Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Άμεση χωροθέτηση υπηρεσιών με μετακινούμενες υπηρεσίες

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΔΙΟΝΥΣΙΟΣ ΑΡΒΑΝΙΤΑΚΗΣ

Επιβλέπων : Δημήτριος Φωτάκης
Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2023

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Άμεση χωροθέτηση υπηρεσιών με μετακινούμενες υπηρεσίες

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΔΙΟΝΥΣΙΟΣ ΑΡΒΑΝΙΤΑΚΗΣ

Επιβλέπων : Δημήτριος Φωτάκης
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 15η Ιουλίου 2023.

....	....	....
Δημήτριος Φωτάκης Καθηγητής ΣΗΜΜΥ Ε.Μ.Π.	Αριστείδης Παγουρτζής Καθηγητής ΣΗΜΜΥ Ε.Μ.Π.	Αντώνιος Συμβώνης Καθηγητής ΣΕΜΦΕ Ε.Μ.Π.

Αθήνα, Ιούλιος 2023

....

Διονύσιος Αρβανιτάκης

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

Περίληψη

Στην παρούσα διπλωματική μελετάμε το πρόβλημα της άμεσης χωροθέτησης υπηρεσιών (online facility location). Η χωροθέτηση υπηρεσιών (facility location) είναι ένα από τα σημαντικότερα προβλήματα στο πεδίο της επιστήμης υπολογιστών και επιχειρησιακής έρευνας με πληθώρα πρακτικών εφαρμογών. Από θεωρητικής πλευράς το πρόβλημα παρέχει εύφορο έδαφος για την ανάπτυξη και εφαρμογή προσεγγιστικών αλγορίθμων και οι αλγόριθμοι για το πρόβλημα αποτελούν από τις σημαντικότερες κατακτήσεις στο πεδίο της θεωρίας προσεγγιστικών αλγορίθμων. Σε πολλές από τις εφαρμογες η είσοδος δεν είναι απόλυτα γνωστή εκ των προτέρων και τα αιτήματα ερχονται σε διακριτές χρονικές στιγμές και ο (άμεσος) αλγόριθμος πρεπει να τα εξυπηρετεί την στιγμή της άφιξης τους. Το πολύ ενδιαφέρον αυτό πρόβλημα εχει μελετηθεί εκτενώς στην βιβλιογραφία η οποία εχει καταλήξει στην πλήρη κατανόηση του απο θεωρητικής σκοπιάς. Μελετάμε το πρόβλημα της άμεσης χωροθέτησης υπηρεσιών και παραθέτουμε μερικά από τα σημαντικότερα αποτελέσματα σε αυτό.

Στην συνέχεια στρέφουμε την προσοχή μας στο πρόβλημα της άμεσης χωροθέτησης υπηρεσιών με μετακινούμενες υπηρεσίες (online facility location with mobile facilities). Μελετάμε τον ήδη γνωστό αλγόριθμο για το πρόβλημα και αποδεικνύουμε οτι είναι ασυμπτωτικά βέλτιστος στην ευθεία των πραγματικών αριθμών. Υποδεικνύουμε ποιό είναι το εμπόδιο στην απόδειξη της βελτιστότητας του αλγορίθμου σε Ευκλείδιους χώρους μεγαλύτερης διαστασης. Στο τελευταιό κεφάλαιο, χρησιμοποιώντας τεχνικές από το κλασικό πρόβλημα της χωροθέτησης υπηρεσιών, δείχνουμε οτι ο αλγόριθμος είναι πράγματι βέλτιστος σε Ευκλείδους χώρους μεγαλύτερης διάστασησ και αναπτύσουμε έναν καινούργιο αλγόριθμο για γενικούς μετρικούς χώρους. Τέλος δείχνουμε ότι ο αλγόριθμος μας είναι ασυμπτωτικά βέλτιστος σε γενικούς μετρικούς χώρους.

Λέξεις κλειδιά

Χωροθέτηση υπηρεσιών, Άμεση χωροθέτηση υπηρεσιών, Άμεσοι Αλγόριθμοι, Προσεγγιστικοί Αλγόριθμοι

Abstract

In this diploma thesis we study the online variation of the classical facility location problem. The facility location problem is arguably one of the most important and extensively studied problem in the fields of computer science and operations research. In many applications the demand sequence is not known in advance, these applications create the need for the design of online algorithms for the problem. We underline key results of the fascinating and important online facility location problem and study the recent work of Feldkord et al. [22] on the online facility location with mobile facilities problem. We give a new algorithm for the problem and apply tools and techniques from the classical literature of the online facility location problem to show that our algorithm is asymptotically optimal on general metric spaces.

Key words

Facility Location, Online Facility Location, Online Algorithms, Approximation Algorithms

Ευχαριστίες

Θα ήθελα πρωτίστως να ευχαριστήσω τον επιβλέποντα της διπλωματικής μου εργασίας καθηγητή Δημήτρη Φωτάκη για τις συμβουλές του, την υποστήριξη του και για το οτι με εμπιστεύθηκε με ένα τόσο ενδιαφέρον θέμα, πίστεψε σε εμένα και ήταν πάντα δίπλα μου στα πρώτα μου ερευνητικά βήματα. Χρησιμοποιώντας τις τεχνικές που ο ίδιος είχε αναπτύξει στο παρελθόν καταφέραμε να λύσουμε ένα πολύ ενδιαφέρον ανοιχτό ερώτημα. Θα ήθελα επίσης να τον ευχαριστήσω για την άψογη συνεργασία μας και γιατί αποτελεί πρότυπο για νέους επιστήμονες/ερευνητές σαν εμένα. Ένα μεγάλο ευχαριστώ οφείλω στον καθηγητή Άρη Παγουρτζή γιατί τα μαθήματα του με ενέπνευσαν να ασχοληθώ και να ενδιαφερθώ με θέματα θεωρητικής πληροφορικής τα οποία τελικά με κέρδισαν. Θέλω επίσης να ευχαριστήσω τον καθηγητή Στάθη Ζάχο για τις όμορφες συζητήσεις μας, τις συμβουλές του και την αγάπη που δείχνει στους φοιτητές. Ευχαριστώ επίσης τον καθηγητή της ΣΕΜΦΕ Αντώνη Συμβώνη που δέχτηκε να είναι μέλος της επιτροπής αξιολόγησης μου. Πέρα από τα σύνορα του Πολυτεχνείου θέλω να ευχαριστήσω τους καθηγητές Βάγγο Χατζηαφράτη και Δημήτρη Αχλιόπτα για τις συμβουλες τους και τα όμορφα πραγματα που μου μάθανε. Ευχαριστώ όλα τα μέλη του εργαστηρίου corelab για το υπέροχο κλίμα που δημιουργούν στο εργαστήριο, το κλίμα αυτό θέτει τις βάσεις για το σημαντικό ερευνητικό και διδακτικό έργο που παράγει το εργαστήριο. Τους ευχαριστώ επίσης γιατί ήταν πάντα πρόθυμοι να δώσουν συμβουλες και να συζητήσουν μαζί μου.

Θέλω επίσης να ευχαριστήσω τους φίλους μου από την σχολή, τον Ηλία, τον Δημήτρη και τον Μάριο για τις ομορφες συζητήσεις μας, για τα πράγματα που μου έμαθαν, για την υποστήριξη τους και γιατί αποτελούν έμπρακτα παραδείγματα του σημαντικότερου (κατα την γνώμη μου) προτερήματος της σχολής ΗΜΜΥ, το οποίο είναι η ποιότητα των φοιτητών. Θέλω να ευχαριστήσω επίσης τους φίλους μου που δεν είναι από την σχολή Ορέστη και Γιώργο για την υποστήριξη τους και την βαθιά φιλία τους μαζί μου.

Στους πιο κοντινούς μου ανθρώπους οφείλω ίσως το μεγαλύτερο ευχαριστώ. Ευχαριστώ την οικογένεια μου, τον πατέρα μου Αλέκο, για την αγάπη του, γιατί αυτός καλλιέργησε την αγάπη μου για τα μαθηματικά και γιατί προσπαθούμε πάντα μαζί να αντιμετωπίσουμε οτιδήποτε με προβληματίζει, την μητέρα μου Μαρία για την υποστήριξη της και γιατί είναι πάντα εκεί όταν την χρειάζομαι και την αδερφή μου Μαρίνα γιατί με νοιάζεται και είναι δίπλα μου σε κάθε μου βήμα. Ευχαριστώ από καρδιάς την Ευαγγελία γιατι με αγαπάει, με νοιάζεται, με βοηθάει να γίνομαι καλύτερος και με στηρίζει από τα πρώτα βήματα μου στην σχολή μέχρι την ολοκλήρωση των σπουδών μου. Η παρούσα διπλωματική είναι αφιερωμένη σε αυτούς.

Διονύσιος Αρβανιτάκης,

Αθήνα, 15η Ιουλίου 2023

Περιεχόμενα

Εκτεταμένη Ελληνική Περίληψη

Το βασικό σκέλος της παρούσας διπλωματικής εργασίας έχει αποδοθεί στην αγγλική γλώσσα. Σε αυτό το κομμάτι, συνοψίζουμε το περιεχόμενό της, δίνοντας έμφαση στους βασικούς ορισμούς, τις μεθοδολογίες και τα θεωρήματα, αλλά χωρίς τις μαθηματικές αποδείξεις. Η δομή της ενότητας αυτής είναι σε ένα προς ένα αντιστοίχηση με το (αγγλικό) περιεχόμενο της διπλωματικής εργασίας. Να σημειωθεί ότι διπλά σε κάθε μετάφραση ενός αγγλικού όρου στα ελληνικά θα αναγράφεται μέσα σε παρένθεση ο αντίστοιχος όρος στα αγγλικά.

0.1 Άμεση χωροθέτηση υπηρεσιών (Online Facility Location)

Στο κεφάλαιο αυτό της παρούσας διπλωματικής εργασίας μελετάμε το ιδιαίτερα ενδιαφέρον πρόβλημα της άμεσης χωροθέτησης υπηρεσιών το οποίο εισήγαγε ο Meyerson ([50]). Το πρόβλημα χωροθέτησης υπηρεσιών είναι από τα σημαντικότερα και εκτενέστερα μελετημένα προβλήματα στα πεδία της επιχειρησιακής έρευνας (operations research) και της επιστήμης υπολογιστών (computer science). Η εκτενής μελέτη του προβλήματος οφείλεται αφενός στις ιδιαίτερα σημαντικές εφαρμογές του προβλήματος σε πρακτικά προβλήματα αφού αποτελεί φυσική μοντελοποίηση σε πολλά πρακτικά προβλήματα μείζονος ενδιαφέροντος και αφετέρου στο ιδιαίτερο θεωρητικό του ενδιαφέρον υπό το πρίσμα των προσεγγιστικών αλγορίθμων (approximation algorithms) και της ανάλυσης πέραν της χειρότερης περίπτωσης αλγορίθμων (beyond the worst case analysis of algorithms).

Η είσοδος του προβλήματος αποτελείται από έναν υποκείμενο μετρικό χώρο (metric space) $(\mathcal{X}, d)$, εναν πραγματικό αριθμό $c_f(i)$ για κάθε σημείο του μετρικού χώρου i που αντιπροσωπεύει το κόστος του να ανοίξει μια υπηρεσία (facility) σε αυτό το σημείο και ένα σύνολο από αιτήματα (demands) $\mathcal{D}$ που είναι σημεία του μετρικού χώρου. Στόχος του προβλήματος είναι να επιλεχθεί ένα σύνολο από σημεία του μετρικού χώρου στα οποία θα ανοίξουν υπηρεσίες ετσι ώστε να ελαχιστοποιείται το ακόλουθο κριτήριο, το άθροισμα του κόστους για τις υπηρεσίες που ανοιξαν συν την απόσταση του κάθε αιτήματος από την κοντινότερη σε αυτό ανοιχτή υπηρεσία. Τυπικά το κριτήριο είναι το ακόλουθο:

$$\min_{\mathcal{F} \subseteq \mathcal{X}} \sum_{i \in \mathcal{F}} c_f(i) + \sum_{j \in \mathcal{D}} \min_{i \in \mathcal{F}} d(i, j)$$

Σε πολλά πρακτικά προβλήματα η είσοδος του προβλήματος δεν είναι γνωστή εκ των προτέρων, πιο συγκεκριμένα το σύνολο από αιτήματα δεν είναι γνωστο΄στον αλγόριθμο εξ αρχής αλλά παρέχεται σε αυτόν σε μια ακόλουθία αιτημάτων, ένα κάθε στιγμή. Τη στιγμή που ένα αίτημα παρέχεται στον αλγόριθμο αυτός πρεπει να το αντιστοιχίσει σε μια ήδη ανοιχτή υπηρεσία ή να ανοίξει μια υπηρεσία και να το αντιστοιχίσει σε αυτή. Εφαρμογές που εφάπτονται στην μοντελοποιήση αυτού το προβλήματος συμπεριλαμβάνουν την σχεδίαση ενός δικτύου υπολογιστών (network design) και την συσταδοποίηση σελίδων στον παγκόσμιο ιστό (clustering web pages). Αφορμώμενος από τέτοιες εφαρμογές ο Meyerson είσηγαγε το πρόβλημα της άμεσης χωροθέτησης υπηρεσιών. Τυπικά το πρόβλημα αυτό αποτελείται και πάλι από έναν υποκείμενο μετρικό χώρο $(\mathcal{X}, d)$ και ένα κόστος c_f για κάθε σημείο του μετρικού χώρου (να σημειωθεί το επικεντρονόμαστε στην ομοίομορφη εκδοχή (uniform case) του προβλήματος όπου κάθε σημείο του μετρικού χώρου εχει το ίδιο κόστος για το άνοιγμα μιας υπηρεσίας). Τα αιτήματα ερχονται σε μια ακολουθια σ και με u_i συμβολίζουμε το αίτημα που ερχεται την χρονική στιγμή i. Σε κάθε χρονική στιγμή ο αλγόριθμος διατηρεί ένα σύνολο από ανοιχτές υπηρεσίες και όποτε ενα καινούργιο αίτημα παρεχεται στον αλγόριθμο αυτός πρεπει είτε να το αναθέσει σε μια ανοιχτή υπηρεσία είτε να ανοίξει μια καινούργια υπηρεσία και να το αναθέσει σε αυτή. Η ανάθεση των αιτημάτων σε υπηρεσίες είναι αμετάκλητη και οι ανοιχτές υπηρεσίες δεν μπορούν να κλείσουν αφού ανοίξουν. Χρησιμοποιούμε το συμβολισμό $\mathcal{F}_i$ για τις υπηρεσίες που εχουν ανοίξει αφού το αίτημα u_i εχει εξυπηρετηθεί. Το κόστος του αλγορίθμου είναι:

$$\sum_{i=1}^{n} d(\mathcal{F}_i, u_i) + c_f|\mathcal{F}_n|$$

Υπενθυμίζουμε ότι c_f είναι τό κόστος για να ανοίξει μια υπηρεσία, το οποίο είναι το ίδιο για κάθε σημείο του μετρικού χώρου.

Στο κεφάλαιο 2 της παρούσας διπλωματικής παραθέτουμε κάποια από τα βασικά αποτελέσματα σε αυτό το πρόβλημα τα οποία προέρχονται από μια ακολουθία εργασίων που οδήγησε στην πλήρη κατανόηση του προβλήματος, από θεωρητικής πλευράς τουλάχιστον.

Στην εργασία στην οποία είσηγαγε το πρόβληματα ο Meyerson, ο ίδιος πρότεινε έναν εναν κομψό και διαισθητικό αλγόριθμο (RANDOFL) για το πρόβλημα και απέξειξε οτι αλγόριθμος αυτός πετυχαίνει προσεγγιστικό λόγο $O(\log(n))$ (competitive ratio) (να σημειωθεί οτι συνήθως ο όρος προσεγγιστικός λόγος χρησιμοποιείται για προσεγγιστικούς αλγορίθμους). Στην συνέχεια στην εργασία [26] ο Φωτάκης απέδειξε ότι ο αλγόριθμος στην πραγματικότητα είναι $O(\frac{\log(n)}{\log(\log(n))})$ προσεγγιστικός και οτι κανένας αλγόριθμος δεν μπορεί να πετύχει ασυμπτωτικά καλύτερο λόγο προσεγγισης ακόμα και σε απλούς μετρικούς χώρους όπως η ευθεία των πραγματικών αριθμών.

Ο αλγοριθμός RANDOFL παρουσιάζεται παρακάτω, η λειτουργία του είναι η εξής: κάθε φορά που ένα καινούργιο αίτημα παρέχεται στον αλγόριθμο αυτός υπολογίζει την απόσταση του αιτήματος από την κοντινότερη ανοιχτή υπηρεσία, έστω d, και ανοίγει μια νεα υπηρεσία πάνω στο αίτημα, με πιθανότητα $\min(1, \frac{d}{c_f})$.

Algorithm 1: Meyerson's algorithm (RANDOFL)

for $i = 1$ *to* n **do**
  let $d = d(u_i, \mathcal{F}_{i-1})$
  w.p $\min(1, \frac{d}{c_f})$ open a facility at u_i
end

Η ανάλυση του αλγορίθμου μας δίνει το ακόλουθο θεώρημα:

Theorem 0.1.1. *Ο αλγόριθμος RANDOFL είναι* $O(\frac{\log(n)}{\log(\log(n))})$ *προσεγγιστικός*

Από την άλλη μεριά το ακόλουθο κάτω όριο στην επίδοση οποιουδήποτε αλγορίθμου μας βεβαίωνει οτι η ανάλυση αυτή είναι εφαρμοστή και μάλιστα οτι ο αλγόριθμος είναι βέλτιστος:

Theorem 0.1.2. *Κανένας τυχαιοποιημένος αλγόριθμος δεν μπορεί να πετύχει προσεγγστικό λόγο καλύτερο από* $\Omega(\frac{\log(n)}{\log(\log(n))})$ *ακόμα και σε απλούς μετρικούς χώρους όπους η ευθεία των πραγματικών αριθμών*

Η εργασία του Φωτάκη επίσης εισάγει και έναν ντετερμινιστικό αλγόριθμο ο οποίος είναι επίσης ασυμπτωτικά βέλτιστος.

Ο αλγόριθμος αυτός, DETOFL, αποδεικνύεται οτι είναι επίσης ασυμπτωτικά βέλτιστος:

Theorem 0.1.3. *Ο αλγόριθμος DETOFL πετυχαίνει προσεγγιστικό λόγο* $O(\frac{\log(n)}{\log(\log(n))})$

Στην παρούσα διπλωματική παρουσιάζουμε την ανάλυση του RANDOFL καθώς και του κάτω φράγματος (lower bound) του προβλήματος. Επίσης αποδεικνυούμε ότι ο αλγόριθμος DETOFL είναι ασυμπτωτικά βέτλιστος στην περίπτωση που η βέλτιστη λύση αποτελείται από μια μόνο υπηρεσία. Παραλείπουμε ωστόσο την απόδειξη της βελτιστότητας του αλγορίθμου στην γενική περίπτωση που η βέλτιστη λύση αποτελέιται από περισσότερες υπηρεσίες.

0.2 Προυπάρχουσες εργασίες στην αμεση χωροθέτηση υπηρεσιών με μετακινούμενες υπηρεσίες (Previous work on online facility location with mobile facilities)

Το ενδεχομένως πεσιμιστικό κάτω φράγμα στους αλγορίθμους για το πρόβλημα της αμεσης χωροθέτησης υπηρεσιών εχει ωθήσει τους ερευνητές να μελετήσουν διάφορες λίγοτερο ”αυστηρες” παραλλαγές του προβλήματος με στόχο την βελτίωση της επίδοσης των αλγορίθμων. Η προσπάθεια αυτή ανήκει σε ένα γενικότερο και ειδικά ενδιαφέρον και εξελισσόμενο πεδίο της θεωρίας αλγορίθμων που ονομάζεται ανάλυση πέραν της χειρότερης περίπτωσης (beyond the worst case analysis). Υπό αυτό το πρίσμα, εχουν προταθεί αρκετές παραλλαγές του προβλήματος και εχει γίνει εκτενής προσπάθεια προς αυτή την κατεύθυνση. Το πρώτο αποτέλεσμα ήταν στην αρχική εργασία του Meyerson που μελετούσε την περίπτωση που ο αντίπαλος επιλέγει την ακόλουθία από αιτήματα και στην συνέχεια αυτά ανακατεύονται τυχαία (randomly permuted) πριν δοθούν στον αλγόριθμο. Σε αυτό το μοντέλο ο αλγόριθμος του Meyerson δίνει σταθερό λόγο προσεγγισης. Ένα από τα σημαντικότερα αποτελέσματα είναι αυτό του Φωτάκη στην αυξητική χωροθέτηση υπηρεσιών (incremental facility location) [23] οπου παρουσιάζεται ένας σταθερού λόγου προσέγγισης αλγόριθμος Μια άλλη τέτοια γενική κατεύθυνση είναι αυτή στην οποία οι

Algorithm 2: Deterministic Algorithm (DETFL)

Let x be an appropriately chosen constant
$\mathcal{F}_0 \leftarrow \emptyset$
$L \leftarrow \emptyset$
for $i = 1$ *to* n **do**
  let $d = d(u_i, \mathcal{F}_{i-1})$
  $L \leftarrow L \cup \{u_i\}$
  $r_{u_i} \leftarrow d/x$
  $B_{u_i} \leftarrow \text{Ball}(u_i, r_{u_i}) \cap L$
  $\text{Pot}(B_{u_i}) \leftarrow \sum_{u \in B_{u_i}} d(\mathcal{F}_{i-1}, u)$
  if $\text{Pot}(B_{u_i}) \geq c_f$ **then**
    if $d(\mathcal{F}_{i-1}, u_i) < c_f$ **then**
      Let $\nu \geq 0$ be the smallest integer such that:
        Either there exists exactly one point $u \in B_{u_i}$, such that:
        $\text{Pot}(B_{u_i} \cap \text{Ball}(u, r_{u_i}/2^{\nu})) > \text{Pot}(B_{u_i})/2$
        Or for every $u \in B_{u_i}$, $\text{Pot}(B_{u_i} \cap \text{Ball}(u, r_{u_i}/2^{\nu+1})) \leq \text{Pot}(B_{u_i})/2$
      Let $\hat{w}$ be any point in B_{u_i} such that:
        $\text{Pot}(B_{u_i} \cap \text{Ball}(\hat{w}, r_{u_i}/2^{\nu})) \geq \text{Pot}(B_{u_i})/2$
    end
    else
      $\hat{w} \leftarrow u_i$
    end
    $\mathcal{F}_i = \mathcal{F}_{i-1} \cup \{\hat{w}\}$
    $L \leftarrow L \setminus B_{u_i}$
  end
  Assign u_i to the nearest facility in $\mathcal{F}_i$
end

υπηρεσίες δεν μπορούν μεν να κλείσουν, μπορούν όμως να μετακινηθούν της οποίας αφετηρία ήταν η δουλειά τον Diveki και Imreh [18].

