το μέρος στο οποίο θα μετακινηθούν να είναι κοντινό σε αυτό που βρίσκονται σε ένα χρονικό βήμα, σε σχέση με το πόση συνάφεια έχουν με το μέρος αυτό και άλλοτε η συνάφεια να παίζει σημαντικότερο ρόλο. Εμείς, χάριν απλότητας, θεωρούμε ότι η βαρύτητα για τους χρήστες ως προς τις δύο ποσότητες είναι η ίδια οπότε χρησιμοποιούμε την παρακάτω έκφραση:

$$p_{ij}(u) = \frac{sim(u, o_j) + 1 - d'_{ij}}{\sum_{j=1}^{p} (sim(u, o_j) + 1 - d'_{ij})} \tag{5.4}$$

Υποθέτουμε ότι οι πιθανότητες που ορίζονται από την παραπάνω ισότητα 5.4 είναι μη μηδενικές. Αυτό εξασφαλίζεται απαιτώντας, για μέρη $o_i, o_j \in O$ τα οποία προσπίπτουν σε μία διάμετρο του γράφου G_o, αφού σε πλήρες γράφημα η διάμετρος είναι η ακμή με το μεγαλύτερο βάρος, (δηλαδή $d'_{ij} = 1$), κάθε χρήστης να έχει μη μηδενική συνάφεια ως προς τα μέρη αυτά (δηλαδή $sim(u, o_i) \neq 0$ και $sim(u, o_j) \neq 0, \forall u \in U$) ώστε $p_{ij}(u) \neq 0$ και $p_{ij}(u) \neq 0, \forall u \in U$.

5.3 Επιλογή των Clusterheads

5.3.1 Δημιουργία Γράφου Χρηστών

Για την επιλογή των Clusterheads οι οποίοι θα είναι υπεύθυνοι για την αποθήκευση των αντικειμένων, θα δημιουργήσουμε ένα έμβαρο γράφημα μεταξύ των χρηστών, το οποίο θα έχει ως βάρη τα εκτιμώμενα πλήθη χρονικών βημάτων *(expected inter-contact time)* μεταξύ διαδοχικών συναντήσεων των χρηστών κατά τους τυχαίους περιπάτους τους. Διαισθητικά τα βάρη του γράφου αυτού θα αντιστοιχούν σε μια έννοια "απόστασης" μεταξύ των χρηστών ως προς το πόσο συγγενικοί είναι ως προς τις προτιμήσεις τους.

Εφόσον το γράφημα των μερών G_o στο οποίο οι χρήστες εκτελούν τους τυχαίους περιπάτους είναι πλήρες και για κάθε χρήστη, κάθε μέρος είναι προσβάσιμο από οποιοδήποτε άλλο, λόγω των παραπάνω μη μηδενικών πιθανοτήτων μετάβασης που ορίζονται στην 5.4, έχουμε ότι το γράφημα G_o είναι συνεκτικό και μη διμερές. Από το θεώρημα 2.2.1 έχουμε ότι ο τυχαίος περίπατος κάθε χρήστη $u \in U$ στο γράφο αυτό, έχει μοναδική στάσιμη κατανομή π_u (όπου το πλήθος των συντεταγμένων του διανύσματος π_u είναι ίσο με το πλήθος των μερών του συστήματος), οπότε κάθε χρήστης έχει σταθερή πιθανότητα μετάβασης προς κάθε μέρος ανεξάρτητα από το

πού βρισκόταν την προηγούμενη χρονική στιγμή του τυχαίου περιπάτου του.

Έστω $\{X_{uv}(i)\}_i$ οικογένεια τυχαίων μεταβλητών (που περιγράφουν ανεξάρτητες αλλά όχι πανομοιότυπα κατανεμημένες δοκιμές Bernoulli) για τις οποίες ισχύει:

$$X_{uv}(i) = \begin{cases} 1, & \text{αν ο } u \text{ συναντά τον } v \text{ τη χρονική στιγμή i,} \\ 0, & \text{αν ο } u \text{ δε συναντά τον } v \text{ τη χρονική στιγμή i.} \end{cases} \tag{5.5}$$

Οι χρήστες επισκέπτονται ανεξάρτητα τα μέρη ενδιαφέροντος, οπότε η πιθανότητα δύο χρήστες $u, v \in U$ να συναντηθούν μετά από i χρονικά βήματα, όπου $i \in [0, t]$ είναι:

$$P(X_{uv}(i) = 1) = \pi_u \cdot \pi_v^\intercal, \quad \forall i \in [0, t] \subset \mathbb{N} \tag{5.6}$$

Ορίζουμε την τυχαία μεταβλητή Y_{uv}, η οποία αναπαριστά το πλήθος των χρονικών βημάτων που μεσολαβούν μεταξύ δύο διαδοχικών συναντήσεων (inter-contact time) δύο χρηστών $u, v \in U$. Μπορούμε να θεωρήσουμε αυτή την τυχαία μεταβλητή ως το πλήθος των δοκιμών Bernoulli με πιθανότητα επιτυχίας $\pi_u \cdot \pi_v^\intercal$ που απαιτούνται μέχρι την πρώτη επιτυχία, δηλαδή τη συνάντηση των χρηστών. Τότε το inter-contact time Y_{uv} ακολουθεί γεωμετρική κατανομή και η πιθανότητα του ενδεχομένου το inter-contact time να ισούται με k χρονικά βήματα (θεωρώντας ότι συναντιούνται στο $k+1$-στο βήμα) είναι:

$$P(Y_{uv} = k+1) = P(X_{uv}(k+1) = 1) \prod_{i=1}^{k} (1 - P(X_{uv}(i) = 1) = (1 - \pi_u \cdot \pi_v^\intercal)^k \cdot \pi_u \cdot \pi_v^\intercal \tag{5.7}$$

με μέση τιμή:

$$E(Y_{uv}) = \frac{1 - \pi_u \cdot \pi_v^\intercal}{\pi_u \cdot \pi_v^\intercal} \tag{5.8}$$

Η παραπάνω μέση τιμή 5.8 εκφράζει το εκτιμώμενο inter-contact time μεταξύ δύο χρηστών $u, v \in U$. Δεδομένης, λοιπόν, αυτής της ποσότητας για κάθε ζεύγος χρηστών, θεωρούμε το πλήρες γράφημα $G_u = (U, E_u, W)$, με κορυφές τους χρήστες U του συστήματος και ακμές μεταξύ τους, βάρους $w(u, v) = E(Y_{uv}), \forall u, v \in U$. Ακμές ορίζονται και μεταξύ ίδιων κορυφών (self-loops), αν και μηδενικού βάρους, οπότε το γράφημα G_u είναι μη απλό. Επίσης, τα βάρη αυτά, από τον τρόπο που ορίζονται, είναι μη αρνητικά και συμμετρικά, δηλαδή $w(u, v) = w(v, u) \forall u, v \in U$. Στο έμβαρο αυτό γράφημα θα εφαρμόσουμε το k-median πρόβλημα για να επιλέξουμε τους Clusterheads του μοντέλου μας, χρησιμοποιώντας την κατάλληλη εκδοχή του (metric ή non-metric) ανάλογα με το αν τα βάρη του γράφου G_u ικανοποιούν την τριγωνική ανισότητα.

5.3.2 Συνθήκη ικανοποίησης τριγωνικής ανισότητας

Έχοντας αναλύσει στο Κεφάλαιο 2 τους προσεγγιστικούς αλγορίθμους επίλυσης του metric και του non-metric k-median προβλήματος αντιλαμβανόμαστε ότι ο αλγόριθμος για το μετρικό πρόβλημα υπερτερεί του αλγορίθμου για το μη μετρικό αφού

έχει πολύ καλό και σταθερό λόγο προσέγγισης και το πλήθος των facilities (στην περίπτωσή μας Clusterheads) που υπολογίζει είναι ίσο με k. Αντίθετα, ο αλγόριθμος για το μη μετρικό k-median πρόβλημα επιτρέπει την επιλογή το πολύ $k \cdot ln(n+\frac{n}{\varepsilon})$ Clusterheads και η προσέγγισή του ως προς το κόστος δεν είναι σταθερή. Ασφαλώς δεν μπορούμε να ελέγξουμε τις στάσιμες κατανομές των τυχαίων περιπάτων των χρηστών και κατά συνέπεια τα βάρη του γραφήματος των χρηστών G_u, ώστε να ικανοποιούν την τριγωνική ανισότητα και κατ'επέκταση να εξασφαλίσουμε ότι χρησιμοποιούμε τον αλγόριθμο για το metric πρόβλημα, γιατί έτσι θα κατευθύναμε τη συμπεριφορά των χρηστών, το οποίο δεν είναι ρεαλιστικό και θεμιτό. Ωστόσο μπορούμε να διατυπώσουμε τη συνθήκη που πρέπει να ικανοποιούν οι στάσιμες κατανομές των χρηστών ώστε η τριγωνική ανισότητα στο γράφημα G_u να ισχύει πάντα.

Προκειμένου να ισχύει η τριγωνική ανισότητα για κάθε τριάδα χρηστών στο γράφημα G_u, θα πρέπει να ισχύει:

$$\begin{aligned} &w(u,v) + w(v,z) \geq w(z,u),\ \forall u,v,z \in U \iff \\ &E(Y_{uv}) + E(Y_{vz}) \geq E(Y_{zu}),\ \forall u,v,z \in U \iff \\ &\frac{1-\pi_u \cdot \pi_v^\intercal}{\pi_u \cdot \pi_v^\intercal} + \frac{1-\pi_v \cdot \pi_z^\intercal}{\pi_v \cdot \pi_z^\intercal} \geq \frac{1-\pi_z \cdot \pi_u^\intercal}{\pi_z \cdot \pi_u^\intercal},\ \forall u,v,z \in U \end{aligned} \tag{5.9}$$

Επίσης ισχύει ότι:

$$0 \leq \pi_u \cdot \pi_v^\intercal \leq 1,\ \forall u,v \in U \tag{5.10}$$

Από τις παραπάνω ανισότητες 5.9, 5.10 παρατηρούμε ότι όσο πιο κοντά είναι οι τιμές των εσωτερικών γινομένων των στάσιμων κατανομών των χρηστών, τόσο πιο πιθανό είναι η τριγωνική ανισότητα να ισχύει στο γράφημα. Αν οι συντεταγμένες των διανυσμάτων των στάσιμων κατανομών παίρνουν οποιαδήποτε δυνατή τιμή, τότε το εσωτερικό γινόμενο δύο διανυσμάτων μπορεί να πάρει οποιαδήποτε τιμή μεταξύ του 0 και του 1. Αν, όμως, φράξουμε τις συντεταγμένες του διανύσματος της στάσιμης κατανομής κάθε χρήστη κατάλληλα, ώστε όλες να έχουν ένα κάτω όριο (έστω ε), τότε οι τιμές των εσωτερικών γινομένων των κατανομών θα είναι πιο συγκεντρωμένες προς τις ενδιάμεσες τιμές του διαστήματος $[0,1]$. Οπότε αν τα γινόμενα αυτά δεν έχουν μεγάλη απόκλιση μεταξύ τους, τότε και τα βάρη των ακμών του γραφήματος που εκφράζονται μέσω των εσωτερικών γινομένων, όπως φαίνεται και στην εξίσωση 5.9, δε θα έχουν ούτε αυτά μεγάλη απόκλιση, οπότε η τριγωνική ανισότητα θα ισχύει με μεγάλη πιθανότητα. Στην ενότητα αυτή εξετάζουμε πόσο φραγμένες πρέπει να είναι οι συντεταγμένες αυτές, βρίσκουμε δηλαδή το εύρος που κυμαίνεται το ε, για κάθε πλήθος συνιστωσών $n \in \mathbb{N}$ του διανύσματος των στάσιμων κατανομών, ώστε η τριγωνική ανισότητα να ισχύει πάντα.

Ορισμός 5.3.1 (ε-κάτω φραγμένο διάνυσμα)**.** *Έστω $\varepsilon > 0$. Ένα διάνυσμα x ονομάζεται ϵ-κάτω φραγμένο, αν κάθε συντεταγμένη του είναι $\geq \varepsilon$.*

Για να προχωρήσουμε την ανάλυση, αναφέρουμε τους δύο παρακάτω ορισμούς [35]:

Ορισμός 5.3.2 (Κυρτή θήκη)**.** *Η* ***κυρτή θήκη*** *ενός πεπερασμένου συνόλου* $X = \{x^1, ..., x^n\}$ *σημείων στο* $\mathbb{R}^d$ *ορίζεται ως εξής:*

$$Q = conv(X) := \{\sum_{i=1}^{n} \lambda_i x^i \mid \lambda_i \geq 0, \ \sum_{i=1}^{n} \lambda_i = 1\}$$

Με άλλα λόγια η κυρτή θήκη του X *είναι το σύνολο* $conv(X)$ *που αποτελείται από όλους τους κυρτούς συνδυασμούς των σημείων του* X*.*

Ορισμός 5.3.3 (Εναλλακτικός ορισμός για την κυρτή θήκη)**.** *Η κυρτή θήκη ενός πεπερασμένου συνόλου* $X = \{x^1, ..., x^n\}$ *σημείων στο* $\mathbb{R}^d$ *είναι το μικρότερο κυρτό σύνολο που περιέχει το* X*.*

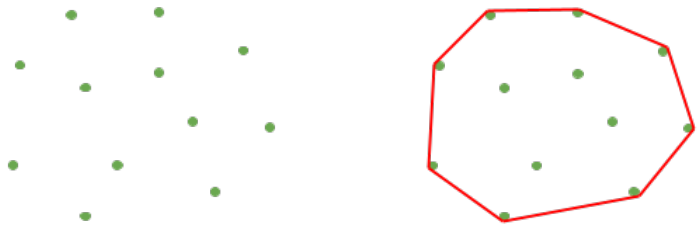


Figure 5.1: Κυρτή θήκη συνόλου σημείων (κόκκινο περίγραμμα)

