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

Το Πρόβλημα Βελτιστοποίησης Εξωτερικών Επιδράσεων από τη Σκοπιά της Μεγιστοποίησης της Διαφοράς μιας submodular και μιας modular Συνάρτησης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΠΑΝΑΓΙΩΤΗΣ Γ. ΑΪΒΑΣΙΛΙΩΤΗΣ

Επιβλέπων : Αριστείδης Παγουρτζής
Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2022

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

Το Πρόβλημα Βελτιστοποίησης Εξωτερικών Επιδράσεων από τη Σκοπιά της Μεγιστοποίησης της Διαφοράς μιας submodular και μιας modular Συνάρτησης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΠΑΝΑΓΙΩΤΗΣ Γ. ΑΪΒΑΣΙΛΙΩΤΗΣ

Επιβλέπων : Αριστείδης Παγουρτζής
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 22η Ιουλίου 2022.

.....	.....	.....
Αριστείδης Παγουρτζής Καθηγητής Ε.Μ.Π.	Δημήτριος Φωτάκης Καθηγητής Ε.Μ.Π.	Ευάγγελος Μαρκάκης Αν. Καθηγητής Ο.Π.Α.

Αθήνα, Ιούλιος 2022

Παναγιώτης Γ. Αϊβασιλιώτης

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

Περίληψη

Από τις πρώτες κιόλας ερευνητικές μελέτες σε προβλήματα βελτιστοποίησης submodular συναρτήσεων, το συνεχές ενδιαφέρον για τεχνικές βελτιστοποίησης με μεγάλη πρακτική βαρύτητα σε περιοχές με την μεγαλύτερη επιρροή όπως Μηχανική Μάθηση, Αλγοριθμική Θεωρία Παιγνίων, Κοινωνικά Δίκτυα κ.α., έχει κάνει επιτακτική την ανάγκη εύρεσης νέων τρόπων έκφρασης τέτοιων συναρτήσεων. Το ενδιαφέρον μας πηγάζει από μια μελέτη πάνω σε συναρτήσεις της μορφής $g = f - c$, όπου f είναι μια γ-weakly submodular συνάρτηση που αποτελεί γενίκευση των submodular συναρτήσεων και c μια modular συνάρτηση. Ειδικότερα, θεωρούμε το πρόβλημα μεγιστοποίησης μιας συνάρτησης $g = f - c$, με την c να είναι συνάρτηση πληθικότητας και την f να αποτελεί την αντικειμενική συνάρτηση μιας οικογένειας προβλημάτων που ορίζουμε ονόματι p-Max k-hop Domination, όπου δοθέντων ενός γραφήματος G και ακεραίων p, k ο στόχος είναι να βρεθεί ένα σύνολο D από p κορυφές του G, έτσι ώστε το σύνολο των κορυφών που βρίσκονται σε απόσταση το πολύ k ακμές από κάποια κορυφή του D, να μεγιστοποιείται. Παρέχουμε μια μέθοδο για να προσεγγίσουμε την βέλτιστη τιμή της g, με διάφορους αλγορίθμους οι οποίοι διαφέρουν ως προς τον τρόπο με τον οποίο αποκτούμε αυτό που αποκαλούμε *αποσύνθεση* ενός στιγμιότυπου. Τα αποτελέσματά μας μπορούν να χρησιμοποιηθούν για να βελτιώσουν τον λόγο προσέγγισης του προβλήματος OPT-EXT(0,1). Στο γενικό πρόβλημα OPT-EXT, δοθέντων ενός γραφήματος και μιας συλλογής αντικειμένων με αξία, ο στόχος είναι να τοποθετήσουμε τα αντικείμενα στις κορυφές ώστε να μεγιστοποιηθεί η *συνολική εξωτερική επίδραση*. Στην πραγματικότητα, σχετίζουμε το πρόβλημα p-Max 1-hop Domination με το OPT-EXT(0,1) (στο οποίο οι αξίες των αντικειμένων είναι 0 ή 1) και βελτιώνουμε το προηγούμενο λόγο $\frac{e-1}{e+1}$ στον νέο $\frac{6e-5}{6e+5}$. Επιπλέον μελετούμε θέματα προσεγγισιμότητας του OPT-EXT με τις γενικές αξίες καθώς και μιας παραλλαγής του OPT-EXT(0,1). Τέλος, σχετικά με το πρόβλημα p-Max K-hop Domination, με σταθερό (fixed) $K \geq 1$, δείχνουμε ότι ο καλύτερος δυνατός λόγος προσέγγισης του είναι $(1 - \frac{1}{e})$, γενικεύοντας το ήδη γνωστό αποτέλεσμα για $K = 1$. Για την απόδειξη χρησιμοποιούνται ειδικές αναγωγές, ονόματι *gap-preserving reductions*.

Λέξεις κλειδιά— προσεγγιστικοί αλγόριθμοι, submodular συνάρτηση, τεχνικές βελτιστοποίησης, μη προσεγγισιμότητα, μέγιστη κάλυψη, προβλήματα κυριαρχίας, εξωτερικές επιδράσεις, αποσύνθεση.

Abstract

Since the first studies on maximization of submodular functions, the fast-growing interest on optimization techniques with deep practical consequence on many of the most influential areas of research like machine learning, algorithmic game theory, social networks etc., has been leading research towards new ways of expressing such functions. Our interest sources from a study on functions g that can be expressed as the difference of a γ-weakly submodular function f and a modular function c, where a γ-weakly submodular function is a very promising extension of the standard definition of submodular functions. More specifically, we consider the problem of maximizing a function $g = f - c$ under cardinality constraints, with f being a coverage function and c a cardinality function. For this we consider a family of problems which we call p-Max k-hop Domination, where given a graph G and integers p, k, the objective function f is to find a set $D \subseteq V(G)$ with $|D| = p$, such that the total number of vertices that are within k hops from at least one vertex in D, is maximized. We provide a method to approximate the optimal value of $g = f - c$, using various approximation algorithms depending on what we call *decomposition* of an instance. We show that our results can be used to improve the current approximability results for the problem OPT-EXT(0,1) which is a special case of the more general problem OPT-EXT. In OPT-EXT, a collection of valued objects is given and the goal is to allocate each object to the vertices of a given graph so that the objective function called *graph externality* is maximized. In the variant OPT-EXT(0,1) each object has either a value 0 or 1. In fact, we can relate the p-Max 1-hop Domination problem to OPT-EXT(0,1) and improve the approximation ratio of $\frac{e-1}{e+1}$ to the new ratio of $\frac{6e-5}{6e+5}$. We also provide a scheme which is close to the notion of a *PTAS* for a variant of OPT-EXT(0,1) called α-(OPT-EXT(0,1)) and finally regarding the general problem OPT-EXT we give the first proof for a approximation ratio that a rather naturally considered algorithm achieves. Our last contribution is that of proving that the p-Max K-hop Domination problem has the inapproximability bound $(1 - \frac{1}{e})$, for any fixed $K \geq 1$, generalizing the known result for $K = 1$. To show this, we establish appropriate gap-preserving reductions.

Keywords— approximation algorithm, submodular function, optimization techniques, hardness of approximation, maximum coverage, domination problem, graph externalities, approximation schemes, decomposition, gap-preserving reductions.

Ευχαριστίες

Το πρώτο μου ερευνητικό ταξίδι φτάνει στο τέλος του με την ολοκλήρωση της διπλωματικής μου εργασίας, η οποία κράτησε το ενδιαφέρον και τον ενθουσιασμό μου αμείωτο όλους τους μήνες εκπόνησής της. Αρχικά θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή κ. Αριστείδη Παγουρτζή για τον χρόνο που διέθετε αφειδώλευτα για να με καθοδηγεί επισταμένως, να ακούει τις ιδέες και προτάσεις μου, να τις φιλτράρει με την διδακτική και ερευνητική του πείρα ώστε να μπορέσουν τελικά να ενσωματωθούν στο σώμα της εργασίας συνοδευόμενες από την κατάλληλη επιστημονική τεκμηρίωση. Ευχαριστώ και τα υπόλοιπα μέλη της επιτροπής για τις υποδείξεις τους και τις ενδιαφέρουσες προτάσεις τους πάνω σε νέες πιθανές επεκτάσεις της παρούσας εργασίας. Δεν θα μπορούσα να μην ευχαριστήσω επίσης τους καθηγητές εκείνους που ξεχώρισα καθ' όλη την φοίτησή μου στο Πολυτεχνείο, οι οποίοι με προετοίμασαν για αυτό το ερευνητικό ταξίδι ήδη από τα πρώτα μόλις έτη, παρέχοντάς μου την απαραίτητη επιστημονική κατάρτιση αλλά και μεταδίδοντάς μου την πηγαία αγάπη τους για την έρευνα. Τέλος θα ήθελα να ευχαριστήσω από καρδιάς όλους όσους στάθηκαν και στέκονται στο πλάι μου, την οικογένεια μου, τους φίλους μου.

Παναγιώτης Αϊβασιλιώτης
Αθήνα, 22η Ιουλίου 2022

Contents

List of Figures

Part I

Text in Greek

Κεφάλαιο 1

Εκτεταμένη Περίληψη στα Ελληνικά

1.1 Θεωρητικό υπόβαθρο

1.1.1 Βασικές έννοιες από την θεωρία γραφημάτων

Ένα γράφημα είναι ένα διατεταγμένο ζέυγος (V, E), όπου V είναι το *σύνολο των κορυφών* και E το *σύνολο των ακμών*. Για ένα *απλό μη κατευθυνόμενο γράφημα*, μια ακμή είναι ένα μη διατεταγμένο ζεύγος $\{u, v\}$ όπου $u, v \in V$ $(u \neq v)$, ενώ για ένα *απλό κατευθυνόμενο γράφημα* μια ακμή είναι ένα διατεταγμένο ζεύγος (u, v) εννοώντας ότι οι ακμές $(u, v), (v, u)$ είναι διαφορετικές ενώ οι ακμές $\{u, v\}, \{v, u\}$ ταυτίζονται. Επίσης βάσει ορισμού σε ένα απλό (μη) κατευθυνόμενο γράφημα δεν επιτρέπονται *πολλαπλές ακμές* ή *βρόχοι* (δηλ. ακμές της μορφής $\{u, u\}$ ή (u, u)).

Στην παρούσα εργασία μας ενδιαφέρουν μόνο απλά μη κατευθυνόμενα γραφήματα, για τα οποία παραθέτουμε τους ορισμούς κάποιων βασικών απαραίτητων εννοιών.