Στην πρόσφατη εργασία τους οι Knollman et al. προτείνουν νέα μοντέλα για το πρόβλημα χωροθέτησης υπηρεσιών με μετακινούμενες υπηρεσίες. Το ένα από αυτά τα μοντέλα είναι το κύριο σημείο ενδιαφέροντος της παρούσας διπλωματικής. Σε αυτό το μοντέλο οι υπηρεσίες μπορούν να μετακινηθούν πληρώνοντας ένα κόστος D επί την απόσταση κατα την οποία μετακινήθηκαν.

Ας δούμε τώρα ποιο είναι το κόστος ενός αλγορίθμου για το πρόβλημα χωροθέτησης υπηρεσιών με μετακινούμενες υπηρεσίες. Το κόστος για να ανοίξει μια υπηρεσία και το κόστος μετακίνησης είναι φυσικά το κόστος των ανοιχτών υπηρεσιών (ο αλγόριθμος δεν επιτρέπεται να κλείνει ανοιχτές υπηρεσίες, μόνο να τις μετακινεί) και η συνολική απόσταση που έχουν μετακινηθεί οι υπηρεσίες επί D. Ωστόσο, υπάρχουν 2 φυσικοί τρόποι για τον καθορισμό του κόστους ανάθεσης. Και οι δύο λαμβάνονται υπόψη στο [22]:

1. *Το μοντέλο της άμεσης εξυπηρετησης (Instant service model)*: Το κόστος ανάθεσης για ένα αίτημα u_i είναι η απόστασή της από την πλησιέστερη υπηρεσία στο $\mathcal{F}_i$. Σημειώστε ότι αυτό θα μπορούσε και, στη γενική περίπτωση, θα είναι διαφορετικό από την απόστασή του από την πλησιέστερη υπηρεσία στο $\mathcal{F}_n$ (το σύνολο των υπηρεσιών στο τέλος της ακολουθίας αιτημάτων). Αυτό το μοντέλο ταιριάζει σε εφαρμογές όπου τα αιτήματα έρχονται με την πάροδο του χρόνου και εξυπηρετούνται κατά την άφιξή τους

2. *Το μοντέλο εξυπηρέτησης με καθυστέρηση έως το τέλος (Delayed to end service model)*: Το κόστος ανάθεσης για ένα αίτημα u_i είναι η απόστασή της από την υπηρεσία στην οποία είχε αρχικά ανατεθεί το αίτημα αλλά στο τέλος της ακολουθίας. Με άλλα λόγια, όταν φτάνει ένα αίτημα, αντιστοιχίζεται αμετάκλητα σε μια υπηρεσία και το κόστος ανάθεσης είναι η απόστασή του από την υπηρεσία αυτή αλλά στο τέλος της ακολουθίας αιτημάτων. Αυτό το μοντέλο ταιριάζει καλύτερα στις εφαρμογές που σχετίζονται με την ομαδοποίηση όπου θέλουμε να έχουμε μια καλή ομαδοποίηση των αιτημάτων στο τέλος της ακολουθίας.

Είναι σημαντικό να σημειωθεί ότι και στις δύο περιπτώσεις η απόδοση του αλγορίθμου μετριέται σε σύγκριση με τη στατική βέλτιστη λύση όπου η ακολουθία είναι γνωστή εκ των προτέρων, δηλαδή με την βέλτιστη λύση του αντίστοιχου offline προβλήματος χωροθέτησης. Αυτή είναι μια ισχυρή υπόθεση για το πρώτο μοντέλο, καθώς υπάρχουν περιπτώσεις όπου ο αλγόριθμος μπορεί πράγματι να ξεπεράσει την βέλτιστη offline λύση. Ωστόσο, στην χειρότερη περίπτωση, οποιοσδήποτε αμεσος αλγόριθμος είναι χειρότερος από τη στατική βέλτιστη offline λύση.

Στην διπλωματική αυτή επικεντρωνόμαστε στο πρώτο μοντέλο. Στο κεφάλαιο 3 της διπλωματικής παρουσιάζουμε τα βασικά αποτελέσματα που ήταν γνωστά κατά την διεξαγωγή της εργασίας αυτής. Έχει γίνει προσπάθεια αυτά να παρουσιαστούν υπό την οπτική πλευρά μας και κάποιες αποδείξεις εχουν διαφορές από τις αντίστοιχες στο [22].

Το πρώτο ερώτημα που ερχεται στο μυαλό όταν μελετάμε αυτό το καινούργιο μοντέλο είναι ”πόσο καλύτερα μπορεί να τα πάει ενας αλγόριθμος σε αυτό το μοντέλο σε σχέση με το κλασικό πρόβλημα αμεσης χωροθέτησης υπηρεσιών;”. Οι Knollman et al. χρησιμοποιώντας ιδέες παρόμοιες με αυτές για το κάτω φράγμα στο κλασικό πρόβλημα αμεσης χωροθέτησης υπηρεσιών αποδεικνύουν το ακόλουθο αποτέλεσμα:

Theorem 0.2.1. *Κανένας αλγόριθμος του οποίου ο λόγος προσεγγισης είναι ανεξάρτητος από τον αριθμό των αιτημάτων δεν μπορεί να πετύχει λόγο προσέγγισης καλύτερο από $\Omega(compm)$, ακόμα και σε απλούς μετρικούς χώρους όπως η ευθεία των πραγματικών αριθμών.*

Έτσι αν θέλουμε να ξεπεράσουμε το κάτω φράγμα της αμεσης χωροθέτησης υπηρεσιών και ο λόγος προσέγγισης να είναι ανεξάρτητος από τον αριθμό των αιτημάτων δεν μπορούμε να περιμένουμε έναν λόγο προσέγγισης καλύτερο από $O(\frac{\log(D)}{\log(\log(D))})$. Οι ερευνητές που εισήγαγαν το προβλημα το μελέτησαν αποκλειστικά σε ευκλείδιους χώρους. Παραθέτουμε τον αλγόριθμο τον οποίο προτείνουν.

Είναι εύκολο να δει κανείς οτι ο αλγόριθμος αυτός λειτουργεί μόνο σε ευκλείδιους μετρικούς χώρους. Η ανάλυση που παραθέτουν οι Knollman et al αποδεικνύει οτι ο παραπάνω αλγόριθμος είναι πραγματι βέλτιστος στην ευθεία των πραγματικών.

Algorithm 3: The algorithm for Euclidean metric spaces (EucOFLM)

Let β be an appropriately chosen constant
$\mathcal{F}_0^m \leftarrow \emptyset$
$\mathcal{F}_0^s \leftarrow \emptyset$
for $i = 1$ *to* n **do**
  Let $a = \arg\min_{a'}\{d(u_i, a') : a' \in \mathcal{F}^s\}$
  if $d(a, u_i) \leq \frac{2c_f}{D}$ **then** /* u_i is a close demand */
    Let $z = \mathrm{mob}(a)$
    w.p. $\frac{d(z,u_i)}{\beta c_f}$: $\mathcal{F}_i^s \leftarrow \mathcal{F}_{i-1}^s \cup \{u_i\}$, $\mathcal{F}_i^m \leftarrow \mathcal{F}_{i-1}^m \cup \{u_i\}$
    $move(z \rightarrow \frac{D-1}{D} z + \frac{1}{D} u_i)$
    Assign u_i to z
  end
  else /* u_i is a far demand */
    w.p. $\frac{d(a,u_i)}{\beta c_f}$: $\mathcal{F}_i^s \leftarrow \mathcal{F}_{i-1}^s \cup \{u_i\}$, $\mathcal{F}_i^m \leftarrow \mathcal{F}_{i-1}^m \cup \{u_i\}$
    Assign u_i to the facility opened at u_i.
  end
end

Theorem 0.2.2. *Ο EucOFL είναι πετυχαίνει λόγο προσέγγισης $O(\frac{\log(D)}{\log(\log(D))})$ στην ευθεία των πραγματικών αριθμών, είναι επομένως ασυμπτωτικά βέλτιστος.*

Ωστόσο, σε ευκλείδιους χώρους μεγαλύτερης διαστασης στην ανάλυση του αλγορίθμου εμπεριεχεται και ένας όρος $\sqrt{k}$ οπου k ο αριθμός των υπηρεσιών στην βέλτιστη λύση. Οι ερευνητές αφησαν σαν ανοιχτό ερωτημα αν ο αλγόριθμος τους είναι πραγματι βελτιστος. Το κύριο εμπόδιο για την απόδειξη αυτού του ισχυρισμού είναι η ανάλυση του κόστους των απαγορευμένων αιτημάτων (prohibited demands) τα οποία είναι εύκολο να αναλυθουν στην ευθεία των πραγματικών αριθμών αλλά οχι σε χωρους μεγαλύτερης διάστασης.

0.3 Αμεση χωροθέτηση υπηρεσιών σε γενικούς μετρικούς

Αυτή η παραγραφος αποτελεί περίληψη του τελευταίου κεφαλαίου της διπλωματικής εργασίας και εμπεριέχει τα καινούργια αποτελέσματα που ειναι προϊοντα της διπλωματικής εργασίας.

Οπως αναφέρθηκε στην προηγούμενη παραγραφο το κύριο εμπόδιο για την ανάλυση του αλγορίθμου EucOFLM είναι η ανάλυση του κόστους των απαγορευμένων αιτημάτων. Χρησιμοποιώντας τεχνικές του Φωτάκη για την ανάλυση του ντετερμινιστικού αλγορίθμου για το κλασικό πρόβλημα της άμεσης χωροθέτησης υπηρεσιών καταφέρνουμε να φράξουμε το κόστος των απαγορευμένων αιτημάτων. Αναλυτικότερα εχουμε το ακόλου λήμμα:

Lemma 0.3.1. *Το αναμενόμενο κόστος των απαγορευμένων αιτημάτων (prohibited demands) φράσσεται από το κόστος της βέλτιστης λύσης*

Η απόδειξη του παραπάνω λήμματος αποδεικνύει την βελτιστότητα του αλγόριθμου των Knollman et al σε ευκλείδιους χώρους οποιασδήποτε διάστασης.

Στην συνέχεια μελετάμε το πρόβλημα της αμεσης χωροθέτησης υπηρεσιών με μετακινούμενες υπηρεσίες σε γενικούς μετρικούς χώρους. Να σημειωθεί οτι το πρόβλμα αυτό δεν είχε, τουλάχιστον από όσο γνωρίζουμε, μελετηθεί στο παρελθόν. Για το πρόβλημα αυτό παραθέτουμε έναν καινούργιο αλγόριθμο και αποδεικνύουμε ότι εχει ασυμπτωτικά βέλτιστο λόγο προσεγγισης σε γενικούς μετρικούς χώρους.

Algorithm 4: The algorithm for general metric spaces GenOFLM

Let β be an appropriately chosen constant
$\mathcal{F}_0^m \leftarrow \emptyset$
$\mathcal{F}_0^s \leftarrow \emptyset$
for $i = 1$ *to* n **do**
    Let $a = \arg\min_{a'} \{d(u_i, a') : a' \in \mathcal{F}^s\}$
    if $d(a, u_i) \leq \frac{2c_f}{D}$ **then** /* u_i is a close demand */
        Let $z = \text{mob}(a)$
        w.p. $\frac{d(z,u_i)}{\beta c_f}$: $\mathcal{F}_i^s \leftarrow \mathcal{F}_{i-1}^s \cup \{u_i\}$, $\mathcal{F}_i^m \leftarrow \mathcal{F}_{i-1}^m \cup \{u_i\}$
        w.p $\frac{1}{D}$: $move(z \rightarrow u_i)$
        Assign u_i to z
    end
    else /* u_i is a far demand */
        w.p. $\frac{d(a,u_i)}{\beta c_f}$: $\mathcal{F}_i^s \leftarrow \mathcal{F}_{i-1}^s \cup \{u_i\}$, $\mathcal{F}_i^m \leftarrow \mathcal{F}_{i-1}^m \cup \{u_i\}$
        Assign u_i to the facility opened at u_i.
    end
end

Για τον αλγόριθμο αυτόν αποδεικνύουμε το ακόλουθο θεώρημα:

Theorem 0.3.2. *Ο αλγόριθμος GenOFLM είναι ασυμπτωτικά βέλτιστος σε γενικούς μετρικούς χώρους. Πετυχαίνει δηλαδή λόγο προσέγγισης* $O(\frac{\log(D)}{\log(\log(D))})$.

Κείμενο στα αγγλικά

Chapter 1

Introduction

1.1 The Uncapacitated Metric Facility Location problem (UMFL)

The metric uncapacitated facility location problem is one of the most extensively studied problems in the computer science and operations research literature and has been studied since the second half of the previous century, dating back to the work of [5], [38], [45], [54]. This is, in part, due to its substantial practical application and in part due to its theoretical interest (being NP-hard) from the approximation algorithms and beyond the worst case analysis point of view. The facility location problem is a natural and accurate abstraction to many practical problems related to network design ([17], [47]) and clustering (techniques for facility location have been applied to the closely related, and widely accepted as one of the main clustering objectives, the k-median problem, for example in [33] and [12]). Facility location has also found applications in healthcare ([1], [2]), supply chain ([48]), planning of warehouses and public transportation terminals to name but a few. Despite its applicability it can be shown to be NP-hard (by a reduction from set cover for example ([56])) and therefore much of the work on this problem focuses on either finding exact solutions for special cases or on approximation algorithms for the problem. The problem consists of a metric space $(\mathcal{X}, d)$, a subset of points in the metric space that are the demands, $\mathcal{D} \subseteq \mathcal{X}$ and a facility cost $c_f(i)$ for every point i in the metric space. The goal is to open facilities on the metric space so as to minimize the sum of the facility cost (the sum of the costs of the facilities opened) plus the assignment cost (the distance between any demand and its closest open facility). In other words a set $F \subseteq \mathcal{X}$ must be selected in order to minimize $\sum_{i\in F} c_f(i) + \sum_{j\in\mathcal{D}} \min_{i\in F} d(i,j)$. As an integer program the problem can be formulated as follows:

$$\begin{aligned}
\text{minimize } & \sum_{i\in\mathcal{X}} y_i c_f(i) + \sum_{i\in\mathcal{X}, j\in D} d(i,j)x_{ij} \\
\text{subject to : } & \sum_{i\in\mathcal{X}} x_{ij} = 1, \forall i \in \mathcal{D} \\
& x_{ij} \le y_i, \forall i \in \mathcal{X}, j \in \mathcal{D} \\
& x_{ij} \in \{0,1\}, \forall i \in \mathcal{X}, j \in \mathcal{D} \\
& y_i \in \{0,1\}, \forall i \in \mathcal{X}
\end{aligned}$$

The y_i variables determine whether the facility at point i will open while the x_{ij} variables determine whether the demand at point j will be assigned to a facility at point i.

The corresponding LP relaxation that is widely used in approximation algorithms for the problem simply relaxes the conditions $x_{ij} \in \{0,1\}$ and $y_i \in \{0,1\}$ to $x_{ij} \geq 0$ and $y_i \geq 0$.:

$$\begin{aligned}
\text{minimize} & \sum_{i \in \mathcal{X}} y_i c_f(i) + \sum_{i \in \mathcal{X}, j \in D} d(i,j) x_{ij} \\
\text{subject to :} & \sum_{i \in \mathcal{X}} x_{ij} = 1, \forall i \in \mathcal{D} \\
& x_{ij} \leq y_i, \forall i \in \mathcal{X}, j \in \mathcal{D} \\
& x_{ij} \geq 0, \forall i \in \mathcal{X}, j \in \mathcal{D} \\
& y_i \geq 0, \forall i \in \mathcal{X}
\end{aligned}$$

The facility location problem has been proven to be a very fruitful problem for the development and application of approximation algorithms techniques. Virtually all techniques have been successfully used to design approximation algorithms for the problem (see [58]) The first approximation algorithm for the problem was an $O(\log(n))$ approximation ([30]). Lin and Vitter ([42]) used LP rounding and the filtering technique to get another $O(\log(n))$-approximation algorithm for the problem and for the more general non-metric case. The first constant factor approximation algorithm for the problem was due to Shmoys, Tardos and Aardal ([53]) and was a 3.16 approximation algorithm. The work of Guha and Khuller ([29]) used LP rounding together with local search/greedy techniques to improve the approximation ratio to 2.408. This paper also gave the first (to our knowledge) complexity characterization of the problem with respect to its approximability. It was proven that the problem is MAX-SNP-hard ([52]) via a reduction from the *B-vertex cover* problem. It was also shown that if there is an approximation algorithm better than 1.463 then this implies an algorithm with approximation ratio $c \ln |X|$ for the set cover problem which would in turn imply that $NP \subseteq DTIME(n^{O(\log(\log(n)))})$ by Feige ([20]).

The papers discussed thus far require the solution of a linear program which, while theoretically requires polynomial time, is very costly in practice. Korupolu, Plaxton and Rajamaran ([37]) also analysed local search algorithms for the facility location problem and gave a $5+\epsilon$ approximation ratio guarantee but with better running times that did not demand solving a linear program. Jain and Vazirani ([33]) proposed an approximation algorithm for the problem based on the primal dual schema that achieved a 3 approximation and had a running time of $O(m \log(m))$ where m is the number of edges in the underlying graph of the metric space. This paper not only significantly improved the running time with only a marginal increase in the approximation ratio but also laid the groundwork for further research on problems pertaining to facility location. Their work also provided a 6 approximation for the k-median problem and the core ideas and techniques were also used in variants of the facility location problem ([49], [13], [35])

To the best of our knowledge, currently the best approximation algorithm for the uncapacitated metric facility location problem is due to Li ([41]) building on work of Byrka [10] and achieves an approximation ratio of 1.488. This is almost tight due to the result of Guha and Khuller ([29]) (which was later strengthened by Sviridenko ([55]) who proved that no approximation algorithm can achieve a competitive ratio better than 1.463 unless $P = NP$)

1.2 The Online Facility Location Problem

This diploma thesis intents to study the online version of the metric uncapacitated facility location problem. Before providing formal proofs for the Online Facility Location problem (Chapter 2) we will provide a brief overview of the problem's motivation and highlight key results. Consider the case of designing a network, which is, as discussed earlier, a very fruitful application of the Facility Location Problem. The designer should select the server locations in the underlying metric space in order to minimize the cost. More often than not, the set of clients is not completely known to the designer in advance, and therefore an online computation is involved. Another example is that of clustering pages of the web, in that case new webpages are created in an online fashion and the algorithm should maintain a good clustering of them over time. Motivated by such applications Meyerson, [50], introduced the Online Facility Location problem, where demands arrive one at a time and the goal is to design an online algorithm that minimizes the competitive ratio of the online algorithm against the optimal offline solution.

Let's now give a more formal definition of the Online Facility Location problem. Similarly to the offline case we consider a metric space (X, d). At distinct points in time $t = 1, 2, ...n$, (not necessarily distinct) demands $u_1, u_2, ...u_n$ arrive as points in the metric space X. We denote by $\mathcal{F}_i$ the set of facilities opened by the algorithm after demand u_i is processed. It is important to note that we will mainly study the uniform case where the cost of opening a facility is the same for every point $p \in X$, another important thing to note is that the algorithm is not allowed to close facilities, only to open them, therefore $\mathcal{F}_{i-1} \subseteq \mathcal{F}_i$. The cost incurred to the algorithm is, as in the offline case, the sum of the assignment cost and the facility opening cost. The key difference from the offline case is that the assignment cost for a demand u_i is the distance to the closest open facility in F_i (i.e. the distance to the nearest facility after demand u_i was processed). The assignment cost on the other hand is naturally $c_f \cdot |\mathcal{F}_n|$, where by c_f we denote the cost of opening a single facility. The cost of the algorithm is:

$$\sum_{i=1}^{n} d(\mathcal{F}_i, u_i) + c_f|\mathcal{F}_n| \tag{1.1}$$

The goal is to design online algorithms such that the competitive ratio between the online and the offline optimal solution, where every demand is known in advance is minimized.

Meyerson proposed an elegant and intuitive algorithm for the problem and showed its competitive ratio to be $O(\log(n))$ where n is the number of demands. He also showed that the competitive ratio can not be independent of the number of demands. Moreover, Meyerson's algorithm had the interesting property of being constant factor competitive in the random order model where the adversary chooses the demand sequence, but it is randomly permuted before it is provided to the algorithm. Fotakis ([26]) showed that the algorithm's competitive ratio is in fact $O(\frac{\log(n)}{\log(\log(n))})$ and provided a matching lower bound for randomized algorithms against oblivious adversaries. It is important to note that the lower bound holds even for very simple metric spaces like tree metrics and the real line (where the offline problem can be solved optimally in polynomial time). Fotakis ([26]) also proposed an asymptotically optimal derandomization of Meyerson's algorithm which led to the complete theoretical understanding of the Online Facility Location problem. However, in order to achieve an asymptotically optimal competitive ratio the algorithm is fairly complicated and it is unlikely that it could be useful in practice. For practical purposes the algorithms of [4] and the primal dual algorithm of [25] seem to outperform the algorithm of [26] even though their theoretical guarantees are slightly worse. More specifically, the primal dual algorithm is $O(\log(n))$-competitive while the algorithm of Anagnostopoulos et al. [4] works only for Euclidean metric spaces and its competitive ratio is $O(2^d \log(n))$, where d is the dimension of the Euclidean space and n is the number of demands. The primal dual algorithm on the other hand has a competitive ratio of $O(\log(n))$ regardless of the underlying metric space and works in the non-uniform case of the problem where different points of the metric space have different costs of opening facilities. It is worth noting that the primal dual algorithm has found application on other problems like the facility leasing problem [51]

1.3 Relaxed versions of the Online Facility Location Problem

It is evident from the lower bound that we will present in Chapter 2 that the lack of knowledge about future demands forces any algorithm to open facilities that will be useless in the future. It is therefore natural to study relaxed versions of the problem where the opening of a facility at some point of the metric space is not completely irrevocable. There has been a particularly interesting line of work on this direction.