Έστω διανύσματα n συντεταγμένων, τότε προκειμένου όλες τους οι συντεταγμένες να φράσσονται από ε, δηλαδή τα διανύσματα να είναι ε-κάτω φραγμένα, πρέπει να ανήκουν στην κυρτή θήκη του συνόλου των σημείων
$A = \{(\underbrace{\varepsilon, ..., \varepsilon}_{\text{i}}, 1 - (n-1)\varepsilon, \underbrace{\varepsilon, ..., \varepsilon}_{\text{n-i-1}}) : i \in \{0, ..., n-1\}\}$. Λόγω συμμετρίας, κανένα σημείο του A δεν περιέχεται στην κυρτή θήκη των άλλων. Διατυπώνουμε το ακόλουθο θεώρημα:

Θεώρημα 5.3.1. *Έστω* $\varepsilon > 0$ *και έστω το σύνολο* n *σημείων*
$A = (x_1, ..., x_n) = \{(\underbrace{\varepsilon, ..., \varepsilon}_{i}, 1 - (n-1)\varepsilon, \underbrace{\varepsilon, ..., \varepsilon}_{n-i-1}) : i \in \{0, ..., n-1\}\}$ *και*
$Q = conv(A)$*, τότε* $\forall p, q \in Q$ *ισχύει ότι* p, q *ε-κάτω φραγμένα και*

$$c \leq p \cdot q \leq C$$

όπου $c = (n-2)\varepsilon^2 + 2\varepsilon(1 - (n-1)\varepsilon)$ *και* $C = (n-1)\varepsilon^2 + (1 - (n-1)\varepsilon)^2$

Proof. Έχουμε ότι $p, q \in Q$, τότε από τον ορισμό της κυρτής θήκης 5.3.2 τα διανύσματα εκφράζονται ως:

$$p = \{\sum_{i=1}^{n} a_i x_i \mid a_i \geq 0, \ \sum_{i=1}^{n} a_i = 1\}$$

και

$$q = \{\sum_{i=1}^{n} b_i x_i \mid b_i \geq 0,\ \sum_{i=1}^{n} b_i = 1\}$$

Οπότε

$$\begin{cases} p = & ((a_1(1-(n-1)\varepsilon) + a_2\varepsilon + ... + a_{n-1}\varepsilon + a_n\varepsilon), \\ & (a_1\varepsilon + a_2(1-(n-1)\varepsilon) + ... + a_{n-1}\varepsilon + a_n\varepsilon), ..., (a_1\varepsilon + a_2\varepsilon + ... + a_{n-1}\varepsilon + a_n(1-(n-1)\varepsilon))) \\ q = & ((b_1(1-(n-1)\varepsilon) + b_2\varepsilon + ... + b_{n-1}\varepsilon + b_n\varepsilon), \\ & (b_1\varepsilon + b_2(1-(n-1)\varepsilon) + ... + b_{n-1}\varepsilon + b_n\varepsilon), ..., (b_1\varepsilon + b_2\varepsilon + ... + b_{n-1}\varepsilon + b_n(1-(n-1)\varepsilon))) \end{cases}$$

άρα

$$\begin{cases} p = & ((a_1(1-(n-1)\varepsilon) + (1-a_1)\varepsilon), \\ & (a_2(1-(n-1)\varepsilon) + (1-a_2)\varepsilon), ..., (a_n(1-(n-1)\varepsilon) + (1-a_n)\varepsilon)) \\ q = & ((b_1(1-(n-1)\varepsilon) + (1-b_1)\varepsilon), \\ & (b_2(1-(n-1)\varepsilon) + (1-b_2)\varepsilon), ..., (b_n(1-(n-1)\varepsilon) + (1-b_n)\varepsilon)) \end{cases}$$

Επομένως έχουμε ότι

$$\begin{aligned} p \cdot q &= \sum_{i=1}^{n} (a_i(1-(n-1)\varepsilon) + (1-a_i)\varepsilon)(b_i(1-(n-1)\varepsilon) + (1-b_i)\varepsilon) \\ &= \sum_{i=1}^{n} (a_i b_i(1-(n-1)\varepsilon)^2 + (a_i + b_i - 2a_i b_i)\varepsilon(1-(n-1)\varepsilon) + (1-a_i-b_i+a_i b_i)\varepsilon^2) \\ &= (1-(n-1)\varepsilon)^2 \sum_{i=1}^{n} a_i b_i + \varepsilon(1-(n-1)\varepsilon)(\sum_{i=1}^{n} a_i + \sum_{i=1}^{n} b_i - 2\sum_{i=1}^{n} a_i b_i) + \\ &\qquad + \varepsilon^2(\sum_{i=1}^{n} 1 - \sum_{i=1}^{n} a_i - \sum_{i=1}^{n} b_i + \sum_{i=1}^{n} a_i b_i), \qquad \text{όπου } \sum_{i=1}^{n} a_i = \sum_{i=1}^{n} b_i = 1 \\ &= (1-(n-1)\varepsilon)^2 \sum_{i=1}^{n} a_i b_i + 2\varepsilon(1-(n-1)\varepsilon)(1 - \sum_{i=1}^{n} a_i b_i) + \varepsilon^2(n-2+\sum_{i=1}^{n} a_i b_i) \end{aligned}$$

Ισχύει ότι $0 \leq \sum_{i=1}^{n} a_i b_i \leq 1$ και αν αντιμετωπίσουμε το παραπάνω γινόμενο σαν συνεχή γραμμική συνάρτηση με μεταβλητή το $\sum_{i=1}^{n} a_i b_i$, τότε η μέγιστη και η ελάχιστη τιμή της συνάρτησης εμφανίζονται στα άκρα του διαστήματος στο οποίο ανήκει η μεταβλητή:

Για $\sum_{i=1}^{n} a_i b_i = 0$:

$$p \cdot q = (n-2)\varepsilon^2 + 2\varepsilon(1-(n-1)\varepsilon) = c \tag{5.11}$$

Για $\sum_{i=1}^{n} a_i b_i = 1$:

$$p \cdot q = (n-1)\varepsilon^2 + (1-(n-1)\varepsilon)^2 = C \tag{5.12}$$

Από 5.11, 5.12

$$c \leq p \cdot q \leq C$$

□

Παρακάτω βλέπουμε την αναπαράσταση της κυρτής θήκης του συνόλου A για $n = 2$. Το σύνολο A γίνεται τότε $A = \{(1-\varepsilon, \varepsilon), (\varepsilon, 1-\varepsilon)\}$ και τα διανύσματα p, q προσπίπτουν στην κυρτή θήκη $Q = conv(A)$, η οποία αντιστοιχεί στην κόκκινη ευθεία του παρακάτω σχήματος. Η κυρτή θήκη του A είναι ένα κανονικό πολύτοπο simplex, οπότε για $n = 1$ το Q είναι σημείο, για $n = 2$ το Q είναι ευθεία (όπως φαίνεται και παρακάτω), για $n = 3$ το Q είναι ισόπλευρο τρίγωνο, για $n = 4$ το Q είναι κανονικό τετράεδρο κ.ο.κ

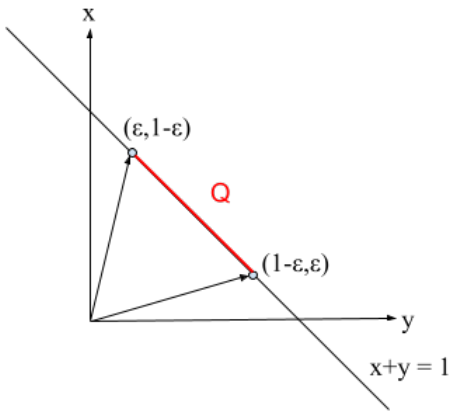


Figure 5.2: Κυρτή θήκη του συνόλου A για $n = 2$ (κόκκινη ευθεία)

Έχοντας βρει, μέσω του παραπάνω θεωρήματος 5.3.1, το άνω και κάτω φράγμα του εσωτερικού γινομένου δύο τυχαίων διανυσμάτων που προσπίπτουν στην κυρτή θήκη του A, επανερχόμαστε στη συνθήκη ισχύος της τριγωνικής ανισότητας στο γράφο G_u. Υπενθυμίζουμε ότι για να ισχύει η τριγωνική ανισότητα πρέπει:

$$\frac{1 - \pi_u \cdot \pi_v^\intercal}{\pi_u \cdot \pi_v^\intercal} + \frac{1 - \pi_v \cdot \pi_z^\intercal}{\pi_v \cdot \pi_z^\intercal} \geq \frac{1 - \pi_z \cdot \pi_u^\intercal}{\pi_z \cdot \pi_u^\intercal}, \ \forall u, v, z \in U$$

Από το 5.3.1:

$$\frac{1 - \pi_u \cdot \pi_v^\intercal}{\pi_u \cdot \pi_v^\intercal} + \frac{1 - \pi_v \cdot \pi_z^\intercal}{\pi_v \cdot \pi_z^\intercal} \geq 2 \cdot \frac{1 - C}{C}$$

και

$$\frac{1 - \pi_z \cdot \pi_u^\intercal}{\pi_z \cdot \pi_u^\intercal} \leq \frac{1 - c}{c}$$

Επομένως αρκεί:

$$2 \cdot \frac{1-C}{C} \geq \frac{1-c}{c} = 2 \cdot \frac{1-\frac{2c}{1+c}}{\frac{2c}{1+c}} \iff$$
$$C \leq \frac{2c}{1+c} \tag{5.13}$$

αφού η συνάρτηση $f(x) = \frac{1-x}{x}$ είναι γνησίως φθίνουσα

Αντικαθιστώντας, η ανισότητα 5.13 γίνεται:

$$(n-1)\varepsilon^2 + (1-(n-1)\varepsilon)^2 \leq 2 \cdot \frac{(n-2)\varepsilon^2 + 2\varepsilon(1-(n-1)\varepsilon)}{1+(n-2)\varepsilon^2 + 2\varepsilon(1-(n-1)\varepsilon)} \tag{5.14}$$

και λύνοντας ως προς ε προκύπτουν οι παρακάτω δυνατές λύσεις:

$$\begin{cases} \varepsilon = 1, \text{ για } n = 1 \\ \frac{2\sqrt{n-1}+n-\sqrt{n^2+4n-4}}{2n\sqrt{n-1}} \leq \varepsilon \leq \frac{2\sqrt{n-1}-n+\sqrt{n^2+4n-4}}{2n\sqrt{n-1}}, \text{ για } n > 1 \end{cases} \tag{5.15}$$

Αναφέρουμε ότι οι λύσεις αυτές περιέχουν την τιμή $\frac{1}{n}$, αφού $\frac{2\sqrt{n-1}-n+\sqrt{n^2+4n-4}}{2n\sqrt{n-1}} \geq \frac{1}{n},\ n > 1,\ 0 < \varepsilon \leq 1$. Η τιμή $\frac{1}{n}$ είναι η μεγαλύτερη που μπορεί να πάρει όμως το ε, αφού έχουμε διανύσματα κατανομών όπου οι συντεταγμένες αθροίζουν στο 1. Οπότε για $\varepsilon = \frac{1}{n}$ όλα τα διανύσματα του συνόλου A γίνονται $(\frac{1}{n}, \frac{1}{n}, ..., \frac{1}{n})$ οπότε η κυρτή θήκη γίνεται σημείο και όλα τα διανύσματα κατανομών θα είναι της μορφής $(\frac{1}{n}, \frac{1}{n}, ..., \frac{1}{n})$ που στην περίπτωση αυτή η τριγωνική ανισότητα ισχύει σε όλο το γράφο G_u.

Επομένως το εύρος τιμών στο οποίο κυμαίνεται το κάτω φράγμα ε των συντεταγμένων των διανυσμάτων στάσιμων κατανομών των χρηστών, προκειμένου η τριγωνική ανισότητα να ισχύει στο γράφημα G_u με πιθανότητα 1, παραλείποντας την τετριμμένη περίπτωση για $n = 1$, είναι:

$$\frac{2\sqrt{n-1}+n-\sqrt{n^2+4n-4}}{2n\sqrt{n-1}} \leq \varepsilon \leq \frac{1}{n}, \quad n > 1,\ 0 < \varepsilon \leq 1 \tag{5.16}$$

Συνοψίζοντας έχουμε το εξής αποτέλεσμα:

Θεώρημα 5.3.2. *Έστω ότι π_v ϵ-κάτω-φραγμένο διάνυσμα στάσιμης κατανομής n συντεταγμένων για κάθε $v \in U$, όπου*

$$\frac{2\sqrt{n-1}+n-\sqrt{n^2+4n-4}}{2n\sqrt{n-1}} \leq \varepsilon \leq \frac{1}{n}, \quad n > 1$$

Τότε ισχύει η τριγωνική ανισότητα στον γράφο G_u.

5.3.3 Εφαρμογή k-Median Προβλήματος στο γράφο χρηστών

Όπως αναφέραμε, το πρόβλημα της επιλογής των Clusterheads μπορεί να αντιστοιχιστεί στο πρόβλημα k-median του οποίου ο στόχος είναι να επιλεγούν k από τους χρήστες του συνόλου U ως facilities και να ανατεθεί κάθε χρήστης του U σε ένα από τα k επιλεγμένα facilities, έτσι ώστε να ελαχιστοποιηθεί το συνολικό κόστος ανάθεσης που προκύπτει.