Ορισμός 1.1.1. *Δοθέντος ενός γραφήματος $G = (V, E)$, η **γειτονιά** μιας κορυφής $u \in V$, συμβολικά $\mathcal{N}_G(u)$ ή απλά $\mathcal{N}(u)$, είναι το σύνολο $\{v \in V : \{u, v\} \in E\}$ και η **κλειστή γειτονιά**, συμβολικά $\mathcal{N}[u]$, είναι το σύνολο $\{u\} \cup \mathcal{N}(u)$.*

Ορισμός 1.1.2. *Ο **βαθμός** μιας κορυφής u, συμβολικά $deg(u)$, ισούται με την πληθικότητα της γειτονιάς της, δηλ. $deg(u) = |\mathcal{N}(u)|$.*

Ορισμός 1.1.3. *Ένα **μονοπάτι** μήκους $l-1$, αποτελείται από μία ακολουθία l διαφορετικών κορυφών $u_1, ..., u_l$ έτσι ώστε $\{u_i, u_{i+1}\} \in E$ για κάθε $1 \leq i < l$.*

Ορισμός 1.1.4. *Η **απόσταση** μεταξύ δύο κορυφών $u, v \in V$ ενός γραφήματος $G = (V, E)$, συμβολικά $d(u, v)$, είναι το μήκος του συντομότερου μονοπατιού που ενώνει τις δύο κορυφές. Σε ένα μη κατευθυνόμενο γράφημα, $d(u, v) = d(v, u)$. Σε περίπτωση που δεν υπάρχει μονοπάτι που να ενώνει τις δύο κορυφές γράφουμε $d(u, v) = \infty$.*

Ορισμός 1.1.5. *Ένας* ***κύκλος*** *μήκους l, αποτελείται από μία ακολουθία l διαφορετικών κορυφών $u_1, ..., u_l$ έτσι ώστε $\{u_i, u_{i+1}\} \in E$ για κάθε $1 \leq i < l$ και επιπλέον $\{u_1, u_l\} \in E$.*

Ορισμός 1.1.6. *Ένα* ***υπογράφημα*** *ενός γραφήματος $G = (V, E)$ είναι ένα ζεύγος (V', E') τέτοιο ώστε $V' \subseteq V$ και $E' \subseteq E$.*

Ορισμός 1.1.7. *Ένα γράφημα λέγεται* ***συνεκτικό*** *αν και μόνο αν οποιεσδήποτε δύο κορυφές του ενώνονται με κάποιο μονοπάτι.*

Γίνεται προφανές ότι εάν ένα γράφημα δεν είναι συνεκτικό τότε αποτελεί την ένωση συνεκτικών γραφημάτων τα οποία ονομάζονται *συνεκτικές συνιστώσες* του γραφήματος.

Ορισμός 1.1.8. *Ένα συνεκτικό γράφημα του οποίου κανένα υπογράφημα δεν αποτελεί κύκλο, λέγεται* ***δέντρο*** *και μια συλλογή από δέντρα λέγεται* ***δάσος****.*

Ορισμός 1.1.9. *Ένα υπογράφημα $T = (V', E')$ ενός συνεκτικού γραφήματος $G = (V, E)$ με $V' = V$, το οποίο είναι δέντρο, λέγεται* ***συνεκτικό δέντρο*** *του G. Αντιστοίχως ορίζεται και το* ***συνεκτικό δάσος*** *ενός μη συνεκτικού γραφήματος.*

1.1.2 Εισαγωγή στις submodular συναρτήσεις

Οι submodular συναρτήσεις είναι ευρέως γνωστές τόσο για την μεγάλη τους θεωρητική σημασία όσο και για την πληθώρα των εφαρμογών τους. Ενδεικτικά αναφέρουμε κάποιες από τις περιοχές με μεγάλη επιρροή στην σύγχρονη εποχή, στις οποίες οι submodular συναρτήσεις είναι καίριας σημασίας όπως Μηχανική Μάθηση, Αλγοριθμική Θεωρία Παιγνίων, Κοινωνικά Δίκτυα και Οικονομικά. Προχωρούμε σε μια εισαγωγή στις submodular συναρτήσεις και σε γενικά προβλήματα βελτιστοποίησης.

Αρχικά θεωρούμε ένα σύνολο X από n στοιχεία και μια συνάρτηση $f : 2^X \to \mathbb{R}$.

Ορισμός 1.1.10. *Μια συνάρτηση $f : 2^X \to \mathbb{R}$ είναι submodular αν για οποιαδήποτε σύνολα $A \subseteq B \subseteq X$ και για κάθε στοιχείο $e \in X \backslash B$, ισχύει ότι*

$$f(A \cup \{e\}) - f(A) \geq f(B \cup \{e\}) - f(B)$$

Στην περίπτωση που στον προηγούμενο ορισμό, η σχέση ισχύει με ισότητα τότε η συνάρτηση f ονομάζεται modular.

Μια άλλη ιδιότητα η οποία συνοδεύει σε πολλές εφαρμογές τις submodular συναρτήσεις είναι αυτή της μονοτονίας.

Ορισμός 1.1.11. *Μια συνάρτηση $f : 2^X \to \mathbb{R}$ είναι μονότονη αν για οποιαδήποτε σύνολα $A \subseteq B \subseteq X$ ισχύει ότι*

$$f(A) \leq f(B)$$

Σχετικά με το πρόβλημα βελτιστοποίησης submodular συναρτήσεων, υπάρχει μια ειδοποιός διαφορά ανάμεσα σε προβλήματα ελαχιστοποίησης και σε προβλήματα μεγιστοποίησης. Πιο συγκεκριμένα, αποδεικνύεται ότι τα προβλήματα ελαχιστοποίησης επιλύονται σε πολυωνυμικό χρόνο [IFF01; Sch00] ενώ τα προβλήματα μεγιστοποίησης είναι **NP**-δύσκολα, καθώς υπάρχει μια πληθώρα γνωστών δύσκολων προβλημάτων των οποίων η αντικειμενική συνάρτηση είναι submodular. Ενδεικτικά αναφέρουμε το πρόβλημα *Μέγιστης Τομής* [GW95; FG95]. Όπως συμβαίνει με όλα τα δύσκολα προβλήματα, το ενδιαφέρον περιστρέφεται γύρω από την προσεγγισιμότητα αυτών των προβλημάτων. Ειδικότερα, κάποια από τα αποτελέσματα γύρω από την μεγιστοποίηση submodular συναρτήσεων αναφέρονται παρακάπτω.

- Για μια θετική μονότονη submodular συνάρτηση, το πρόβλημα υπολογισμού της τιμής
$$\max_{A \subseteq X : |A| \leq l} f(A)$$
δύναται να προσεγγιστεί με λόγο $(1 - \frac{1}{e})$ χρησιμοποιώντας ένα απλό άπληστο κριτήριο [NWF78].

- Για την μεγιστοποίηση μιας θετικής, (γενικά) μη μονότονης, submodular συνάρτησης οι πρώτοι προσεγγιστικοί αλγόριθμοι που προτάθηκαν περιλαμβάνουν έναν ντετερμινιστικό τοπικής αναζήτησης $\frac{1}{3}$-προσεγγιστικό αλγόριθμο κι ένα τυχαιοκρατικό (randomized) $\frac{2}{5}$-προσεγγιστικό αλγόριθμο [FMV11].

- Για την μεγιστοποίηση μιας (γενικά) μη μονότονης submodular συνάρτησης με περιορισμούς και ειδικότερα για το πρόβλημα εύρεσης της τιμής
$$\max_{A \subseteq X : |A| \leq l} f(A)$$
υπάρχει προσεγγιστικός αλγόριθμος με λόγο προσέγγισης στο διάστημα $[\frac{1}{e} + 0.004, \frac{1}{2}]$ ενώ για το πρόβλημα εύρεσης της τιμής
$$\max_{A \subseteq X : |A| = l} f(A)$$
υπάρχει προσεγγιστικός αλγόριθμος με λόγο προσέγγισης στο διάστημα $[0.356, \frac{1}{2}]$ [Buc+14].

Όπως έγινε αντιληπτό, κάποια από τα προηγούμενα προβλήματα μεγιστοποίησης υπόκεινται σε περιορισμούς πληθικότητας (cardinality constraints). Πέραν αυτών, στην βιβλιογραφία μελετώνται κι άλλοι περιορισμοί όπως *matroid* και *knapsack* περιορισμοί.

1.2 Μέθοδος προσέγγισης ειδικών submodular συναρτήσεων της μορφής $g = f - c$

1.2.1 Maximum Coverage Problem

Θεωρούμε ένα σύνολο X αποτελούμενο από n στοιχεία και μια συλλογή συνόλων $\mathcal{S} = \{S_1, ..., S_m\}$, όπου $S_i \subseteq X$, $1 \leq i \leq m$. Έστω ότι κάθε στοιχείο έχει ένα βάρος το οποίο δίνεται από την συνάρτηση $w : X \to \mathbb{R}^+$. Επίσης, ορίζουμε την συνάρτηση $f : 2^{\mathcal{S}} \to \mathbb{R}^+$, ως εξής :

$$f(\mathcal{S}') = \sum_{e_j \in \bigcup_{S_i \in \mathcal{S}'} S_i} w(e_j)$$

Η συνάρτηση f όπως ορίστηκε είναι μία από τις πλέον αντιπροσωπευτικές submodular συναρτήσεις.

Ορισμός 1.2.1 (weighted Maximum Coverage Problem)**.** *Δοθέντος ενός στιγμιότυπου το οποίο αποτελείται από τα σύνολα $X, \mathcal{S}$, την συνάρτηση $w : X \to \mathbb{R}^+$ και έναν ακέραιο l, ο στόχος είναι η εύρεση ενός υποσυνόλου $\mathcal{S}' \subseteq \mathcal{S}$ αποτελούμενου από το πολύ l σύνολα, έτσι ώστε η τιμή $f(\mathcal{S}')$ να είναι μέγιστη.*

Με το όνομα MCP στο εξής θα αναφερόμαστε στην ειδική περίπτωση όπου όλα τα στοιχεία έχουν βάρος ίσο με 1, το οποίο πρόβλημα έχει αποδειχθεί ότι είναι **NP**-δύσκολο [Fei98].

Ένα άπληστο κριτήριο για την εύρεση μια προσεγγιστικής λύσης για το MCP, είναι να επιλέγεται κάθε φορά εκείνο το σύνολο το οποίο καλύπτει (δηλ. περιλαμβάνει) το μεγαλύτερο πλήθος από ακάλυπτα στοιχεία (δηλ. που δεν περιλαμβάνονται σε κανένα σύνολο από αυτά που έχουν ήδη επιλεγεί).

Θεώρημα 1.2.1 ([Hoc96])**.** *Ο άπληστος αλγόριθμος είναι ένας $(1 - \frac{1}{e})$-προσεγγιστικός πολυωνυμικού χρόνου αλγόριθμος για το MCP.*

Στην πραγματικότητα ο λόγος $(1 - \frac{1}{e})$ είναι η καλύτερη δυνατή προσέγγιση όπως διατυπώνεται στο επόμενο θεώρημα.

Θεώρημα 1.2.2. *Δεν υπάρχει αλγόριθμος πολυωνυμικού χρόνου ο οποίος να επιτυγχάνει καλύτερη προσέγγιση από $(1 - \frac{1}{e})$ για το MCP, υποθέτοντας ότι* **P** $\neq$ **NP**.

Στην εργασία μας, θεωρούμε μια παραλλαγή του προβλήματος στην οποία *ακριβώς* l σύνολα πρέπει να επιλεγούν. Αν υποθέσουμε ότι υπάρχει βέλτιστο υποσύνολο $\mathcal{S}^*$ με $|\mathcal{S}^*| < l$, επειδή για τη συνάρτηση f μπορεί εύκολα να αποδειχθεί ότι είναι μονότονη, μπορούμε να προσθέσουμε στο $\mathcal{S}^*$ τυχαία σύνολα που δεν έχουν επιλεγεί και να προκύψει ένα επίσης βέλτιστο σύνολο $S^{*\prime}$, με ακριβώς l σύνολα. Έτσι, τα δύο προηγούμενα θεωρήματα, διατυπωμένα αναλόγως, είναι αληθή και για την παραλλαγή που μελετούμε.

1.2.2 Ορισμός του προβλήματος

Για την παραλλαγή του MCP που μελετούμε, εστιάζουμε το ενδιαφέρον μας σε συγκεκριμένα στιγμιότυπα για τα οποία δείχνουμε ότι η συνάρτηση f μπορεί ισοδύναμα να οριστεί ως εξής :

$$f(A) = g(A) + |A|, A \subseteq V$$

Στην πραγματικότητα το σύνολο V ταυτίζεται με το σύνολο X της εισόδου.