To our knowledge, the first and perhaps the most outstanding work on this direction is that on incremental facility location. On incremental facility location, the algorithm is allowed to merge pairs of facilities (and clusters of demands) as time passes by closing one facility and reassigning every demand to the other facility. The cost of the algorithm on that setting is the sum of distances from demands to the facilities they are assigned to, plus the facility costs of the open facilities by the end of the sequence. This problem can be viewed as an instance of the more general setting of incremental clustering. The framework of incremental clustering was introduced by Charikar, Chekuri, Feder and Motwani ([11]). Charikar and Panigrahy revisited the incremental clustering problem ([15]) and presented a constant factor competitive incremental algorithm for the sum k-radius problem using $O(k)$ centers, they also proved that for the Incremental k-median any algorithm that maintains at most k centers has a competitive ratio of $\Omega(k)$. They left the following question as an open problem: does a constant factor competitive algorithm for Incremental k-median that maintains $O(k)$ centers exist?. Fotakis ([23]) gave a deterministic constant factor competitive algorithm using $O(k)$ centers for the incremental k-median problem thus resolving the problem posed in [15]. The algorithm of [14] can also be interpreted as an algorithm for the incremental problem under certain conditions but opens $O(k \log^2(n))$ centers. The natural problem of Incremental facility location was also introduced by Fotakis in [23] who surprisingly showed that there exists a constant factor competitive algorithm for the problem (despite the fact that the closely related problem of online facility location has a lower bound of $\Omega(\frac{\log(n)}{\log(\log(n))})$).

The idea of facilities being able to move was introduced by Diveki and Imreh ([18]). In their model the facilities are able to move with no extra cost. In [18] a constant factor competitive ratio for the problem was provided by computing an approximation of the optimal clustering at each time step and moving the facilities accordingly. In their recent work Feldkord et al. ([22]), motivated by the problem of page migration ([8], [7], [57]) introduced a new model for the online facility location with mobile facilities problem. In their model, whenever a facility is moved by the algorithm from a point a to a point b a cost of $Dd(a,b)$ is incurred. The constant $D \geq 1$ measures how costly the movements are, in instances where D is close to 1 in the optimal solution the facilities are expected to move a lot while in instances where D is large the problem can be reduced to the classical online facility location problem.

Another particularly interesting line of work towards surpassing the pessimistic lower bounds of online algorithms is that of learning augmented algorithms initiated (to our knowledge) in [43] and [46]. The framework of learning augmented algorithms has been applied to the facility location problem as well ([34], [28], [3])

It is important to note that that an effort to make the constraints of the algorithm less strict was introduced in [50] by Meyerson, who considered the random order model for online facility location where the sequence of demands is selected adversarially but is also randomly permuted before being provided to the algorithm. In that model [50] gave a constant factor competitive ratio of 8. Lang ([39]) strengthened this to show that against a t-bounded adversary the algorithm achieves a competitive ratio of $O(\log(t)/\log(\log(t)))$. The recent work of Kaplan, Naori and Raz ([36]) showed that Meyerson's algorithm is in fact 4 competitive in the random order model.

Let's now see what the cost of an algorithm is on the setting of online facility location with mobile facilities. The facility and movement cost is naturally the cost of the opened facilities (the algorithm is not allowed to close open facilities, only to move them) and the total distance that the facilities have moved times D. There are however 2 natural ways to define the assignment cost and both are considered in [22]:

1. *The instant service model*: The assignment cost for a demand u_i is its distance to the closest facility in $\mathcal{F}_i$. Note that this could and, in the general case will, be different from its distance from the closest facility in $\mathcal{F}_n$ (the facility configuration at the end of the demand sequence). This model captivates applications where clients arrive over time and they are serviced at their arrival time

2. *The delayed to end service model*: The assignment cost for a demand u_i is its distance from the facility that the demand was initially assigned to but by the end of the demand sequence. In other words when a demand arrives it is assigned irrevocably to a facility, the assignment cost is its distance from that facility by the end of the demand sequence. This model better captures applications related to clustering where we want to have a good clustering of demands by the end of the demand sequence.

It is important to note that in both cases the performance of the algorithm is measured compared to the static optimal offline solution, namely to the optimal facility location solution of the demand sequence. This is a strong assumption for the first model since there are instances where the algorithm can indeed outperform the offline benchmark. In worst case instances however any offline algorithm is worse than the static offline optimal solution.

In this thesis we will focus more on the first model, the instant service model. In Chapter 3 we will present the algorithm of [22] for the problem. We will consider the case where the metric space is a Euclidean space (this is essential for the algorithm presented in [22]) and we will show that their algorithm, EucOFLM for short, is optimal for the real line and postulate what needs to be proven in order for the algorithm to be asymptotically optimal for Euclidean spaces of arbitrary dimension. We will revisit the problem in Chapter 4 and show that the algorithm is indeed optimal for Euclidean spaces of arbitrary dimension.

The online facility location with mobile facilities problem was not studied on general metric space prior to this diploma thesis. In chapter 4 we provide an algorithm for that problem and show that it is asymptotically optimal

1.4 Our contribution

In this diploma thesis we propose the first (to our knowledge) asymptotically optimal algorithm for the online facility location with mobile facilities problem on general metric spaces. In our effort to do so we also resolve the open question of [22] on whether their algorithm is asymptotically optimal on Euclidean spaces of arbitrary dimension. The barrier we have to overcome in order to prove the optimality of the algorithm is that of analysing the cost of the prohibited demands (see Chapters 3 and 4 for a formal definition of prohibited demands). Prior to our work, the cost of the prohibited demands was shown to be $O(kc_f)$ (where k is the number of facilities in the optimal solution and c_f is the cost of opening a facility) only for the case of the real line and a weaker bound of $O(k^{3/2}c_f)$ was shown for Euclidean spaces of arbitrary dimension. We use the hierarchical decomposition lemma of [26] to show that the cost of prohibited demands is $O(kc_f)$ for any euclidean space. Furthermore, we generalize the algorithm of [22] to general metric spaces and show that the algorithm that we propose is asymptotically optimal for general metric spaces.

1.5 Organization

We will explore the following in the Chapters 2,3 and 4 of this thesis:

Chapter 2: We present the main results on the online facility location problem. First we present the lower bound of [26] and show that no randomized algorithm can achieve a competitive ratio better than $\Omega(\frac{\log(n)}{\log(\log(n))})$ even on simple metric spaces like tree metrics and the real line against an oblivious adversary. We continue to present Meyerson's algorithm [50] and prove that its competitive ratio matches the lower bound (up to constants). We introduce the optimal deterministic algorithm of [26]. We prove that the algorithm achieves an optimal competitive ratio (again up to constants) in the case that the optimal solution consists of a single facility.

Chapter 3: This chapter focuses on previous work on the online facility location with mobile facilities problem. We discuss how the lower bound for the online facility location problem can be adapted to show that in the model discussed no algorithm whose competitive ratio is independent of the number of demands can achieve a competitive ratio better than $O(\frac{\log(D)}{\log(\log(D))})$. We continue by presenting the algorithm of [22] and analyse its competitive ratio in the case where the underlying metric space is the real line. In that case we can rather easily exploit the structure of the real line to cope with the challenge of prohibited demands.

Chapter 4: This chapter contains our contribution to the online facility location with mobile facilities problem. We prove that the cost of prohibited demands is $O(kc_f)$ on any metric space, thus, answering the open problem of [22]. We also consider the problem of online facility location with mobile facilities on general metric spaces. We introduce a new algorithm for the problem and prove that our algorithm is asymptotically optimal for general metric spaces.

1.6 Preliminaries

1.6.1 Metric Spaces

We will introduce notions that will be useful throughout this diploma thesis. The first notion is that of a metric space. A metric space is a set $\mathcal{X}$ equipped with a function $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that the following conditions hold for every $x, y, z \in \mathcal{X}$:

1. *Non-negativity*: $d(x, y) \geq 0$
2. *Symmetry*: $d(x, y) = d(y, x)$
3. *Triangle inequality* $d(x, y) \leq d(x, z) + d(z, y)$
4. $d(x, y) = 0 \Leftrightarrow x = y$.

The elements of $\mathcal{X}$ are called points of the metric space and the function d is called the distance or metric function. It is often very important to consider sets for which the last condition does not hold (a prime example is cut metrics, for example [40]). We call such spaces semi-metrics or pseudometrics.

Let S be a subset of the metric space $\mathcal{X}$. By $d(S, u)$ we call the smallest possible distance between points of S and point u:

$$d(S, u) = \inf_{s \in S}\{d(s, u)\}$$

If the metric space is finite then:

$$d(S, u) = \min_{s \in S}\{d(s, u)\}$$

For a set S we denote $\Delta(S)$ the diameter of S:

$$\Delta(S) = \sup_{s, s' \in S}\{d(s, s')\}$$

Again if the metric space is finite we can have max instead of sup and:

$$\Delta(S) = \max_{s, s' \in S}\{d(s, s')\}$$

For a set $S \subseteq$ we define the separation of S denoted by $\text{sep}(S)$ as follows:

$$\text{sep}(S) = \inf_{s \in S, s' \notin S}\{d(s, s')\}$$

For finite metric spaces:

$$\text{sep}(S) = \min_{s \in S, s' \notin S}\{d(s, s')\}$$

1.6.2 Online Algorithms

In this diploma thesis we will study online algorithms for the online facility location problem (with and without mobile facilities). Before we concern ourselves with such algorithms lets back up a little bit to see what online algorithms are and how we measure their performance in the framework of competitive analysis.

In many practical problems the input is not known in advance but rather arrives in items, one item at a time. In such scenarios the algorithm must make irrevocable decisions in an online fashion every time a new ”item” of the input arrives. Consider as an example the following situation. You have taken up skiing as a hobby. Whenever you go skiing you can either buy the equipment for skiing or you can rent the equipment. Obviously buying the equipment is much more expensive that renting it, once you buy it however you need never again pay for it. How do you decide whether to buy the equipment or rent it when you go skiing. If someone knew in advance the exact number of times he would go skiing it would be easy for him to deduce whether the equipment needed to be bought or not. Without knowing the future however the problem is not as trivial and an online algorithm is involved.

In online computation the input consist of a sequence σ of items arriving over time. Whenever a new item arrives the algorithm must make an irrevocable decision without knowledge of future demands. A natural question is ”how do we measure the quality of an online algorithm?”. It is evident that in most interesting problems we cannot expect the algorithm to be the best possible, but how much worse can it be from the optimal. This is exactly the measure used in competitive analysis, the performance of the algorithm is compared to the offline optimal algorithm that had access to the entire sequence in advance. Furthermore, we assume that an adversary selects the input sequence in order to cause the algorithm to have a cost as high as possible when compared to the offline optimal. This way of analysing the performance of online algorithms falls under the category of worst case analysis.

Formally an online algorithm has a competitive ratio $C = C(n)$ if on any demand sequence the cost of the algorithm is at most C times the cost of an offline optimal algorithm that knew the entire sequence in advance. The goal is to design algorithms with as small as possible competitive ratio.

Chapter 2

Online Facility Location

2.1 Problem definition

In the online version of the classical facility location problem we consider an underlying metric space (X, d). At distinct points in time $t = 1, 2, ..., n$, (not necessarily distinct) demands $u_1, u_2, ..., u_n$ arrive as points of the metric space X. At every point in time the algorithm maintains a set of open facilities and whenever a new demand arrives the algorithm must either assign the demand to an open facility or open a new facility and assign the demand to it. The assignment of demands to facilities is irrevocable and open facilities cannot be closed once they are opened. We use $\mathcal{F}_i$ to denote the set of facilities opened by the algorithm right after demand u_i was processed. The cost of the algorithm is:

$$\sum_{i=1}^{n} d(\mathcal{F}_i, u_i) + c_f|\mathcal{F}_n|$$

Where c_f is the (uniform) cost of opening a facility.

2.2 Notation

In this section we introduce some notation. We will usually use $\mathcal{F}^*$ to denote the set of facilities in the optimal solution and we will denote facilities in $\mathcal{F}^*$ by f^*. Let u be a demand, then by f_u^* we denote the facility in the optimal solution that u is assigned to, we use $d_u^* = d(u, f_u)$ to denote the assignment cost for a demand u in the optimal solution. Furthermore, we will use Asg* to denote the optimal assignment cost and for a facility $f^* \in \mathcal{F}^*$ we use $\text{Asg}^*(f^*)$ to denote the cost of demands assigned to f^* in the optimal solution, that is if by S_{f^*} we denote the set of demands u assigned to f^* then:

$$\text{Asg}^*(f^*) = \sum_{u \in S_{f^*}} d(u, f^*) = \sum_{u \in S_{f^*}} d(u, f^*)$$

More generally if S is a set of demands then by $\text{Asg}^*(S)$ we denote the assignment cost of the demands in S in the optimal solution, that is:

$$\text{Asg}^*(S) = \sum_{u \in S} d_u^*$$

2.3 The lower bound

We will now proceed to show the lower bound for the competitive ratio of the online facility location problem. The metric space we employ for the construction of the lower bound is a Hierarchically-well-separated tree (see for example [6], [19]). More specifically, we consider the metric space induced by a complete binary tree of height h, such that the root's distance from its children is l and it decreases by a factor of m at every level, which means that at level j, the distance between nodes at level j and their children (at level $j+1$) is $\frac{l}{m^{j-1}}$ (See figure **??**). h, l, and m will be determined later. We use T_u to denote the subtree that has vertex u as its root. The adversary creates demands in phases. In the $j-$th phase , $0 \leq j \leq h$, the adversary creates a set of demands at level j. The total number of phases is $h+1$, one phase for every level of the tree. At every phase the adversary creates m^j demands on vertex v_j, meaning that the total number of demands is:

$$\sum_{i=0}^{h} m^i =$$
$$\frac{m^{h+1}-1}{m-1} =$$
$$\Theta(m^h)$$

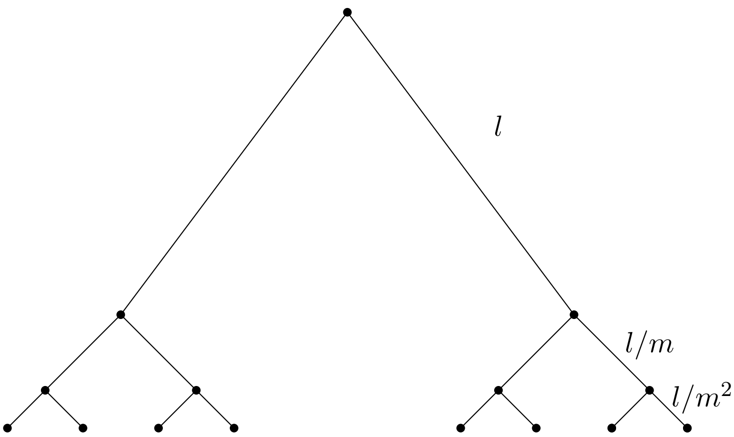


Figure 2.1: The hierarchically well separated tree

We will now specify how the vertices v_j are selected. The first vertex v_0 is the root of the tree, subsequently every vertex v_{j+1} is either the left or the right child of v_j. Specifically, v_{j+1} is the right child of v_j if the algorithm does not have a facility on the right subtree of v_j and the left child otherwise. What we wish to show is that the algorithm pays at least $\min(c_f, ml)$ per phase so that the total cost paid by the algorithm will be at least:

$$(h+1)\min(c_f, ml)$$

In order to do so we will show that for every phase we can either charge the algorithm with an assignment cost for the demands that is at least ml or we can charge it with a facility cost for a facility that has not been charged so far. For the first phase (phase 0) our charging scheme is simple: the algorithm has to open at least one facility to service the demands at v_0 so we charge the algorithm with the cost of opening that facility, c_f. Our charging scheme for phases $j \geq 1$ will be as follows:

1. If the algorithm does not have any facilities on the subtree T_{v_j} then it either has to pay an assignment cost that is at least:

$$\begin{aligned} m^j d(u_{j-1}, u_j) &= \\ m^j \cdot \frac{l}{m^{j-1}} &= \\ ml & \end{aligned}$$

   Or it opens a new facility to service the demands at v_j in which case we charge the algorithm with the cost of opening that facility

2. If the algorithm has an open facility on the subtree T_{v_j} then because of the policy under which we select the vertices v_j, we have that $v_j = left(v_{j-1})$ and that there is also a facility on the subtree $T_{right(v_{j-1})}$. We will show that we have not charged the algorithm for both facilities and therefore we can charge the cost of one of the facilities to the algorithm for that phase. To show that we will show that at any phase j the algorithm is charged for at most one facility on the subtree T_{v_j}

We have the following lemma:

Lemma 2.3.1. *At every phase j there is at most one facility of the algorithm charged to the algorithm in the subtree T_{u_j}*

Proof. We will prove the lemma by induction on j. For $j = 0$ we have that only one facility is charged to the algorithm. Suppose that the lemma holds for j. That is there is at most one facility of the algorithm charged to the algorithm at the subtree T_{u_j}. To prove that the lemma holds for $j + 1$ we divide between cases:

- If the algorithm has a facility in both $T_{left(v_j)}$ and $T_{right(v_j)}$ then without loss of generality we can assume that the facility charged to the algorithm is the one on the right subtree (by the induction hypothesis at most one of them is already charged to the algorithm). Then $v_{j+1} = left(v_j)$ and there is no facility charged to the algorithm on $T_{v_{j+1}}$. We charge the algorithm with the facility on the left subtree and the lemma holds.

- If the algorithm has a facility in only one of the two subtrees $T_{left(v_j)}$, $T_{right(v_j)}$ then the adversary will create the demands on the other subtree. In that case the algorithm either opens a facility (we charge the algorithm for that facility) or does not open a facility on that subtree and in both cases the lemma holds.

□

With the above lemma we conclude that any algorithm has a cost of at least $(h+1)\min(c_f, ml)$. On the other hand, an algorithm that opens a single facility on v_h pays a facility cost of c_f and an assignment cost for phase j:

$$\begin{aligned} m^j \sum_{i=j}^{h-1} \frac{l}{m^i} &\leq \\ m^j l \sum_{i=j}^{\infty} \frac{1}{m^i} &= \\ l \sum_{i=0}^{\infty} \frac{1}{m^i} &= \\ l \frac{1}{1-m^{-1}} &= \\ l \frac{m}{m-1} & \end{aligned}$$

The assignment cost over all phases is therefore at most $(h+1)l\frac{m}{m-1}$. Combining the assignment and the facility cost for the offline algorithm that opens a single facility on v_h we get $c_f + (h+1)l\frac{m}{m-1}$. We now determine the values of the parameters in order to get the lower bound. We set $m = h$ and $l = \frac{c_f}{h}$. The total number of demands ($\Theta(m^h)$ should not exceed n so we have that $m = h = \Theta(\frac{\log(n)}{\log(\log(n))}))$. Comparing the cost of the online and offline algorithms we get that the ratio of their costs is $\Omega(h) = \Omega(\frac{\log(n)}{\log\log(n)})$. It is not hard to show that a similar lower bound holds if the vertices v_j are not chosen according to the policy described but uniformly at random from the children of v_{j-1}. Assuming that, we can apply Yao's Principle (see for example [9], chapter 6, [59]) to get the same lower bound for randomized algorithms against an oblivious adversary. The metric space described above can be embedded to the real line with a small distortion, proving that the lemma also holds for the very simple metric space of the real line.

2.4 The randomized algorithm

We will now present and analyze the algorithm of [50] and prove that the algorithm achieves an asymptotically optimal competitive ratio of $O(\frac{\log(n)}{\log(\log(n))})$. The algorithm opens facilities according to the following simple facility opening rule: whenever a new demand u_i arrives it calculates the distance of the demand to the nearest open facility, $d = d(u_i, \mathcal{F}_{i-1})$ (recall that by $\mathcal{F}_i$ we denote the set of open facilities of the algorithm after demand u_i was processed), and opens a new facility at point u_i with probability $\min(1, \frac{d}{c_f})$.