Όπως είδαμε στην ενότητα 5.3.2, ανάλογα με τις στάσιμες κατανομές των χρηστών, η τριγωνική ανισότητα μπορεί να ισχύει ή να μην ισχύει ενώ, αν οι συντεταγμένες των κατανομών έχουν κάτω φράγμα του οποίου το εύρος δίνεται από την ανισότητα 5.16, η τριγωνική ανισότητα ισχύει με πιθανότητα 1 (θεώρημα 5.3.2). Επειδή όμως για κάποιο τυχαίο στιγμιότυπο του συστήματος δεν μπορούμε να διασφαλίσουμε τις απαραίτητες συνθήκες για να ισχύει η τριγωνική ανισότητα, χρησιμοποιούμε τη metric ή την non-metric εκδοχή του k-median προβλήματος στο γράφο των χρηστών G_u. Υπενθυμίζουμε ότι όλοι οι χρήστες του συστήματος είναι υποψήφιοι Clusterheads επομένως το σύνολο των facilities και το σύνολο των clients, τα οποία δέχεται ως είσοδο το k-median πρόβλημα, ταυτίζονται και είναι ίσα με το σύνολο των χρηστών U. Το k-median πρόβλημα απαιτεί να ορίζονται κόστη εξυπηρέτησης μεταξύ οποιουδήποτε client και facility. Επομένως μπορούμε να εφαρμόσουμε τον αλγόριθμο στο γράφο G_u, αφού το γράφημα αυτό είναι πλήρες και υπάρχουν ακμές μεταξύ οποιωνδήποτε δύο χρηστών, δηλαδή οποιουδήποτε client και facility. Τα κόστη εξυπηρέτησης που θεωρούμε για το k-median πρόβλημα στη δική μας περίπτωση, αντιστοιχούν στα βάρη μεταξύ των χρηστών και έχουν και αυτά την έννοια της απόστασης, δείχνουν δηλαδή πόση "απόσταση" έχουν οι χρήστες ως προς τη συνάφεια, δηλαδή πόσο κοινές προτιμήσεις έχουν. Διαισθητικά, αν δύο χρήστες έχουν κοινές προτιμήσεις, θα μεταβαίνουν με μεγαλύτερη πιθανότητα σε κοινά μέρη, άρα το εκτιμώμενο inter-contact time τους (το βάρος της ακμής που τους συνδέει στο γράφο G_u) θα είναι μικρό, δηλαδή θα συναντιούνται συχνά με μεγάλη πιθανότητα. Οπότε αν οι προτιμήσεις δύο χρηστών είναι αρκετά συναφείς, τότε οι χρήστες θα συναντιούνται συχνά, οπότε το βάρος θα είναι μικρό (μικρό κόστος εξυπηρέτησης), ενώ αν έχουν μεγάλη απόκλιση, το βάρος θα είναι μεγάλο. Διακρίνουμε τις εξής δύο περιπτώσεις:

- **Αν ισχύει η τριγωνική ανισότητα για τα βάρη του γράφου G_u**, δηλαδή $w(u,v)+w(v,z) \geq w(z,u),\ \forall u,v,z \in U$ τότε καλούμε τον αλγόριθμο 2.2 για το metric k-median πρόβλημα με σύνολο facilities και σύνολο clients, το σύνολο U των χρηστών, κόστη εξυπηρέτησης τα βάρη w του γράφου G_u και την παράμετρο k που ορίζει το πλήθος των Clusterheads που θέλουμε να έχει το σύστημα. Συγκεκριμένα θα καλούμε ως εξής:

  Metric_k_Median (U, U, w, k)

  Ο αλγόριθμος επιστρέφει k Clusterheads και συνδέει κάθε χρήστη με τον πιο κοντινό του από τους Clusterheads που επιλέχθηκαν, επιστρέφοντας τον πίνακα που περιέχει αυτή την πληροφορία.

- **Αν δεν ισχύει η τριγωνική ανισότητα για τα βάρη του γράφου** G_u, δηλαδή $\exists u, v, z \in U$ έτσι ώστε $w(u,v)+w(v,z) < w(z,u)$, τότε καλούμε τον αλγόριθμο 2.3 για το non-metric k-median πρόβλημα, με σύνολο facilities και σύνολο clients το σύνολο U των χρηστών, κόστη εξυπηρέτησης τα βάρη w του γράφου G_u, το κόστος της fractional λύσης d που μας δίνει η επίλυση του LP-fractional γραμμικού προγράμματος 2.13 με κάποιον από τους γνωστούς αλγορίθμους επίλυσης γραμμικών προγραμμάτων (π.χ. αλγόριθμος Simplex) θεωρώντας όμως ως σύνολο facilities $F \leftarrow U$ και σύνολο clients $D \leftarrow U$ και τέλος τη σταθερά $\varepsilon_1 > 0$. Συγκεκριμένα καλούμε ως εξής:

  Non_metric_k_median $(U, U, w, d, \varepsilon_1)$

  Ο αλγόριθμος επιστρέφει το πολύ $k \cdot ln(n + \frac{n}{\varepsilon_1})$ Clusterheads και συνδέει κάθε χρήστη με τον πιο κοντινό από τους Clusterheads που επιλέχθηκαν, επιστρέφοντας τον πίνακα που περιέχει αυτή την πληροφορία. Όπως είχαμε δει και στο κεφάλαιο 2, ο αλγόριθμος αυτός δεν εξασφαλίζει την επιλογή ακριβώς k Clusterheads, αλλά όπως έχουμε αναφέρει οι αλγόριθμοι του προβλήματος σε μη μετρικό χώρο δεν μπορούν να επιτύχουν προσέγγιση με μεγαλύτερη ακρίβεια από $O(logn)$. Ο αλγόριθμος αυτός επιτυγχάνει καλή προσέγγιση ως προς το κόστος της λύσης που υπολογίζει, αλλά επιτρέπει η επιλογή των Clusterheads να μπορεί να φτάσει σε πλήθος τους $k \cdot ln(n + \frac{n}{\varepsilon_1})$.

5.4 Τοποθέτηση αντικειμένων στους Clusterheads

Έχοντας επιλέξει τους Clusterheads, μπορούμε λοιπόν να προχωρήσουμε στην επιλογή των αντικειμένων που θα αποθηκεύσουμε σε αυτούς, βρίσκοντας παράλληλα σε ποιον Clusterhead θα αποθηκεύσουμε κάθε αντικείμενο που επιλέξαμε. Για το σκοπό αυτό θα χρησιμοποιήσουμε τον άπληστο αλγόριθμο για το Generalized Assignment Problem 2.5, το οποίο διατυπώσαμε και αναλύσαμε στο κεφάλαιο 2, θεωρώντας εδώ ως αντικείμενα, τα αντικείμενα του συνόλου I του συστήματος και ως σακίδια, τους Clusterheads που επέλεξε το k-median πρόβλημα. Ως μεγέθη των αντικειμένων, θεωρούμε τα μεγέθη s_i των αντικειμένων του συνόλου I, τα οποία είναι διαφορετικά για κάθε αντικείμενο, αλλά θεωρούμε ότι κάθε αντικείμενο έχει σταθερό μέγεθος ως προς όλα τα σακίδια (Clusterheads). Επίσης ως χωρητικότητες των σακιδίων θεωρούμε τις χωρητικότητες των Clusterheads C_u. Ωστόσο οι αξίες των αντικειμένων ορίζονται λίγο διαφορετικά. Έστω H ο πίνακας ανάθεσης των χρηστών στους επιλεγμένους Clusterheads που επέστρεψε το k-median πρόβλημα, τότε η αξία ενός αντικειμένου $i \in I$ ως προς τον Clusterhead $j \in X \subseteq U$, όπου X το σύνολο των Clusterheads που επέστρεψε το k-median, εξαρτάται από τη συνάφεια που έχουν οι χρήστες που εξυπηρετεί ο Clusterhead j, δηλαδή που έχουν ανατεθεί σε αυτόν, ως

προς το αντικείμενο i. Συγκεκριμένα:

$$v_{ij} = \sum_{u:H[u]=j} sim(u,i)$$

Οπότε καλούμε τον αλγόριθμο 2.5 με τα ορίσματα που αναφέραμε παραπάνω και με παράμετρο λάθους $\varepsilon_2 > 0$:

$$\textbf{GAP}\ (I, X, v_{ij}, s_i, C_u, \varepsilon_2)$$

Υπενθυμίζουμε ότι ο παραπάνω αλγόριθμος επιστρέφει έναν πίνακα, έστω T, ο οποίος περιέχει τον Clusterhead στον οποίο θα αποθηκεύσουμε το κάθε αντικείμενο. Υπενθυμίζουμε ότι αν $T[i] = -1$, όπου $i \in I$, τότε το αντικείμενο i δεν αποθηκεύεται σε κανέναν Clusterhead.

5.5 Προτάσεις αντικειμένων στους χρήστες

Θέλουμε να κάνουμε R προτάσεις αντικειμένων σε κάθε χρήστη του συστήματος, με τα οποία ο χρήστης να έχει υψηλή συνάφεια και τα οποία να μπορεί να τα προσπελάσει γρήγορα (μεγάλο Quality Of Experience). Θεωρούμε ότι κάθε χρήστης $u \in U$ έχει μια παράμετρο $0 \leq \beta_u \leq 1$, η οποία σχετίζεται με το πόσο σημαντικό είναι για ένα χρήστη να παίρνει περιεχόμενο με το οποίο να έχει υψηλή συνάφεια σε σχέση με το να το παίρνει σε μικρό χρόνο και αντίστροφα (μία ανάλογη χρήση της παραμέτρου β_u γίνεται κι εδώ [33]). Όσο πιο μεγάλη είναι αυτή η παράμετρος τόσο πιο σημαντική είναι η συνάφεια για το χρήστη έναντι της γρήγορης προσπέλασης, ενώ όσο μικραίνει, η προσπέλαση γίνεται πιο σημαντική συγκριτικά με τη συνάφεια. Σκοπός είναι να κάνουμε προτάσεις στους χρήστες έτσι ώστε να μεγιστοποιηθεί το μέσο Quality of Experience για το σύστημα.

Η συνάφεια ενός χρήστη $u \in U$ με ένα αντικείμενο $i \in I$ ισούται με το $sim(u,i)$. Τον χρόνο προσπέλασης του αντικειμένου από το χρήστη μπορούμε να τον δούμε ως το εκτιμώμενο inter-contact time του χρήστη με τον Clusterhead που περιέχει το αντικείμενο αυτό. Υπενθυμίζουμε ότι οι χρήστες μπορούν να προσπελάσουν αντικείμενα και από άλλους Clusterheads εκτός από τον δικό τους, αλλά προφανώς επειδή έχουν μικρότερο inter-contact time με τον δικό τους Clusterhead, τα αντικείμενα που είναι αποθηκευμένα σε αυτόν τα προσπελάζουν γρηγορότερα. Επίσης, όπως έχουμε αναφέρει νωρίτερα, αντικείμενα τα οποία δεν είναι αποθηκευμένα σε κανέναν Clusterhead, οι χρήστες μπορούν να τα προσπελάσουν μέσω ενός σταθερού Base Station χωρίς μνήμη, ο οποίος επικοινωνεί με το core network και προσπελάζει τα αντικείμενα αυτά. Θεωρούμε ότι το κόστος προσπέλασης C_{BS} του αντικειμένου στην περίπτωση αυτή, είναι αρκετά μεγαλύτερο από το μέγιστο εκτιμώμενο inter-contact time μεταξύ δύο χρηστών στο γράφο G_u. Συμβολίζουμε: $C_{BS}\ (>> max_{u,v \in V(G_u)} w(u,v))$.

Διατυπώνουμε τον ακόλουθο αλγόριθμο. Ως ορίσματα δέχεται το σύνολο των χρηστών U, το σύνολο των αντικειμένων I, το πλήθος R των προτάσεων που επιθυμούμε να κάνουμε στους χρήστες, την παράμετρο β_u, $\forall u \in U$, τα βάρη w του γράφου

G_u που αναπαριστούν το εκτιμώμενο inter-contact time μεταξύ των χρηστών, το χρόνο προσπέλασης C_{BS} αντικειμένου από το core network, τον πίνακα T ο οποίος περιέχει ως πληροφορία τον Clusterhead που είναι αποθηκευμένο κάθε αντικείμενο και τέλος τη συνάφεια μεταξύ κάθε χρήστη με κάθε αντικείμενο $sim(u,i)$, $\forall u \in U, \forall i \in I$:

Recommender $(U, I, R, \beta_u, w, C_{BS}, T, sim(u,i))$

$QOE[u][i] \leftarrow 0, \ \forall u \in U, \forall i \in I$
$recommendations[u][r] \leftarrow 0, \ \forall u \in U, \forall r \in [1, R]$
$mean\_QOE_u[u] \leftarrow 0, \ \forall u \in U$
$mean\_QOE \leftarrow 0$
Για κάθε $u \in U$
  Για κάθε $i \in I$
    Αν $T[i] \neq -1$ **τότε**
      $QOE[u][i] = \beta_u sim(u,i) + (1-\beta_u)(1 - \frac{w(u,T[i]) - w_{min}}{C_{BS} - w_{min}})$
    αλλιώς
      $QOE[u][i] = \beta_u sim(u,i)$
Για κάθε $u \in U$
  Ταξινόμηση του πίνακα $QOE[u]$ σε φθίνουσα σειρά
  $recommendations[u] \leftarrow$ πρώτοι R δείκτες που οδηγούν στην ταξινόμηση του $QOE[u]$ σε φθίνουσα σειρά
  $mean\_QOE_u[u] \leftarrow \frac{\sum_{i=1}^{R} QOE[u][i]}{R}$
$mean\_QOE \leftarrow \frac{\sum_{u \in U} mean\_QOE_u[u]}{|U|}$
Επιστροφή των πινάκων $recommendations$, $mean\_QOE_u$ και του μέσου QOE του συστήματος $mean\_QOE$

Algorithm 5.1: Αλγόριθμος συστάσεων στους χρήστες

Όπως αναφέραμε και παραπάνω, το Quality of Experience για κάθε χρήστη εξαρτάται από τη συνάφεια που έχει με κάθε αντικείμενο που του προτείνουμε και από το πόσο γρήγορα μπορεί να το προσπελάσει. Για κάθε $u \in U$ και για κάθε $i \in I$, ορίζουμε, όπως φαίνεται στον αλγόριθμο 5.1, ένα δισδιάστατο πίνακα QOE, ο οποίος είναι κυρτός συνδυασμός της συνάφειας του χρήστη u ως προς το αντικείμενο i και του χρόνου προσπάλασης του αντικειμένου από αυτόν. Ο συντελεστής για τη συνάφεια είναι η παράμετρος β_u, που αναφέραμε στην πρώτη παράγραφο της ενότητας 5.5, ενώ ο συντελεστής για το χρόνο προσπέλασης είναι $1-\beta_u$, έτσι ώστε για μεγάλο β_u η συνάφεια να παίζει μεγαλύτερο ρόλο ενώ για μικρό β_u να είναι σημαντικότερος ο χρόνος προσπέλασης για το χρήστη u. Ωστόσο, επειδή η συνάφεια και η παράμετρος β_u ανήκουν στο διάστημα $[0,1]$, κανονικοποιούμε και το χρόνο προσπέλασης ώστε να ανήκει στην ίδια τάξη μεγέθους με τις υπόλοιπες ποσότητες με τον εξής τρόπο: $\frac{w(u,T[i]) - w_{min}}{C_{BS} - w_{min}}$, αν $T[i] \neq -1$, όπου $w_{min} = min_{u,v \in U} w(u,v)$ και θεωρώντας ως μέγιστο βάρος, το μέγιστο δυνατό χρόνο προσπέλασης αντικειμένου από χρήστη, ο οποίος συμβαίνει όταν προσπελάζεται αντικείμενο από το core network, δηλαδή C_{BS}.