Αποδεικνύουμε ότι το σύνολο όλων των υποσυνόλων $A \subseteq V$ με ακριβώς p στοιχεία (χρησιμοποιούμε πλέον το p στην θέση του l) τα οποία μεγιστοποιούν την συνάρτηση f, μεγιστοποιούν την g και αντιστρόφως. Το πρόβλημα το οποίο μας ενδιαφέρει είναι να προσεγγίσουμε την τιμή :

$$\max_{A \subseteq V : |A| = p} \{g(A)\}$$

Μπορούμε να αποδείξουμε ότι η συνάρτηση g ανήκει στην οικογένεια των submodular συναρτήσεων, όχι όμως και των μονότονων submodular συναρτήσεων. Επίσης η συνάρτηση g εκφράζεται ως η διαφορά μια submodular monotone συνάρτησης και μιας modular συνάρτησης. Με τις κατηγοριοποιήσεις αυτές της g, το πρόβλημα μπορεί να αναχθεί σε γενικότερα προβλήματα που έχουν μελετηθεί και έχουν προταθεί (προσεγγιστικές) λύσεις για αυτά, τα οποία αναφέρονται στην βιβλιογραφία. Παρόλα αυτά, παρουσιάζουμε μια μέθοδο για την επίλυση του προβλήματος εκμεταλλευόμενοι το γεγονός ότι επικεντρωνόμαστε σε πιο ειδικές εφαρμογές.

1.2.3 Περιγραφή της μεθόδου

Υπενθυμίζεται ότι ο άπληστος αλγόριθμος υπολογίζει μια ακολουθία συνόλων $(A_0, A_1, ..., A_p)$ όπου

$$A_0 = \emptyset$$

$$A_{i+1} = argmax_{v \in V \setminus A_i} \{f(A_i \cup \{v\})\}, 0 \leq i < p$$

Επίσης ο άπληστος αλγόριθμος είναι $(1 - \frac{1}{e})$-προσεγγιστικός, το οποίο πρακτικά μεταφράζεται ως εξής:

$$f(A_p) \geq (1 - \frac{1}{e}) f(A_p^*)$$

από όπου προκύπτει ότι

$$g(A_p) \geq (1 - \frac{1}{e}) g(A_p^*) - \frac{p}{e}$$

, όπου A_p^* είναι ένα βέλτιστο σύνολο με ακριβώς p στοιχεία.
Αν ορίσουμε $\theta_p := \frac{g(A_p)}{p}$, τότε η προηγούμενη σχέση γίνεται

$$g(A_p) \geq \frac{\theta_p (e-1)}{1 + \theta_p e} g(A_p^*)$$

Επίσης ορίζουμε $\sigma_p := \frac{g(A_p)}{n-p}$. Έτσι καταλήγουμε στην επόμενη σχέση

$$g(A_p) \geq \max\{\frac{\theta_p(e-1)}{1+\theta_p e}, \sigma_p\} g(A_p^*)$$

Επιπλέον, αποδεικνύουμε ότι $\sigma_1 \leq \sigma_2 \leq ... \leq \sigma_n$ κι ότι $\theta_1 \geq \theta_2 \geq ... \geq \theta_n$.

Στην συνέχεια, η μέθοδος απαιτεί την κατασκευή ενός βοηθητικού στιγμιότυπου το οποίο σχετίζεται με το αρχικό και στο οποίο εφαρμοζόμενος ο άπληστος αλγόριθμος θα υπολογίσει μια ακολουθία $(\widehat{A}_0, ..., \widehat{A}_p)$ τέτοια ώστε

$$g(\widehat{A}_p) \geq \widehat{g}(\widehat{A}_p)$$

όπου έχουμε θεωρήσει ότι η αντικειμενική συνάρτηση του προβλήματος MCP, για το βοηθητικό στιγμιότυπο είναι η $\widehat{f}(\widehat{A}) = \widehat{g}(\widehat{A}) + |\widehat{A}|$, $\widehat{A} \subseteq \widehat{V}$, με $\widehat{V} = V$. Το βοηθητικό αυτό στιγμιότυπο θα χρησιμοποιηθεί ώστε να μπορέσουμε να μεγιστοποιήσουμε το κάτω όριο της τιμής

$$\min_p \max\{\frac{\widehat{\theta}_p(e-1)}{1+\widehat{\theta}_p e}, \widehat{\sigma}_p\}$$

για κάθε στιγμιότυπο $\mathcal{I}$ και το αντίστοιχο βοηθητικό του $\widehat{\mathcal{I}}$ όπου τα $\widehat{\theta}_p, \widehat{\sigma}_p$ ορίζονται με όμοιο τρόπο με τα θ_p, σ_p και το οποίο κάτω όριο θα δίνεται από τον τύπο $\min\{\frac{\widehat{\theta}(e-1)}{1+\widehat{\theta}e}, \widehat{\sigma}\}$, για κάποιες καθολικές σταθερές $\widehat{\theta}, \widehat{\sigma}$ κι όπως θα αποδείξουμε

$$\frac{\max\{g(A_p), g(\widehat{A}_p)\}}{g(A_p^*)} \geq \min\{\frac{\widehat{\theta}(e-1)}{1+\widehat{\theta}e}, \widehat{\sigma}\}$$

1.2.4 Πρακτική εφαρμογή

Την μέθοδο που περιγράψαμε στην προηγούμενη ενότητα, την εφαρμόζουμε στο πρόβλημα το οποίο ονομάζουμε p-Max k-hop Domination το οποίο ορίζεται αμέσως :

Ορισμός 1.2.2 (p-Max k-hop Domination)**.** *Δοθέντων ενός μη κατευθυνόμενου γραφήματος $G = (V, E)$ και ακεραίων p, k βρες ένα σύνολο $D \subseteq V$, p κορυφών του G, έτσι ώστε ο συνολικός αριθμός των κορυφών που ενώνονται με μονοπάτι το πολύ μήκους k με τουλάχιστον μία κορυφή του D, να είναι ο μέγιστος δυνατός.*

Δείχνουμε ότι το p-Max k-hop Domination ανάγεται στο MCP οπότε και κάθε στιγμιότυπο του p-Max k-hop Domination μετασχηματίζεται σε ένα στιγμιότυπο $\widehat{\mathcal{I}}$ του MCP. Στην συνέχεια, κατασκευάζουμε το στιγμιότυπο $\widehat{\mathcal{I}}$ για το οποίο όπως έγινε προφανές, υπάρχει γράφημα $\widehat{G}$ το οποίο μετασχηματίζεται

στο $\widehat{\mathcal{I}}$ λόγω της προαναφερθείσας αναγωγής, με αντίστοιχες αντικειμενικές συναρτήσεις $f(A) = g(A)+|A|$ και $\widehat{f}(A) = \widehat{g}(A)+|A|$, $A \subseteq V = \widehat{V}$. Προτείνουμε δύο διαδικασίες (Decomposition D1 και Decomposition D2) κατασκευής του γραφήματος $\widehat{G}$, όπου και οι δύο αρχικά υπολογίζουν ένα συνεκτικό δάσος του G. Στην συνέχεια η κάθε μία αποσυνθέτει το συνεκτικό δάσος σε απλούστερες συνιστώσες. Ενδεικτικά, για το γράφημα που προκύπτει μετά την εφαρμογή της Decomposition D1 σε συνεκτικό δέντρο ενός συνεκτικού γραφήματος, αποδεικνύουμε ότι υπάρχει t τέτοιο ώστε $\widehat{\theta} \geq k$, για $p \leq t$ και $\widehat{\sigma} = 1$, για $p \geq t$, οπότε ο λόγος προσέγγισης γίνεται

$$\frac{e-1}{e+\frac{1}{k}}$$

Στην συνέχεια δείχνουμε πως μπορούμε να γενικεύσουμε το ανωτέρω αποτέλεσμα για οποιοδήποτε γράφημα G. Με την Decomposition D2, βελτιώνουμε τον λόγο προσέγγισης σε

$$\frac{e-1}{e+\frac{1}{k+1}}, \text{ για } k \geq 2$$

και σε

$$\frac{6e-5}{6e+5}, \text{ για } k = 1$$

1.3 Αποτελέσματα (μη) προσεγγισιμότητας για το πρόβλημα p-Max k-hop Domination και παραλλαγών του

Το πρόβλημα p-Max k-hop Domination το οποίο είναι ένα πρόβλημα μεγιστοποίησης, σχετίζεται άμεσα με ένα πρόβλημα ελαχιστοποίησης του οποίου ο ορισμός δίνεται παρακάτω :

Ορισμός 1.3.1 (k-hop Dominating set[BM14]). *Δοθέντων ενός μη κατευθυνόμενου γραφήματος $G = (V, E)$ κι ενός ακεραίου k, βρες το μικρότερο σύνολο D κορυφών του G, έτσι ώστε κάθε κορυφή στο $V \backslash D$ να συνδέεται με μονοπάτι μήκους το πολύ k με τουλάχιστον μία κορυφή στο D.*

Και τα δύο προαναφερθέντα προβλήματα είναι **NP**-δύσκολα.

Όπως δείξαμε σε προηγούμενη ενότητα, το p-Max k-hop Domination είναι ειδική περίπτωση του MCP, οπότε μπορεί να προσεγγιστεί με λόγο $(1 - \frac{1}{e})$. Αποδεικνύουμε ότι αυτός ο λόγος είναι ο καλύτερος δυνατός λόγος προσέγγισης για το πρόβλημα p-Max K-hop Domination στο οποίο πρόβλημα ο αριθμός των αλμάτων (hops) είναι σταθερός (fixed) και ίσος με K (υποθέτοντας ότι $\mathbf{P} \neq \mathbf{NP}$). Για $K = 1$, το αποτέλεσμα αυτό είναι ήδη γνωστό [MO11],

οπότε εμείς παρέχουμε την γενικεύση αυτού για κάθε (fixed) $K \geq 1$. Για να αποκτήσουμε το προηγούμενο αποτέλεσμα, χρησιμοποιούνται συγκεκριμένες αναγωγές οι οποίες ονομάζονται *gap-preserving (reductions)* που αποτελούν έναν από τους ευρέως γνωστούς τρόπους απόδειξης δυσκολίας προσεγγισιμότητας (δύσκολων) προβλημάτων.

1.4 Το πρόβλημα των εξωτερικών επιδράσεων

1.4.1 Εισαγωγή

Ένα στιγμιότυπο $\mathcal{I}$ του προβλήματος OPT-EXT (όπως πρωτό-ορίστηκε στην εργασία [Fot+20]) περιγράφεται από μια τριάδα (G, O, val) όπου $G = (V, E)$ είναι ένα μη κατευθυνόμενο γράφημα, O είναι είναι μια συλλογή από m αντικείμενα το καθένα έχον αξία που δίνεται από την συνάρτηση $val : O \to \mathbb{R}^+$. Ο χώρος των εφικτών λύσεων του προβλήματος αποτελείται από όλες εκείνες τις αναθέσεις υπό τις οποίες *κάθε* αντικειμένο ανατίθεται και σε μια *διαφορετική* κορυφή του γραφήματος. Η προαναφερθείσα ανάθεση περιγράφεται από μια συνάρτηση $\pi : V \to O \cup \{\perp\}$. Ο ορισμός απαιτεί $m \leq n = |V|$. Επομένως, για να υποδηλώσουμε ότι σε κάποια κορυφή v δεν έχει ανατεθεί κάποιο αντικείμενο, γράφουμε $\pi(v) = \perp$.