Algorithm 5: Meyerson's algorithm (RANDOFL)

for $i = 1$ *to* n **do**
  let $d = d(u_i, \mathcal{F}_{i-1})$
  w.p $\min(1, \frac{d}{c_f})$ open a facility at u_i
end

The analysis of the algorithm requires the following lemma, which essentially states that the algorithm, on expectation, will pay at most assignment cost c_f on a set S before opening a facility on one of the points of S. There are various ways to prove this lemma, for example using expected waiting time techniques, [50] or potential function arguments, [27]. We follow the proof of [39]

Lemma 2.4.1. *Let S be any subset of demands and X be the random variable denoting the expected assignment cost until a facility is opened on one of the points of S. X is simply the sum of the assignment costs if no facility is opened. Then $\mathbb{E}[X] \leq c_f$. If we also account for the facility cost then the cost is at most $2c_f$.*

Proof. We prove the lemma by induction on the number of demands that S has. Let $v_1, v_2, ..., v_k$ be the demands in S in the order that they appear and $d_1, d_2, ..., d_k$ their respective distances from the nearest open facility at the time when they arrive, conditioned on the event that a facility is not opened on one of the demands of S yet. Note that they do not need to be consecutive demands since we want to prove the lemma for any set S. In the case where the set S has only one demand the lemma is trivial. If $d_1 \geq c_f$ then a facility is opened on v_1 deterministically, therefore the assignment cost in that case is 0. On the other hand, if $d_1 < c_f$, X is equal to 0 with probability $\frac{d_1}{c_f}$ and d_1 otherwise. Therefore:

$$\mathbb{E}[X] = (1 - \frac{d_1}{c_f})d_1 \leq d_1 \leq c_f$$

Suppose now that the lemma holds for any set S with $|S| = k$. Then for a set S' with $|S'| = k+1$ let O be the event that a facility is opened on v_1, using conditional expectation we have:

$$\mathbb{E}[X] = \mathbb{E}[X|O]Pr[O] + \mathbb{E}[X|\overline{O}](1 - Pr[O])$$

Again, we first divide between the cases that $d_1 \geq c_f$ and that $d_1 < c_f$. In the first case, $Pr[O] = 1$ therefore $\mathbb{E}[X] = \mathbb{E}[X|O]$, however $\mathbb{E}[X|O] = 0$ and the lemma holds. In the more interesting case that $d_1 < c_f$ the expectation of X is:

$$\begin{aligned}\mathbb{E}[X] = \frac{d_1}{c_f}\mathbb{E}[X|O] + (1 - \frac{d_1}{c_f})\mathbb{E}[X|\overline{O}] =& \\ (1 - \frac{d_1}{c_f})\mathbb{E}[X|\overline{O}]&\end{aligned}$$

Where we used the fact that by the definition of the random variable X if a facility is opened on v_1 then $X = 0$. Now to analyse the value $\mathbb{E}[X|\overline{O}]$ we make the following simple observation, if a facility is not opened on v_1 then the expected assignment cost for X is d_1 plus the expected assignment cost for the rest of the demands in the set $S' \setminus \{v_1\}$. Let $S = S' \setminus \{v_1\}$ then $|S| = k$. We now use the induction hypothesis. Let Y be the random variable that is equal to the expected assignment cost for demands on the set S until a facility is opened on one of them (Y is equal to the sum of the assignment costs if no facility is opened) using the induction hypothesis we have that $\mathbb{E}[Y] \leq c_f$. Putting everything together:

$$\begin{aligned} \mathbb{E}[X] &= \\ (1 - \frac{d_1}{c_f})(d_1 + \mathbb{E}[Y]) &\leq \\ (1 - \frac{d_1}{c_f})(d_1 + c_f) &= \\ d_1 - d_1 + \frac{d_1^2}{c_f} + c_f &\leq \\ & c_f \end{aligned}$$

If we also account for the cost of opening the facility we have that the cost is at most $2c_f$ $\square$

For the analysis of the algorithm we focus on a single center (facility) of the optimal solution and show that the RANDOFL pays at most $O(\frac{\log(n)}{\log(\log(n))})$ times the optimal solution. The first observation for the analysis is that for every demand that arrives the expected facility and assignment cost is bounded by $2d(u_i, \mathcal{F}_{i-1})$. This can easily be proven as follows: if a facility is opened on u_i then the algorithm pays cost c_f for the facility and no assignment cost. If on the other hand, a facility is not opened on u_i then the algorithm pays cost $d(u_i, \mathcal{F}_{i-1})$. The expected facility and assignment cost is:

$$\begin{aligned} d(u_i, \mathcal{F}_{i-1})(1 - \frac{d(u_i, \mathcal{F}_{i-1}}{c_f}) + c_f \frac{d(u_i, \mathcal{F}_{i-1})}{c_f} &\leq \\ & 2d(u_i, F_{i-1}) \end{aligned}$$

Note that this bound holds even in the case when $d(\mathcal{F}_{i-1}, u_i) > c_f$. Therefore we only need to bound the cost of $d(u_i, F_{i-1})$. Let m, h be parameters such that $m^h \geq n$. Similarly to the analysis of the lower bound, we divide the operation of the algorithm in phases. More specifically, we have $h + 2$ distinct phases. Let f^* be the optimal center that we focus on (by f^* we denote the facility as well as the point in the metric spaces that the facility is opened). Let δ^* be the average assignment cost on the cluster of demands assigned to f^* in the optimal solution. We make the two following simple observations: there is no demand assigned to f^* in distance greater than $n\delta^*$. To see this, let S_{f^*} be the set of demands assigned to f^* in the optimal solution and for the sake of contradiction assume that there is a demand at distance greater that $n\delta^*$ assigned to f^* then:

$$\begin{aligned} n\delta^* &\leq \\ \sum_{u_i \in S_f} d(u_i, f) &= \end{aligned}$$

$$\delta^*|S_f| \leq \\ \delta^* n$$

Where from the first to the second line we have used the fact that the assignment cost of a single demand is less than or equal to the sum of assignment costs over all demands. And from the third to the fourth line we have used the fact that the subset of demands assigned to f^* in the optimal solution is less than n. The second observation is that if a facility of RANDOFL is opened within distance δ^* from f^* then from then on we can bound the assignment cost of every other demand assigned to f^* in the optimal solution by the optimal assignment cost for those demands plus $k\delta^*$ where k is the number of demands assigned to f^* in the optimal solution that arrive after a has opened. The proof of this is again fairly simple, let a be the facility of RANDOFL that is within distance δ^* from f^*. Let $S = \{v_1, v_2, ..., v_k\}$ be the demands assigned to f^* in the optimal solution that arrive after a has opened. The assignment cost of RANDOFL for those demands is at most:

$$\sum_{i=1}^{k} d(a, v_i) \leq \\ \sum_{i=1}^{k} d(a, f^*) + d(f^*, v_i) \leq \\ k\delta^* + \sum_{i=1}^{k} d(f^*, v_i) = \\ k\delta^* + \sum_{i=1}^{k} d^*_{v_i} \leq \\ 2\,\mathrm{Asg}^*(f^*)$$

Where from the first to the second line we have used the triangle inequality, from the second to the third we have used that a is within distance δ^* from f^*. It is evident that after RANDOFL has opened a facility within distance δ^* from f^* it will pay at most twice $\mathrm{Asg}^*(f^*)$ therefore we would like for the algorithm to open a facility within distance δ^* from f^* as soon as possible. Note that this distance is the best we can hope for since the algorithm only opens facilities on demands. Equipped with these two observations we proceed to analyse the algorithm, recall that we are only focusing on a single facility.

We divide the state of the algorithm in phases with respect to the distance from the optimal center f^*. There are $h+2$ phases, we say that the algorithm is in phase j if there is a facility of the algorithm within distance $m^{j+1}\delta^*$ from f^* and no facility of the algorithm within distance $m^j\delta^*$. There is also a last phase -1 that starts after phase 0 and never ends. We divide demands while the algorithm is in phase j in two categories, namely inner and outer demands. Inner demands are demands that are at distance less than $m^j\delta^*$ from f^* the rest of the demands are outer. In the hth phase from our first observation there are only inner demands. We also consider every demand arriving during the last phase of the algorithm to be outer.

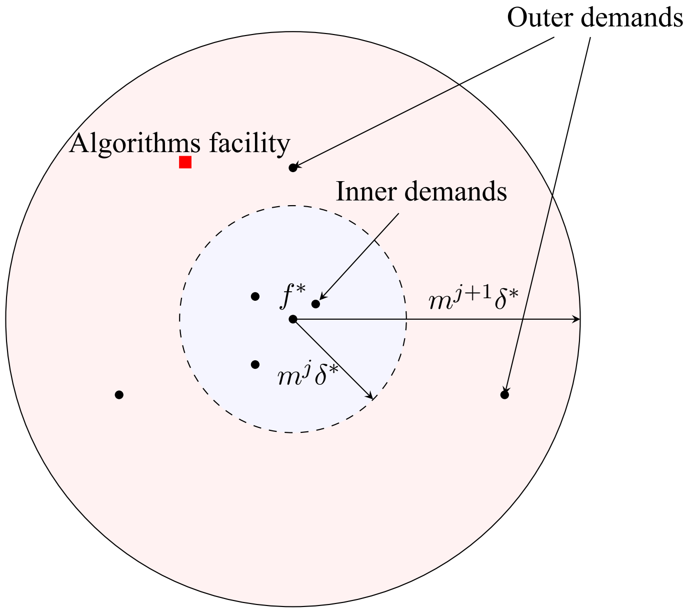


Figure 2.2: Inner and outer demands

It is easy to bound the assignment cost of outer demands using the triangle inequality, intuitively these are the demands that are relatively far from the optimal center. We continue to bound the assignment cost of RANDOFL for those demands. Let u be an outer demand then:

$$\begin{aligned} d(\mathcal{F}_{i-1}, u) \leq \\ d(\mathcal{F}_{i-1}, f) + d^*_u \leq \\ md^*_u + d^*_u = \\ (m+1)d^*_u \end{aligned}$$

Then the assignment and facility cost for outer demands is at most $2(m+1)d^*_u$.

On the other hand for inner demands by lemma 2.2.2 we have that after expected cost at most $2c_f$ a facility of RANDOFL will open within distance $m^j\delta^*$ and the phase will change. For the last phase since the algorithm has a facility opened in distance at most δ^* from f^* by the second observation the assignment cost for demands arriving in the last phase is at most $2\operatorname{Asg}^*(f^*)$. Summing over all phases we have that on expectation the algorithm pays at most:

$$\begin{aligned} 2(m+1)\operatorname{Asg}^*(f^*) + 2(h+1)c_f + 4\operatorname{Asg}^*(f^*) = \\ 2(m+2)\operatorname{Asg}^*(f) + 2(h+1)c_f \end{aligned}$$

We can now sum over all optimal facilities to get the upper bound for the algorithm:

$$2(m+2)\operatorname{Asg}^* + 2(h+1)|\mathcal{F}^*|c_f$$

We let $m = h = \Theta(\frac{\log(n)}{\log(\log(n))})$ and we get that the algorithm is $O(\frac{\log(n)}{\log(\log(n))})$ competitive.

2.5 The optimal deterministic algorithm

We will now present the deterministic algorithm of [26] that achieves the asymptotically optimal competitive ratio $O(\frac{\log(n)}{\log(\log(n))})$ and essentially closes the problem of online facility location. We will present the analysis for the case that the optimal solution consists of a single facility f^* and discuss why the straightforward generalization of this analysis could only yield a competitive ratio of $O(\log(n))$. This will present the need for the hierarchical decomposition lemma of [26] that we will explore further in Chapter 4.

The deterministic algorithm, DETFL for short, is a derandomization of RANDOFL. The algorithm maintains at every point in time a facility configuration $\mathcal{F}_{i-1}$ of the currently open facilities as well as a set L of unsatisfied demands. Unsatisfied demands are demands whose assignment cost has not yet contributed to the opening of a facility. Each unsatisfied demand has a contribution to the opening of a facility equal to its distance from the closest open facility. For an unsatisfied demand u we call this value the "potential" of u, $\text{Pot}(u) = d(\mathcal{F}, u)$. For a set S of unsatisfied demands we denote by $\text{Pot}(S)$ the sum of the potentials of demands in S, $\text{Pot}(S) = \sum_{u \in S} \text{Pot}(u)$. Whenever a new demand u_i arrives the algorithm calculates the potential of unsatisfied demands in a small ball around u_i, namely in a ball with center u_i and radius $\frac{d(\mathcal{F}_{i-1}, u_i)}{x}$ for sufficiently large x. If the potential exceeds c_f then the algorithm will open a new facility on one of the unsatisfied demands in the aforementioned ball. All the demands that contributed to the opening of a facility are removed from the set of unsatisfied demands, therefore each demand can contribute to the opening of a facility at most once. Intuitively the algorithm balances between the assignment and facility cost, whenever the assignment cost of a set of demands close to each other (relative to the facilities of the algorithm) exceeds c_f the algorithm opens a new facility on one of them. Specifically the algorithm opens a facility on the center of the smallest radius ball contributing at least half of the potential, if the distance to u_i is less than the cost of opening a facility and on u_i otherwise.

Similarly to the analysis of RANDOFL we consider parameters m, h such that $m^h > n$. We will then take $m = h = \Theta(\frac{\log(n))}{\log(\log(n))})$ and divide the operation of the algorithm in $h + 2$ distinct phases, namely $h, h-1, ..., 0, -1$. We consider sufficiently large $x \geq 10$ and sufficiently larger $\lambda \geq 3x + 2$. We again focus on the special case of a single optimal facility f^*, however this time this is not without loss of generality since the straightforward generalization of the analysis presented would only yield a suboptimal $O(\log(n))$ competitive ratio. We define δ^* to be the average assignment cost. The divison of phases is similar to the one in the analysis of RANDOFL. We say that the algorithm is in phase $j \neq -1$ if there is a facility of the algorithm within distance $\lambda m^{j+1}\delta^*$ and no facility of the algorithm within distance $\lambda m^j \delta^*$. We consider a final phase that begins after phase 0 and never ends. The demands while the algorithm is in phase j are divided between inner and outer demands. Inner demands u are demands such that $d(\mathcal{F}, u) \leq m^j \delta^*$ and are denoted by $\text{In}(f^*)$, the rest of the demands are outer. The first phase only has inner demands (as established in the analysis of RANDOFL there are no demands at

Algorithm 6: Deterministic Algorithm (DETFL)

Let x be an appropriately chosen constant
$\mathcal{F}_0 \leftarrow \emptyset$
$L \leftarrow \emptyset$
for $i = 1$ *to* n **do**
  let $d = d(u_i, \mathcal{F}_{i-1})$
  $L \leftarrow L \cup \{u_i\}$
  $r_{u_i} \leftarrow d/x$
  $B_{u_i} \leftarrow \text{Ball}(u_i, r_{u_i}) \cap L$
  $\text{Pot}(B_{u_i}) \leftarrow \sum_{u \in B_{u_i}} d(\mathcal{F}_{i-1}, u)$
  if $\text{Pot}(B_{u_i}) \geq c_f$ **then**
    if $d(\mathcal{F}_{i-1}, u_i) < c_f$ **then**
      Let $\nu \geq 0$ be the smallest integer such that:
        Either there exists exactly one point $u \in B_{u_i}$, such that:
        $\text{Pot}(B_{u_i} \cap \text{Ball}(u, r_{u_i}/2^{\nu})) > \text{Pot}(B_{u_i})/2$
        Or for every $u \in B_{u_i}$, $\text{Pot}(B_{u_i} \cap \text{Ball}(u, r_{u_i}/2^{\nu+1})) \leq \text{Pot}(B_{u_i})/2$
      Let $\hat{w}$ be any point in B_{u_i} such that:
        $\text{Pot}(B_{u_i} \cap \text{Ball}(\hat{w}, r_{u_i}/2^{\nu})) \geq \text{Pot}(B_{u_i})/2$
    end
    else
      $\hat{w} \leftarrow u_i$
    end
    $\mathcal{F}_i = \mathcal{F}_{i-1} \cup \{\hat{w}\}$
    $L \leftarrow L \setminus B_{u_i}$
  end
  Assign u_i to the nearest facility in $\mathcal{F}_i$
end

distance greater than $n\delta^*$). The last phase only has outer demands whose cost is at most 2- times the optimal assignment cost. We let Λ be the set of inner unsatisfied demands. Demands in Λ are much closer to the optimal solution and to each other than to the algorithms facilities. The algorithm maintains the following invariant. The potential of demands in Λ never exceeds c_f, that is $\text{Pot}(\Lambda) \leq c_f$. We have the following lemma:

Lemma 2.5.1. *At any point in time the potential of inner unsatisfied demands* $\text{Pot}(\Lambda)$ *does not exceed* c_f. *That is:*

$$\text{Pot}(\Lambda) \leq c_f$$

Proof. In the last phase there are no inner demands, therefore $\text{Pot}(\Lambda) = 0 \leq c_f$. We consider the case where the algorithm is in phase $j \neq -1$. Suppose for the sake of contradiction that at some point this was not the case, namely $\text{Pot}(\Lambda) > c_f$ and consider the last demand in Λ that arrived, say u_i. Since the potential of demands is non-increasing with time (the algorithm is not allowed to close facilities, only to open new ones) when u_i arrived the potential of demands in Λ exceeded c_f. It suffices to show that Ball(u_i, r_{u_i}) contains every point in Λ because then the potential Pot(B_{u_i}) would exceed c_f and a new facility would open making the potential of Λ to go to 0. In order to show that $\Lambda \subseteq$ Ball(u_i, r_{u_i}) we have to show that $r_{u_i} \geq 2m^j\delta^*$ (the radius of the ball must be greater than or equal to the diameter of the ball of inner demands). By the definition of r_{u_i} we have that:

$$\begin{aligned} r_{u_i} = d(\mathcal{F}_{i-1}, u_i)/x \geq \\ (d(\mathcal{F}_{i-1}, f^*) - d(f^*, u_i))/x \geq \\ (\lambda m^j\delta^* - m^j\delta)/x \end{aligned}$$

Where from the first to the second line we have used the triangle inequality and from the second to the third we have used that the algorithm is in phase j as well as that u_i is an inner demand. On the other hand the diameter of inner demands is at most $2m^j\delta^*$. Substituting and using the fact that $\lambda \geq 2x + 1$ we get the result. □

The assignment cost of outer demands is the easiest to bound. The following lemma states that the assignment cost of outer demands is at most within a factor $O(\lambda m)$ of the optimal assignment cost.

Lemma 2.5.2. *Let j be the phase of the algorithm. Then the following hold:*

- *If the algorithm is in the last phase then the cost of outer demands is at most $2\,\mathrm{Asg}^*(f^*)$ (recall that all demands in the last phase are outer)*
- *If the algorithm is in a phase j with $j \neq -1$ then the assignment cost for an outer demand is at most $(\lambda m + 1)d^*_{u_i}$*

Proof. For the first case where the algorithm is in the last phase let u_i be a demand arriving in the last phase of the algorithm. By our assumption that the algorithm is in the last phase we have that $d(\mathcal{F}_{i-1}, f^*) \leq \delta^*$, therefore by the triangle inequality the assignment cost is at most:

$$\begin{aligned} d(\mathcal{F}_{i-1}, u_i) \leq \\ d(\mathcal{F}_{i-1}, f^*) + d^*_{u_i} \leq \\ \delta^* + d^*_{u_i} \end{aligned}$$

Summing over all demands arriving in the last phase we get the bound.

For the second case consider a phase $j \neq -1$ and let u_i be an outer demand arriving while the algorithm is at phase j. Using the triangle inequality the assignment cost is at most:

$$\begin{aligned} d(\mathcal{F}_{i-1}, u_i) \leq \\ d(\mathcal{F}_{i-1}, f^*) + d^*_{u_i} \end{aligned}$$

However since u_i is an outer demand its optimal assignment cost is at least $m^j\delta^*$. Furthermore, the algorithm is in phase j therefore $d(\mathcal{F}_{i-1}, f^*) \leq \lambda m^{j+1}\delta^*$. We can conclude that:

$$d(\mathcal{F}_{i-1}, f^*) \leq \lambda m d^*_{u_i}$$

Hence:

$$d(\mathcal{F}_{i-1}, u_i) \leq (\lambda m + 1) d^*_{u_i}$$

$\square$

By the above lemma the contribution of outer demands to the opening of a facility can be bounded by $O(\lambda m)$ times the optimal assignment cost. This is not entirely the case since outer unsatisfied demands could have been inner at their assignment time and turned to outer when the phase changed. We can however bound the contribution in that case as well. Since the potential of Λ never exceeds c_f and the potential of a demand is non-increasing with time, the total contribution of outer demands that were initially inner is bounded by c_f times the number of phases. In other words, the potential of any such demand when it contributes to the opening of a facility is at most its potential when it turned from inner to outer. Since Pot$(\Lambda) \leq c_f$ for every phase the total contribution is at most $c_f(h+1)$.

Now we have to bound the cost for inner demands. Suppose that whenever the algorithm opened a new facility it also changed phase. If that were the case then since the potential of Λ never exceeds c_f the assignment cost of inner demands would be bounded by $(h+1)c_f$. Unfortunately this is not the case, a new facility might significantly decrease the potential of demands in Λ without changing the phase of the algorithm and we have to account for this decrease as well. In order to circumvent this difficulty we will show that whenever a new facility is opened either the phase changes or the contribution of outer demands in the opening of the facility was at least $cf/2$. We have argued above that the contribution of outer demands to the opening of facilities is within a factor $O(\frac{\log(n)}{\log(\log(n))})$ of the optimal solution (this consists of an $O(\lambda m) = O(\frac{\log(n)}{\log(\log(n))})$ contribution due to demands that were outer at their assignment time as well as an $h+1 = O(\frac{\log(n)}{\log(\log(n))})$ contribution due to outer demands that were initially inner)

Lets consider what are the costs the algorithm pays whenever a new facility is opened due to the arrival of a demand u_i. The total cost consist of the three following costs:

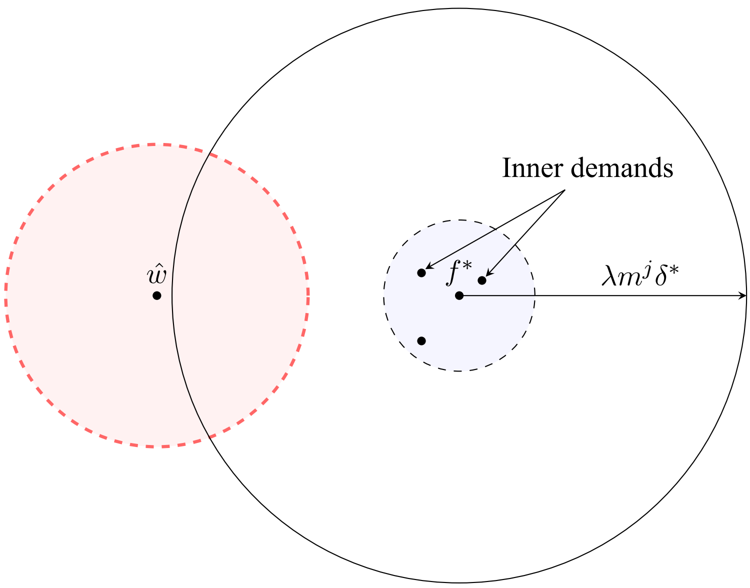
1. The assignment cost of demand u_i. We will show this to be at most $\frac{c_f}{x}$
2. The cost of opening the facility. This is obviously equal to c_f.
3. The decrease in the potential of inner demands. Since the potential of inner demands is at most c_f this is again at most c_f

Therefore ,the total cost paid by the algorithm whenever a new facility is opened is at most $\frac{2x+1}{x}c_f$. The proof that the assignment cost is at most c_f/x is very simple. If $d(u_i, \mathcal{F}_{i-1}) \geq c_f$ then the new facility is opened at u_i therefore the assignment cost is 0. If on the other hand $d(u_i, \mathcal{F}_{i-1}) < c_f$ then the facility is opened on some point of a ball with center u_i and radius $\frac{d(u_i, \mathcal{F}_{i-1})}{x}$ therefore the assignment cost is at most c_f/x. The following lemma formally states that we can charge the above costs either to the change of the phase or to outer demands.

Lemma 2.5.3. *Let u_i be a demand opening a new facility and let $\hat{w}$ be the point on which the facility is opened. Then either this new facility changes the phase of the algorithm or the contribution of outer demands to the opening of the facility is at least $c_f/2$.*

Proof. First consider the case that $d(\mathcal{F}_{i-1}, u_i) > c_f$. If that is the case then if u_i is an inner demand then because $\hat{w} = u_i$ the phase will change. If on the other hand u_i is an outer demand then its contribution to the opening of the facility is $c_f > c_f/2$ and the lemma holds as well.