Παράλληλα, θεωρούμε ότι αν το αντικείμενο προσπελάζεται από το core network, η ποιότητα ως προς το χρόνο είναι η χειρότερη δυνατή οπότε δεν την εξετάζουμε καν, παρά μόνο τη συνάφεια του χρήστη με το αντικείμενο. Τέλος, επειδή μικρό inter-contact time μεταξύ δύο χρηστών αντιστοιχεί σε γρηγορότερο χρόνο προσπέλασης αντικειμένου του ενός χρήστη από τον άλλο, παίρνουμε στην έκφραση του QOE τον κανονικοποιημένο χρόνο προσπέλασης τον οποίο αφαιρούμε από τη μονάδα. Οπότε διακρίνουμε δύο περιπτώσεις:

Για κάθε χρήστη $u \in U$ και για κάθε αντικείμενο $i \in I$:

- Αν το αντικείμενο είναι αποθηκευμένο σε κάποιον Clusterhead (το στοιχείο $T[i]$ δεν είναι -1):

$$QOE[u][i] = \beta_u sim(u,i) + (1-\beta_u)(1 - \frac{w(u,T[i]) - w_{min}}{C_{BS} - w_{min}})$$

  δηλαδή ο χρήστης προσπελάζει το αντικείμενο από τον Clusterhead, ο οποίος αντιστοιχεί στην τιμή του $T[i]$

- Διαφορετικά: Θεωρούμε ότι ο χρήστης προσπελάζει το αντικείμενο από το κεντρικό δίκτυο με το χειρότερο δυνατό χρόνο προσπέλασης (άρα όπως είπαμε τον παραλείπουμε και δε συνεισφέρει στο QOE):

$$QOE[u][i] = \beta_u sim(u,i)$$

Αφού συμπληρώσουμε τον πίνακα, κατά τον τρόπο που αναφέραμε παραπάνω, για κάθε χρήστη $u \in U$, ταξινομούμε τον πίνακα $QOE[u]$, ο οποίος περιέχει την τιμή QOE του χρήστη για κάθε αντικείμενο, σε φθίνουσα σειρά από τις μέγιστες προς τις ελάχιστες τιμές και αποθηκεύουμε σε πίνακα $recommendations[u]$ τα R αντικείμενα που αντιστοιχούν στις πρώτες R μέγιστες τιμές της φθίνουσας ταξινόμησης του $QOE[u]$. Τα αντικείμενα αυτά που περιέχονται στον πίνακα $recommendations[u]$, θα είναι και αυτά που θα προτείνουμε σε κάθε χρήστη $u \in U$. Υπολογίζουμε, έπειτα, το μέσο Quality Of Experience του κάθε χρήστη ($mean\_QOE_u[u]$) ως προς τα R αντικείμενα που επιλέξαμε για αυτόν καθώς και το μέσο Quality Of Experience ($mean\_QOE$) για όλο το σύστημα ως το μέσο όρο των μέσων Quality Of Experience των χρηστών. Επιστρέφουμε τον πίνακα $recommendations$, τον πίνακα $mean\_QOE_u[u]$ καθώς και το μέσο Quality Of Experience για όλο το σύστημα $mean\_QOE$.

5.6 Αλγόριθμος Συστήματος

Έχοντας αναφέρει όλα τα προηγούμενα διατυπώνουμε παρακάτω το συνολικό αλγόριθμο 5.2 για το σύστημά μας σε μορφή ψευδοκώδικα, καλώντας τους αλγορίθμους που αναφέραμε στα κεφάλαια 2 και 5. Ο αλγόριθμος αυτός δέχεται ως ορίσματα το

σύνολο των χρηστών U, το σύνολο των αντικειμένων I, το σύνολο των μερών ενδιαφέροντος O, το πλήθος l των κατηγοριών στις οποίες υπάγονται τα αντικείμενα, το πλήθος R των προτάσεων που επιθυμούμε να κάνουμε στους χρήστες, το πλήθος k των Clusterheads που θέλουμε να έχει το σύστημά μας, τις χωρητικότητες του αποθηκευτικού χώρου των συσκευών των χρηστών C_u, τα μεγέθη των αντικειμένων s_i, την παράμετρο β_u, $\forall u \in U$, το χρόνο προσπέλασης C_{BS} αντικειμένου από το core network και τις παραμέτρους ε_1 και ε_2 που χρησιμοποιούνται ως ορίσματα στο Non-Metric-k-Median Problem και στο Generalized Assignment Problem αντίστοιχα. Ο αλγόριθμος προτείνει R αντικείμενα (αυτά που επιλέχθηκαν) σε κάθε χρήστη. Είναι σημαντικό να αναφέρουμε εδώ ότι θεωρούμε ότι οι χρήστες μπορούν να προσπελάσουν κάποιο αντικείμενο που τους προτείνεται εντός του χρονικού βήματος της (πιθανοτικής) συνάντησής τους με τον Clusterhead που περιέχει το αντικείμενο αυτό. Το μέσο Quality Of Experience του συστήματος ($mean\_QOE$) θα αναπαραστήσουμε γραφικά παρακάτω για διαφορετικούς συνδυασμούς παραμέτρων ώστε να αξιολογήσουμε το μοντέλο που αναπτύξαμε.

System Algorithm $(U, I, O, l, R, k, C_u, s_i, \beta_u, C_{BS}, \varepsilon_1, \varepsilon_2)$

Υπολογισμός $s(u,i)$, $\forall u \in U, \forall i \in I$ μέσω των f_u, f_i
Υπολογισμός $s(u,o)$, $\forall u \in U, \forall o \in O$ μέσω των f_u, f_o
Υπολογισμός πίνακα μετάβασης P_u, $\forall u \in U$ μέσω της 5.4
Υπολογισμός στάσιμης κατανομής π_u, $\forall u \in U$ μέσω των P_u
$w(u,v) \leftarrow \frac{1-\pi_u \cdot \pi_v^\intercal}{\pi_u \cdot \pi_v^\intercal}$
Αν ισχύει η τριγωνική ανισότητα για τα βάρη w **τότε**
    $CH, H \leftarrow Metric\_k\_Median(U, U, w, k)$
αλλιώς
    Υπολογισμός fractional λύσης d για το LP-relaxed k-median
    $CH, H \leftarrow Non\_Metric\_k\_median(U, U, w, d, \varepsilon_1)$
$v_{ij} \leftarrow \sum_{u:H[u]=j} sim(u,i)$
$T \leftarrow GAP(I, CH, v_{ij}, s_i, C_u, \varepsilon_2)$
$recommendations, mean\_QOE\_u, mean\_QOE \leftarrow$
        $Recommender(U, I, R, \beta_u, w, C_{BS}, T, sim(u,i))$
Προτάσεις $recommendations[u]$, $\forall u \in U$

Algorithm 5.2: Αλγόριθμος Συστήματος

Κεφάλαιο 6

Αξιολόγηση Συστήματος μέσω Προσομοίωσης

Στο κεφάλαιο αυτό, παρουσιάζονται τα αποτελέσματα της αξιολόγησης του μοντέλου, που αναπτύξαμε στα προηγούμενα κεφάλαια, μέσω προσομοίωσης σε συνθετικά δεδομένα. Συγκεκριμένα, για διαφορετικό συνδυασμό στις τιμές των παραμέτρων που επηρεάζουν το σύστημα, παρατηρούμε τη μεταβολή του μέσου QOE (Quality Of Experience) που προσφέρει το σύστημα στους χρήστες, δηλαδή της συνάφειας που έχουν τα προτεινόμενα αντικείμενα με τους χρήστες και του χρόνου προσπέλασης τους από αυτούς (είτε από τον Clusterhead που είναι αποθηκευμένα είτε από το core network). Παράλληλα εξετάζουμε και άλλες δύο μετρικές, το μέσο QOR (Quality Of Recommendations) και το μέσο QOS (Quality Of Service), ο κυρτός συνδυασμός των οποίων συνιστά το QOE. Το QOR εκφράζει τη μέση συνάφεια των προτεινόμενων αντικειμένων με τους χρήστες και το QOS το μέσο χρόνο προσπέλασής τους από αυτούς. Εκτός από την απόδοση του συστήματός μας ως προς τις μετρικές αυτές, παρέχουμε και τις βέλτιστες δυνατές τιμές των μετρικών αυτών, οι οποίες προκύπτουν μέσω προτάσεων σε κάθε χρήστη των αντικειμένων με τα οποία έχει τη μέγιστη συνάφεια και θεωρώντας ότι μπορεί να τα προσπελάσει στο μικρότερο δυνατό χρόνο, δηλαδή από τον Clusterhead με τον οποίο έχει το μικρότερο expected inter-contact time. Αναπαριστούμε το μέσο όρο των βέλτιστων τιμών των χρηστών για κάθε μετρική, ώστε να αξιολογήσουμε την απόδοση του αλγορίθμου μας σε σύγκριση με τη βέλτιστη δυνατή απόδοση.

Γενικά για το Quality Of Experience ισχύει:

$$QOE_{ui} = \beta_u \cdot QOR_{ui} + (1 - \beta_u) \cdot QOS_{ui}, \quad \forall u \in U, \forall i \in I$$

όπου β_u ενδεικτική παράμετρος του κάθε χρήστη ως προς το πόσο σημαντική είναι για αυτόν η συνάφεια, έναντι του χρόνου προσπέλασης ως προς τα προτεινόμενα αντικείμενα, όπως αναφέραμε και στο κεφάλαιο 5. Οπότε οι μετρικές του συστήματος ορίζονται όπως φαίνεται παρακάτω:

Για κάθε χρήστη $u \in U$ και για κάθε αντικείμενο $i \in I$:

- Αν το αντικείμενο i είναι αποθηκευμένο σε κάποιον από τους επιλεγμένους Clusterheads:
  - $QOE_{ui} = \beta_u sim(u,i) + (1-\beta_u)(1 - \frac{w(u,T[i]) - w_{min}}{C_{BS} - w_{min}})$
  - $QOR_{ui} = sim(u,i)$
  - $QOS_{ui} = 1 - \frac{w(u,T[i]) - w_{min}}{C_{BS} - w_{min}}$
- Αν το αντικείμενο i δεν είναι αποθηκευμένο σε κανέναν από τους επιλεγμένους Clusterheads (οπότε ο χρήστης το προσπελάζει από το core network μέσω Base Station):
  - $QOE_{ui} = \beta_u sim(u,i)$
  - $QOR_{ui} = sim(u,i)$
  - $QOS_{ui} = 0$

Από τις παραπάνω εκφράσεις βλέπουμε ότι οι μετρικές αυτές παίρνουν τιμές στο διάστημα $[0,1]$, οπότε η καλύτερη δυνατή τιμή που μπορούν να πάρουν είναι 1. Ο τρόπος μέσω του οποίου προκύπτει ο τύπος για το QOE εξηγείται αναλυτικά στο κεφάλαιο 5. Υπενθυμίζουμε επίσης, ότι ο χρόνος προσπέλασης αντικειμένου από το κεντρικό δίκτυο, θεωρείται ο χειρότερος δυνατός, οπότε αφού η ποιότητα εξυπηρέτησης αποτελεί μετρική μεγιστοποίησης, παίρνουμε το χρόνο, σε αυτή την περίπτωση, μηδενικό ($QOS = 0$).

Στην προσομοίωση, τα γραφικά αποτελέσματα της οποίας παρατίθενται στις παρακάτω ενότητες, εξετάζουμε πως επιδρά το πλήθος των χρηστών του δικτύου, το πλήθος των αντικειμένων, το πλήθος των θεματικών κατηγοριών στις οποίες υπάγονται τα αντικείμενα, το πλήθος των προτεινόμενων αντικειμένων προς τους χρήστες, το πλήθος των Clusterheads που επιλέγονται καθώς και το μέγεθος της cache του κάθε Clusterhead, στο προτεινόμενο σύστημα. Συγκεκριμένα, μελετάμε τη συμπεριφορά του αλγορίθμου 5.2 ως προς τη μεταβολή των μετρικών QOE, QOR, QOS που αναφέραμε παραπάνω, για τις διάφορες τιμές των παραμέτρων. Τα αποτελέσματα αφορούν instances στα οποία η τριγωνική ανισότητα ισχύει για τα βάρη του γράφου G_u των χρηστών, τα οποία αναπαριστούν τον εκτιμώμενο χρόνο που μεσολαβεί μεταξύ των διαδοχικών συναντήσεων των χρηστών. Οπότε στην περίπτωση αυτή, όπως έχουμε αναφέρει και στα προηγούμενα κεφάλαια, χρησιμοποιείται ο αλγόριθμος για το metric-k-median πρόβλημα 2.2 και οι Clusterheads που επιλέγονται είναι όσοι ορίζουμε εμείς μέσω της αντίστοιχης παραμέτρου.

Επίσης, εξετάζουμε τη συμπεριφορά του αλγορίθμου μας και στην περίπτωση που δεν ισχύει η τριγωνική ανισότητα για τα βάρη του γράφου G_u των χρηστών. Στην περίπτωση αυτή χρησιμοποιείται ο αλγόριθμος για το non-metric-k-median πρόβλημα 2.3 και οι Clusterheads που επιλέγονται είναι το πολύ $k \cdot ln(n + \frac{n}{\varepsilon})$, όπου k το πλήθος

των Clusterheads που ορίζουμε ως παράμετρο. Στην περίπτωση αυτή εξετάζουμε τη μεταβολή των μετρικών QOE, QOR, QOS για διαφορετικές τιμές της παραμέτρου k, καθώς και το πλήθος των Clusterheads που επιλέγει τελικά το σύστημα.