Ορισμός 1.4.1. *Υπό μία ανάθεση π, η εξωτερική επίδραση μιας κορυφής $v \in V$, η οποία συμβολίζεται με $ext_\pi(v)$ υπολογίζεται ως εξής :*

- $ext_\pi(v) = 0$, *if* $\pi(v) = \perp$
- $ext_\pi(v) = val(\pi(u)) - val(\pi(v))$, $u := argmax_{w \in \mathcal{N}[v]:\pi(w) \neq \perp}\{val(w)\}$

Μια κορυφή v λαμβάνει θετική εξωτερική επίδραση από κάποια γειτονική της κορυφή στις περιπτώσεις που αυτό είναι εφικτό, διαφορετικά δεν λαμβάνει εξωτερική επίδραση ή βάσει του ορισμού η επίδραση που λαμβάνει είναι μηδενική.

Υπό μία ανάθεση π, η συνολική εξωτερική επίδραση του γραφήματος δίνεται από την συνάρτηση

$$Ext_\pi(G) = \sum_{v \in V} ext_\pi(v)$$

Στο πρόβλημα OPT-EXT, ο στόχος είναι να βρεθεί μια βέλτιστη ανάθεση π^*, υπό την οποία η συνολική εξωτερική επίδραση του γραφήματος είναι η μέγιστη δυνατή.

1.4.2 Το πρόβλημα OPT-EXT με δύο αξίες

Στην περίπτωση όπου η αξία κάθε αντικειμένου είναι μια εκ των 0, 1 τότε ανφερόμαστε στο πρόβλημα OPT-EXT με την ονομασία OPT-EXT(0,1).

Αποδεικνύεται ότι η ειδική αυτή περίπτωση είναι ένα **NP**-δύσκολο πρόβλημα [Fot+20]. Για την προσέγγιση αυτού του προβλήματος, ανάγουμε το OPT-EXT(0,1) στο πρόβλημα μεγιστοποίησης της συνάρτησης $g = f - c$ (υπό περιορισμούς πληθικότητας), όπου η f είναι η αντικειμενική συνάρτηση του προβλήματος p-Max 1-hop Domination και η c είναι συνάρτηση πληθικότητας. Αποδεικνύουμε έτσι ότι υπάρχει αλγόριθμος πολυωνυμικού χρόνου που προσεγγίζει το πρόβλημα OPT-EXT(0,1) με λόγο $\frac{6e-5}{6e+5}$.

Επειδή όμως παρά την βελτίωση του λόγου το ερώτημα περί του καλύτερου δυνατού λόγου προσέγγισης για το OPT-EXT(0,1) παραμένει ανοιχτό, μελετούμε μια εκδοχή του προβλήματος OPT-EXT(0,1) στην οποία αναφερόμαστε με το όνομα α-(OPT-EXT(0,1)) για κάποια σταθερή (fixed) τιμή α, $0 < \alpha \leq 1$.

Ορισμός 1.4.2. *Για κάθε $\alpha, 0 < \alpha \leq 1$, ορίζουμε ως* α-(OPT-EXT(0,1)) *μια παραλλαγή του προβλήματος* OPT-EXT(0,1) *για εκείνα τα στιγμιότυπα που ικανοποιούν το εξής : Αν $F - 1$ είναι ένα κάτω όριο των βαθμών των κορυφών του γραφήματος G, δηλ. $F = \delta(G) + 1$, όπου $\delta(G)$ είναι ο ελάχιστος βαθμός κορυφής του G, τότε $\frac{F}{|V(G)|} \geq \alpha$.*

Η παραλλαγή α-(OPT-EXT(0,1)) ορίστηκε κατά αντιστοιχία με τον ορισμό της παραλλαγής α-MCP του MCP (Maximum Coverage Problem) όπως διατυπώνεται παρακάτω:

Ορισμός 1.4.3 ([SF13])**.** *Για κάθε $\alpha, 0 < \alpha \leq 1$, ορίζουμε ως* α-MCP *μια παραλλαγή του προβλήματος* MCP *για εκείνα τα στιγμιότυπα που ικανοποιούν το εξής : Αν F είναι ένα κάτω όριο των συχνοτήτων των στοιχείων, (όπου ως συχνότητα ενός στοιχείου ορίζεται το πλήθος των συνόλων στα οποία εμφανίζεται το στοιχείο αυτό) και υπάρχουν m σύνολα τότε $\frac{F}{m} \geq \alpha$.*

Αποδεικνύουμε ότι υπάρχει β-προσεγγιστικός αλγόριθμος πολυωνυμικού χρόνου για το πρόβλημα α-(OPT-EXT(0,1)) για κάθε σταθερή (fixed) τιμή $\beta \leq \frac{e}{e+1} \approx 0.73$. Το αποτέλεσμα αυτό βασίζεται πάνω σε προηγούμενα αποτελέσματα [SF13] σύμφωνα με τα οποία αποδεικνύεται ότι υπάρχει *PTAS* (Polynomial Time Approximation Scheme) για την παραλλαγή α-MCP του προβλήματος MCP (Maximum Coverage Problem), το οποίο πρακτικά σημαίνει ότι υπάρχει αλγόριθμος πολυωνυμικού χρόνου ο οποίος σε είσοδο $(\mathcal{I}, \varepsilon)$ όπου $\mathcal{I}$ ένα στιγμιότυπο του α-MCP και $\varepsilon > 0$ εγγυάται λόγο προσέγγισης $(1 - \varepsilon)$.

1.4.3 Το γενικό πρόβλημα OPT-EXT

Για το γενικό πρόβλημα, μιας και στην βιβλιογραφία από όσο είμαστε σε θέση να γνωρίζουμε δεν υπάρχει κάποιος αλγόριθμος ο οποίος να συνοδεύεται από την απόδειξη για το λόγο προσέγγισης που επιτυγχάνει, σχεδιάζουμε

έναν $\frac{\rho}{L-1}$-προσεγγιστικό αλγόριθμο όπου L είναι ο πλήθος των διαφορετικών αξιών των αντικειμένων (δηλαδή η πληθικότητα του συνόλου τιμών της συνάρτησης val) κι όπου επίσης έχει υποτεθεί ότι υπάρχει κάποιος γνωστός ρ-προσεγγιστικός αλγόριθμος για το πρόβλημα OPT-EXT(0,1).

1.5 Συμπεράσματα

Όπως είδαμε το πρόβλημα βελτιστοποίησης submodular συναρτήσεων περιβάλλεται από μια πληθώρα ερευνητικών εργασιών οι οποίες μελετούν μια μεγάλη γκάμα παραλλαγών του προβλήματος, κάθε μια με τη δική της θεωρητική και πρακτική βαρύτητα σε ποικίλες ερευνητικές περιοχές. Στην παρούσα εργασία, το ενδιαφέρον μας επικεντρώθηκε εν γένει στο πρόβλημα μεγιστοποίησης μη μονότονων submodular συναρτήσεων υπό περιορισμούς πληθικότητας. Ειδικότερα, μελετήσαμε συναρτήσεις g που μπορούν να εκφραστούν ως η διαφορά $f - c$ μιας submodular συνάρτησης f και μιας modular συνάρτησης c, όπου η συνάρτηση c κωδικοποιεί ένα είδος τιμήματος (penalty) κάποιων επιλογών του εκάστοτε αλγόριθμου. Εξειδικεύοντας έτι περισσότερο, θεωρήσαμε την f ως την αντικειμενική συνάρτηση μιας οικογένειας προβλημάτων μεγιστοποίησης που ορίσαμε ονόματι p-Max k-hop Domination, ενώ το κόστος θεωρήσαμε ότι δίνεται από μια συνάρτηση πληθικότητας (cardinality function). Αναπτύξαμε μια μέθοδο προσέγγισης τέτοιων συναρτήσεων g, η οποία μέθοδος έχει ευελιξία υπό την έννοια ότι παρέχει στον χρήστη την δυνατότητα να την τροποποιήσει ώστε να ορίσει διαφορετικούς αλγόριθμους προσέγγισης. Τα ανωτέρω βρήκαν πρακτική εφαρμογή σε ένα γνωστό πρόβλημα βελτιστοποίησης, το OPT-EXT(0,1) του οποίου βελτιώσαμε τον προηγούμενο λόγο προσέγγισης $\frac{e-1}{e+1}$ στον νέο λόγο $\frac{6e-5}{6e+5}$. Επίσης, μελετήσαμε τα όρια μη προσεγγισιμότητας του προβλήματος p-Max k-hop Domination καθώς και κάποιων παραλλαγών του αλλά και επεκτείναμε την μελέτη μας περισσότερο πάνω σε θέματα προσεγγισιμότητας τόσο για το γενικό πρόβλημα OPT-EXT όσο και για κάποιες ήδη γνωστές αλλά και νέες παραλλαγές του.

Part II

Text in English

Chapter 1

Theoretical background

1.1 Preliminaries on graph theory

A graph is an ordered pair (V, E) of sets, where V is called the *vertex set* and E is called the *edge set*. For a *simple undirected* graph, an *edge* is an unordered pair of vertices $\{u, v\}$ for some $u, v \in V : u \neq v$, while for a *simple directed* graph, an edge is an ordered pair (u, v) meaning that the elements (u, v), (v, u) are distinct, while the elements $\{u, v\}$, $\{v, u\}$ are identical. An undirected edge $\{u, v\}$ can be seen as a line that connects u, v while a directed edge (u, v) as an arrow pointing to v. Also note that in any simple (un)ordered graph, *multiple edges* or *loops* (i.e. elements of the form (u, u) or $\{u, u\}$)) are not allowed.

Since our study deals only with *simple undirected graphs*, we provide the definitions for some essential basic notions regarding them:

Definition 1.1.1. *The **neighborhood** of a vertex u in a graph $G = (V, E)$ denoted as $\mathcal{N}_G(u)$ or simply $\mathcal{N}(u)$ is the set $\{v \in V : \{u, v\} \in E\}$ and the **closed neighborhood** of u denoted as $\mathcal{N}[u]$ is formed by the union of $\{u\}$ and $\mathcal{N}(u)$.*

Definition 1.1.2. *The **degree** of a vertex $u \in V$ denoted as $deg(u)$ is the number of its neighbors in G, that is $deg(u) = |\mathcal{N}(u)|$.*

Definition 1.1.3. *A **path** of length $l - 1$, is formed by a sequence of l distinct vertices $u_1, ..., u_l$ for which $\{u_i, u_{i+1}\} \in E$ stands for each $i < l$.*

Definition 1.1.4. *The **distance** between two vertices $u, v \in V$ in a graph G, is the length of the shortest path from u to v and is denoted as $d(u, v)$. Note that in an undirected graph $d(u, v) = d(v, u)$. We write $d(u, v) = \infty$, if there is no path that connects u and v.*

Definition 1.1.5. *A **circle** of length l, is formed by a sequence of l distinct vertices $u_1, ..., u_l$ for which $\{u_i, u_{i+1}\} \in E$ stands for each $i < l$ and additionally $\{u_1, u_l\} \in E$.*

Definition 1.1.6. *A **subgraph** G' of a graph $G = (V, E)$ is a pair (V', E') where $V' \subseteq V$ and $E' \subseteq E$.*

Definition 1.1.7. *An **induced subgraph** G' of a graph $G = (V, E)$ is a subgraph (V', E') of G such that*

$$\{u, v\} \in E \rightarrow \{u, v\} \in E'$$

is true for all $u, v \in V'$.

Definition 1.1.8. *Two graphs $G_1 = (V_1, E_1), G_2 = (V_2, E_2)$ are called **isomorphic** if there exists a bijection $f : V_1 \rightarrow V_2$ between the vertex sets of G_1 and G_2 such that for any two vertices $u, v \in V_1$,*

$$\{u, v\} \in E_1 \text{ if and only if } \{f(u), f(v)\} \in E_2$$

Definition 1.1.9. *A graph is called **connected** if for any two vertices $u, v \in V$, there is a path that connects them.*

Definition 1.1.10. *A **connected component** of a graph is a maximal induced connected subgraph of G.*

Finally, we provide the definitions for two notions, which as we will see in the following chapters are critical for the problem of graph externalities : the notion of a *spanning tree* and the notion of a *matching*.