We now focus on the most difficult case that $d(\mathcal{F}_{i-1}, u_i) \leq c_f$. The total contribution is divided into the contribution of outer demands and inner demands, which means that at least one of them must contribute at least half of the total contribution . If the outer demands contribute at least half of the potential then since $\text{Pot}(B_{u_i}) \geq c_f$ we are done. Therefore we can assume that outer demands contribute less than half of the potential, which in turn means that inner demands contribute at least half of the potential. We will assume that the phase does not change and this will lead us to a contradiction by showing that outer demands must contribute at least half of the potential (contrary to our assumption). Since every pair of inner demands are at distance at most $2m^j\delta^*$ there is a ball of radius at most $2m^j\delta^*$ that contributes at least half of the potential. In order for the phase to not change however the location of $\hat{w}$ must be at distance at least $\lambda m^j\delta^*$ from f^*. Furthermore the radius of the ball with center $\hat{w}$ containing at least half of the potential cannot exceed $4m^j\delta^*$ (Since otherwise the facility would open on one of the inner demands and the phase would change)



What we want to show now is that the situation is as in the figure, namely that the ball with center $\hat{w}$ that contributes at least half of the potential has no intersection with the ball of inner demands, this would in turn mean that outer demands contribute at least half of the potential and would lead us to a contradiction. Here the fact that we have taken λ sufficiently large comes into play, specifically we have that $\lambda \geq 32$ therefore:

$$d(\hat{w}, f^*) \geq$$
$$\lambda m^j\delta^* \geq$$
$$32m^j\delta^*$$

$$4m^j\delta^* + m^j\delta^*$$

Where $4m^j\delta^*$ is an upper bound for the radius of the ball around $\hat{w}$ and $m^j\delta^*$ is the radius of the ball of inner demands. $\square$

Now lets see more formally how we can use this lemma to bound the assignment cost of inner demands. Consider the set of inner demands S arriving between any two consecutive openings of facilities. Since by our assumption no facility opened in between, the potential of Λ is greater than or equal to the assignment cost of the demands in S and since the potential does not exceed c_f the assignment cost of the demands in S is at most c_f. Now when the algorithm opens the new facility the previous lemma assures us than one of two things happen, namely either the phase changes or the outer demands contribute at least half of the potential. If the phase changes, this can happen at most $h+1$ times and therefore the assignment cost for inner demands when this is the case can be bounded by $(h+1)c_f$ which is $O(\frac{\log(n)}{\log(\log(n))})$ times the optimal cost. On the other hand if the phase does not change then the outer demands contribute at least $c_f/2$ to the opening of a facility. Since the total contribution of outer demands is itself bounded by $O(\frac{\log(n)}{\log(\log(n))})$ times the optimal solution we can also bound the assignment cost of inner demands when this is the case by $O(\frac{\log(n)}{\log(\log(n))})$ times the optimal solution.

Lets now see why this approach can not be generalised to the general case of multiple centers in the optimal solution. Let $f_1^*, f_2^*, ..., f_k^*$ be the centers in the optimal solution. A careful examination of the analysis presented above would lead to the conclusion that all the lemmas proven above also hold for the case of multiple facilities. The difficulty in the analysis of the case with multiple facilities lies in the fact that a new facility opened by the algorithm might decrease the potential of inner demands belonging to multiple optimal centers. In that case while we can certify that for the centers that the phase does not change the decrease in the potential is at most c_f and it is still the case that outer demands contribute at least $c_f/2$ to the opening of a facility this is not enough to bound the decrease in the potential of inner demands of multiple centers that could be as much as $\Omega(kc_f)$.

One approach to circumvent that problem is to relax our requirement on the competitive ratio and more specifically on the phases of the algorithm. We could be less strict on the requirement of the phases being $O(\frac{\log(n)}{\log(\log(n))})$ and allow $O(\log(n))$ phases. In order to do so we will say that the algorithm changes phase, with respect to some optimal center f_j^*, whenever the distance of the closest facility of the algorithm decreases by a constant factor, say 3. Then instead of having $O(\frac{\log(n)}{\log(\log(n))})$ phases with respect to every facility we would have $O(\log(n))$ phases.

Chapter 3

Previous work on Online Facility Location with Mobile Facilities

3.1 Problem Definition and Notation

In this Chapter we explore already known results for the mobile facility location problem with mobile facilities, we focus on the *Instant Service Model*. We again consider an underlying metric space $(\mathcal{X}, d)$. Similarly to the classical online facility location problem at distinct points in time (not necessarily distinct) demands arrive as points of the metric space. Whenever a new demand arrives the algorithm can assign the demand to an already open facility, open a new facility or move a facility from one point of the metric space to another. The cost of opening a new facility is the same for every point of the metric space and the cost of moving the facility is a constant times the distance moved. The cost incurred to the algorithm is the assignment cost, namely the distance between every facility and its closest open facility at its service time, the facility cost, namely the number of facilities opened times the cost of opening a facility and the movement cost.

Lets recall some notation and also introduce some new notation. We will use c_f to denote the cost of opening a facility and D to denote the cost of moving a facility per unit distance, that is if a facility is moved from point a to point b then the algorithm pays a cost of $Dd(a, b)$. We will use f^* to denote centers (facilities) of the optimal solution. For a demand u we will use f^*_u to denote the center that the demand is assigned to in the optimal solution, furthermore we will use d^*_u to denote the assignment cost of demand u in the optimal solution, that is $d(f^*_u, u)$. If a is a facility of the algorithm we will use f^*_a to denote the facility in the optimal solution that was closest to a when the facility opened, we will say that a belongs to f^*_a. We will use $\mathcal{F}_i$ to denote the set of facilities opened by the algorithm right after the demand u_i was served (or right before demand u_{i+1} arrived). We will use $p_i(a)$ to denote the point in the metric space of a facility a after demand u_i was processed. For the case of the real line we will call the interval of some facility f^* the set of points that are closer to f^* than any other facility of the optimal solution.

As mentioned earlier, the cost paid by the algorithm by the end of the sequence is the sum of three different "types" of costs. The assignment cost, that is for every demand the distance to its closest facility right after the demand was processed denoted by Asg, the movement cost that is D times that total movement of the facilities. denoted by Mov and the facility cost that is $|\mathcal{F}_n| c_f$ and is denoted by Fac.

Essential to the analysis is the notion of prohibited and non-prohibited demands denoted by

Proh and NProh respectively. Let $y \geq 8$ be a constant and let f_i^*, f_j^* be two facilities of the optimal solution. We will say that a demand u is a prohibited request of f_i^* with respect to f_j^* if the following hold:

1. The demand u is assigned to f_i^* in the optimal solution ($f_u^* = f_i^*$) and is much closer to f_i^* than to f_j^*. Specifically:

$$d(f_i^*, u) \leq \frac{d(f_j^*, f_i^*)}{y}$$

2. The demand u is assigned to some facility z of the algorithm belonging to f_j^*, ($f_z^* = f_j^*$).

The importance of prohibited requests to the analysis will become clear later. The rest of demands are non-prohibited. In order to understand how prohibited and non-prohibited will be used in the analysis lets formulate what a non-prohibited demand is. Intuitively non-prohibited demands are either assigned to the optimal facility that the facility of the algorithm serving them belongs to or can be considered to be assigned to that optimal facility with only a constant factor loss (this is very easy to verify)

As we will see it can be proven that no algorithm can achieve a competitive ratio better than $\Omega(\frac{\log(D)}{\log\log(D)})$. The algorithm we will present was proven in [22] to achieve this ratio on the line and we will present the proof of this in this chapter.

3.2 The lower bound

The proof of the lower bound is conceptually the same as in the lower bound for the online facility location problem. We can take the parameter D to be approximately equal to the number of demands and get the bound. Another intuitive way to view this lower bound is to consider the hierarchically well separated tree in the construction of the lower bound in the previous chapter. The first demand of the sequence is at distance approximately c_f from the optimal facility. The assignment cost for demands that are "inner" is approximately the same as the distance between the algorithms facilities and the optimal facility. In order to make sense for a facility to move, the distance between the demand and the facility must be less than c_f/D and it would take $\log(D)/\log\log(D)$ phases to get from distance c_f to distance c_f/D while paying cost for outer demands approximately $\log(D)/\log\log(D)$. We have the following theorem:

Theorem 3.2.1. *No algorithm whose competitive ratio does not depend on n can be better than $\Omega(\frac{\log(D)}{\log(\log(D)})$-competitive*

Proof. We omit the proof. The reader is directed to [22] for a formal presentation of the arguments described above. $\square$

The theorem presented states that if we want to improve upon the previous lower bound of $\frac{\log(n)}{\log(\log(n))}$ and be independent of the number of demands then we cannot hope for something better than $O(\frac{\log(D)}{\log(\log(D))})$. Surprisingly however, this exactly what we will achieve.

3.3 The algorithm

The algorithm we will present combines ideas from the online facility location and the page migration problem. Consider for simplicity an instance where the optimal solution opens a single facility f^*. The high level idea is to use RANDOFL ([50]) in order to get at distance approximately c_f/D from the optimal facility and then use ideas from the page migration problem and the ability of the facilities to move in order to converge to the optimal solution.

The algorithm is as follows. First of all the algorithm only opens facilities in pairs and maintains two sets of facilities, namely the mobile and the static facilities denoted by $\mathcal{F}^m$, $\mathcal{F}^s$ respectively. Obviously it is the case that $\mathcal{F} = \mathcal{F}^m \cup \mathcal{F}^s$. Static facilities will never be moved by the algorithm after they are opened. Mobile facilities on the other hand are moved towards the current demand whenever the demand is sufficiently close. When the algorithm decides that a new facility is to be opened at some point a (this decision is made randomly with a rule much similar to the rule of RANDOFL) it in facts opens a pair of facilities on a, adding a both to the set $\mathcal{F}^m$ and to the set $\mathcal{F}^s$. For a facility $a \in \mathcal{F}^s$ we will use $\text{mob}(a)$ to denote the mobile facility in $\mathcal{F}^m$ corresponding to a. Whenever a new demand u_i arrives the algorithm considers the closest open static facility to the demand, say a, and divides between two cases. If the demand is sufficiently close (distance less than or equal to $2c_f/D$) to the static facility then the algorithm either opens a new pair of facilities on the demand or moves the facility towards the demand by a fraction $1/D$ of their distance (or both). We call such demands close demands. This operation of the algorithm makes it unsuitable for general metric spaces since it is essential that any point between to points of the metric space is also on the metric space. On the other hand if the demand is at distance greater than $2c_f/D$ then the algorithm simply opens a pair of facilities on u_i with probability $\frac{d(a,u_i)}{\beta c_f}$. We call such demands far demands. In pseudocode the algorithm is Algorithm 7

3.4 The analysis

We will now present the analysis of the algorithm. We will begin our analysis by sketching the case of a single facility f^* in the optimal solution. We will first prove the bound on outer demands (outer demands are demands that are distance at least c_f/D from their optimal center). Then we will describe the use of a potential function argument to bound the cost of the online algorithm for the inner demands (inner demands are demands that are at distance at most c_f/D from their optimal center). This will lead us to describe what problems arise in the generalization of this analysis, and this will lead us naturally to the notion of prohibited and non-prohibited demands and as we will see it will suffice to show that the cost of prohibited demands is bounded in order to generalize to more facilities in the optimal solution. This is easy to bound in the case of the real line. A bound on the cost of prohibited demands permits us to consider only the case of a single facility in the optimal solution for the rest of the analysis. Then we will again focus our

Algorithm 7: The algorithm for Euclidean metric spaces (EucOFLM)

Let β be an appropriately chosen constant
$\mathcal{F}_0^m \leftarrow \emptyset$
$\mathcal{F}_0^s \leftarrow \emptyset$
for $i = 1$ *to* n **do**
    Let $a = \arg\min_{a'}\{d(u_i, a') : a' \in \mathcal{F}^s\}$
    if $d(a, u_i) \le \frac{2c_f}{D}$ **then** /* u_i is a close demand */
        Let $z = \text{mob}(a)$
        w.p. $\frac{d(z,u_i)}{\beta c_f}$: $\mathcal{F}_i^s \leftarrow \mathcal{F}_{i-1}^s \cup \{u_i\}$, $\mathcal{F}_i^m \leftarrow \mathcal{F}_{i-1}^m \cup \{u_i\}$
        $move(z \rightarrow \frac{D-1}{D} z + \frac{1}{D} u_i)$
        Assign u_i to z
    end
    else /* u_i is a far demand */
        w.p. $\frac{d(a,u_i)}{\beta c_f}$: $\mathcal{F}_i^s \leftarrow \mathcal{F}_{i-1}^s \cup \{u_i\}$, $\mathcal{F}_i^m \leftarrow \mathcal{F}_{i-1}^m \cup \{u_i\}$
        Assign u_i to the facility opened at u_i.
    end
end

attention on a single optimal center and present a formal proof of the potential function argument.

In order to make the analysis more clear we will present the notion of the upper assignment cost denoted by Asg^+. The upper assignment cost is useful because the other costs of the algorithm can be bounded by a constant factor times the upper assignment cost of the algorithm. This reduces the problem of bounding all the costs of the algorithm to only bounding the upper assignment cost. The upper assignment cost of a demand u_i (the upper assignment cost of the algorithm is simply the upper assignment cost over all demands) is essentially the assignment cost that the algorithm would have to pay if it took no action after the demand arrived and simply maintained its previous configuration. More formally the upper assignment cost for a demand u_i is defined as follows. Let $a = \arg\min_{a' \in \mathcal{F}_{i-1}^s}\{d(a', u)\}$, that is the closest open static facility at the arrival time of u_i. The upper assignment cost for demand u_i is:

$$\text{Asg}^+(u_i) = \begin{cases} \min\{d(u_i, a), \beta c_f\} \text{ If } u_i \text{ is a far demand} \\ \min\{d(u_i, \text{mob}(a)), \beta c_f\} \text{ If } u_i \text{ is a close demand} \end{cases}$$

An important ingredient in the algorithm and its analysis is the basic lemma from Meyerson's algorithm. Using the ideas described in Chapter 2 we can easily prove the following lemma, formulated in terms of the upper assignment cost:

Lemma 3.4.1. *Let S be any subset of demands and X be the random variable denoting the expected upper assignment cost until a facility is opened on one of the points of S. X is simply the sum of the assignment costs if no facility is opened. Then $\mathbb{E}[X] \le 2\beta c_f$.*

Using this lemma we can easily bound the cost of outer demands. For inner demands the idea is to use a potential function argument to bound the cost of the solution. However as we will see the potential function charging cost will be a random variable depending on the algorithm's random choices. The constants of the algorithm are chosen in such a way so as to assure that on expectation the potential function charging cost will be at most $\frac{\mathrm{Asg}^+}{2}$. More specifically we charge every mobile facility z of the algorithm at distance at most $2c_f/D$ from the optimal facility that it belongs to with a potential of $\rho d(z, f_z^*)$. Since we only charge with a potential the facilities of the algorithm that are at distance at most $2c_f/D$ the potential initially charged to any such facility is at most $2\rho c_f$.

3.4.1 The costs of the algorithm

The costs of the algorithm consist of the assignment cost, the facility cost, the movement cost and the potential function charging cost. We will now show that the costs of the algorithm can be bounded by a constant factor times the upper assignment cost. We first show that the assignment cost is bounded by the upper assignment cost.

Lemma 3.4.2. *The assignment cost* Asg *is at most the upper assignment cost* Asg^+. *That is:*

$$\mathrm{Asg} \leq \mathrm{Asg}^+$$

Proof. Whenever a new demand u_i arrives the algorithm might do one of the following things if u_i is a far demand:

1. It might open a facility on u_i. In this case the assignment cost for the demand is 0 and therefore the upper assignment cost is greater than or equal to the actual cost.

2. It might not open a new facility in which case the assignment cost is equal to the upper assignment cost

On the other hand if the demand is a close demand then the algorithm might do one of the following things:

1. Open a facility on u_i. Again if this is the case then the upper assignment cost is greater than or equal to the actual assignment cost.

2. Not open a facility on u_i but move the mobile facility serving u_i towards u_i. This also makes the actual assignment cost greater than or equal to the actual assignment cost.

In every case the upper assignment cost of a demand u_i is greater than or equal to the actual assignment cost of the algorithm. Summing over all demand we get the result ☐

Now we will continue with the facility cost of the algorithm.

Lemma 3.4.3. *The expected facility cost of the algorithm* $\mathbb{E}[\mathrm{Fac}]$ *is bounded by a constant factor times the upper assignment cost, that is:*

$$\mathbb{E}[\mathrm{Fac}] \leq \frac{2}{\beta}\mathrm{Asg}^+$$

Proof. Whenever a demand, say u_i, arrives being at distance d from the facility it is assigned to a new pair of facilities is opened with probability min$(\frac{d}{\beta c_f}, 1)$. The expected facility cost if $d < \beta c_f$ is:

$$2c_f \frac{d}{\beta c_f} \leq \frac{2}{\beta} d = \\ \frac{2}{\beta} \mathrm{Asg}^+(u_i)$$

If on the other hand $d \geq \beta c_f$ then the facility cost is:

$$2c_f = \\ \frac{2\beta c_f}{\beta} = \\ \frac{2}{\beta} \mathrm{Asg}^+(u_i)$$

Summing over all demands u_i we get the result. □

Let Φ_0 be the random variable denoting the total potential function charging cost of the algorithm. The following lemma guarantees a bound on the expected value of Φ_0 by the upper assignment cost:

Lemma 3.4.4. $\mathbb{E}[\Phi_0] \leq \frac{\mathbb{E}[\mathrm{Asg}^+]}{2}$

Proof. Every demand u_i being at distance d from the facility it is assigned opens a new pair of facilities with probability min$(\frac{d}{\beta c_f}, 1)$. If $d > \beta c_f$ then a facility pair opens with probability 1 and the potential charging cost is:

$$2\rho c_f \leq \frac{\beta}{2} c_f \leq \frac{\mathrm{Asg}^+(u_i)}{2}$$

On the other hand if $d \leq \beta c_f$ then $2\rho c_f \cdot \frac{d}{\beta c_f} \leq \frac{d}{2} = \frac{\mathrm{Asg}^+(u_i)}{2}$ summing over all demands we have that:

$$\mathbb{E}[\Phi_0] \leq \frac{\mathrm{Asg}^+}{2}$$

□

Finally for the movement cost of the algorithm:

Lemma 3.4.5. *The expected movement cost is bounded by the upper assignment cost, that is:*

$$\mathbb{E}[\mathrm{Mov}] \leq \mathrm{Asg}^+$$

Proof. Let u_i be the demand arriving at round i, the algorithm will move a facility towards the demand only if the demand is a close demand. Let $z \in \mathcal{F}^m$ be the mobile facility servicing the demand u_i. The movement cost whenever u_i is a close demand is

$$\begin{aligned} D\frac{d(z,u_i)}{D} &= \\ d(z,u_i) &= \\ \text{Asg}^+(u_i) \end{aligned}$$

Again summing over all demands we get the result. □

It is evident that we only need to bound the upper assignment cost of the algorithm, then if we manage to show that on expectation the upper assignment cost is $O(\frac{\log(D)}{\log(\log(D))})$ OPT then a competitive ratio of $O(\frac{\log(D)}{\log(\log(D))})$ will easily follow.

3.4.2 The analysis of outer demands

We focus on a single optimal facility f^*. The idea is to use the guarantees of Meyerson's algorithm to prove that after cost $O(\frac{\log(D)}{\log(\log(D))})$ OPT there will be a facility at distance at most c_f/D from f^*.

Inner and outer demands. Here the definition of inner and outer demands is different from the definition of the online facility location problem. We say that a demand u_i is outer if $d(u_i, f^*) > c_f/D$. the rest of the demands are inner. Following the analysis of [27] we can easily bound the assignment cost of outer demands by $O(\frac{\log(D)}{\log(\log(D))})$ times the optimal solution.

A simple observation similar to the one we made in the analysis of RANDOFL and DETOFL is that for every demand u the assignment cost for u in the optimal solution is at most c_f. We will use the technique we used for Meyerson's algorithm to analyze the cost of outer demands.

There are outer demands that are also close demands, we will show that if an outer demand u_i is also a close demand, then the assignment cost for the demand if it is assigned to a mobile facility instead of a static is increased by at most $2d^*_{u_i}$.

Lemma 3.4.6. *The upper assignment cost of any outer demand that is assigned to a mobile facility (is close) is at most $2d^*_{u_i}$ greater than the upper assignment cost if it was assigned to the corresponding static facility*

Proof. Since u_i is an outer demand $d^*_{u_i} \geq c_f/D$, therefore by the triangle inequality it suffices to show that no mobile facility can be at distance greater than $2c_f/D$ from its corresponding static facility. Let a be a facility of the algorithm and $z = \text{mob}(a)$ its corresponding mobile facility. We will prove the statement by an induction on the number of demands served by z. If no demands where served by the mobile facility then the mobile and the static facility are at the same point and the statement holds. Assume that after l demands served by the mobile facility $d(a,z) \leq 2c_f/D$. Let u_i be the $(l+1)$-th demand served by z. The point that z will be after demand u_i is served is:

$$p_i(z) = \frac{D-1}{D}p_{i-1}(z) + \frac{1}{D}u_i$$

The distance between z after u_i is processed will be:

$$\begin{aligned}
d(a, p_i(z)) = \\
||a - p_i(z)|| = \\
||a - \frac{D-1}{D} p_{i-1}(z) - \frac{1}{D} u_i|| \leq \\
||\frac{D-1}{D}(a - p_{i-1}(z)) + \frac{1}{D}(a - u_i)|| \leq \\
\frac{D-1}{D}||a - p_{i-1}(z)|| + \frac{1}{D}||a - u_i|| = \\
\frac{D-1}{D}\frac{2c_f}{D} + \frac{1}{D}\frac{2c_f}{D} = \\
\frac{2c_f}{D}
\end{aligned}$$

Where from the fourth to the fifth line we have used the triangle inequality and from the fifth to the sixth line we have used that $d(a, p_{i-1}(z)) \leq 2c_f/D$ (by the induction hypothesis) and that $d(a, u_i) \leq 2c_f/D$ (because u_i was served by a mobile facility which means that it is a close demand) □

The above lemma guarantees that assigning outer demands to mobile facilities instead of static facilities is not too costly (in fact the algorithm would have asymptotically the same competitive ratio if there were no static facilities). Since this is the case, for the proof of the following lemma we will assume that every outer demand is assigned to the closest open static facility and the actual upper assignment cost will be at most 2 Asg* greater.