Η προσομοίωση πραγματοποιήθηκε σε γλώσσα Python.

6.1 Σύνολο Δεδομένων και Προεπιλεγμένες Παράμετροι

Για την πειραματική αξιολόγηση του μοντέλου, χρησιμοποιήσαμε συνθετικά σύνολα δεδομένων, θεωρώντας ότι τα διανύσματα χαρακτηριστικών των χρηστών, των αντικειμένων και των μερών ως προς τις θεματικές κατηγορίες του συστήματος παράγονται μέσω ομοιόμορφης κατανομής. Επίσης, τα μεγέθη των αντικειμένων s_i και η κατανομή των μερών στο χώρο προκύπτουν και αυτά μέσω ομοιόμορφης κατανομής στο διάστημα $[1, 10]$ τα μεν και στο μοναδιαίο τετράγωνο $[0, 1]^2$ τα δε. Μέσω κατανομής προκύπτουν επίσης και οι παράμετροι β_u για κάθε χρήστη $u \in U$. Με αυτόν τον τρόπο, ενδεχομένως να καλύπτονται χαρακτηριστικά που παρατηρούνται σε ρεαλιστικές συνθήκες (π.χ. biased users), αλλά αναδεικνύεται η προσαρμοστικότητα του μοντέλου ως προς την απόδοση για διαφορετικές περιπτώσεις συστημάτων. Ωστόσο για να μελετήσουμε τη συμπεριφορά του αλγορίθμου στατιστικά, επαναλάβαμε το πείραμα 5 φορές για κάθε συνδυασμό παραμέτρων που εξετάζαμε και πήραμε τη μέση απόδοση που προέκυψε για κάθέναν από αυτούς. Σε όλες τις περιπτώσεις θεωρούμε σταθερό γράφο μερών G_o με 50 κορυφές, σταθερά ε_1 για το non-metric k-median ίση με 0.5 και παράμετρο λάθους για το Generalized Assignment Problem $\varepsilon_2 = 0.1$. Ο χρόνος προσπέλασης αντικειμένων από το core network θεωρούμε ότι είναι πενταπλάσιος από το μέγιστο (χειρότερο) expected inter-contact time μεταξύ δύο χρηστών που απαντάται σε κάθε instance του συστήματος.

Για να διαπιστώσουμε τη συμπεριφορά του αλγορίθμου καθώς μεταβάλλονται οι τιμές κάθε μίας από τις παραμέτρους που απαριθμήσαμε παραπάνω, κρατάμε τις τιμές των υπόλοιπων παραμέτρων σταθερές, μεταβάλλοντας κάθε φορά τις τιμές αυτής που θέλουμε να εξετάσουμε. Παρακάτω βλέπουμε τις default παραμέτρους του πειράματος:

- Πλήθος Χρηστών $|U| = 100$
- Πλήθος Αντικειμένων $|I| = 500$
- Πλήθος θεματικών Κατηγοριών $l = 6$
- Πλήθος Recommendations $R = 6$
- Πλήθος Clusterheads $k = 3$
- Μέγεθος Clusterhead Cache $C_u = 5 \cdot S_{average}$, όπου $S_{average} = \frac{\sum_{i \in I} s_i}{|I|}$, δηλαδή 5 φορές το μέσο μέγεθος των αντικειμένων

6.2 Πλήθος Χρηστών (Number Of Users)

Η πρώτη παράμετρος που μεταβάλουμε ώστε να παρατηρήσουμε τη συμπεριφορά του αλγορίθμου μας, είναι το μέγεθος του συνόλου των χρηστών U. Θεωρούμε ότι οι υπόλοιπες παράμετροι έχουν τις default τιμές που αναφέραμε στην προηγούμενη ενότητα.

Εξετάζουμε το μοντέλο μας για τα παρακάτω πλήθη χρηστών:

- $|U| = 100, 150, 200, 250, 300$

Παρακάτω έχουμε τις γραφικές αναπαραστάσεις των μετρικών QOE, QOR, QOS του συστήματος για τα διάφορα πλήθη χρηστών:

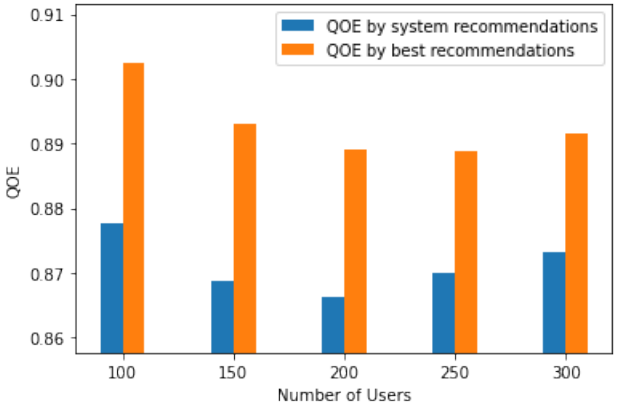
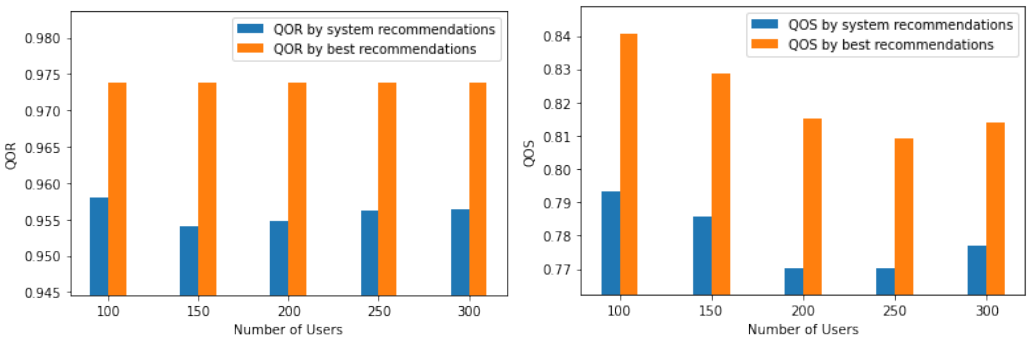


Figure 6.1: Change in Number Of Users: QOE



(a) QOR (b) QOS

Figure 6.2: Change in Number Of Users: QOR and QOS

Παρατηρώντας τα τρία παραπάνω διαγράμματα, τα οποία αναπαριστούν τις τιμές που παίρνουν οι μετρικές του Quality Of Experience (QOE), του Quality Of Recommendations και του Quality Of Service (QOS) για διαφορετικές τιμές του πλήθους

των χρηστών του συστήματος, βλέπουμε ότι οι τιμές που δίνει για τις μετρικές ο αλγόριθμός μας (μπλε) είναι πολύ κοντά στις βέλτιστες μέσες τιμές των μετρικών (πορτοκαλί), γεγονός το οποίο οφείλεται στις καλές προσεγγίσεις των αλγορίθμων που χρησιμοποιούμε, ενώ παράλληλα ακολουθούν τη μονοτονία τους. Ωστόσο, δεν παρατηρούμε αύξουσα ή φθίνουσα συμπεριφορά των τιμών αυτών με την αύξηση του πλήθους των χρηστών του συστήματος. Αυτό συμβαίνει γιατί για κάθε πλήθος χρηστών, τα διανύσματα χαρακτηριστικών των χρηστών ως προς τις κατηγορίες αλλάζουν και κατά συνέπεια αλλάζουν όλες οι ποσότητες του συστήματος που επηρεάζουν τη συμπεριφορά των μετρικών QOE, QOR, QOS (η συνάφεια των χρηστών προς τα αντικείμενα, το expected inter-contact time μεταξύ των χρηστών, κ.λπ), κάνοντας τα instances του συστήματος για διαφορετικό πλήθος χρηστών μη άμεσα συγκρίσιμα. Το γεγονός ότι επαναλάμβάνουμε το πείραμα περισσότερες φορές (5 φορές για 100 χρήστες, 5 για 150, κ.ο.κ) και υπολογίζουμε το μέσο όρο των τιμών των μετρικών για καθένα από τα πλήθη χρηστών, δεν μπορεί να οδηγήσει σε ένα πιο ξεκάθαρο στατιστικό αποτέλεσμα γιατί οι δυνατές κατανομές των διανυσμάτων των χρηστών είναι άπειρες. Παρ'όλα αυτά βλέπουμε ότι τα αποτελέσματα του αλγορίθμου μας ακολουθούν την ίδια μονοτονία με τις βέλτιστες τιμές σε όλα τα διαγράμματα και για όλα τα πλήθη χρηστών, γεγονός που υποδεικνύει το σωστό τρόπο υλοποίησης και λειτουργίας του αλγορίθμου μας. Επίσης αξίζει να παρατηρήσουμε το πόσο κοντά είναι για τα διάφορα πλήθη χρηστών όλες οι τιμές (και του αλγορίθμου μας και οι βέλτιστες), το οποίο δείχνει ότι για διαφορετικά πλήθη χρηστών, το σύστημα προσαρμόζεται και προβαίνει σε σύσταση πολύ ικανοποιητικού και γρήγορα προσπελάσιμου περιεχομένου.

6.3 Πλήθος Αντικειμένων (Number Of Items)

Στη συνέχεια, παραθέτουμε τα αποτελέσματα της προσομοίωσής μας, μεταβάλοντας το πλήθος των αντικειμένων του καταλόγου I του συστήματος, κρατώντας τις υπόλοιπες τιμές των παραμέτρων σταθερές.

Εξετάζουμε το μοντέλο μας για τα παρακάτω πλήθη αντικειμένων:

- $|I| = 500, 750, 1000, 1250, 1500$

Παρακάτω έχουμε τις γραφικές αναπαραστάσεις των μετρικών QOE, QOR, QOS του συστήματος για τα διάφορα πλήθη διαθέσιμων αντικειμένων:

Παρατηρούμε εδώ, ότι με την αύξηση του πλήθους των διαθέσιμων αντικειμένων για αποθήκευση και πρόταση, η μέση ποιότητα εμπειρίας (QOE) των χρηστών αυξάνεται. Αυτό είναι λογικό αφού έχοντας περισσότερα διαθέσιμα αντικείμενα, υπάρχει μεγαλύτερη πιθανότητα ένα υποσύνολο αυτών να καλύπτει καλύτερα τα γούστα των χρηστών του δικτύου και τα αντικείμενα που αποθηκεύονται και προτείνονται να είναι περισσότερο εξατομικευμένα ως προς κάθε χρήστη. Έτσι οι caches των Clusterheads μπορούν να συμπεριλάβουν συγκεκριμένα, πιο εύστοχα επιλεγμένα, αντικείμενα που να έχουν υψηλή συνάφεια με τους επιμέρους χρήστες της ομάδας τους (σύνολο των

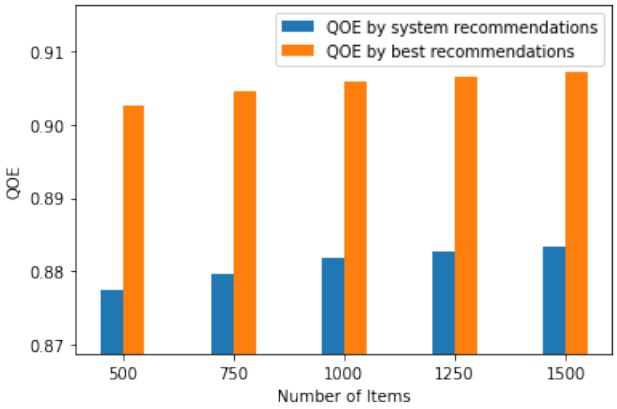
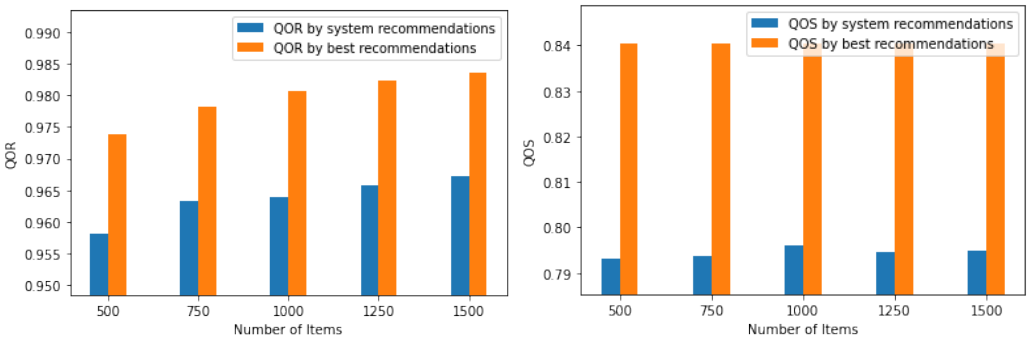


Figure 6.3: Change in Number Of Items: QOE



(a) QOR (b) QOS

Figure 6.4: Change in Number Of Items: QOR and QOS

χρηστών που έχουν ανατεθεί σε αυτούς). Βλέπουμε και πάλι τη μικρή απόκλιση μεταξύ των βέλτιστων τιμών των μετρικών σε σχέση με τις τιμές των μετρικών με εφαρμογή του αλγορίθμου μας και την ανάλογη συμπεριφορά τους ως προς τη μονοτονία.

Λαμβάνοντας υπόψιν τα παραπάνω, βλέπουμε ότι και η ποιότητα των συστάσεων QOR αυξάνεται με την αύξηση του πλήθους των διαθέσιμων αντικειμένων, αφού επιλέγονται προς αποθήκευση και πρόταση αντικείμενα πιο εξατομικευμένα ως προς τους χρήστες του συστήματος, όταν έχουμε μεγάλη γκάμα επιλογών. Αύξηση παρουσιάζουν και οι βέλτιστες τιμές των μετρικών, ενώ διαπιστώνουμε πάλι το πόσο κοντά είναι με τις τιμές που δίνει ο αλγόριθμος.