Definition 1.1.11. *A **tree** is a connected graph which does not contain cycles.*

Definition 1.1.12. *A **forest** is a graph, every connected component of which is a tree.*

Definition 1.1.13. *A **rooted tree** is a tree in which one of its vertices has been designated the **root**.*

Definition 1.1.14. *A **leaf** of a tree is a vertex with degree 1.*

Definition 1.1.15. *An **internal** vertex is a vertex which is not a leaf.*

Definition 1.1.16. *The **depth** of a vertex u in a rooted tree, is the length of the (unique) path from u to the root.*

Now that we have defined the depth of a vertex, we can imagine that the vertices of a tree are placed in levels such that the root is placed at the top, then on the next level (downwards) are placed all vertices with depth 1, and so on. The placement we described allows the definition of the *height* of every vertex and the *height* of the tree itself.

Definition 1.1.17. *The **height** of a vertex u in a rooted tree is the length of the downward path from u to the farthest leaf. The **tree height** is the height of the root.*

Definition 1.1.18. *A **spanning tree** of a connected graph $G = (V, E)$ is a subgraph $T = (V', E')$ of G with $V' = V$, which is a tree.*

Definition 1.1.19. *The **spanning forest** of a graph is the union of the spanning trees of each one of its connected components.*

We can compute a spanning tree of a graph G in polynomial time using the Algorithm 1.

Algorithm 1 An algorithm that computes a spanning tree of a connected graph

Input: A connected graph $G = (V, E)$
Output: A spanning tree $T = (V, E')$ of G

```
1: E' ← ∅
2: for each edge e ∈ E do
3:     if (V, E' ∪ {e}) does not contain cycles then
4:         E' ← E' ∪ {e}
5:     end if
6: end for
7: return (V, E')
```

In fact the previous algorithm is a simplified version of the greedy algorithm that solves the more general problem [GH85] of computing a spanning tree of minimum weight, assuming that each edge e has a weight w_e and the weight of any spanning tree is given by the sum of the weights of edges in it. The greedy algorithm picks an edge of minimum weight such that the subgraph which is computed in each iteration contains no cycles. The Minimum Spanning Tree problem —a typical problem of Combinatorial Optimization— has been widely studied and many improved algorithms have been proposed [Cha00; FW90; PR02], while the first algorithm dates back to 1926 [NMN01].

Definition 1.1.20. *A **matching** of a graph $G = (V, E)$ is a subset $M \subseteq E$ of non-adjacent edges, that is they share no endpoint.*

Definition 1.1.21. *A **maximum matching** is a matching M such that $|M| \geq |M'|$ for any other matching M'.*

Definition 1.1.22. *A matching M is **maximal** if there is no matching M' such that $M \subset M'$.*

The computation of a *maximum matching* can be done in polynomial time and some of the algorithms proposed for computing a maximum matching can be found in [Gab76; MV80; Blu90].

1.2 Introduction to submodular functions

Submodular functions are of great importance in areas like machine learning (sensor placement [KG05; Kra+08; KSG08], document summarization [LB11]), algorithmic game theory (calculating market expansion [DRS09], modelling bidders' valuation functions in combinatorial auctions [HM90; LLN06; BBM08]), social networks [HMS08; KKT03], economics [Top98], statistical design of experiments (maximum entropy sampling problem [Lee+09]) etc. We proceed with a brief introduction to submodular functions, the hardness regarding optimization problems with submodular objective functions and (in)approximability results.

We consider a set X —which we call the *ground set*— of n elements. Let $f : 2^X \to \mathbb{R}$ be a set function.

Definition 1.2.1 (submodularity)**.** *A function $f : 2^X \to \mathbb{R}$ is submodular if for every $A \subseteq B \subseteq X$ and $e \in X \backslash B$ it holds that*

$$f(A \cup \{e\}) - f(A) \geq f(B \cup \{e\}) - f(B)$$

The value $f(A \cup \{e\}) - f(A)$ is known as the *discrete derivative* of f at A with respect to e, or *marginal gain (benefit, increase)* and it is denoted as $\Delta_f(e|A)$.

If for a function f, the former inequality holds with equality, then the function f is called *modular*.

Definition 1.2.2 (monotonicity)**.** *A function $f : 2^X \to \mathbb{R}$ is monotone if for every $A \subseteq B \subseteq X$ it holds that*

$$f(A) \leq f(B)$$

An equivalent definition of submodularity of a monotone set function f which can be very useful for generalizing the property of submodularity (see Chapter 2) is the following.

Definition 1.2.3. *A monotone function $f : 2^X \to \mathbb{R}$ is submodular if for every $A, B \subseteq X$ with $A \cap B = \emptyset$ it holds that*

$$\sum_{e \in B} (f(A \cup \{e\}) - f(A)) \geq f(A \cup B) - f(A)$$

Submodular functions have many interesting properties among which is the property of constructing more complex functions using simple submodular functions with the property of submodularity to be preserved (c.f. [Les+07; KKT03; SK10]). For example the following functions are submodular:

1. $f(A) = \sum_{i=1}^{k} a_i g_i(A)$, where $a_i > 0$ and g_i is submodular for each $i \in [1, k]$.

2. $f(A) = \min\{g(A), c\}$, where g is monotone submodular and c is any constant.

For many optimization problems such as Minimum/Maximum (Hypergraph) Cut [KS96; BMN14; VBK20], Maximum Coverage [HP98], Maximum Group Coverage etc. the objective function can be shown to be submodular. But there is a critical difference between minimum and maximum optimization problems with a submodular objective function; it is shown that minimizing a submodular function can be done in polynomial time [IFF01; Sch00], while maximizing a submodular function is an **NP**-hard optimization problem as there is a wealth of **NP**-hard maximization problems whose objective functions are submodular, like Max-Cut [GW95; FG95]. The results regarding the approximability of maximizing a submodular function f vary, depending for example on the properties that f may have, one of which is monotonicity or some constraints subject to which the optimization interests, such as cardinality constraints where the goal is to find a set $A^* \subseteq X$ of at most l elements so that $f(A^*)$ is maximized (among other constraints that have been studied are *matroid constraints* [Cal+11] and *knapsack constraints* [KST13; Lee+09]).

As we mentioned, maximizing a submodular function is an **NP**-hard optimization problem and the problem remains hard even in cases of simple cardinality constraints for many classes of submodular functions such as *weighted coverage* (which we study thoroughly in the next chapter). Nevertheless, there are many notable results regarding the approximability of this problem. More specifically, for the problem of maximizing a nonnegative submodular monotone function subject to cardinality constraints, there exists the following well known theorem, regarding the approximation ratio achieved using a naturally considered greedy heuristic:

Theorem 1.2.1 ([NWF78])**.** *For a nonnegative submodular monotone function $f : 2^X \to \mathbb{R}^+$ and a positive integer l, let $A_i = \{e_1, ..., e_i\}$, $1 \leq i \leq l$, denote the set of the first i elements selected by the greedy algorithm which, each time selects the element e_i that provides the maximum marginal gain $\Delta_f(e|A_{i-1})$. It holds that*

$$f(A_l) \geq (1 - \frac{1}{e}) f(A^*)$$

, where A^* maximizes f over all sets $A \subseteq X$ with $|A| \leq l$.

The study of the problem of approximately maximizing a submodular nonmonotone function has yielded some notable approximability results. The first constant factor approximation algorithms designed for the problem of maximizing a nonnegative submodular nonmonotone function include

a deterministic local search $\frac{1}{3}$-approximation algorithm and a randomized $\frac{2}{5}$-approximation algorithm [FMV11].

For completeness we provide the *local search* algorithm *LS* which achieves an approximation factor of $\frac{1}{3}$, as stated in the following theorem,

Theorem 1.2.2 ([FMV11]). *Algorithm LS is a* $(\frac{1}{3} - \frac{\epsilon}{n})$*-approximation algorithm for maximizing nonnegative submodular functions.*[1] *The algorithm uses at most* $O(\frac{1}{\epsilon}n^3 \log n)$ *oracle calls*[2].

First, the definitions of a *local optimum* and a *approximate local optimum* set S follow, with respect to a submodular function $f : 2^X \to \mathbb{R}$:

Definition 1.2.4. *A set* S *is called a local optimum if for any* $u \in X \backslash S$ *and* $v \in S$,

$$f(S) \geq f(S \cup \{u\}) \text{ and } f(S) \geq f(S \backslash \{v\})$$

Definition 1.2.5. *A set* S *is called a* $(1+\alpha)$*-approximate local optimum, if for any* $u \in X \backslash S$ *and* $v \in S$,

$$(1+\alpha)f(S) \geq f(S \cup \{u\}) \text{ and } (1+\alpha)f(S) \geq f(S \backslash \{v\})$$

Algorithm 2 Local search algorithm *LS*

1: $v \leftarrow argmax_{u \in X}\{f(\{u\})\}$
2: $S \leftarrow \{v\}$
3: **do**
4: **while** there exists $a \in X \backslash S : f(S \cup \{a\}) > (1 + \frac{\epsilon}{n^2})f(S)$ **do**
5: $S \leftarrow S \cup \{a\}$
6: **end while**
7: **if** there exists $a \in S : f(S \backslash \{a\}) > (1 + \frac{\epsilon}{n^2})f(S)$ **then**
8: $S \leftarrow S \backslash \{a\}$
9: **end if**
10: **while** *if-condition* in step 7 is satisfied
11: **return** $\max\{f(S), f(X \backslash S)\}$

The algorithm searches for a $(1+\alpha)$-approximate local optimum instead of a local optimum because of the hardness of finding local optimum for the *Maximum Cut* problem[3] [Sch91]. In fact, if the algorithm terminates, then

[1] The theorem also states that for maximizing nonnegative symmetric submodular functions, the algorithm *LS* is a $(\frac{1}{2} - \frac{\epsilon}{n})$-approximation algorithm. A nonmonotone function $f : 2^X \to \mathbb{R}$ is symmetric if $f(S) = f(X \backslash S)$ stands for any $S \subseteq X$.

[2] An oracle can be seen as a black box which answers queries of the form : *What is the value* $f(S)$*?*.

[3] Finding a local optimum for Maximum Cut is **PLS**-complete, which means that if there exists a polynomial time algorithm that finds a local optimum for Maximum Cut then there is a polynomial time algorithm for every **PLS**-complete problem, but no such polynomial time algorithm is known.

S is a $(1 + \frac{\epsilon}{n^2})$-approximate local optimum.

The last case with which we complete our brief summary of maximizing submodular functions, is that of maximizing a nonmonotone submodular function subject to some given constraints. The work of [Buc+14] has given improved approximations for two variants of the problem of maximizing a nonmonotone submodular function subject to cardinality constraints, in the first of which at most k sets can be chosen and exactly k sets must be chosen in the latter, with respective approximation factors in the range $[\frac{1}{e} + 0.004, \frac{1}{2}]$ and $[0.356, \frac{1}{2}]$.

The reader is encouraged to refer to [KG14; BFS17] for a summary of maximizing submodular functions and to [Iwa08] for submodular minimization.

Chapter 2

Current knowledge

A definition that was deliberately not included in the introduction, which is the last definition we need so that the reader can be familiar with all the notions regarding submodular functions, used in the present work, is the following.