Lemma 3.4.7. *The expected upper assignment cost for outer demands is* $O(\frac{\log(D)}{\log(\log(D))})OPT$

Proof. We consider a single optimal facility of the algorithm f^*. We divide the points of the metric space at distance greater than $\frac{c_f}{D}$ with respect to their distance from f^*. We consider $h+1$ zones, the 0 zone is consisted of the points that are at distance greater than c_f from f^*. For $j \geq 1$ the j-th zone is consisted of all points p such that $\frac{c_f}{m^j} < d(f^*, p) \leq \frac{c_f}{m^{j-1}}$. We take $m = \frac{\log(D)}{\log(\log(D))}$ and $h = \Theta(\frac{\log(D)}{\log\log(D)})$ such that $m^h = D$ which means that every outer demand belongs to some zone. We say that the algorithm is in phase j with respect to the facility f^* if there is a static facility of the algorithm in zone j and no static facility of the algorithm in any zone j' with $j' > j$. Because there are no demands assigned to f^* in zone 0 after expected upper assignment cost $O(c_f)$ a facility of the algorithm will open inside a zone j with $j > 0$.

We will bound the assignment cost of the algorithm while in phase j. Consider a demand u_i arriving while the algorithm is in phase j. If the demand belongs to a zone j' with $j' \leq j$ then by the triangle inequality the upper assignment cost for u_i is at most $(m+1)d^*_{u_i}$. On the other hand for demands that belong to zone $j' > j$ after expected upper assignment cost at most $2\beta c_f$ a facility will open on one of these points and phase j will end. If we sum over all optimal centers and all phases it follows that the expected upper assignment cost of outer demands is at most $(m+1)Asg^* + (h+1)Fac^* = (\frac{\log(D)}{\log(\log(D))})OPT$.

If we now take into consideration that the demands might not be assigned to a static facility but rather to a mobile facility the cost grows only by at most 2 Asg*, therefore we can conclude that the upper assignment cost on outer demands is again on expectation $O(\frac{\log(D)}{\log(\log(D))}OPT)$. $\square$

3.4.3 The analysis of inner demands

Recall that an inner demand u_i is a demand such that $d(u_i, f^*_{u_i}) \leq c_f/D$. We wish to show that for every optimal center the cost of the algorithm on inner demands assigned to f^* is a constant times the optimal assignment cost. We focus on a single facility center f^*. After expected cost at most $O(c_f)$ on inner demands a facility is opened within distance $\frac{c_f}{D}$ from f^*. Thereafter any inner demand will be assigned to a facility being at distance at most $\frac{2c_f}{D}$ from f^*. Any such facility a is charged with a potential $\phi_a = \rho D d(z_a, f_a) \leq 2\rho c_f$. We call this the potential function cost of a. We call the sum of the potential function costs over every facility opened by the algorithm within distance $\frac{2c_f}{D}$ the potential function charging cost of the algorithm, Φ_0. We say that a demand is close if it is within distance $\frac{2c_f}{D}$ from its closest static facility. Note that by the time that a facility is opened within distance $\frac{c_f}{D}$ from f^* every inner demand is also a close demand whose closest static facility is at distance at most $2c_f/D$. So it suffices to bound the cost of close demands assigned to some mobile facility z whose corresponding static facility is at distance at most $2c_f/D$ from the optimal facility (and thus is charged with a potential $\rho D d(f^*_z, z)$).

The high level idea is to use a potential function argument to balance between two cases. The first case is that the inner demand u_i that arrived is far from the optimal center. When this is the case the algorithm pays approximately the same as the offline optimal solution, however the potential function of the mobile facility serving this demand might increase, the optimal assignment cost being high however compensates for that increase as well. The second case is that the inner demand u_i is much closer to the optimal facility than to the algorithms facility. When that is the case the upper assignment cost paid by the algorithm is much greater than the optimal assignment cost. However, the facility of the algorithm will move towards the demand and therefore towards the optimal facility (since those are very close) and the potential of the facility will decrease. This decrease in the potential will be enough to compensate for the assignment cost of the algorithm. Before we continue to formalize this proof sketch we will see why the straightforward generalization of this analysis to multiple facilities in the optimal solution is not enough to guarantee an asymptotically optimal competitive ratio.

There is a setback when considering multiple facilities in the optimal solution, a demand u_i assigned to a facility $f^*_{u_i}$ in the optimal solution might be assigned to a mobile facility z belonging to some different facility of the optimal solution, $f^*_z \neq f^*_{u_i}$. If this is the case but the demand is non-prohibited then we can assume that the demand was assigned to f^*_z in the optimal solution, this would only increase the cost of the "optimal" solution by a constant factor and the same arguments can be applied:

$$\begin{aligned} d(f^*_z, u_i) &\leq \\ d(f^*_z, f^*_{u_i}) + d(f^*_{u_i}, u_i) &\leq \\ y d^*(u_i) + d^*(u_i) &= \\ (y+1) d^*(u_i) \end{aligned}$$

Where we have used that u_i is a non prohibited demand therefore $d(u_i, f^*_{u_i}) > d(f^*_z, f^*_{u_i})/y$.

On the other hand prohibited demands are misleading for the algorithm: they direct the facilities of the algorithm towards a different center while they also have small assignment cost. Consider however the simple case of two facilities in the optimal solution, f_1^* and f_2^*.

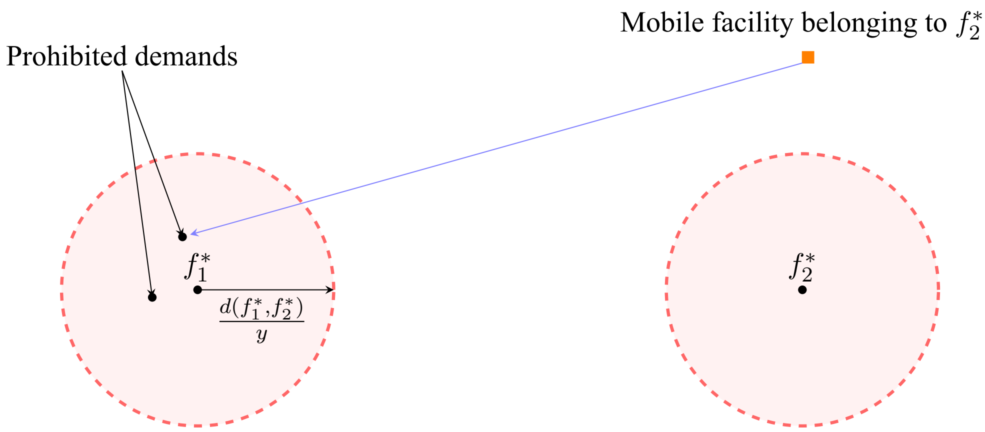


Figure 3.1: Prohibited demands of f_1^* with respect to f_2^*

Consider the prohibited demands of f_1^* with respect to f_2^*. After expected upper assignment cost $O(\beta c_f)$ on prohibited demands, a facility of the algorithm will open at distance at most $d(f_1^*, f_2^*)/y$ from f_1^*. Now consider a demand of f_1^* that could potentially be prohibited, meaning that it is at distance less than $d(f_1^*, f_2^*)/y$. Every static facility a belonging to f_2^* is at distance at least $d(f_1^*, f_2^*)/2$ from f_1^*. This is easy to verify:

$$\begin{aligned} d(f_1^*, f_2^*) &\leq \\ d(f_1^*, a) + d(f_2^*, a) &\leq \\ 2d(f_1^*, a) & \end{aligned}$$

Where we have used that a belongs to f_2^* therefore $d(f_2^*, a) \leq d(f_1^*, a)$. Rearranging we get the result. However when a static facility a' is opened at distance at most $d(f_1^*, f_2^*)/y$ from f_1^* the distance of any new demand, say u_i at distance at most $d(f_1^*, f_2^*)/y$ from f_1^* (a potentially prohibited demand) will not be assigned to a facility belonging to f_2^*. Recall that the algorithm finds the closest static facility to the new demand u_i, we will show that a' is closer from any static facility belonging to f_2^*. The distance between such a demand u_i and any static facility a belonging to f_2^* is at least:

$$\begin{aligned} d(a, u_i) &\geq \\ d(a, f_1^*) - d(u_i, f_1^*) &\geq \\ \frac{d(f_1^*, f_2^*)}{2} - \frac{d(f_1^*, f_2^*)}{y} &\geq \\ \frac{3}{8} d(f_1^*, f_2^*) & \end{aligned}$$

Where from the first to the second line we have used the triangle inequality, from the second to the third line we have used that a facility belonging to f_2^* is at distance at least $d(f_1^*, f_2^*)/2$ from f_1^*, finally from the third to the fourth line we have used that $y \geq 8$. On the other hand the distance of a' from u_i is, using the triangle inequality, at most $\frac{1}{4}d(f_1^*, f_2^*)$.

This analysis guarantees us that the expected upper assignment cost for the case of two facilities is $O(2\beta c_f)$. Furthermore it guarantees that if we select any pair of facilities f_i^*, f_j^* the expected upper assignment cost of prohibited demands of f_i^* with respect to f_j^* is at most $O(c_f)$. This analysis however does not immediately give us a good bound for the general case of k facilities in the optimal solution. If we naively used the same arguments the bound we would get would be $O(k^2 c_f)$ (this is constituted of an expected upper assignment cost of O(c_f) for every pair of optimal facilities), this is somewhat natural, when we generalize this argument to more facilities we make no use of the fact that the the optimal centers lie in an underlying metric space. For the case where the metric space is the real line we can exploit the structure of $\mathbb{R}$ to improve this to $O(kc_f)$.

Lemma 3.4.8. *The upper assignment cost of prohibited demands on the real line is $O(kc_f)$*

Proof. We let $f_1^* < f_2^* < ... < f_k^*$ be the positions of the optimal facilities on the real line. We follow the configuration of static facilities of the algorithm by a binary tree. Every node in the tree will correspond to a subset of optimal facilities that are consecutive. The tree will grow dynamically according to the openings of facilities of the algorithm. The root of the tree will contain all facilities of the optimal solution and at first the tree only contains the root. Consider a node u of the tree containing facilities $f_i^* < ... < f_j^*$. If a facility is opened at the interval of some facility f_l^* (recall that the interval of a facility is the set of points for which the facility is closer than any other facility in the optimal solution) with $i < l < j$ then we create two children for node u containing the facilities $f_i^* < ... < f_l^*$ and $f_l^* < ... < f_j^*$. After this event happens there will be no prohibited demands of f_m^* with respect to some facility $f_{m'}^*$ with $m < l < m'$. Similarly there will be no prohibited demands of $f_{m'}^*$ with respect to f_m^* (with $m < l < m'$). If on the other hand a leaf contains only two facilities, f_i^*, f_j^* then the node is split after a facility of the algorithm is opened at the prohibited area of f_i^* with respect to f_j^* (at distance at most $\frac{d(f_i^*, f_j^*)}{8}$ from f_i^*) and vice versa.

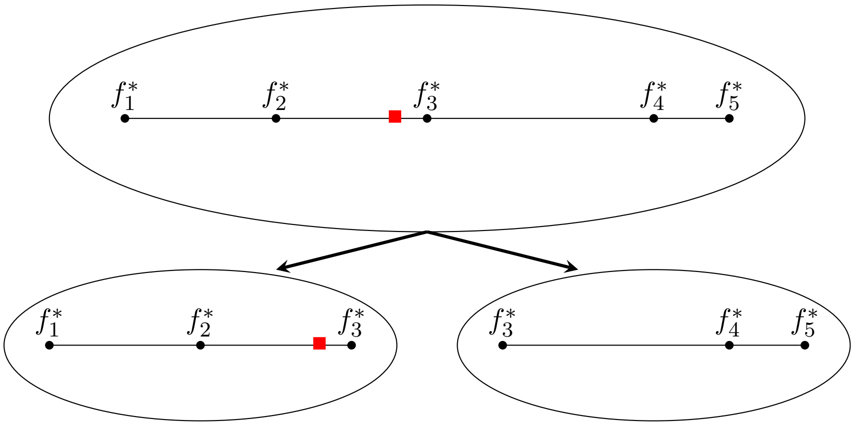


Figure 3.2: How the tree splits nodes when a facility is opened (red rectangle)

Note that for every prohibited demand of some facility f_i^* with respect to some facility f_j^* there is a unique leaf of the tree containing both f_i^* and f_j^*, therefore for any prohibited demand there is a unique leaf containing the two optimal facilities involved. We only charge leaves of the tree with the costs of prohibited demands. Our goal is to bound the cost on prohibited demands until a node is split (is no more a leaf) while also showing that the number of nodes is $O(k)$. Lets see what is the expected upper assignment cost until a leaf is split. Let u be a leaf with the facilities $f_i^* < ... < f_j^*$. We are only looking at prohibited demands from facilities within the node. Since the node is not yet split there are no facilities of the algorithm belonging to f_l^* for $i < l < j$. Therefore the only prohibited demands can be of f_i^* with respect to f_j^* or vice versa or from f_l^* with respect to either f_i^* or f_j^*. The cost of prohibited demands involving f_i^* and f_j^* is at most $2 \cdot 2\beta c_f$ (this accounts for $2\beta c_f$ for prohibited demands of f_i^* with respect to f_j^* and $2\beta c_f$ for prohibited demands of f_j^* with respect to f_i^*). On the other hand after expected upper assignment cost $2\beta c_f$ on prohibited demands of facilities f_l^* with respect to f_i^* and f_j^* a facility will be opened on the interval of one of them and the node will be split (a perfectly similar argument can show that after expected cost $4\beta c_f$ a node with two facilities will be split). We can conclude that the expected upper assignment cost on prohibited demands of every node until it is split is at most $6\beta c_f$. The number of leaves at any point is at most $2k$ which means that at any point the number of nodes of the tree is $O(k)$. By the end of the demand sequence the expected upper assignment cost on prohibited demands is $O(k)6\beta c_f = O(kc_f)$ □

Now that we have dealt with prohibited demands (at least on the real line) we are ready to present the main analysis. The proof we will show will go according to the sketch we gave at the beginning of the section. We will be able to show that when the upper assignment cost is much greater than the the actual assignment cost the configuration of the algorithm is significantly improved (the potential function is decreased). On the other hand when the algorithm pays approximately the same as the offline optimal solution then this compensates for the potential worsening of the configuration of the algorithm.

Lemma 3.4.9. *For every close inner demand u_i assigned to a mobile facility* $\mathrm{mob}(a)$ *the upper assignment cost plus the potential function difference is at most a constant times the optimal* *assignment cost if the demand is non prohibited. That is:*

$$d(u_i, \text{mob}(a)) + \Delta\Phi \leq (\rho + 1)x \cdot d^*_{u_i}$$

If the demand is prohibited the potential function difference is bounded by the upper assignment cost (meaning that the assignment plus the potential difference on every prohibited demand is bounded by the optimal facility cost)

Proof. We will focus on the simple case of the line (since our bound of prohibited demands also works only for this case), the general case is similar but a bit more technical. Let u_i be a close inner demand and $a = \arg\min_{a' \in \mathcal{F}^s}\{d(a, u_i)\}$, let $z = p_{i-1}(\text{mob}(a))$ and z' the place of the mobile facility after the movement ($z' = p_i(\text{mob}(a))$). We will first divide between the cases that the demand u_i is prohibited and the case that the demand u_i is non-prohibited.

Lets focus first on the case that the demand is non-prohibited, this means that either u_i is assigned to f^*_a in the optimal solution or it can be considered to be assigned to f^*_a in the optimal solution with only a constant factor increase of the cost. We will further divide between two cases. The first case is that the assignment cost for the demand u_i is much lower than its upper assignment cost, namely for large enough x:

$$d(u_i, f^*_a) \leq \frac{d(z, u_i)}{x}$$

What we wish to show when that is the case is that the expected upper assignment cost plus the potential function difference is less than or equal to 0. Note that the only facility that is moved is $\text{mob}(a)$ therefore $\Delta\Phi = \rho D(d(z', f^*_a) - d(z, f^*_a))$.

It is easy to verify that:

$$d(f^*_a, z') \leq \max\{d(f^*_a, z) - \frac{d(u_i, z)}{D}, \frac{d(u_i, f^*_a)}{x}\}$$

If $d(f^*_a, z') \leq d(f^*_a, z) - \frac{d(u_i,z)}{D}$ holds then the potential function difference is:

$$\begin{aligned} \Delta\Phi &= \\ \rho D(d(z', f^*_a) - d(z, f^*_a)) &\leq \\ -\rho D(\frac{d(u_i, z)}{D} &= \\ -\rho d(u_i, z) \end{aligned}$$

Note that $d(u_i, z)$ is precisely the upper assignment cost, meaning that for $\rho \geq 1$:

$$d(u_i, z) + \Delta\Phi \leq 0$$

If on the other hand $d(f^*_a, z') \leq \frac{d(u_i, f^*_a)}{x}$ then the potential function difference is:

$$\begin{aligned} \Delta\Phi &= \\ \rho D(d(z', f^*_a) - d(f^*_a, z)) &\leq \end{aligned}$$

$$
\begin{aligned}
\rho D(d(u, f_a^*) - d(f_a^*, z)) &\leq \\
\rho D(d(u_i, f_a^*) - d(z, u_i) + d(u, f_a^*)) &= \\
\rho D(2d(u_i, f_a^*) - d(z, u_i)) &\leq \\
\rho D(\frac{2}{x} d(z, u_i) - d(z, u_i)) &= \\
\rho D(\frac{2-x}{x} d(z, u_i))
\end{aligned}
$$

We take $x > 2$ therefore (because $D \geq 1$):

$$
\begin{aligned}
\rho D(\frac{2-x}{x} d(z, u_i)) &\leq \\
\rho \frac{2-x}{x} d(z, u_i) &= \\
-\rho \frac{x-2}{x} d(z, u_i)
\end{aligned}
$$

We take $\rho \geq \frac{x-2}{x}$ and we have:

$$d(z, u_i) + \Delta\Phi \leq 0$$

Now lets focus on the case that the upper assignment cost of the demand is approximately the same as the optimal assignment cost, namely:

$$d(z, u_i) \leq x \cdot d(f_a^*, u_i)$$

The increase in the potential function is:

$$
\begin{aligned}
\Delta\Phi &\leq \\
D\rho \frac{d(z, u_i)}{D} &= \\
\rho x d(f_a^*, u_i)
\end{aligned}
$$

Therefore the upper assignment cost plus the potential function difference is:

$$d(z, u_i) + \Delta\Phi \leq (\rho + 1) x d(u_i, f_a^*)$$

Since the demand is non-prohibited $d(u_i, f_a^*)$ is within a constant factor of the optimal assignment cost, meaning that $d(z, u_i) + \Delta\Phi$ is also within a constant factor of the optimal assignment cost.

Now for the case that the demand is prohibited. Since the expected upper assignment cost for prohibited demands is bounded by $O(kc_f)$ it suffices to show that the increase in the potential function is also bounded by the upper assignment cost. Then we can easily conclude that the expected upper assignment cost plus the potential function difference over all prohibited demands is $O(kc_f)$. This is very simple:

$$\Delta\Phi \leq D\rho \frac{d(z, u_i)}{D} \leq \rho d(z, u_i)$$

□

3.4.4 The Theorem

We have now concluded the proof of the asymptotically optimal competitive ratio of the algorithm on the real line. We will summarize and combine the lemmas in the following theorem to conclude that the competitive ratio of the algorithm for the real line is $O(\frac{\log(D)}{\log(\log(D))}OPT)$.

Theorem 3.4.10. *The expected cost of EucOFLM on the real line is:*

$$O(\frac{\log(D)}{\log(\log(D))})OPT$$

Proof. Recall that the expected costs of the algorithm are bounded by a constant factor times the upper assignment cost of the algorithm. Let C denote the cost of the algorithm:

$$\mathbb{E}[C] = \mathbb{E}[\text{Asg}] + \mathbb{E}[\text{Fac}] + \mathbb{E}[\text{Mov}] \leq O(\text{Asg}^+)$$

The expected upper assignment cost is:

$$\begin{aligned} \mathbb{E}[\text{Asg}^+] = \\ \mathbb{E}[\text{Asg}^+_{inner}] + \mathbb{E}[\text{Asg}^+_{outer}] \leq \\ O(\frac{\log(D)}{\log(\log(D))}OPT)O(kc_f) + (\rho+1)x\,\text{Asg}^* + \mathbb{E}[\Phi_0] \end{aligned}$$

We now use the fact that $\mathbb{E}[\Phi_0] \leq \mathbb{E}[\text{Asg}^+]$ we can get the bound:

$$\text{Asg}^+ \leq O(\frac{\log(D)}{\log(\log(D))}OPT)$$

□

Chapter 4

Online Facility Location with Mobile Facilities in general Metric Spaces

4.1 Introduction

In this section we present our work on the online facility location with mobile facilities problem. Our results include a generalization of the algorithm of [22]. We analyse the algorithm using the hierarchical decomposition lemma of [26] and prove that our algorithm is asymptotically optimal on general metric spaces. Our analysis also proves the optimality of the algorithm of [22] in Euclidean metric spaces of arbitrary dimension.

It is evident from the analysis of the EucOFLM algorithm that the main problem in generalizing to dimensions higher than 1 is the analysis of the cost of prohibited demands. In Chapter 3 we exploited the structure of the real line to prove that in that case the expected cost of prohibited demands is $O(kc_f)$ where k is the number of facilities in the offline optimal solution and c_f is the cost of opening a single facility (recall that we are focusing on the uniform case where the cost of opening a facility is the same for every point of the metric space). Our main result is the use of the hierarchical decomposition lemma of [26] to analyse the cost of prohibited demands in Euclidean spaces of arbitrary dimension and general metric spaces.

4.2 The algorithm

We will now present our algorithm that achieves the asymptotically optimal competitive ratio on general metric spaces. Similarly to the EucOFLM algorithm our algorithm maintains to set of facilities: the mobile facilities denoted by $\mathcal{F}^m$ and the static facilities $\mathcal{F}^s$. The mobile facilities will be moved by the algorithm with the goal to increase its configuration. The static facilities on the other hand will never move. Whenever the algorithm decides to open a facility at some point of the metric space (the decision will be made randomly according to the opening rule of RANDOFL, Meyerson's randomized algorithm [50]) it will instead open a pair of facilities at the same point: a static facility, that will never be moved and a mobile facility that will be moved throughout the execution of the algorithm. The way that the facilities are opened implies a pairing between mobile and static facilities. For a static facility a we denote by $\mathrm{mob}(a)$ the corresponding mobile facility, that is the mobile facility that was opened at the same time as a.

The algorithm classifies demands into two categories, close demands and far demands. Let

u_i be a demand, the algorithm calculates the closest open static facility to demand u_i, $a = \arg\min_{a' \in \mathcal{F}^s}\{d(a', u_i)\}$. If it holds that $d(a, u_i) \leq 2c_f/D$ then u_i is classified as a close demand. If on the other hand $d(a, u_i) > 2c_f/D$ then the demand is classified as a far demand. Far demands are treated in exactly the same way as in Meyerson's algorithm and are served by the closest static facility. Close demands on the other hand are served by mobile facilities and they might cause a facility to move.