Την αύξηση που βλέπουμε στις προηγούμενα διαγράμματα, δεν την διαπιστώνουμε στην περίπτωση της ποιότητας εξυπηρέτησης (QOS). Αυτό συμβαίνει γιατί, παρόλο που μπορούμε να αποθηκεύσουμε πιο συναφή και εξατομικευμένα αντικείμενα στις caches των χρηστών και να κάνουμε καλύτερες προτάσεις, οι χρήστες συναντούν τους Clusterheads με την ίδια εκτιμώμενη καθυστέρηση. Οπότε παρόλο που ένας

Clusterhead μπορεί να έχει πιο relevant αντικείμενα ως προς τους χρήστες που εξυπηρετεί, αυτοί θα προσπελάζουν το περιεχόμενο με την ίδια καθυστέρηση, αφού το expected inter-contact time είναι το ίδιο. Οπότε βλέπουμε ότι η μετρική QOS μένει σχετικά αμετάβλητη ως προς τη μεταβολή του πλήθους αντικειμένων του συστήματος. Ίδια συμπεριφορά παρουσιάζουν και οι βέλτιστες τιμές της μετρικής.

Επομένως, η συμπεριφορά της μετρικής QOE επηρεάζεται κυρίως από τη συμπεριφορά της QOR, αφού η ποιότητα των συστάσεων αυξάνεται για τους χρήστες, χωρίς όμως να επηρεάζεται κατά μεγάλο βαθμό η ποιότητα εξυπηρέτησής τους από το δίκτυο.

6.4 Πλήθος Θεματικών Κατηγοριών (Number Of Categories)

Η επόμενη παράμετρος που εξετάζουμε ως προς το πώς η μεταβολή της επηρεάζει την απόδοση του μοντέλου, είναι το πλήθος των θεματικών κατηγοριών στο οποίο υπάγονται τα αντικείμενα του συστήματος. Θεωρούμε και πάλι ότι οι υπόλοιπες παράμετροι έχουν τις default τιμές τους.

Εξετάζουμε το μοντέλο μας για τα παρακάτω πλήθη θεματικών κατηφοριών:

- $l = 6, 8, 10, 12, 14$

Παρακάτω έχουμε τις γραφικές αναπαραστάσεις των μετρικών QOE, QOR, QOS του συστήματος για τα διάφορα πλήθη θεματικών κατηγοριών:

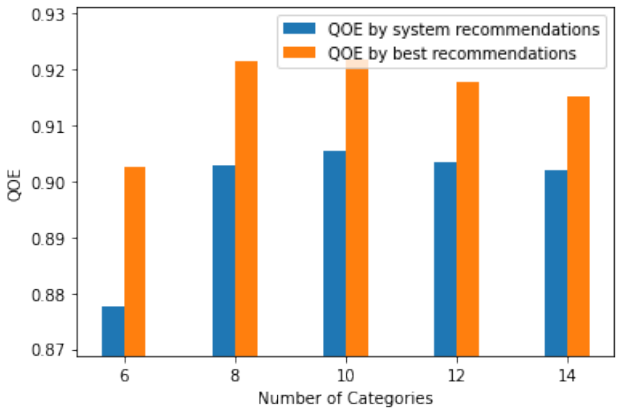
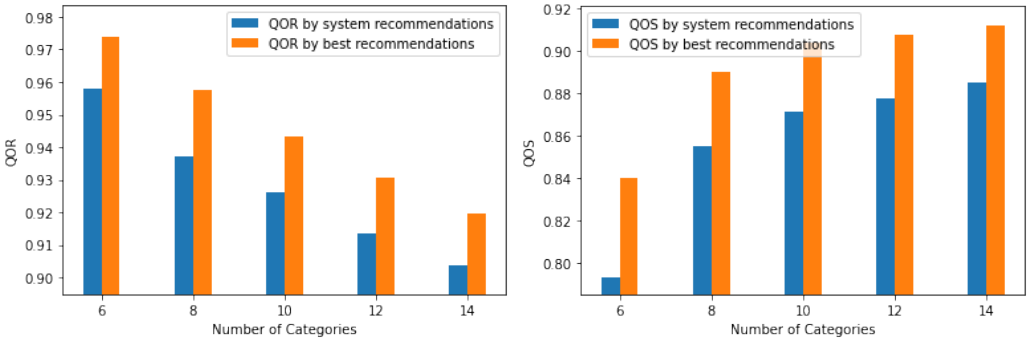


Figure 6.5: Change in Number Of Categories: QOE

Υπενθυμίζουμε ότι κάθε χρήστης, κάθε αντικείμενο και κάθε μέρος χαρακτηρίζονται από ένα διάνυσμα χαρακτηριστικών ως προς τις θεματικές κατηγορίες του συστήματος, το οποίο αντικατοπτρίζει τη συνάφεια του χρήστη, του αντικειμένου ή του μέρους ως προς την κάθε θεματική κατηγορία. Οπότε η αλλαγή στο πλήθος των



(a) QOR (b) QOS

Figure 6.6: Change in Number Of Categories: QOR and QOS

κατηγοριών αλλάζει όλα αυτά τα διανύσματα και επηρεάζει όλες τις ποσότητες που καθορίζουν την τιμή των μετρικών QOE, QOR, QOS. Συνεπώς, για τους λόγους που αναφέραμε και στην ενότητα 6.2, οι τιμές των μετρικών για διαφορετικά πλήθη κατηγοριών δεν είναι άμεσα συγκρίσιμες οπότε δεν μπορούμε να προβούμε σε ξεκάθαρες παρατηρήσεις. Ωστόσο, επιβεβαιώνεται κι εδώ η σωστή συμπεριφορά και η πολύ καλή απόδοση του αλγορίθμου αφού οι τιμές που προκύπτουν από την εφαρμογή στο σύστημα, του αλγορίθμου μας, είναι πολύ κοντά στις βέλτιστες και ακολουθούν τη μονοτονία τους.

6.5 Πλήθος Προτάσεων (Number Of Recommendations)

Στο σημείο αυτό, θεωρούμε ως παράμετρο που μεταβάλλεται, το πλήθος των προτάσεων αντικειμένων που κάνουμε στους χρήστες του συστήματος (κάθε χρήστης λαμβάνει το ίδιο πλήθος συστάσεων σε σχέση με τους υπόλοιπους). Οι υπόλοιπες παράμετροι έχουν τις προεπιλεγμένες τιμές που απαριθμίζονται στην ενότητα 6.1.

Εξετάζουμε το μοντέλο μας για τα παρακάτω πλήθη προτάσεων:

- $R = 6, 8, 10, 12, 14$

Παρακάτω έχουμε τις γραφικές αναπαραστάσεις των μετρικών QOE, QOR, QOS του συστήματος για τα διάφορα πλήθη προτάσεων :

Παρατηρούμε ότι το μέσο QOE που προσφέρει το σύστημα στους χρήστες μειώνεται, καθώς αυξάνουμε τα προτεινόμενα αντικείμενα. Ο αλγόριθμος προτείνει σε κάθε χρήστη τα R πρώτα αντικείμενα που μεγιστοποιούν το QOE του. Αυξάνοντας το πλήθος, προτείνουμε και αντικείμενα που βρίσκονται σε πιο χαμηλές θέσεις ως προς τη συνάφεια με το χρήστη και το χρόνο προσπέλασής τους από αυτόν, με αποτέλεσμα

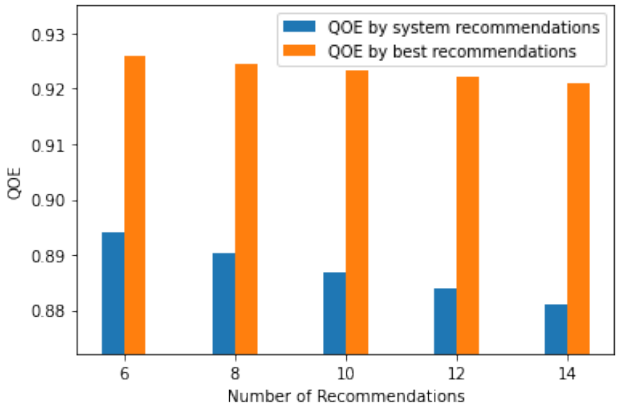
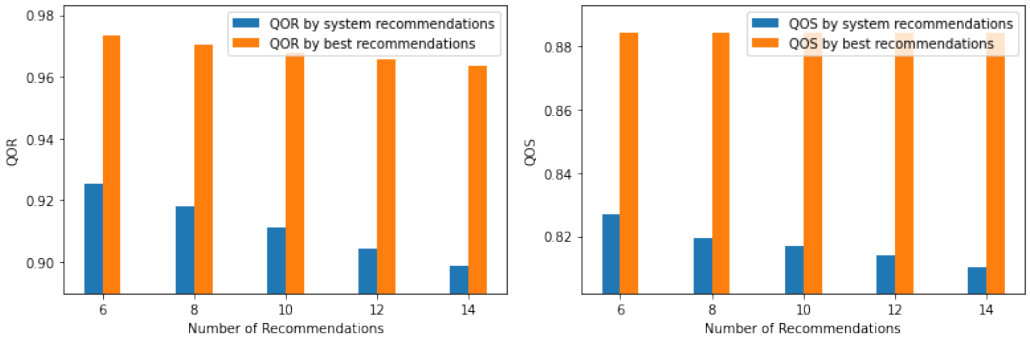


Figure 6.7: Change in Number Of Recommendations: QOE



(a) QOR (b) QOS

Figure 6.8: Change in Number Of Recommendations: QOR and QOS

το μέσο QOE για κάθε χρήστη να μειώνεται, οδηγώντας και το μέσο QOE όλων των χρηστών σε μείωση.

Μείωση βλέπουμε και στα άλλα δύο διαγράμματα που αναπαριστούν την ποιότητα των συστάσεων και την ποιότητα εξυπηρέτησης. Αυξάνοντας το πλήθος των προτάσεων, κάθε χρήστης δέχεται και συστάσεις περιεχομένου το οποίο πιθανότατα δεν βρίσκεται στον Clusterhead στον οποίο ο χρήστης έχει ανατεθεί και επομένως δεν έχει γίνει cached με κριτήριο τη συνάφεια που παρουσιάζει με αυτόν. Παράλληλα, όσο αυξάνονται οι προτάσεις, αυξάνονται και τα αντικείμενα που ο χρήστης προσπελάζει από άλλους Clusterheads, με τους οποίους έχει φυσικά χειρότερο expected inter-contact time. Αυτό εξηγείται άμεσα και από το γεγονός ότι η default τιμή μεγέθους της cache των Clusterheads είναι πενταπλάσια του μέσου μεγέθους των αντικειμένων, δηλαδή κατά μέσο όρο κάθε cache χωράει 5 αντικείμενα. Όταν εμείς προτείνουμε 12 ή 14 αντικείμενα (που είναι οι μεγαλύτερες τιμές της παραμέτρου που δοκιμάζουμε), προφανώς πολλά από αυτά οι χρήστες τα προσπελάζουν από άλλους Clusterheads και

όχι από τον δικό τους. Τα παραπάνω, έχουν ως αποτέλεσμα τόσο το QOR όσο και το QOS να μειώνονται. Παρόλα αυτά βλεπουμε πάλι μικρές αποκλίσεις και μεταξύ των τιμών των μετρικών που δίνει το μοντέλο και των βέλτιστων τιμών των μετρικών αυτών και μεταξύ των διάφορων τιμών του μοντέλου καθώς αυξάνεται το πλήθος των προτάσεων.

6.6 Πλήθος Clusterheads (Number Of Clusterheads)

Η επόμενη παράμετρος που μεταβάλλουμε είναι το πλήθος των Clusterheads που επιλέγει το σύστημα. Υπενθυμίζουμε ότι τα αποτελέσματα αφορούν metric χώρο, συνεπώς για την επιλογή των Clusterheads χρησιμοποιείται ο αλγόριθμος για το metric-k-median 2.2, συνεπώς επιλέγονται ακριβώς τόσοι Clusterheads όσοι υπαγορεύει το πλήθος των Clusterheads που θέτουμε ως παράμετρο. Θεωρούμε ότι οι υπόλοιπες παράμετροι λαμβάνουν τις default τιμές τους.