Definition 2.0.1. *A nonnegative monotone set function $f : 2^X \to \mathbb{R}^+$ is γ-weakly submodular for some parameter $0 \leq \gamma \leq 1$, if for any pair of disjoint sets $A, B \subseteq X$ (i.e. $A \cap B = \emptyset$), it satisfies*

$$\sum_{e \in B} (f(A \cup \{e\}) - f(A)) \geq \gamma (f(A \cup B) - f(A))$$

The parameter γ is called the *submodularity ratio* and it captures how close to submodular a function is. It easily follows that f is submodular if and only if f is γ-weakly submodular with $\gamma = 1$. The definition is used in [SY20; Bia+17; CFK18; Ele+17], while the original definition is given in [DK11; DK18]. Approximability and inapproximability results can be found in [DK11; Har+19] respectively. More specifically, in [DK11] it is shown that the greedy algorithm guarantees an approximation ratio $(1 - e^{-\gamma})$ for the problem of maximizing a monotone γ-weakly submodular function subject to a cardinality constraint, while in [Har+19] it is shown that for any $\varepsilon > 0$ and $\gamma, 0 < \gamma \leq 1$, no polynomial time algorithm achieves $(1 - e^{-\gamma} + \varepsilon)$-approximation for the problem of maximizing a non-negative monotone γ-weakly submodular function subject to a cardinality constraint in the *value oracle model* (where the only way to have access to the value $f(S)$ is by querying an *oracle*).

The fast-growing literature regarding weak submodularity is a proof of the practical success of these type of functions in areas such as interpretation of deep neural networks [Ele+17], high dimensional sparse regression [Ele+18], black-box interpretation of images [CFK18] etc.

Also in [Har+19] the problem of maximizing a function $g : 2^X \to \mathbb{R}$ that can be expressed as the difference $g = f - c$, where f is a monotone, non-negative and γ-weakly submodular and c is non-negative modular is studied. Formally, the problem is stated as follows,

$$\max_{S \subseteq X : |S| \leq k} f(S) - c(S)$$

and a deterministic algorithm is provided, which makes $O(nk)$ function evaluations and returns a set S such that

$$f(S) - c(S) \geq (1 - e^{-\gamma}) f(OPT) - c(OPT)$$

, where OPT is the optimal set.

A similar result to the previous one, is obtained in [SVW17] where it is studied the problem of maximizing a function $g = f + l$ where f is a nonnegative monotone submodular function and l an arbitrary modular function, under cardinality constraints and a randomized polynomial time algorithm is given, which outputs a set S satisfying,

$$f(S) + l(S) \geq (1 - \frac{1}{e}) f(OPT) + l(OPT)$$

, where OPT is the optimal set.

A representative example is that of the Maximum Facility Location Problem [CFN77b; CFN77a; AS99] where the objective function can be expressed as the difference of a monotone submodular function and a modular function. More specifically, given a set of facilities the objective is to open a subset of facilities so as to maximize the total profit from the clients minus the cost of opening the facilities.

Another example is that of OPT-EXT(0,1) [Fot+20]. In OPT-EXT(0,1) we are given an undirected graph G and a collection of objects with a valuation of 0 or 1 and the objective is to allocate each object to the vertices so that the number of vertices with valued 0 objects that neighbor with vertices with valued 1 objects is maximized. It can be shown that the objective function Ext of OPT-EXT(0,1) can be expressed as

$$Ext(S) = \min\{z, g(S)\}$$

, where S denotes the subset of vertices where valued 1 objects are to be allocated to (of course $|S| \leq p$, where p is the number of valued 1 objects), z is the number of valued 0 objects and $g(S)$ is the number of vertices in $V(G) \backslash S$ that have a neighbor in S. Using a simple counterexample, they show that Ext is not submodular (neither monotone). Nevertheless,

we observed that maximizing Ext subject to $|S| \leq p$ it is equivalent to maximizing g subject to $|S| = p$, when $p + z = V(G)$. What's more when it comes to approximating OPT-EXT(0,1) it is equivalent to assume that $p + z = V(G)$ [Fot+20]. We show that g is submodular and it can be expressed as the difference of a *uniform weighted coverage* function (which is monotone submodular) and a *cardinality* function (which is modular). In our work, we give an improved approximation ratio for OPT-EXT(0,1), providing a method for approximately maximizing functions of the latter form subject to cardinality constraints of the form $|S| = p$.

It is made obvious that encoding a cost for every "action" we make, which is to add some sort of penalty—e.g., the cost of opening a specific facility, has indeed real-world applications and as a matter of fact when it comes to optimizing a nonmonotone submodular function, such expressions are well-celebrated.

Chapter 3

A method for approximating some special submodular functions of the form $g = f - c$

3.1 Maximum Coverage Problem

Consider a set of n elements X and a collection of m sets $\mathcal{S} = \{S_1, ..., S_m\}$, with $S_i \subseteq X$, for each $i = 1, ..., m$. Let $w : X \to \mathbb{R}^+$ be a weight function. Extending the notion of weight of an element, we define the function $f : 2^{\mathcal{S}} \to \mathbb{R}^+$, where for any $\mathcal{S}' \subseteq \mathcal{S}$ the value $f(\mathcal{S}')$ can be considered as the total weight of all distinct elements appearing in $\mathcal{S}'$. Formally,

$$f(\mathcal{S}') = \sum_{e_j \in \bigcup_{S_i \in \mathcal{S}'} S_i} w(e_j)$$

The function f is a *weighted coverage function* which is a well-known family of submodular monotone functions.

Definition 3.1.1 (weighted Maximum Coverage Problem)**.** *Given an instance which is described by the sets $X, \mathcal{S}$, the function $w : X \to \mathbb{R}^+$ and an integer l, the objective is to maximize the function f over all sets $\mathcal{S}' \subseteq \mathcal{S}$, with $|\mathcal{S}'| \leq l$.*

In cases where each element $e \in X$ has a weight 1, we simply call it MCP, which is known to be **NP**-hard [Fei98]. A greedy algorithm for the (weighted) MCP is given in Algorithm 3, below.

Observe that the value $f(\mathcal{S}' \cup \{S\}) - f(\mathcal{S}')$ is the total weight of all elements of S which have not been covered by any set of $\mathcal{S}'$.

There are two well known results regarding the approximability and inapproximability of the (weighted) MCP stated in the following two theorems:

Theorem 3.1.1 ([Hoc96])**.** *The greedy algorithm achieves an approximation factor of $(1 - \frac{1}{e})$ for* Maximum Coverage Problem*.*

Algorithm 3 A greedy algorithm for the (weighted) MCP

1: $\mathcal{S}' \leftarrow \emptyset$
2: **while** $|\mathcal{S}'| < l$ **do**
3: $\quad S^* \leftarrow argmax_{S \in \mathcal{S} \setminus \mathcal{S}'} \{f(\mathcal{S}' \cup \{S\}) - f(\mathcal{S}')\}$
4: $\quad \mathcal{S}' \leftarrow \mathcal{S}' \cup \{S^*\}$
5: **end while**
6: **return** $\mathcal{S}'$

Theorem 3.1.2 ([Fei98]). *No polynomial time approximation algorithm with a constant approximation ratio better than* $(1 - \frac{1}{e})$ *exists for* (weighted) Maximum Coverage Problem *unless* $\mathbf{P} = \mathbf{NP}$*, even if the weights of elements are restricted to 1.*

In fact, we are interested in a variant of the (weighted) MCP where the objective is to maximize $f(\mathcal{S}')$ subject to $|\mathcal{S}'| = l$. Of course, this variant is as hard as the original problem. Furthermore, since f is monotone we can easily see that every optimal set $\mathcal{S}^*$ with $|\mathcal{S}^*| < l$, can be transformed into an optimal set $\mathcal{S}^{*\prime}$, with $|\mathcal{S}^{*\prime}| = l$, by including random unpicked sets. Thus, the previous two theorems also hold for the variant that we study.

3.2 Problem definition

Recall that for any instance $\mathcal{I} = (p, \{S_1, ..., S_m\})$ of the (unweighted) MCP, the objective is to maximize the function

$$f(\mathcal{S}) = \Big| \bigcup_{S_i \in \mathcal{S}} S_i \Big|$$

, over all sets of subsets $\mathcal{S} \subseteq \{S_1, ..., S_m\}$ with $|\mathcal{S}| = p$.

Our interest lies in instances for which it is guaranteed that they have the following property (**A**): *We can chose one element $v_i \in S_i$ out of every set so that no two elements v_i, v_j are the same (for $i \neq j$).* Let $V = \{v_1, ..., v_m\}$ be the set of these elements —which verify that the previous property is true— and s be a function such that $s(v_i) = i, v_i \in V$. In fact we will be given the set V and the function s. We further assume that $V = \bigcup_{v \in V} S_{s(v)}$. If we consider that each $v \in V$ covers all the elements of $S_{s(v)}$, including itself, we can find an equivalent definition of f with f being defined over 2^V:

$$f(A) = |Coverage(A)| = \Big| \bigcup_{v \in A} S_{s(v)} \Big|^1$$

Observe that for the instances we study, the function f can be written as

$$f(A) = g(A) + |A|, A \subseteq V$$

[1]The definition of $Coverage : 2^V \to 2^V$ is clear from the context.

Proposition 3.2.1. *Every set that maximizes f, maximizes g as well and vice versa (over all sets $A \subseteq V : |A| = p$).*

Proof. Assume that $A_f \subseteq V$ maximizes f and $A_g \subseteq V$ maximizes g. Since $|A_f| = |A_g| = p$, we have that $f(A_f) - g(A_f) = f(A_g) - g(A_g)$. If $g(A_f) < g(A_g)$, then $f(A_f) < f(A_g)$ or if $f(A_f) > f(A_g)$, then $g(A_f) > g(A_g)$, which is a contradiction in either case. □

Proposition 3.2.2. *The function $g : 2^V \to \mathbb{R}^+$ is submodular*

Proof. For any $A \subseteq B \subseteq V$ and $v \in V \backslash B$,

$$f(A \cup \{v\}) - f(A) \geq f(B \cup \{v\}) - f(B) \Rightarrow$$

$$g(A \cup \{v\}) - g(A) + |A \cup \{v\}| - |A| \geq g(B \cup \{v\}) - g(B) + |B \cup \{v\}| - |B|$$

Since $v \notin B$, we have $|A \cup \{v\}| = |A| + 1$ and $|B \cup \{v\}| = |B| + 1$, which completes the proof. □

The function g is nonnegative submodular but generally nonmonotone. Let $A^* \subset V$ be an optimal set for g which covers V for some instance $\mathcal{I}$, then for any $v \in V \backslash A^*$, $g(A^* \cup \{v\}) = g(A^*) - 1$, hence g is nonmonotone. Throughout this chapter, we consider the problem of maximizing[2] the function g which can be expressed as the difference of a monotone submodular function f and a modular function c ($c(A) = |A|, A \subseteq V$) subject to cardinality constraints. Of course, the problem is hard, therefore we are interested in approximating it. The method we provide can be used to approximate the optimal value of g for a family of problems which are MCP's special cases, called p-Max k-hop Domination, using various approximation algorithms.

3.3 Description of the method

It is well known[3] that the MCP can be approximated within $1 - \frac{1}{e}$ using a greedy algorithm which produces the chain $(A_0, A_1, ..., A_p)$ where $A_0 = \emptyset$ and for each $i \in [0, p-1]$,

$$A_{i+1} = A_i \cup \{argmax_{v \in V \backslash A_i} f(A_i \cup \{v\})\}$$

Proposition 3.3.1. *If $f(A_{i+1}) = f(A_i)$ for some $i < p$, then an optimal solution can be found.*

[2]See Section 1.2 (*Introduction to submodular functions*) for the problem of approximately maximizing a general nonegative nonmonotone submodular function subject to cardinality constraints.