Since we are trying to tackle the problem of general metric spaces, moving a facility to an arbitrary point on the line between 2 points of the metric space is no longer an option (this is what the algorithm of [22] does). We instead apply a very simple facility moving rule, we move the mobile facility to the point of the demand with probability $\frac{1}{D}$. In other words whenever a close demand u_i arrives if z is the mobile facility serving that demand then the algorithm flips a coin and with probability $\frac{1}{D}$ moves the facility on the point of the demand. In pseudocode the algorithm is Algorithm 8.

Algorithm 8: The algorithm for general metric spaces GenOFLM

Let β be an appropriately chosen constant
$\mathcal{F}_0^m \leftarrow \emptyset$
$\mathcal{F}_0^s \leftarrow \emptyset$
for $i = 1$ *to* n **do**
    Let $a = \arg\min_{a'}\{d(u_i, a') : a' \in \mathcal{F}^s\}$
    if $d(a, u_i) \leq \frac{2c_f}{D}$ **then** /* u_i is a close demand */
        Let $z = \text{mob}(a)$
        w.p. $\frac{d(z,u_i)}{\beta c_f}$: $\mathcal{F}_i^s \leftarrow \mathcal{F}_{i-1}^s \cup \{u_i\}$, $\mathcal{F}_i^m \leftarrow \mathcal{F}_{i-1}^m \cup \{u_i\}$
        w.p $\frac{1}{D}$: $move(z \to u_i)$
        Assign u_i to z
    end
    else /* u_i is a far demand */
        w.p. $\frac{d(a,u_i)}{\beta c_f}$: $\mathcal{F}_i^s \leftarrow \mathcal{F}_{i-1}^s \cup \{u_i\}$, $\mathcal{F}_i^m \leftarrow \mathcal{F}_{i-1}^m \cup \{u_i\}$
        Assign u_i to the facility opened at u_i.
    end
end

4.3 The analysis

We will now proceed to analyze the algorithm and show that it achieves an asymptotically optimal competitive ratio of $O(\frac{\log(D)}{\log(\log(D))})$.

4.3.1 Notation

Lets recall some notation that we also used on the previous chapter. We will use c_f to denote the cost of opening a facility and D to denote the cost of moving a facility per unit distance. For the set of facilities of the optimal offline solution we use the notation $\mathcal{F}^*$ and we use f^* to denote optimal facilities. Let u be a demand, we use f_u^* to denote the optimal facility that u is assigned to in the optimal solution. More generally for a point p of the metric space we use f_p^* to denote the closest facility of the optimal solution. For a demand u we use d_u^* to denote the optimal assignment cost of u ($d_u^* = d(u, f_u^*)$).

4.3.2 An outline of the analysis and definitions

We again define the notion of upper assignment cost, let u_i be a demand and a the closest open static facility at the arrival time of u_i. We define the upper assignment cost as:

$$\mathrm{Asg}^+(u_i) = \begin{cases} \min\{d(u_i, a), \beta c_f\} \text{ If } u_i \text{ is a far demand} \\ \min\{d(u_i, \mathrm{mob}(a)), \beta c_f\} \text{ If } u_i \text{ is a close demand} \end{cases}$$

We can bound the expected costs of the algorithm by a constant factor times the upper assignment cost, therefore if we show that the expected upper assignment cost is bounded by $O(\frac{\log(D)}{\log(\log(D))}OPT)$ then a bound of $O(\frac{\log(D)}{\log(\log(D))}OPT)$ will easily follow (by OPT we denote the optimal offline solution of the problem). Our analysis will go as follows. Lets focus our attention on a single facility of the optimal solution f^*. We will first categorize the demands into inner and outer. Inner demand are demands that are within a distance c_f/D from the facility they are assigned to in the optimal solution. The rest of the demands are outer. We will use the ideas from the analysis of Meyerson's algorithm [27] to show that the upper assignment cost of outer demands is bounded by $O(\frac{\log(D)}{\log(\log(D))}OPT)$

For inner demands we will show that after expected upper assignment cost $O(c_f)$ a pair of facilities will be opened at distance at most c_f/D from f^*. Thereafter every inner demand will be a close demand and furthermore every inner demand will be assigned to some mobile facility whose corresponding static is at distance at most $2c_f/D$ from f^*. We charge any such mobile facility with a potential of $\rho d(f^*, z)$. For inner demands after a pair of facilities was opened at distance at most c_f/D we will show the following if a demand is much closer to f^* than to the mobile facility of the algorithm then the configuration of the algorithm is significantly improved on expectation (the expected potential function difference compensates for the large upper assignment cost). If on the other hand the upper assignment cost of the algorithm is comparable to the optimal assignment cost then this also compensates for the (possible) increase of the potential function.

As mentioned earlier every mobile facility initially opened at distance at most $2c_f/D$ from its nearest optimal facility is charged with a potential charging cost of $\rho d(z, f_z^*)$. We call the sum over all such facilities the potential function charging cost of the algorithm, denoted by Φ_0. The expected potential function cost is bounded by $\frac{\mathrm{Asg}^+}{2}$. We will show that on expectation:

$$\mathbb{E}[\mathrm{Asg}^+] = O(\frac{\log(D)}{\log(\log(D))})\,\mathrm{OPT} + \mathbb{E}[\Phi_0] \Rightarrow$$

$$\mathbb{E}[\text{Asg}^+] = O(\frac{\log(D)}{\log(\log(D))})\,\text{OPT} + \mathbb{E}[\text{Asg}^+]/2 \Rightarrow$$
$$\mathbb{E}[\text{Asg}^+] = O(\frac{\log(D)}{\log(\log(D))})\,\text{OPT}$$

The main technical difficulty in our analysis is the analysis of prohibited demands. We recall now the definition of prohibited demands (denoted by Proh): Let $y \geq 8$ be a constant and let f_i^*, f_j^* be two facilities of the optimal solution. We will say that a demand u is a prohibited demand of f_i^* with respect to f_j^* if the following hold:

1. The demand u is assigned to f_i^* in the optimal solution $f_u^* = f_i^*$ and is much closer to f_i^* than to f_j^*. Specifically:

$$d(f_i^*, u) \leq \frac{d(f_j^*, f_i^*)}{y}$$

2. The demand u is assigned to some facility z of the algorithm belonging to f_j^*, ($f_z^* = f_j^*$).

The rest of the demands are the non-prohibited demands, denoted by NProh. Let u_i be a close demand and z the mobile facility that serves u_i, then either u_i is assigned to f_z^* in the optimal solution or it can be considered to be assigned to f_z^* in the optimal solution with only a constant factor increase of the cost.

Our algorithm heavily relies on Meyerson's algorithm to bound the cost of outer demands. It is therefore natural that the following lemma is essential to our analysis. Its proof is very similar to the one in lemma 2.4.1 presented in Chapter 2.

Lemma 4.3.1. *Let S be a set of demands. The expected upper assignment cost until a facility is opened at one of them is bounded by $2\beta c_f$.*

4.3.3 The costs of the Algorithm

We will now prove that the expected costs of the algorithm are bounded by a constant factor times the upper assignment cost, assuming that, we can only focus on the upper assignment cost for the analysis of our algorithm. It is very easy to see that the actual assignment cost of the algorithm is upper bounded by the upper assignment cost (for a formal proof see Lemma 3.4.2 from Chapter 3). For the facility cost of the algorithm:

Lemma 4.3.2. *The expected facility cost of the algorithm $\mathbb{E}[\text{Fac}]$ is bounded by a constant factor times the upper assignment cost, that is:*

$$\mathbb{E}[\text{Fac}] \leq \frac{2}{\beta}\,\text{Asg}^+$$

Proof. Whenever a demand, say u, arrives being at distance d from its serving facility (if the demand is far then d is the distance from the nearest open static facility while if the demand is close d is the distance from the mobile facility corresponding to the nearest open static facility).

A new pair of facilities is opened with probability $\min(\frac{d}{\beta c_f}, 1)$ so the expected facility cost if $d < \beta c_f$ is:

$$2c_f \frac{d}{\beta c_f} \leq \frac{2}{\beta} d = \frac{2}{\beta} \text{Asg}^+(u)$$

If on the other hand $d \geq \beta c_f$ then the facility cost is:

$$2c_f = \frac{2\beta c_f}{\beta} = \frac{2}{\beta} \text{Asg}^+(u)$$

Summing over all demands u we get the result. □

We now bound the potential function charging cost Φ_0 by the upper assignment cost of the algorithm:

Lemma 4.3.3. $\mathbb{E}[\Phi_0] \leq \frac{\mathbb{E}[\text{Asg}^+]}{2}$

Proof. Every demand u being at distance d from the facility it is assigned to opens a new pair of facilities with probability $\min(\frac{d}{\beta c_f}, 1)$. If $d > \beta c_f$ then a facility pair opens with probability 1 and the potential charging cost is:

$$2\rho c_f \leq \frac{\beta}{2} c_f \leq \frac{\text{Asg}^+(u)}{2}$$

On the other hand if $d \leq \beta c_f$ then $2\rho c_f \cdot \frac{d}{\beta c_f} \leq \frac{d}{2} = \frac{\text{Asg}^+(u)}{2}$. Summing over all demands we have that:

$$\mathbb{E}[\Phi_0] \leq \frac{\text{Asg}^+}{2}$$

□

Finally for the movement cost:

Lemma 4.3.4. *The expected movement cost is bounded by the upper assignment cost, that is:*

$$\mathbb{E}[\text{Mov}] \leq \text{Asg}^+$$

Proof. Let u be a close demand, a the closest static facility to u and z be the mobile facility serving u. The upper assignment cost for u is $d(u, z)$ ($\text{Asg}^+(u) = d(u, z)$). The facility z is moved with probability $\frac{1}{D}$ making the expected moving cost:

$$\frac{1}{D} D d(u, z) = \text{Asg}^+(u)$$

Summing over all demands we get the result. □

4.3.4 The analysis of outer demands

We will use the analysis of [27] of Meyerson's algorithm to analyze the expected cost of outer demands.

We say that a demand u is outer if $d(f_u^*, u) > \frac{c_f}{D}$. Following the analysis of [22] which in turn follows the analysis of [27] for Meyerson's algorithm we can easily prove that the expected upper assignment cost on outer demands can be bounded by $O(\frac{\log(D)}{\log\log(D))})OPT$. First of all we will show that for an outer demand u, the assignment cost if the demand is assigned to a mobile facility instead of the closest static facility is increased only by at most $2d_u^*$, this is guaranteed by the following lemma:

Lemma 4.3.5. *Let u be an outer close demand and let $a = \arg\min_{a'\mathcal{F}^s}\{d(a', u)\}$. Then the following inequality holds:*

$$d(u, \text{mob}(a)) \leq d(u, a) + d_u^*$$

Proof. Since u is an outer demand we have that $d_u^* > c_f/D$, furthermore since the mobile facility $\text{mob}(a)$ only moves on close demands we have that $d(\text{mob}(a), a) \leq 2c_f/D$ meaning that $d(\text{mob}(a), a) \leq 2d_u^*$. Using the triangle inequality we get the result □

In the following, for clarity, we will assume that every outer demand is also far. This, in fact, is not necessarily the case. However the cost of the algorithm can increase by at most $2\,\text{Asg}^*$ if outer demands are assigned to mobile facilities.

We consider a single optimal facility of the algorithm f^*. We divide the points of the metric space at distance greater than $\frac{c_f}{D}$ with respect to their distance from f^*. We consider $h+1$ zones. The 0th zone is consisted of the points that are at distance greater than c_f from f^*. For $i \geq 1$ the ith zone is consisted of all points p such that $\frac{c_f}{m^i} < d(f^*, p) \leq \frac{c_f}{m^{i-1}}$. We take $m = \frac{\log(D)}{\log(\log(D))}$ and $h = \Theta(\frac{\log(D)}{\log\log(D)})$ such that $m^h = D$ which means that every outer demand belongs to some zone. We say that the algorithm is in phase i with respect to facility f^* if there is a static facility of the algorithm in zone i and no static facility of the algorithm in any zone j with $j > i$. Because there are no demands assigned to f^* in zone 0 after expected upper assignment cost $O(c_f)$ a facility of the algorithm will open inside a zone i with $i > 0$.

We will bound the assignment cost of the algorithm while in phase j. Consider a demand u_i arriving while the algorithm is in phase j. If the demand belongs to a zone j' with $j' \leq j$ then by the triangle inequality the upper assignment cost for u_i is at most $(m+1)d_{u_i}^*$. On the other hand for demands that belong to a zone $j' > j$ by lemma 4.3.1 we have that after expected upper assignment cost at most $2\beta c_f$ a facility will open on one of these points and phase j will end. From the above analysis if we sum over all optimal centers and all phases it follows that the expected upper assignment cost on outer demands is at most $(m+1)\,\text{Asg}^* + (h+1)\,\text{Fac}^* = (\frac{\log(D)}{\log(\log(D))})\,\text{OPT}$

4.3.5 The analysis of inner demands

Now we will analyse the cost of inner demands. We will start by proving that the expected upper assignment cost of prohibited demands is $O(kc_f)$ where k is the number of facilities in the optimal solution.

We begin by giving some definitions related to prohibited demands. Let K be a set of facilities of the optimal solution. We call external prohibited demands of the set K the prohibited demands that are assigned to some facility $f_i^* \in K$ in the optimal solution but the algorithm assigns them to some facility belonging to some optimal facility f_j^* not in K, $f_j^* \notin K$. On the other hand internal prohibited demands are prohibited demands that are assigned to some facility $f_i^* \in K$ in the optimal solution but they are assigned to some facility of the algorithm belonging to some different optimal facility $f_j^* \in K$, $f_i^* \neq f_j^*$.

We now present the main tool of our analysis, namely the hierarchical decomposition lemma of [26]. A hierarchical decomposition is a complete laminar set system. A set system is laminar if it has no intersecting sets. A pair of sets K, K' form an intersecting pair if neither $K \setminus K'$, $K' \setminus K$, $K \cap K'$ is empty. In other words a set system is laminar if for any pairs of sets K, K' the sets are either disjoint or related by containment. A complete set system is a set system containing every singleton element $\{c\}$. We will think of this hierarchical decomposition as a rooted tree where every node of the tree represents a subset of points of the metric space. The root of the tree is the entire metric space and every node is either a leaf which means that it is a singleton set or the union of its children is equal to the node it self, furthermore the intersection between any pair of its children is the empty set. Formally, for a node u of the tree representing a subset $K \subseteq \mathcal{M}$ of the metric space either $K = \{c\}$ for some element $c \in \mathcal{M}$ or u is not a leaf and $K = \cup_{i=1}^{l} K_i$, where $K_1, ..., K_l$ are the children of set K. It also holds that $K_i \cap K_j = \emptyset$ for $1 \leq i < j \leq l$. It is easy to show that any complete laminar set system of a set S has at most $2|S| - 1$ sets. For a set K in the hierarchical decomposition other than $\mathcal{M}$ we use $\text{par}(K)$ to denote the parent of K in the corresponding tree representation.

Lemma 4.3.6. *For every $\gamma \geq 16$ every finite metric space $(\mathcal{M}, d)$ has a hierarchical decomposition $\mathcal{K}$ such that for every $K \in \mathcal{K}$ which is not $\mathcal{M}$ one of the following hold:*

1. *Either* $\Delta(K) > \frac{\Delta(\text{par}(K))}{\gamma^2}$
2. *Or* $\text{sep}_{\mathcal{M}}(K) > \frac{\Delta(\text{par}(K))}{4\gamma}$

We consider the metric space induced by the facilities in the optimal solution and apply lemma 4.3.6 on it. This gives a hierarchical decomposition of the facilities of the optimal solution. The idea is to charge every set of the hierarchical decomposition for the prohibited demands while the set is active (we will define later what a set being active means). Since there are $O(k)$ sets in the hierarchical decomposition if we show that the expected upper assignment cost on prohibited demands per set is $O(c_f)$ we will have that the expected cost on prohibited demands is $O(kc_f)$

We define two types of sets in the hierarchical decomposition. We call a set $K \in \mathcal{K}$ a set of type $\mathcal{A}$ if the following holds:

$$\Delta(K) > \min\{\frac{\Delta(\text{par}(K)}{\gamma^2}, \frac{\Delta(\text{par}(K))}{64\gamma}\}$$

We call a set $K \in \mathcal{K}$ a set of type $\mathcal{B}$ if:

$$\text{sep}(K) > \frac{\Delta(\text{par}(K))}{4\gamma}$$

and also:

$$\frac{\text{sep}(K)}{16} > \Delta(K)$$

Lemma 4.3.7. *Consider the hierarchical decomposition $\mathcal{K}$ of optimal centers guaranteed by lemma 4.3.6. For every $K \in \mathcal{K}$ and $K \neq \mathcal{M}$ is either a set of type $\mathcal{A}$ or a set of type $\mathcal{B}$*

Proof. Let $K \in \mathcal{K}$ be a set different than $\mathcal{M}$. Note that by lemma 4.3.6 one of the two conditions hold. If the first condition holds ($\Delta(K) > \frac{\Delta(\text{par}(K))}{\gamma^2}$) then K is trivially a set of type $\mathcal{A}$ since:

$$\Delta(K) > \frac{\Delta(\text{par}(K))}{\gamma^2} \geq \min\{\frac{\Delta(\text{par}(K))}{\gamma^2}, \frac{\Delta(\text{par}(K)}{64\gamma}\}$$

If on the other hand the second condition holds ($\text{sep}(K) > \frac{D(\text{par}(K))}{4\gamma}$) then we divide between two cases. If $\text{sep}(K) > \frac{D(\text{par}(K))}{4\gamma}$ then the set is of type $\mathcal{B}$.

If not, then $\text{sep}(K) \leq \frac{\Delta(\text{par}(K))}{4\gamma}$ which means that:

$$\begin{aligned} 16 \cdot \Delta(K) \geq sep_M(K) > \frac{\Delta(par(K))}{4 \cdot \gamma} \Leftrightarrow \\ \Delta(K) > \frac{\Delta(par(K))}{64 \cdot \gamma} \end{aligned}$$

Which makes K a set of type $\mathcal{A}$. $\square$

As mentioned earlier we will consider the metric space induced by the set of optimal facilities $\mathcal{F}^*$ and apply the lemma to this set. Let $\mathcal{K}$ be the hierarchical decomposition of $\mathcal{F}^*$. For every set $K \in \mathcal{K}$ we arbitrarily choose an optimal facility $f^*_K \in K$ to be the representative of the set. We say that a set K is active if $d(\mathcal{F}^s, f^*_K) > \lambda\Delta(K)$ while for every superset of K, $K' \in \mathcal{K}$ it holds that $d(\mathcal{F}^s, f^*_{K'}) \leq \lambda\Delta(K')$ where λ is a constant that is larger than $\frac{9}{8}$. Note that for every prohibited request of some center f^*_i with respect to some other center f^*_j there exists some active set $K \in \mathcal{K}$ such that $f^*_i \in K$. This holds because $\{f^*_i\} \in \mathcal{K}$. If the algorithm has opened a facility on f^*_i then a prohibited request of f^*_i with respect to some other facility can never occur. If on the other hand the algorithm has not opened a facility on f^*_i then $d(\mathcal{F}^s, f^*_i) > 0 = \lambda\Delta(\{f^*_i\})$ ($\Delta(\{f^*_i\}) = 0$) which means that the only way for the set $\{f^*_i\}$ to not be active is if a superset K' of it is active which in turn means that there is an active set containing f^*_i.