Εξετάζουμε το μοντέλο μας για τα παρακάτω πλήθη Clusterheads:

- $k = 3, 5, 7, 9, 11$

Παρακάτω έχουμε τις γραφικές αναπαραστάσεις των μετρικών QOE, QOR, QOS του συστήματος για τα διάφορα πλήθη Clusterheads:

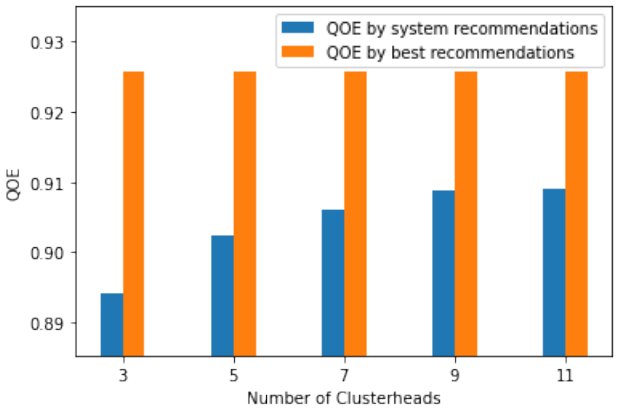
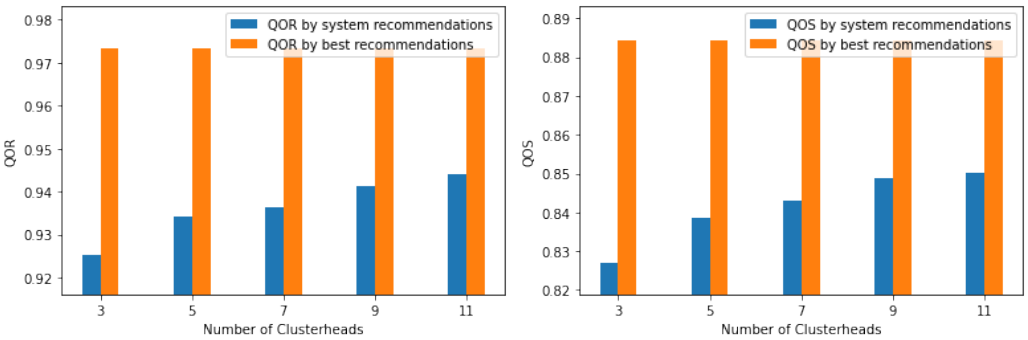


Figure 6.9: Change in Number Of Clusterheads (k): QOE



(a) QOR (b) QOS

Figure 6.10: Change in Number Of Clusterheads (k): QOR and QOS

Όπως ήταν αναμενόμενο, με την αύξηση του πλήθους των Clusterheads του συστήματός μας, η μέση ποιότητα εμπειρίας (QOE) αυξάνεται. Όταν έχουμε περισσότερους χρήστες στους οποίους μπορούμε να τοποθετήσουμε περιεχόμενο ενδιαφέροντος, ενώ παράλληλα το πλήθος των χρηστών του δικτύου που πρέπει να εξυπηρετήσουμε και των διαθέσιμων, προς αποθήκευση και πρόταση, αντικειμένων διατηρείται σταθερό, μπορούμε να πετύχουμε καλύτερη ποιότητα υπηρεσίας. Όσο αυξάνεται το πλήθος των Clusterheads, σε καθέναν από αυτούς αντιστοιχίζονται λιγότεροι χρήστες και γίνεται cached περιεχόμενο που είναι πιο εξατομικευμένο σε αυτούς, το οποίο μπορούν να το πρεσπελάσουν γρήγορα. Παράλληλα, ακόμα κι αν κάποια προτεινόμενα αντικείμενα δεν είναι αποθηκευμένα στον Clusterhead στον οποίο έχει ανατεθεί ένας χρήστης, επειδή συνολικά έχουμε περισσότερους Clusterheads αλλά με ίδιο μέγεθος cache, άρα και περισσότερα αντικείμενα αποθηκευμένα κοντά στους χρήστες, μπορεί ένας χρήστης να προσπελάσει περιεχόμενο από κάποιον άλλο Clusterhead γρήγορα και με το οποίο παρουσιάζει με μεγάλη πιθανότητα υψηλή συνάφεια. Τα παραπάνω δικαιολογούν την αύξηση που βλέπουμε στην ποιότητα των συστάσεων (QOR) και στην ποιότητα εξυπηρέτησης (QOS) που λαμβάνουν οι χρήστες και κατά συνέπεια στην ποιότητα εμπειρίας τους (QOE).

Λίγο πιο απότομη παρουσιάζεται η αύξηση στο QOS σε σχέση με το QOR, αφού η βελτιωμένη απόδοση που παρατηρείται στο σύστημα με την αύξηση του πλήθους των Clusterheads, οφείλεται κυρίως στην αποθήκευση περισσότερων (και με μεγάλη πιθανότητα πιο συναφών) αντικειμένων κοντά στους χρήστες, μειώνοντας το χρόνο προσπέλασης αντικειμένων. Έτσι, οι χρήστες επιβαρύνονται με μικρότερη πιθανότητα από τη μεγάλη καθυστέρηση που επιβάλει η προσπέλαση αντικειμένων από το κεντρικό δίκτυο, όταν αυτά δεν είναι αποθηκευμένα σε κάποιον Clusterhead.

Σε όλα τα διαγράμματα, ωστόσο, βλέπουμε ότι όσο αυξάνεται το πλήθος των Clusterheads μειώνεται η αύξηση των τιμών των μετρικών, γεγονός που οφείλεται στο ότι για ένα μικρό πλήθος χρηστών (default 100) καθοριστική επιρροή στο σύστημα μπορεί να έχει η μεταβολή μιας παραμέτρου εντός διαστήματος μικρών τιμών.

Δηλαδή για 9 Clusterheads έχουμε τόσο καλές τιμές, που για 11, παρότι βελτιώνεται η απόδοση του συστήματος, δεν σημειώνεται πολύ μεγάλη πρόοδοες. Σημειώνουμε ότι κι εδώ βλέπουμε μικρές αποκλίσεις και μεταξύ των τιμών των μετρικών που δίνει το μοντέλο και των βέλτιστων τιμών των μετρικών αυτών και μεταξύ των διάφορων τιμών του μοντέλου καθώς αυξάνεται το πλήθος των Clusterheads.

6.7 Μέγεθος Clusterhead Cache (Size Of Clusterhead Cache)

Συνεχίζουμε με την παρατήρηση της επίδρασης που σημειώνει η μεταβολή του μεγέθους της cache των Clusterheads στην απόδοση του συστήματος. Κρατάμε τις υπόλοιπες παραμέτρους σταθερές στις default τιμές τους.

Εξετάζουμε το μοντέλο μας για τους παρακάτω λόγους του μεγέθους μιας cache ως προς το μέσο μέγεθος των αντικειμένων:

- $C_u/S_{average} = 5, 10, 15, 20, 25$

Παρακάτω έχουμε τις γραφικές αναπαραστάσεις των μετρικών QOE, QOR, QOS του συστήματος για τα διάφορα μεγέθη cache.

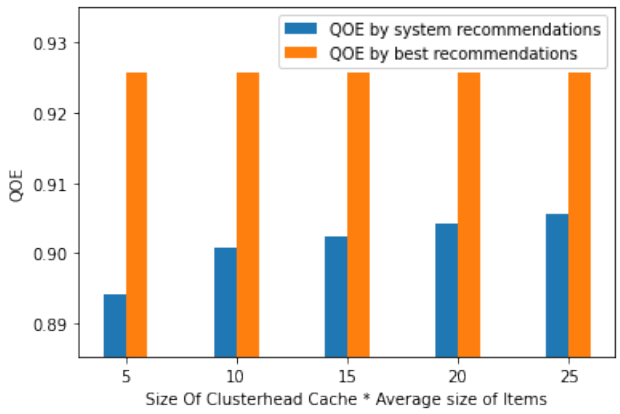
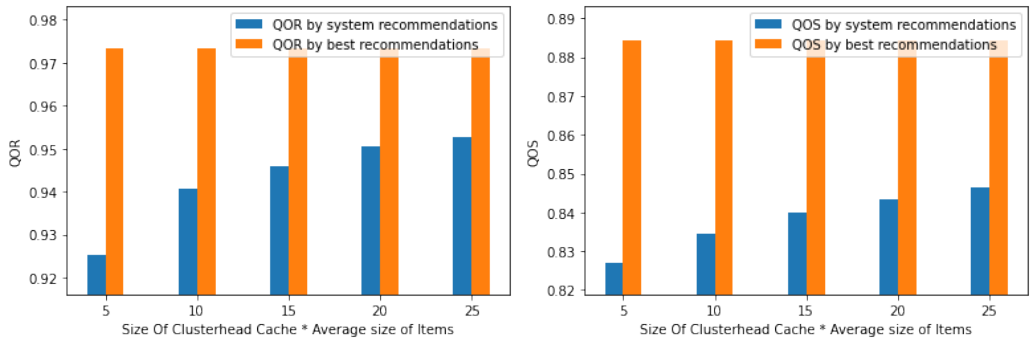


Figure 6.11: Change in Size Of Clusterhead Cache: QOE

Από τα διαγράμματα, παρατηρούμε αύξηση στην ποιότητα υπηρεσίας (QOE) που προσφέρει το σύστημα, καθώς αυξάνεται το μέγεθος των cache των επιλεγμένων Clusterheads. Όταν αυξάνεται το μέγεθος του αποθηκευτικού χώρου, αυξάνεται και το πλήθος των αντικειμένων που μπορούν να αποθηκευτούν στους Clusterheads, οπότε σε κάθε Clusterhead μπορούν να αποθηκευτούν αντικείμενα ανταποκρινόμενα σε περισσότερα ενδιαφέροντα και με υψηλή συνάφεια προς περισσότερους χρήστες. Παράλληλα, όπως και στην προηγούμενη ενότητα 6.6, περισσότερα αντικείμενα αποθηκεύονται σε Clusterheads, δηλαδή κοντά σε χρήστες, οπότε τέτοια αντικείμενα που



(a) QOR (b) QOS

Figure 6.12: Change in Size Of Clusterhead Cache: QOR and QOS

προσπελάζονται σε μικρό χρόνο (και με υψηλή συνάφεια όπως αναφέραμε παραπάνω), θα είναι αυτά που θα προταθούν και στους χρήστες, αποφεύγοντας έτσι τους μεγάλους χρόνους προσπέλασης αντικειμένων από το κεντρικό δίκτυο. Λόγω των παραπάνω, παρατηρούμε αύξηση και στις άλλες δύο μετρικές που συνιστούν το QOE, δηλαδή την ποιότητα συστάσεων (QOR) και την ποιότητα εξυπηρέτησης (QOS). Πάλι βλέπουμε μικρές αποκλίσεις και μεταξύ των τιμών των μετρικών που δίνει το μοντέλο και των βέλτιστων τιμών των μετρικών αυτών και μεταξύ των διάφορων τιμών του μοντέλου καθώς αυξάνεται το μέγεθος των caches.

6.8 Μη μετρικός χώρος

Τέλος, εξετάζουμε το προτεινόμενο μοντέλο στην περίπτωση όπου τα βάρη στο γράφο G_u των χρηστών, δηλαδή τα expected inter-contact times μεταξύ των χρηστών, δεν επαληθεύουν την τριγωνική ανισότητα. Στην περίπτωση αυτή χρησιμοποιούμε τον αλγόριθμο του non-metric-k-median προβλήματος 2.3 για την επιλογή των Clusterheads του συστήματος, το οποίο βρίσκει όμως το πολύ $k \cdot ln(n+\frac{n}{\varepsilon})$, όπου k το πλήθος των Clusterheads που ορίζουμε ως παράμετρο. Μελετάμε τη συμπεριφορά του συστήματος ως προς τη μεταβολή του πλήθους k των Clusterheads για instances στα οποία δεν ισχύει η τριγωνική ανισότητα, χρησιμοποιώντας για τις υπόλοιπες παραμέτρους τις default τιμές που χρησιμοποιοιήσαμε και στο μετρικό χώρο. Για οικονομία χρόνου, δεν εξετάζουμε τη συμπεριφορά του αλγορίθμου ως προς τη μεταβολή όλων των παραμέτρων όπως κάναμε στις προηγούμενες ενότητες , και για αυτά τα non-metric instances, καθώς είναι εύκολο να αντιληφθούμε ότι θα οδηγούμασταν σε ανάλογα συμπεράσματα. Περιοριζόμαστε στην εξέταση της απόδοσης του μοντέλου ως προς τη μεταβολή της παραμέτρου που σχετίζεται με το σημείο που διαφοροποιείται ο αλγόριθμος που χρησιμοποιούμε, δηλαδή το πλήθος των Clusterheads που επιλέγει το σύστημα.

Σημειώνουμε παράλληλα, ότι για 1000 επαναλήψεις παραγωγής των συνθετικών

δεδομένων που απαριθμήσαμε στην ενότητα 6.1 και για τις default τιμές των παραμέτρων του συστήματος, περίπου το 15% των instances δεν επαλήθευαν την τριγωνική ανισότητα. Αναφέρουμε, επίσης, ότι ο αλγόριθμος 2.3 για μέγεθος δικτύου $n = 100$ και σταθερά $\varepsilon_1 = 0.5$ μπορεί να επιλέξει έως και $k \cdot ln(300) \approx 5.7k$.

Εξετάζουμε τη συμπεριφορά του μοντέλου σε non-metric χώρο για τις παρακάτω τιμές της μεταβλητής k:

- $k = 3, 5, 7, 9, 11$

Παρακάτω έχουμε τις γραφικές αναπαραστάσεις των μετρικών QOE, QOR, QOS του συστήματος για τα διάφορα πλήθη Clusterheads που θέτουμε ως παράμετρο:

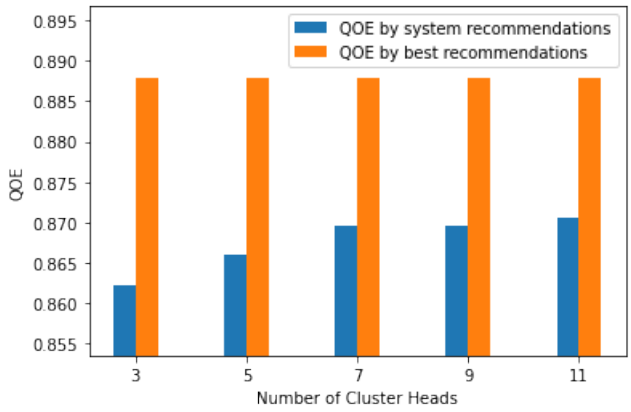
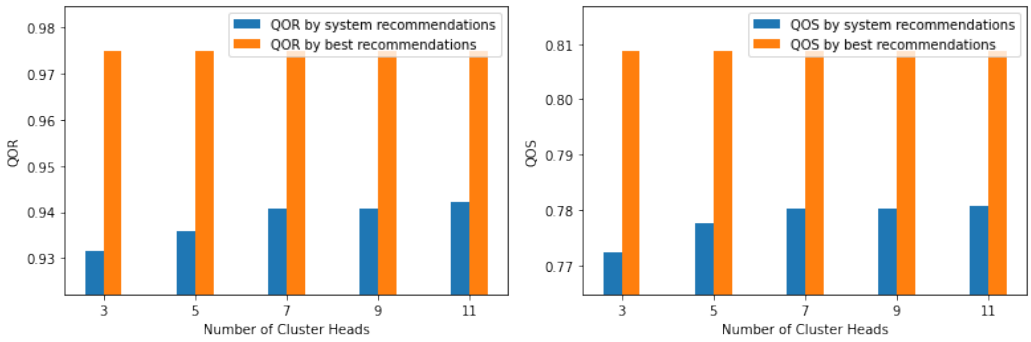


Figure 6.13: Change in Number Of Clusterheads (k) in Non-Metric Space: QOE



(a) QOR (b) QOS

Figure 6.14: Change in Number Of Clusterheads (k) in Non-Metric Space: QOR and QOS

***Σημείωση**: Στα παραπάνω διαγράμματα ο οριζόντιος άξονας αφορά τις τιμές της παραμέτρου k και όχι το πλήθος των Clusterheads που επιλέγει τελικά ο αλγόριθμος.*

Παραθέτουμε για κάθε τιμή k που θέσαμε ως παράμετρο, το πλήθος των Clusterheads που βρήκε ο αλγόριθμος:

- Για $k = 3$, Πλήθος επιλεγμένων Clusterheads $\rightarrow 3$
- Για $k = 5$, Πλήθος επιλεγμένων Clusterheads $\rightarrow 4$
- Για $k = 7$, Πλήθος επιλεγμένων Clusterheads $\rightarrow 6$
- Για $k = 9$, Πλήθος επιλεγμένων Clusterheads $\rightarrow 6$
- Για $k = 11$, Πλήθος επιλεγμένων Clusterheads $\rightarrow 7$

Από τα παραπάνω, βλέπουμε ότι καθώς το πλήθος των Clusterheads αυξάνεται, παρατηρείται αύξηση και στις τιμές των μετρικών QOE, QOR, QOS και γενικά καταλήγουμε στις ίδιες παρατηρήσεις και συμπεράσματα που διατυπώσαμε στην ενότητα 6.6, όπου εκεί μεταβάλαμε το πλήθος των Clusterheads σε μετρικό χώρο.