[3]See Theorem 1.2.1

Proof. Assume that for some $u \in V \backslash A_i$, $Coverage(\{u\}) - Coverage(A_i) \neq \emptyset$. It follows that, $f(A_i \cup \{u\}) \geq f(A_i) + 1$, which is a contradiction. Thus, A_i covers all elements and therefore for each $j > i$, A_j covers all elements as well. □

Since f is monotone we will consider instances for which it holds that : $f(A_{i+1}) > f(A_i)$, $i < p$ and consequently $g(A_{i+1}) \geq g(A_i)$, $i < p$.

Let $h(p) = \left|\bigcup_{i=1}^{n} S_i\right| - p$.

Proposition 3.3.2. *If, for some $j \leq p$, $f(A_j) - j \geq h(p)$, then an optimal solution can be found.*

Proof. It's $f(A_p^*) \geq f(A_p) \geq f(A_j) + (p - j)$, since $f(A_1) < ... < f(A_p)$, where A_p^* is an optimal solution for the instance $\mathcal{I}$. Furthermore, $f(A_j) - j \geq h(p) \geq f(A_p^*) - p \Rightarrow f(A_p^*) \leq h(p) + p \leq f(A_j) + (p - j)$. Thus, it must be $f(A_p) = f(A_p^*) = f(A_j) + (p - j) = h(p) + p$. □

It is clear that it suffices to study only the instances $\mathcal{I}$ for which

$$f(A_i) - i < h(p), \text{ for each } i \leq p$$

, which implies that $f(A_1) < ... < f(A_p)$.

Lemma 3.3.1. *Under the previous assumptions regarding the instances we study, there exists a sequence of integers $w_1, ..., w_p$ such that $g(A_i) = \sum_{j=1}^{i} w_j$, for each $i \leq p$, with $w_1 \geq w_2 \geq ... \geq w_p$ and $w_i \geq 0$, for each $i \leq p$.*

Proof. $g(A_i) = f(A_i) - i = (f(A_1) - f(A_0) - 1) + (f(A_2) - f(A_1) - 1) + ... + (f(A_i) - f(A_{i-1}) - 1)$.
Let $w_j = f(A_j) - f(A_{j-1}) - 1$, for each $j \leq i$.
For each i, let $v_{\alpha_i} = argmax_{v \in V \backslash A_i} f(A_i \cup v)$.
We have, $f(A_{j-1} \cup v_{\alpha_{j-1}}) - f(A_{j-1}) \geq f(A_{j-1} \cup v_{\alpha_j}) - f(A_{j-1}) \geq f(A_j \cup v_{\alpha_j}) - f(A_j) \Rightarrow f(A_j) - f(A_{j-1}) \geq f(A_{j+1}) - f(A_j) \Rightarrow w_j \geq w_{j+1}$.
Since f is monotone it holds $w_i \geq -1$, but $w_j = -1 \Rightarrow f(A_j) = f(A_{j-1})$, which contradicts our assumptions for the instances we study. Thus $w_i \geq 0$, for each $i \leq p$. □

Let $\mathcal{I}_p = (p, \{S_1, ..., S_n\})$, $p = 1, ..., p'$, where p' is the maximum index so that for each $p \leq p'$ it holds that $f(A_p) - p < h(p')$. For $p > p'$, the greedy algorithm will find a (trivial) optimal solution, for each instance $\mathcal{I}_p$.

We do know that the greedy algorithm achieves an approximation factor of $(1 - \frac{1}{e})$. Formally we have $f(A_p) \geq (1 - \frac{1}{e}) f(A_p^*)$, where A_p^* is an optimal solution for the instance $\mathcal{I}_p$. Using this result, we get the following:

$$g(A_p) + p \geq (1 - \frac{1}{e})(g(A_p^*) + p) \Rightarrow$$

$$g(A_p) \geq (1 - \frac{1}{e})g(A_p^*) - \frac{p}{e}$$

If we define $\theta_p := \frac{g(A_p)}{p} (> 0)$, the former inequality becomes:

$$g(A_p) \geq \frac{\theta_p(e-1)}{1+\theta_p e} g(A_p^*)$$

Proposition 3.3.3. *It holds,* $\theta_1 \geq \theta_2 \geq ... \geq \theta_{p'}$.

Proof.

$$p \cdot w_1 + ... + p \cdot w_p + p \cdot w_{p+1} \leq p \cdot w_1 + ... + p \cdot w_p + (w_1 + ... + w_p)$$

$$\Rightarrow p \sum_{j=1}^{p+1} w_j \leq (p+1) \sum_{j=1}^{p} w_j$$

$$\Rightarrow \frac{g(A_{p+1})}{p+1} \leq \frac{g(A_p)}{p}$$

□

Note that it also holds $\theta_{p'} \geq \theta_{p'+1} \geq ... \geq \theta_n$. Now, let $\sigma_p = \frac{g(A_p)}{h(p)}$, $p = 1, ..., n$. Since $h(1) > ... > h(p')$ and $g(A_1) \leq ... \leq g(A_{p'})$, it follows that $\sigma_1 \leq ... \leq \sigma_{p'}$. Furthermore, $g(A_p) < h(p)$, for each $p \leq p'$, while for $p > p'$, $g(A_p) = h(p)$ (that is, a trivial optimal solution can be found) which implies that $\sigma_p = 1$. Thus, when it comes to maximizing the function g, the approximation ratio for each instance $\mathcal{I}_p$, becomes

$$\max\{\frac{\theta_p(e-1)}{1+\theta_p e}, \sigma_p\}$$

Now, we continue with the definition of an auxiliary instance $\widehat{\mathcal{I}}_p = (p, \{\widehat{S}_1, ..., \widehat{S}_n\})$, satisfying:

i. $\widehat{S}_i \subseteq S_i$, for each $i = 1, ..., n$.

ii. $v_i \in \widehat{S}_i$, for each $v_i \in V (\Rightarrow \widehat{V} \equiv V$ and $\widehat{s}(v_i) = s(v_i) = i)$.

Let $\widehat{f}(A) = \big|\bigcup_{v \in A} \widehat{S}_{\widehat{s}(v)}\big|$. It's easy to prove that, for every $A \subseteq V$ it is true that,

$$\big|\bigcup_{v \in A} S_{s(v)}\big| \geq \big|\bigcup_{v \in A} \widehat{S}_{\widehat{s}(v)}\big| \Leftrightarrow f(A) \geq \widehat{f}(A)$$

Let $(\widehat{A}_1, ..., \widehat{A}_p)$ be the chain produced by the greedy algorithm when it is applied on instance $\widehat{\mathcal{I}}_p$. Let $\widehat{\theta}_p, \widehat{\sigma}_p$, be defined similar to θ_p, σ_p. We have,

$$\max\{f(A_p), \widehat{f}(\widehat{A}_p)\} \geq (1-\frac{1}{e}) f(A_p^*) \Rightarrow \max\{g(A_p), \widehat{g}(\widehat{A}_p)\} \geq (1-\frac{1}{e}) g(A_p^*) - \frac{p}{e}$$

and

$$\frac{\max\{g(A_p), \widehat{g}(\widehat{A}_p)\}}{p} \geq \widehat{\theta}_p$$

, from which we get

$$\max\{g(A_p), \widehat{g}(\widehat{A}_p)\} \geq \frac{\widehat{\theta}_p(e-1)}{1+\widehat{\theta}_p e} g(A_p^*)$$

Furthermore, $\max\{g(A_p), \widehat{g}(\widehat{A}_p)\} \geq \widehat{\sigma}_p \cdot g(A_p^*)$. Finally, since $\max\{g(A_p), g(\widehat{A}_p)\} \geq \max\{g(A_p), \widehat{g}(\widehat{A}_p)\}$, the approximation ratio for the instance $\mathcal{I}_p$ for the problem of maximizing g becomes,

$$\max\{\frac{\widehat{\theta}_p(e-1)}{1+\widehat{\theta}_p e}, \widehat{\sigma}_p\}$$

As it is rather unlikely that we will be able to determine the values θ_p, σ_p (or bound them satisfactorily) due to the complicacy of the "inner structure" of $\mathcal{I}_p$, we construct another instance $\widehat{\mathcal{I}}_p$ of much simpler structure, with respect to $\mathcal{I}_p$ for which we will be able to know a (good) lower bound $\widehat{\theta}$ (resp. $\widehat{\sigma}$) of the corresponding value $\widehat{\theta}_p$ (resp. $\widehat{\sigma}_p$). More specifically, we want to prove that there exists some integer d (which depends on $\widehat{\mathcal{I}}$) such that

$$\widehat{\theta}_p \geq \widehat{\theta}, \text{ for } p \leq d$$

and

$$\widehat{\sigma}_p \geq \widehat{\sigma}, \text{ for } p > d$$

The solution that the greedy algorithm will produce, when it will be applied on instance $\widehat{\mathcal{I}}_p$, will be then transformed into a solution for the instance $\mathcal{I}_p$ (and it will yield at least as good coverage as in instance $\widehat{\mathcal{I}}_p$). This solution in combination with the solution produced by the greedy algorithm, when applied on the initial instance $\mathcal{I}_p$ (that is taking the maximum of the coverages yielded by the two solutions), will give an approximation ratio, which is lower-bounded by

$$\min\{\frac{\widehat{\theta}(e-1)}{1+\widehat{\theta}e}, \widehat{\sigma}\}$$

3.4 Practical application on maximizing the function g of p-Max k-hop Domination

Definition 3.4.1 (p-Max k-hop Domination[4]). *Given an undirected graph $G = (V, E)$ and integers p, k, find a set D of p vertices of G so that the total number of vertices that can be reached from some vertex in D in at most k hops (i.e. traversing at most k edges) is maximized.*

[4] see Chapter 4

Let $\mathcal{I}_{(p,k)} = (p, \{S_1, ..., S_{|V|}\})$ an instance of MCP, that can be constructed by applying the following procedure on an instance (G, p, k) of p-Max k-hop Domination:

1. For each vertex $v_i \in V(G) = \{v_1, ..., v_n\}$, create a set S_i that contains it.

2. For each pair of vertices v_i, v_j, the distance $d(v_i, v_j)$ of which is at most k, add vertex v_i in S_j and vertex v_j in S_i.

Note that the set V of vertices verifies that the property (**A**) is true for the instance $\mathcal{I}_{(p,k)}$. Hence the objective function f can be written as : $f(A) = g(A) + |A|$, $A \subseteq V$.

We now proceed with the construction of the instance $\widehat{\mathcal{I}}_{(p,k)}$. Note that there must be a graph $\widehat{G}$ which is a subgraph of G, so that the instance $\widehat{\mathcal{I}}_{(p,k)}$ can be constructed by applying the previous procedure (steps 1 and 2) on $\widehat{G}$.

3.4.1 Decomposition D1

To begin with, we assume that G is connected. Let T be one of its spanning trees. It will be helpful to consider a representation of the tree in the plane with one of its vertices being the root and the others —the descendants— arranged in levels (as described in section 1.1). We will construct a subgraph $\widehat{G}$ of G with $V(\widehat{G}) = V(G)$, using the following procedure D1:

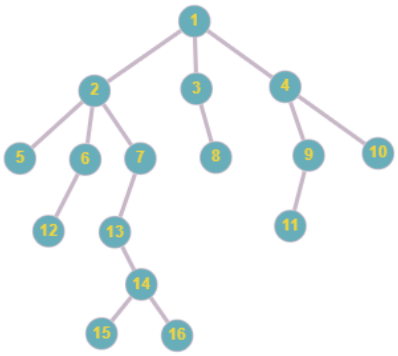
D1

1. Choose a vertex v whose height is k.

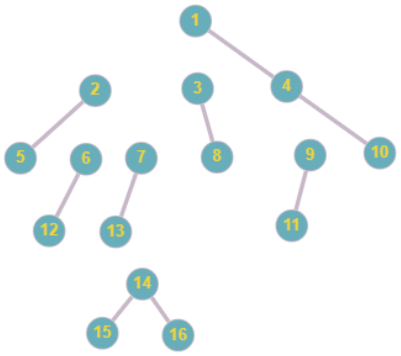
2. Remove the subtree that is formed by the vertex v and all its descendants (we will call it $\widehat{G}_i-$ it's the i-th tree produced by this procedure).