If we manage to show that for every set $K \in \mathcal{K}$ the expected upper assignment cost of prohibited demands assigned to some facility of K in the optimal solution while K is active is $O(c_f)$ then since $|\mathcal{K}| \leq 2k - 1$ the expected upper assignment cost of prohibited demands will be at most $O(kc_f)$ (recall that k is the number of facilities in the optimal solution). We have the following main lemma:

Lemma 4.3.8. *Let K be a set in $\mathcal{K}$. The upper assignment cost of prohibited demands while K is active that are assigned to a facility of K is $O(c_f)$.*

Proof. We divide between three cases for the set K. If K is the set of all optimal facilities $\mathcal{F}^*$ then since K is active we have that $d(\mathcal{F}^m, f^*_{\mathcal{F}^*}) > \lambda\Delta(\mathcal{F}^*)$. By lemma 4.3.1 after expected upper assignment cost at most $2\beta c_f$ on prohibited demands the algorithm opens a pair of facilities on one of them. Let u_i be the demand that caused this pair of facilities to open (both facilities are opened on u_i). Since u_i is a prohibited demand we have that there exist optimal facilities f^*_j, f^*_l such that:

$$d(u_i, f^*_j) \leq \frac{1}{8} d(f^*_j, f^*_l) \leq \Delta(K)$$

We have that:

$$d(f^*_{\mathcal{F}^*}, u_i) \leq d(f^*_{\mathcal{F}^*}, f^*_j) + d(f^*_j, u_i) \leq \Delta(K)$$

The algorithm opens a pair of facilities on u_i and since we have taken $\lambda \geq \frac{9}{8}$ the set $\mathcal{F}^*$ stops being active after expected upper assignment cost at most $2\beta c_f$. If K is not $\mathcal{F}^*$ then by lemma 4.3.7 K is either of type $\mathcal{A}$ or of type $\mathcal{B}$. Lets first consider the case that K is a set of type $\mathcal{A}$. By the definition of $\mathcal{A}$ sets:

$$\Delta(K) > \min\{\frac{\Delta(\text{par}(K))}{\gamma^2} \frac{\Delta(\text{par}(K))}{64\gamma}\} \Leftrightarrow$$
$$\max(\gamma^2, 64\gamma)\Delta(K) > \Delta(\text{par}(K))$$

We set for convenience $\Gamma = \max(\gamma^2, 64\gamma)$ and we have:

$$\Gamma \cdot \Delta(K) > \Delta(\text{par}(K))$$

Because K is active for $\text{par}(K)$ we have that:

$$d(\mathcal{F}^s, f^*_{\text{par}(K)}) \leq \lambda\Delta(\text{par}(K))$$

Therefore:

$$\begin{aligned} d(\mathcal{F}^s, f^*_K) &\leq \\ d(\mathcal{F}^s, f^*_{\text{par}(K)}) + d(f^*_K, f^*_{\text{par}(K)}) &\leq \\ \lambda\Delta(\text{par}(K)) + \Delta(\text{par}(K)) &\leq \\ (\lambda + 1)\Delta(\text{par}(K)) &\leq \\ (\lambda + 1)\Gamma \cdot \Delta(K) \end{aligned}$$

Overall since K is still active we have:

$$\lambda\Delta(K) < d(\mathcal{F}^s, f^*_K) \leq (\lambda + 1)\Gamma\Delta(K)$$

Let u be a prohibited demand assigned to some facility in K in the optimal solution and let a be the static facility of the algorithm that is closest to u. Since u is a prohibited demand a does not belong to $f_u^* \in K$ but to some other facility f_a^*. If for u it holds that $d(u, f_K^*) > \lambda\Delta(K)$ then its distance from f_u^* is by the triangle inequality:

$$d(u, f_u^*) \geq d(u, f_K^*) - d(f_u^*, f_K^*) \geq (\lambda - 1)\Delta(K)$$

On the other hand the distance to the nearest open static facility can be upper bounded as follows:

$$\begin{aligned} d(a, u) \leq \\ d(\mathcal{F}^s, f_K^*) + d(f_K^*, f_u^*) + d(f_u^*, u) \leq \\ (\lambda + 1)\Gamma \cdot \Delta(K) + \Delta(K) + d(f_u^*, u) \leq \\ ((\lambda + 1)\Gamma + 1)\Delta(K) + d(f_u^*, u) \leq \\ \frac{1}{\lambda - 1}((\lambda + 1)\Gamma + 1)d(f_u^*, u) + d(f_u^*, u) = \\ (\frac{(\lambda + 1)\Gamma + 1}{\lambda - 1} + 1)d(f_u^*, u) \end{aligned}$$

If the demand is a far demand then the upper assignment cost for it is only a constant factor greater than the optimal assignment cost. If on the other hand the demand is close this might not hold since the mobile facility might have moved and significantly increased its distance from the demand. We can show however that u can be considered to be assigned to f_a^* in the optimal solution with only a constant factor increase in the cost. (meaning that essentially the demand is non-prohibited only with a greater constant). In other words we will show that the distance from u to f_a^* is only a constant factor greater than the optimal assignment cost. First of all since a is a facility belonging to f_a^*, we have that:

$$\begin{aligned} d(f_u^*, f_a^*) \leq \\ d(f_u^*, a) + d(f_a^*, a) \leq \\ 2d(f_u^*, a) \end{aligned}$$

Now the distance between a and f_u^* is, by the triangle inequality, at most:

$$\begin{aligned} d(a, f_u^*) \leq \\ d(a, u) + d(u, f_u^*) \leq \\ (\frac{(\lambda + 1)\Gamma + 1}{\lambda - 1} + 2)d(f_u^*, u) \end{aligned}$$

Combining these two inequalities together we get:

$$d(f_u^*, f_a^*) \leq 2(\frac{(\lambda + 1)\Gamma + 1}{\lambda - 1} + 2)d(f_u^*, u)$$

Finally using the triangle inequality once again we can bound the distance from f_a^* to u by a constant factor times the optimal assignment cost:

$$d(f_a^*, u) \leq$$

$$
\begin{aligned}
d(f_u^*, f_a^*) + d(f_u^*, u) \leq \\
(2(\frac{(\lambda+1)\Gamma+1}{\lambda-1} + 2) + 1) d(f_u^*, u)
\end{aligned}
$$

Therefore assigning u to f_a^* can only increase the cost by a constant factor.

On the other hand if $d(f_K^*, u) < \lambda\Delta(K)$ then after expected upper assignment cost at most $2\beta c_f$ on such demands a facility will open on one of them and the set K will stop being active.

Now we focus our attention on sets of type $\mathcal{B}$. There are two types of prohibited demands that we have to consider for that case, namely external and internal prohibited demands. For external prohibited demands we will show that after expected upper assignment cost $O(c_f)$ the set will be "separated" and no more external prohibited demands will occur. On the other hand after expected cost $O(c_f)$ on internal prohibited demands the set will stop being active.

First of all since K is active we have that:

$$
d(\mathcal{F}^s, f_{\text{par}(K)}^*) \leq \lambda\Delta(K)
$$

Therefore, for the distance between the algorithms configuration and the optimal facility f_K^* we have:

$$
\begin{aligned}
d(\mathcal{F}^s, f_K^*) \leq \\
d(\mathcal{F}^s, f_{\text{par}(K)}^*) + d(f_{\text{par}(K)}^*, f_K^*) \leq \\
\lambda\Delta(\text{par}(K)) + \Delta(\text{par}(K)) \leq \\
(\lambda+1)\Delta(\text{par}(K)) \leq \\
(\lambda+1)4\gamma \, \text{sep}(K)
\end{aligned}
$$

Lets start with external prohibited demands. Let u be an external prohibited demand of set K and a the closest open static facility at the arrival time of u. If $d(f_u^*, u) \geq \text{sep}(K)/16$ then we have that:

$$
\begin{aligned}
d(a, u) \leq \\
d(\mathcal{F}^s, f_K^*) + d(f_K^*, f_u^*) + d(f_u^*, u) \leq \\
(\lambda+1)\Delta(\text{par}(K)) + \Delta(K) + d(u, f_u^*) \leq \\
(\lambda+1)4\gamma \, \text{sep}(K) + \frac{\text{sep}(K)}{16} + d(f_u^*, u) \leq \\
64(\lambda+1)\gamma d(f_u^*, u) + d(f_u^*, u) + d(f_u^*, u) = \\
(64(\lambda+1)\gamma + 2) d(f_u^*, u)
\end{aligned}
$$

Similarly to the case of $\mathcal{A}$ sets if u is a far demand for the algorithm then the upper assignment cost is within a constant factor of the optimal assignment cost. If on the other hand u is a close demand assigned to a mobile facility then we can show that u can be considered to be assigned to f_a^* with only a constant factor increase in the cost.

Lets now focus on the case that $d(u, f_u^*) \leq \frac{\text{sep}(K)}{16}$. After expected upper assignment cost $O(c_f)$ on such demands a pair of facilities of the algorithm will open at distance at most $\text{sep}(K)/16$

from some optimal facility of K. Let r be the position of the opened facilities that are at distance at most sep$(K)/16$ from an optimal center of K. Let u' be a prohibited demand after a static facility at r was opened and a' the closest open static facility at the arrival time of u'. If $d(f^*_{u'}, u') \geq$ sep$(K)/8$ then similarly to the previous case we can either show that the upper assignment cost is bounded by a constant factor times the optimal assignment cost (if u' is a far demand) or we can show that it can be considered to be assigned to f^*_a with only a constant factor increase of the cost (if u' is a close demand).

We have now to consider the case that $d(u', f^*_{u'}) \leq$ sep$(K)/8$, however, as we will show this can not be the case as long as u' is an external prohibited demand. Assume for the sake of contradiction that u' was indeed an external prohibited demand. Again, let a' be the closest open static facility at the arrival time of u', since u' is an external prohibited demand we have that $f^*_{a'} \notin K$. Since $f^*_{a'} \notin K$ the distance between $f^*_{u'}$ and $f^*_{a'}$ is at least sep(K) meaning that the distance between a' and $f^*_{u'}$ can be bounded from below by sep$(K)/2$. By the triangle inequality, the distance between a' and u' is at least:

$$\begin{aligned} d(u', a') \geq \\ d(f^*_{u'}, a') - d(f^*_{u'}, u') \geq \\ \text{sep}(K)/2 - \text{sep}(K)/8 = \\ \frac{3\,\text{sep}(K)}{8} \end{aligned}$$

On the other hand the distance between r and u' can be bounded from above as follows:

$$\begin{aligned} d(r, u') \leq \\ d(r, f^*_r) + d(f^*_r, f^*_{u'}) + d(f^*_{u'}, u') \leq \\ \frac{\text{sep}(K)}{16} + \Delta(K) + \frac{\text{sep}(K)}{8} \leq \\ \frac{\text{sep}(K)}{16} + \frac{\text{sep}(K)}{16} \frac{\text{sep}(K)}{8} \leq \\ \frac{\text{sep}(K)}{4} \end{aligned}$$

Which means that the static facility opened at r is closer to u' than a' leading to a contradiction since we assumed that a' is the closest open static facility to u'.

The last case to consider is that of internal prohibited demands. After expected upper assignment cost $O(c_f)$ on internal prohibited demands a static facility will open on one of them. Let u be the demand that caused the facility to open. We have assumed that u is a prohibited demand meaning that there is a pair of facilities $f^*_j \in K$ and $f^*_l \in K$ with u being assigned to f^*_j in the optimal solution but assigned to a facility of the algorithm belonging to f^*_l. By the definition of prohibited demands the facility opened at u will be at distance at most $\frac{d(f^*_j, f^*_l)}{8} \leq \frac{\Delta(K)}{8}$ from f^*_j. If we recall that we have taken λ to be greater than or equal to $\frac{9}{8}$ then:

$$\begin{aligned} d(f^*_K, u) \leq \\ d(f^*_K, f^*_u) + d(f^*_u, u) \leq \end{aligned}$$

$$\Delta(K) + \frac{\Delta(K)}{8} =$$
$$\frac{9}{8}\Delta(K) \leq$$
$$\lambda\Delta(K)$$

Making K no longer active. □

Having proved that the expected upper assignment cost of prohibited demands is indeed bounded by $O(kc_f)$ we can continue to analyse the cost of inner demands. We return our attention to a single optimal facility f^*. Recall that after expected upper assignment cost at most $O(c_f)$ on inner demands every inner demand is also a close demand assigned to some facility at distance at most $2c_f/D$ from the f^*. We will therefore focus our attention to close demands assigned to a mobile facility whose corresponding static facility is at distance at most $2c_f/D$ from f^* (these are the facilities charged with a potential).

We have the following lemma:

Lemma 4.3.9. *Let u be a close demand assigned to a mobile facility z whose corresponding static facility is at distance at most $2c_f/D$ from f^*. Then the upper assignment cost plus the expected potential function difference is at most a constant times the optimal assignment cost if the demand is non prohibited. That is:*

$$d(u,z) + \mathbb{E}[\Delta\Phi] \leq (\rho+1)x \cdot d_u^*$$

If the demand is prohibited the expected potential function difference is bounded by the upper assignment cost (meaning that the upper assignment cost plus the potential function difference on every prohibited demand is bounded by the optimal facility cost)

Proof. Let u be a close demand and $a = \arg\min_{a' \in \mathcal{F}^s}(d(a',r))$ In the following by z we mean the place of the mobile facility before the movement to distinguish between the place of the mobile facility after the facility movement which is the place of the demand r (if such a movement occurs). Because u is a close demand $d(z,u)$ is the upper assignment cost of demand u. We first analyse the value of $\mathbb{E}[\Delta\Phi]$. The only facility that might change its position is z in this iteration. Since the facility is moved with probability $\frac{1}{D}$, $\Delta\Phi$ is a random variable with distribution:

$$\Delta\Phi = \begin{cases} \rho D(d(u,f^*) - d(z,f^*)) \textbf{ w.p } \frac{1}{D} \\ 0 \textbf{ w.p } \frac{D-1}{D} \end{cases}$$

So we can easily deduce that $\mathbb{E}[\Delta\Phi] = \rho(d(u,f^*) - d(z,f^*))$ We begin with the case that the demand is prohibited. We have already shown that the expected cost on prohibited demands is $O(\text{Fac}^*)$ but we also have to account for the potential function difference due to those demands. It suffices to show that the potential function difference is bounded by a constant factor times the upper assignment cost of prohibited demands. Indeed using the triangle inequality:

$$\mathbb{E}[\Delta\Phi] = \rho(d(u,f^*) - d(z,f^*)) \leq \rho d(z,u)$$

And thus the expected upper assignment cost as well as the expected potential function difference for prohibited demands are bounded by $O(\text{Fac}^*)$.

We now turn our attention to non prohibited demands. We divide between two cases. We first consider the case that $d(u, f^*) \leq \frac{d(u,z)}{x}$ so that the demand is much closer to the optimal center than it is to the mobile facility z. If that is the case then the expected potential function difference will suffice to show that $d(u, z) + \mathbb{E}[\Delta\Phi] \leq 0$. Specifically:

$$\begin{aligned}\mathbb{E}[\Delta\Phi] \leq \rho(d(u, f^*) - d(z, f^*)) \leq \\ \rho(d(u, f^*) - d(z, u) + d(u, f^*)) \leq \\ \rho(2d(u, f^*) - d(z, u)) \leq \\ \rho\frac{2-x}{x}d(z, u) = \\ -\rho\frac{x-2}{x}d(z, u)\end{aligned}$$

Taking $\rho \geq \frac{x-2}{x}$ we have that:

$$d(z, u) + \mathbb{E}[\Delta\Phi] \leq 0 \leq (\rho + 1)x \cdot d_u^*$$

On the other hand if $d(u, f^*) > \frac{d(u,z)}{x} \Leftrightarrow d(u, z) < x \cdot d(u, f^*)$ then we can use that, together with the fact that $\mathbb{E}[\Delta\Phi] \leq \rho d(z, u)$ to bound the algorithm's assignment cost together with the expected potential function difference:

$$\begin{aligned}d(u, z) + \mathbb{E}[\Delta\Phi] \leq \\ (\rho + 1)d(u, z) \leq (\rho + 1)x \cdot d(u, f^*) = \\ (\rho + 1)x \cdot d_u^*\end{aligned}$$

□

4.3.6 Putting everything together

We will now combine all the above lemmas to prove the theorem that our algorithm obtains an asymptotically optimal competitive ratio.

Theorem 4.3.10. *On Expectation the algorithm achieves a competitive ratio of* $O(\frac{\log(D)}{\log(\log(D))})$

Proof. The cost of the algorithm is: (by Alg we denote the cost of the algorithm)

$$\mathbb{E}[\text{Alg}] = \mathbb{E}[\text{Mov}] + \mathbb{E}[\text{Fac}] + \mathbb{E}[\text{Asg}] \leq O(\text{Asg}^+)$$

Therefore as we also mentioned earlier we only need to bound the upper assignment cost of the algorithm, furthermore it suffices to bound the upper assignment cost for inner and outer demands:

$$\mathbb{E}[\text{Asg}^+] = \mathbb{E}[\text{Asg}^+_{inner}] + \mathbb{E}[\text{Asg}^+_{outer}] \tag{4.1}$$

As we have shown, the expected upper assignment cost of outer demands is $O(\frac{\log(D)}{\log(\log(D))})OPT$. On the other hand the expected upper assignment cost of inner demands can be bounded by:

$$\mathbb{E}[\text{Asg}^{+}_{inner}] \leq O(\text{Fac}^{*}) + \mathbb{E}[\text{Asg}^{+}(\text{Proh})] + \mathbb{E}[\text{Asg}^{+}(\text{NProh})]$$

Where the $O(\text{Fac}^{*})$ term comes from the $O(c_f)$ cost for every optimal facility until a static facility is opened at distance at most c_f/D from that facility. We can use lemma 4.3.9 and lemma 4.3.8 to bound the cost of the sum of prohibited and non prohibited demands:

$$\begin{aligned}\mathbb{E}[\text{Asg}^{+}(\text{Proh})] + \mathbb{E}[\text{Asg}^{+}(\text{NProh})] \leq \\ O(\text{Fac}^{*}) + O(\text{Asg}^{*}) + \mathbb{E}[\Phi_0] \leq \\ O(\text{Fac}^{*}) + O(\text{Asg}^{*}) + \mathbb{E}[\text{Asg}^{+}]/2\end{aligned}$$

Substituting into (4.1) we get:

$$\begin{aligned}&\mathbb{E}[\text{Asg}^{+}] \leq O(\frac{\log(D)}{\log(\log(D))})OPT + O(\text{Fac}^{*}) + O(\text{Asg}^{*}) + \mathbb{E}[\text{Asg}^{+}]/2 \Leftrightarrow \\ &\mathbb{E}[\text{Asg}^{+}] \leq O(\frac{\log(D)}{\log(\log(D))})OPT\end{aligned}$$

□

Bibliography

[1] Hamid Afshari and Qingjin Peng. Challenges and solutions for location of healthcare facilities, 2014.

[2] Amir Ahmadi-Javid, Pardis Seyedi, and Siddhartha S. Syam. A survey of healthcare facility location, 2017.

[3] Matteo Almanza, Flavio Chierichetti, Silvio Lattanzi, Alessandro Panconesi, and Giuseppe Re. Online facility location with multiple advice, 2021.

[4] Aris Anagnostopoulos, Russell Bent, Eli Upfal, and Pascal Van Hentenryck. A simple and deterministic competitive algorithm for online facility location, 2004.

[5] Michel Louis Balinski. On finding integer solutions to linear programs, 1964.

[6] Yair Bartal. Probabilistic approximation of metric spaces and its algorithmic applications, 1996.

[7] Marcin Bienkowski, Jarosław Byrka, and Marcin Mucha. Dynamic beats fixed: On phase-based algorithms for file migration, 2019.

[8] David L Black and Daniel D Sleator. *Competitive algorithms for replication and migration problems*. Carnegie-Mellon University. Department of Computer Science, 1989.

[9] Allan Borodin and Ran El-Yaniv. *Online computation and competitive analysis*. cambridge university press, 2005.

[10] Jaroslaw Byrka and Karen Aardal. An optimal bifactor approximation algorithm for the metric uncapacitated facility location problem, 2010.

[11] Moses Charikar, Chandra Chekuri, Tomás Feder, and Rajeev Motwani. Incremental clustering and dynamic information retrieval, 1997.

[12] Moses Charikar, Sudipto Guha, Éva Tardos, and David B. Shmoys. A constant-factor approximation algorithm for the k-median problem (extended abstract), 1999.

[13] Moses Charikar, Samir Khuller, David M Mount, and Giri Narasimhan. Algorithms for facility location problems with outliers, 2001.

[14] Moses Charikar, Liadan O'Callaghan, and Rina Panigrahy. Better streaming algorithms for clustering problems, 2003.

[15] Moses Charikar and Rina Panigrahy. Clustering to minimize the sum of cluster diameters, 2001.

[16] Fabián A Chudak and David P Williamson. Improved approximation algorithms for capacitated facility location problems, 1999.

[17] MS Daskin, AP Hurter, and MG Van Buer. Toward an integrated model of facility location and transportation network design, the transportation center, 1993.

[18] Gabriella Divéki and Csanád Imreh. Online facility location with facility movements, 2011.

[19] Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics, 2003.

[20] Uriel Feige. A threshold of ln n for approximating set cover, jul 1998.

[21] Björn Feldkord and Friedhelm Meyer Auf Der Heide. The mobile server problem, 2019.

[22] Björn Feldkord, Till Knollmann, and Friedhelm Meyer auf der Heide. Online facility location with mobile facilities. *Theoretical Computer Science*, 2022.

[23] Dimitris Fotakis. Incremental algorithms for facility location and k-median, 2006.

[24] Dimitris Fotakis. Memoryless facility location in one pass, 2006.

[25] Dimitris Fotakis. A primal-dual algorithm for online non-uniform facility location, 2007.

[26] Dimitris Fotakis. On the competitive ratio for online facility location, 2008.

[27] Dimitris Fotakis. Online and incremental algorithms for facility location, 2011.

[28] Dimitris Fotakis, Evangelia Gergatsouli, Themis Gouleakis, and Nikolas Patris. Learning augmented online facility location, 2023.

[29] Sudipto Guha and Samir Khuller. Greedy strikes back: Improved facility location algorithms, 1999.

[30] Dorit S Hochbaum. Heuristics for the fixed cost median problem, 1982.

[31] Kamal Jain, Mohammad Mahdian, Evangelos Markakis, Amin Saberi, and Vijay V Vazirani. Greedy facility location algorithms analyzed using dual fitting with factor-revealing lp, 2003.

[32] Kamal Jain, Mohammad Mahdian, and Amin Saberi. A new greedy approach for facility location problems, 2002.

[33] Kamal Jain and Vijay V Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and lagrangian relaxation, 2001.

[34] Shaofeng H. C. Jiang, Erzhi Liu, You Lyu, Zhihao Gavin Tang, and Yubo Zhang. Online facility location with predictions, 2022.

[35] Jain Kamal and Vijay V Vazirani. An approximation algorithm for the fault tolerant metric facility location problem, 2000.

[36] Haim Kaplan, David Naori, and Danny Raz. Almost tight bounds for online facility location in the random-order model, 2022.

[37] Madhukar R Korupolu, C Greg Plaxton, and Rajmohan Rajaraman. Analysis of a local search heuristic for facility location problems, 2000.

[38] Alfred A Kuehn and Michael J Hamburger. A heuristic program for locating warehouses, 1963.

[39] Harry Lang. Online facility location against a t-bounded adversary, 2018.

[40] Tom Leighton and Satish Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms, nov 1999.

[41] Shi Li. A 1.488 approximation algorithm for the uncapacitated facility location problem, 2013.

[42] Jyh-Han Lin and Jeffrey Scott Vitter. Approximation algorithms for geometric median problems, 1992.

[43] Thodoris Lykouris and Sergei Vassilvitskii. Competitive caching with machine learned advice, 2020.

[44] Mohammad Mahdian, Yinyu Ye, and Jiawei Zhang. Improved approximation algorithms for metric facility location problems, 2002.

[45] Alan S Manne. Plant location under economies-of-scale—decentralization and computation, 1964.

[46] Andrés Muñoz Medina and Sergei Vassilvitskii. Revenue optimization with approximate bid predictions, 2017.

[47] Sanjay Melkote and Mark S. Daskin. Capacitated facility location/network design problems, 2001.

[48] M.T. Melo, S. Nickel, and F. Saldanha da Gama. Facility location and supply chain management – a review, 2009.

[49] Ramgopal R Mettu and C Greg Plaxton. The online median problem, 2003.

[50] Adam Meyerson. Online facility location, 2001.

[51] Chandrashekhar Nagarajan and David P Williamson. Offline and online facility leasing, 2013.

[52] Christos Papadimitriou and Mihalis Yannakakis. Optimization, approximation, and complexity classes, 1988.

[53] David B Shmoys, Éva Tardos, and Karen Aardal. Approximation algorithms for facility location problems, 1997.

[54] John F Stollsteimer. A working model for plant numbers and locations, 1963.

[55] Maxim Sviridenko. An improved approximation algorithm for the metric uncapacitated facility location problem, 2002.

[56] Jens Vygen. *Approximation algorithms facility location problems*. Forschungsinstitut für Diskrete Mathematik, Rheinische Friedrich-Wilhelms …, 2005.

[57] Jeffery Westbrook. Randomized algorithms for multiprocessor page migration, 1994.

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

[59] Andrew Chi-Chin Yao. Probabilistic computations: Toward a unified measure of complexity, 1977.