Αυτό που αξίζει ωστόσο να σχολιάσουμε είναι το πλήθος των Clusterheads το οποίο βρίσκει ο αλγόριθμος 2.3, για τις διάφορες παραμέτρους του k που θέτουμε. Παρατηρούμε ότι ο αλγόριθμος βρίσκει το ίδιο πλήθος Clusterheads με την τιμή του k που ορίζουμε, όταν το k που θέτουμε έιναι μικρό ($k = 3$). Αυτό συμβαίνει γιατί η απόδοση είναι ικανοποιητικά καλή για το σύστημα ακόμα και για μικρό πλήθος Clusterheads. Αυτό υποδεικνύεται με πιο άμεσο τρόπο και από το γεγονός ότι για μεγαλύτερες τιμές του $k(= 5, 7, 9, 11)$, ο αλγόριθμος όχι μόνο δε βρίσκει περισσότερους Clusterheads από την τιμή του k, αλλά βρίσκει λιγότερους, γιατί έχει ήδη φτάσει σε πολύ καλά επίπεδα απόδοσης και με την επιλογή λίγων Clusterheads, το κόστος που επιτυγχάνει διατηρείται εντός προσέγγισης. Μάλιστα βλέπουμε ότι για $k = 7, 9, 11$ το σύστημα έρχεται σε έναν "κορεσμό" ως προς την απόδοση που μπορεί να φτάσει με εφαρμογή του non-metric αλγορίθμου για το συγκεκριμένο πλήθος χρηστών, αφού παρότι αυξάνουμε το μέγεθος της παραμέτρου k, άρα και το πλήθος των Clusterheads που μπορεί να επιλέξει ο αλγόριθμος, με την επιλογή περισσότερων Clusterheads, δεν επιτυγχάνεται αισθητή βελτίωση στην απόδοση του μοντέλου ως προς τις τρεις μετρικές QOE, QOR, QOS.

Κεφάλαιο 7

Συμπεράσματα και Μελλοντική Εργασία

7.1 Σύνοψη και Συμπεράσματα

Στην παρούσα εργασία μελετήθηκε το πρόβλημα της προσωρινής αποθήκευσης και σύστασης περιεχομένου στα άκρα του δικτύου λαμβάνοντας υπόψιν την κινητικότητα των χρηστών εντός ενός χώρου ενδιαφέροντος. Προσομοιάζοντας την κίνηση των χρηστών μέσω Τυχαίων Περιπάτων σε γράφο, χωρίσαμε στη συνέχεια το πρόβλημα στις εξής συνιστώσες: Στην επιλογή των κατάλληλων Clusterheads, για προσωρινή αποθήκευση περιεχομένου στις κινητές τους συσκευές, στην επιλογή του περιεχομένου που θα αποθηκεύσουμε σε αυτούς και στη σύσταση των κατάλληλων αντικειμένων σε κάθε χρήστη που μεγιστοποιούν την ποιότητα εμπειρίας του, εκφρασμένη ως συνάρτηση της συνάφειας του προτεινόμενου περιεχομένου με το χρήστη και του αναμενόμενου χρόνου προσπέλασής του. Αντιστοιχίσαμε τα παραπάνω προβλήματα σε γνωστά αλγοριθμικά προβλήματα και χρησιμοποιήσαμε αποδοτικούς αλγορίθμους επίλυσής τους που επιτυγχάνουν πολύ καλή προσέγγιση και γρήγορο χρόνο εκτέλεσης. Τέλος, αξιολογήσαμε το μοντέλο που αναπτύξαμε μέσω προσομοίωσης για διάφορετικούς συνδυασμούς παραμέτρων και μελετήσαμε την συμπεριφορά του αλγορίθμου μας ως προς τις μέσες τιμές των μετρικών Quality Of Experience (QOE), Quality Of Recommendations (QOR) και Quality Of Service (QOS) που προσφέρει το σύστημα στους χρήστες, κάνοντας παρατηρήσεις και σχόλια. Παράλληλα, παρατηρήσαμε την απόδοση του αλγορίθμου μας σε σύγκριση με τις βέλτιστες δυνατές τιμές των μετρικών αυτών και είδαμε ότι τα αποτελέσματα του αλγορίθμου μας ακολουθούν την ίδια μονοτονία με τις βέλτιστες τιμές για όλους τους συνδυασμούς παραμέτρων, γεγονός που υποδεικνύει το σωστό τρόπο υλοποίησης και λειτουργίας του αλγορίθμου μας. Επίσης διαπιστώσαμε το πόσο κοντά είναι όλες οι τιμές (και του αλγορίθμου μας και οι βέλτιστες) για τους διάφορους συνδυασμούς παραμέτρων, το οποίο υποδεικνύει ότι το σύστημα προσαρμόζεται και προβαίνει σε σύσταση πολύ ικανοποιητικού και γρήγορα προσπελάσιμου περιεχομένου, για διάφορετικούς συνδυασμούς των παραμέτρων

που συμμετέχουν στο μοντέλο μας.

7.2 Μελλοντική Εργασία

Προκειμένου το προτεινόμενο σύστημα προσωρινής αποθήκευσης και σύστασης περιεχομένου να μπορεί να υλοποιηθεί σε πραγματικό επίπεδο, παραθέτουμε τα παρακάτω σημεία στα οποία θα ήταν χρήσιμο να επικεντρωθεί μελλοντικά η σχετική έρευνα:

- Προσεκτική συμπερίληψη στο πρόβλημα, και κατ'επέκταση αντιμετώπιση, των περιορισμών που εισάγουν οι υποψήφιοι Clusterheads στο σύστημα ως προς τη χωρητικότητα του αποθηκευτικού χώρου των συσκευών τους, τη διάρκεια ζωής της μπαταρίας των συσκευών αυτών αλλά και τη διαθεσιμότητα και προθυμία των χρηστών να γίνουν Clusterheads στο δίκτυο, με την παροχή προνομίων από τους διαχειριστές δικτύου και τους παρόχους περιεχομένου, τη διασφάλιση σεβασμού των προσωπικών τους δεδομένων και τη μη υπερφόρτωση των συσκευών τους.

- Εισαγωγή στο μοντέλο πραγματικών συνόλων δεδομένων που προσομοιάζουν πιστότερα τις ρεαλιστικές κατανομές των προτιμήσεων των χρηστών, των αντικειμένων ως προς τις κατηγορίες που υπάγονται και εφαρμογή του μοντέλου σε πραγματικούς γεωγραφικούς χώρους.

- Εξασφάλιση συνεργασίας μεταξύ των διαχειριστών δικτύου και των παρόχων περιεχομένου ώστε να αντιμετωπιστεί αποδοτικότερα το κοινό πρόβλημα προσωρινής αποθήκευσης και συστάσεων.

- Εύρεση κατάλληλων χρονικών διαστημάτων ανανέωσης των caches ώστε να προστίθεται νέο δημοφιλές υλικό χωρίς μεγάλη επιβάρυνση του δικτύου αλλά και εξασφάλιση ίσης αντιμετώπισης ως προς όλα τα αντικείμενα με αποθήκευση και λιγότερο δημοφιλούς περιεχομένου.

Bibliography

[1] R. Serfozo, Basics of Applied Stochastic Processes: http://www.stat.yale.edu/~jtc5/251/readings/Basics%20of%20Applied%20Stochastic%20Processes_Serfozo.pdf

[2] J.Chang, Markov Chains: http://www.stat.yale.edu/~pollard/Courses/251.spring2013/Handouts/Chang-MarkovChains.pdf

[3] Brémaud, Pierre. Markov chains: Gibbs fields, Monte Carlo simulation, and queues. Vol. 31. Springer Science & Business Media, 2013.

[4] Lovász, László. "Random walks on graphs." Combinatorics, Paul erdos is eighty 2.1-46 (1993): 4.

[5] wikipedia, Random Walks. https://en.wikipedia.org/wiki/Random_walk

[6] Williamson, David P., and David B. Shmoys. The design of approximation algorithms. Cambridge university press, 2011.

[7] Hochbaum, Dorit S. "Heuristics for the fixed cost median problem." Mathematical programming 22.1 (1982): 148-162.

[8] Jain, Kamal, et al. "Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP." Journal of the ACM (JACM) 50.6 (2003): 795-824.

[9] Guha, Sudipto, and Samir Khuller. "Greedy strikes back: Improved facility location algorithms." Journal of algorithms 31.1 (1999): 228-248.

[10] Li, Shi. "A 1.488 approximation algorithm for the uncapacitated facility location problem." International Colloquium on Automata, Languages, and Programming. Springer, Berlin, Heidelberg, 2011.

[11] Lin, Jyh-Han, and Jeffrey Scott Vitter. "e-Approximations with minimum packing constraint violation." Proceedings of the twenty-fourth annual ACM symposium on Theory of computing. 1992.

[12] Jain, Kamal, and Vijay V. Vazirani. "Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation." Journal of the ACM (JACM) 48.2 (2001): 274-296.

[13] Arya, Vijay, et al. "Local search heuristics for k-median and facility location problems." SIAM Journal on computing 33.3 (2004): 544-562.

[14] Young, Neal E. "K-medians, facility location, and the Chernoff-Wald bound." arXiv preprint cs/0205047 (2002).

[15] Motwani, Rajeev, and Prabhakar Raghavan. Randomized algorithms. Cambridge university press, 1995.

[16] Karloff, Howard. Linear programming. Springer Science & Business Media, 2008.

[17] David P. Williamson, NP-Complete Problems and General Strategy: `https://people.orie.cornell.edu/dpw/orie6300/Lectures/lec25.pdf`

[18] Özbakir, Lale, Adil Baykasoğlu, and Pınar Tapkan. "Bees algorithm for generalized assignment problem." Applied Mathematics and Computation 215.11 (2010): 3782-3795.

[19] Vazirani, Vijay V. Approximation algorithms. Vol. 1. Berlin: springer, 2001.

[20] Cohen, Reuven, Liran Katzir, and Danny Raz. "An efficient approximation for the generalized assignment problem." Information Processing Letters 100.4 (2006): 162-166.

[21] Fleischer, Lisa, et al. "Tight approximation algorithms for maximum general assignment problems." Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm. 2006.

[22] Yao, Jingjing, Tao Han, and Nirwan Ansari. "On mobile edge caching." IEEE Communications Surveys & Tutorials 21.3 (2019): 2525-2553.

[23] Wang, Shuo, et al. "A survey on mobile edge networks: Convergence of computing, caching and communications." Ieee Access 5 (2017): 6757-6779.

[24] Waqas, Muhammad, et al. "A comprehensive survey on mobility-aware D2D communications: Principles, practice and challenges." IEEE Communications Surveys & Tutorials 22.3 (2019): 1863-1886.

[25] Yang, Shusen, et al. "Using social network theory for modeling human mobility." IEEE network 24.5 (2010): 6-13.

[26] Lee, Kyunghan, et al. "Slaw: A new mobility model for human walks." IEEE INFOCOM 2009. IEEE, 2009.

[27] Karyotis, Vasileios, et al. "Efficient and socio-aware recommendation approaches for big data networked systems." Big Data Recommender Systems 1 (2019): 58-87.

[28] Chatzieleftheriou, Livia Elena, Merkouris Karaliopoulos, and Iordanis Koutsopoulos. "Jointly optimizing content caching and recommendations in small cell networks." IEEE Transactions on Mobile Computing 18.1 (2018): 125-138.

[29] Gomez-Uribe, Carlos A., and Neil Hunt. "The netflix recommender system: Algorithms, business value, and innovation." ACM Transactions on Management Information Systems (TMIS) 6.4 (2015): 1-19.

[30] Zhou, Renjie, Samamon Khemmarat, and Lixin Gao. "The impact of YouTube recommendation system on video views." Proceedings of the 10th ACM SIGCOMM conference on Internet measurement. 2010.

[31] Isinkaye, Folasade Olubusola, Yetunde O. Folajimi, and Bolande Adefowoke Ojokoh. "Recommendation systems: Principles, methods and evaluation." Egyptian informatics journal 16.3 (2015): 261-273.

[32] towardsdatascience, Recommender.Systems. `https://towardsdatascience.com/brief-on-recommender-systems-b86a1068a4dd`.

[33] Tsigkari, Dimitra, and Thrasyvoulos Spyropoulos. "An approximation algorithm for joint caching and recommendations in cache networks." IEEE Transactions on Network and Service Management (2022).

[34] Vitoropoulou, Margarita, et al. "CAUSE: Caching Aided by USer Equipment." 2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics). IEEE, 2020.

[35] Henk, Martin, Jürgen Richter-Gebert, and Günter M. Ziegler. Basic properties of convex polytopes. Chapman and Hall/CRC, 2017.