3. Unless the remaining tree is null, if the height of its root is at least k, continue from step 1. If the height is at most $k-1$, then we distinguish two cases : (i) if the number of vertices (of the remaining tree) is at least $k+1$, then name the remaining tree $\widehat{G}_{i+1}$ and halt (ii) otherwise, connect again the vertex u_i to the remaining tree and consider this union as the graph $\widehat{G}_i$ (observe that any vertex of the remaining tree is within k hops from u_i).

The output of the decomposition D1 is the graph $\widehat{G} = \bigcup_{i=1}^{t} \widehat{G}_i$. Similar to the construction of the instance $\mathcal{I}_{(p,k)}$ we can construct an instance $\widehat{\mathcal{I}}_{(p,k)} = (p, \{\widehat{S}_1, ..., \widehat{S}_n\})$ from $\widehat{G}$. Then, it is easy to see that the following are true,



A rooted tree T



The output of Decomposition D1 on input T

Figure 3.1: Decomposition D1 - An example (priority rule : (1) vertices of largest depth (2) leftmost vertices).

i. $\widehat{S}_i \subseteq S_i$, for each $i = 1, ..., n$.

ii. $v_i \in \widehat{S}_i$, for each $v_i \in V (\Rightarrow \widehat{V} \equiv V$ and $\widehat{s}(v_i) = s(v_i) = i)$.

Recall that $(\widehat{A}_1, ..., \widehat{A}_p)$ is the chain produced by the greedy algorithm when applied on instance $\widehat{\mathcal{I}}_{(p,k)}$. For the value $\widehat{g}(\widehat{A}_p)$ it's easy to prove that,

$$\widehat{g}(\widehat{A}_p) \geq k \cdot p, p \leq t \Rightarrow \widehat{\theta}_p \geq k, p \leq t$$

$$\widehat{g}(\widehat{A}_p) = n - p, p \geq t$$

Thus, when it comes to maximizing the function g the approximation ratio (for any connected graph) becomes,

$$\frac{e-1}{e+\frac{1}{k}}$$

This result holds for any graph (not necessarily connected), but first we will improve the approximation ratio using a different decomposition of the spanning tree and then we will prove that we can generalize for any graph.

3.4.2 Decomposition D2

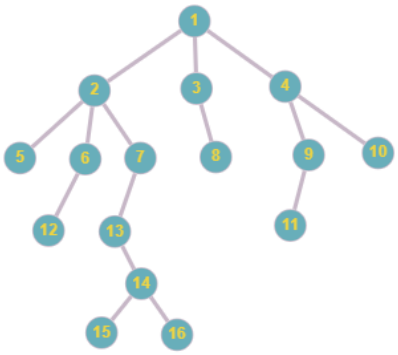
Improving the approximation ratio depends on the construction of the instance $\widehat{\mathcal{I}}_{(p,k)}$ (or equivalently of the graph $\widehat{G}$). We now describe a different decomposition of the spanning tree of a connected graph (again we consider connected graphs, for the moment), in the following procedure D2:

D2

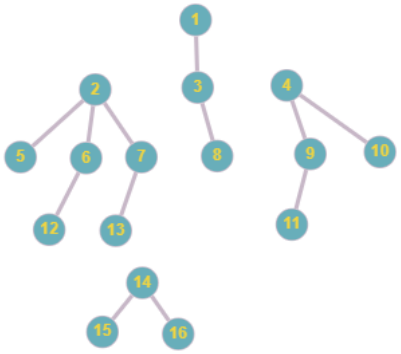
1. Choose a vertex u_i such that the subtree (we will call it $\widehat{G}_i$– it's the i-th tree produced by this procedure) that is formed by the vertex u_i and all its descendants has at least $k+2$ vertices. Note that step 1 must ensure that $\widehat{G}_i$ is minimal, which means that every one of its subtrees has at most $k+1$ vertices.

2. Remove the subtree.

3. Unless the remaining tree is null, if its total number of vertices is at least $k+2$, continue from step 1. Otherwise, connect again the vertex u_i to the remaining tree and consider this union as the graph $\widehat{G}_i$. Observe that if we assume that the edge $e = \{u_i, v\}$ is the one that connects the two graphs, then the subtree formed by v (see v as a child of u_i) and all its descendants has either a maximum height of $k-1$ or is a path of $k+1$ vertices.



A rooted tree T



The output of Decomposition D2 on input T

Figure 3.2: Decomposition D2 - An example (priority rule : (1) vertices of largest depth (2) leftmost vertices).

Let u_i be the vertex chosen in step 1 at the i-th iteration and $\widehat{G}_i$ the subtree of which is the root. It is obvious that the height of u_i is at most $k+1$, because otherwise $\widehat{G}_i$ wouldn't be minimal. If the height of u_i is at most k then u_i can cover all its descendants (since they are within k hops from u_i). Otherwise, if the height of u_i is $k+1$ then every subtree of $\widehat{G}_i$ with height equal to k is in fact a path of $k+1$ vertices, and that's because $\widehat{G}_i$ is minimal. As for the number of vertices that can be covered (for the latter case where the height of $\widehat{G}_i$ is $k+1$), there are two cases (now, we don't distinguish in which step each graph is produced as the following apply to every graph $\widehat{G}_i$):

1. $\widehat{G}_i$ *is a path of* $k+2$ *vertices.*
   Any internal vertex (except the root) can cover the rest of the vertices.

2. *The subtrees formed by the children of* u_i *form the graph* $X_i \cup Y_i$*, where* X_i *is a union of* λ_i *paths of* $k+1$ *vertices and* Y_i *is a union of the rest subtrees which are of height of at most* $k-1$.
   The vertex u_i covers $\lambda_i \cdot k + |V(Y_i)| \geq k+1$ vertices (not counting itself). Observe that $\frac{\lambda_i \cdot k + |V(Y_i)|}{|V(\widehat{G}_i)|-1} = \frac{\lambda_i \cdot k + |V(Y_i)|}{\lambda_i \cdot (k+1) + |V(Y_i)|} \geq \frac{k}{k+1}$.

Let again $(\widehat{A}_1, ..., \widehat{A}_p)$ be the chain produced by the greedy algorithm when applied on instance $\widehat{\mathcal{I}}_{(p,k)}$. For the value $\widehat{g}(\widehat{A}_p)$, we can prove,

$$\widehat{g}(\widehat{A}_p) \geq (k+1)p, p \leq t \Rightarrow \widehat{\theta}_p \geq k+1, p \leq t$$

$$\widehat{g}(\widehat{A}_t) \geq \frac{k}{k+1} \sum_{j=1}^{t} (|V(\widehat{G}_j)| - 1) = \frac{k}{k+1}(V(\widehat{G}) - t)$$

$$\Rightarrow \widehat{\sigma}_t \geq \frac{k}{k+1} \Rightarrow \widehat{\sigma}_p \geq \frac{k}{k+1}, p \geq t$$

Hence the approximation ratio for any connected graph, becomes

$$\min\{\frac{e-1}{e+\frac{1}{k+1}}, \frac{k}{k+1}\}$$

Note that $k \geq 2 \Rightarrow \frac{k}{k+1} \geq \frac{2}{3}$, hence for $k \geq 2$ the approximation ratio is

$$\frac{e-1}{e+\frac{1}{k+1}}$$

For $k = 1$ the approximation ratio improves to $\frac{1}{2}$, which we will further improve providing a deeper analysis on the lower bounds of the values $\widehat{\theta}_p, \widehat{\sigma}_p$, focusing mostly on those graphs $\widehat{G}_i$ (of the second case) for which the corresponding graph Y_i is null.

3.5 More on maximizing the function g of p-Max 1-hop Domination

Recall that every instance of p-Max 1-hop Domination can be transformed into an instance of MCP, for which the objective function f is expressed as $f(A) = g(A) + |A|, A \subseteq V$. The goal here is to further improve the lower-bounds of the values $\widehat{\theta}_p, \widehat{\sigma}_p$, which we achieved in the previous section.

Definition 3.5.1 (the class $S_{n,m}$)**.** *$S_{n,m}$ contains the trees (V, E):*

$$V = \{u_0, ..., u_n, ..., u_{n+m}\} \cup \{v_1, ..., v_m\}$$

$$E = \{\{u_0, u_1\}, ..., \{u_0, u_{n-1}\}, \{u_0, u_n\}, ..., \{u_0, u_{n+m}\}\} \cup \{\{u_{n+1}, v_1\}, ..., \{u_{n+m}, v_m\}\}$$

For the needs of our analysis we give the following definition for the *center* of a tree $T \in S_{n,m}$:

Definition 3.5.2 (center)**.** *The center of a tree $T \in S_{n,m}$ is the vertex u_0 in cases where $T \notin S_{0,2}$, while if $T \in S_{0,2}$ then the center of T is defined to be any of the two neighbors of u_0.*

Let G be a connected graph and $T_{spanning}$ one of its spanning trees. Let $\widehat{G} = \{\widehat{G}_1, ..., \widehat{G}_t\}$ be the output of the decomposition D2 on input $T_{spanning}$. With the analysis that has preceded on Decomposition D2 and specifically for the case where $k = 1$ we can prove the following lemma:

Lemma 3.5.1. *Every tree T of $n \geq 3$ nodes can be partitioned into a collection $\widehat{G}$ of pairwise vertex-disjoint graphs $\widehat{G}_1, ..., \widehat{G}_t$ so that (i) $V(G) = V(\widehat{G})$ (ii) $\widehat{G}_i \in S_{k_i,l_i}$, with $k_i + l_i \geq 2$ for $i \in [1, t]$.*

An algorithm that has been studied in [Fot+20] and it can be useful for our analysis is algorithm *Aux*, is presented below in pseudocode.

Algorithm 4 *Aux*

Input: *p, G of order n*

Output: *A coloring of the vertices of G*

1: Every vertex of G is white.
2: $sol(0) = \emptyset$
3: **for** $i = 1, 2, \ldots, p$ **do**
    Color in blue a vertex v that is not already blue, and if possible, color in red some white neighbors of v. Do this so as to maximize the number of newly covered vertices, under the constraint that the total number of red vertices is at most $n - p$.
4: $sol(i) \leftarrow sol(i-1) \cup \{\text{newly covered vertices}\}$
5: **end for**
6: **return** $sol(p)$

Let $g_{Aux}(p, G) = |sol(p)| - p$ be the total number of red vertices (covered by p blue vertices), derived from the coloring which algorithm *Aux* produces when it is applied on graph G. During the execution of algorithm *Aux*, we know that ties are broken arbitrarily. But in the case of the decomposed graph $\widehat{G}$ it will be necessary for our analysis, some ties to be broken based on the following priority rule: *Among equivalent vertices any vertex that has been designated the center of a graph $\widehat{G}_i$ is preferred over any other vertex, and an uncovered (white) vertex is preferred over any covered (red) vertex.*

Observation 3.5.1. *Let $B_1, ..., B_p$ where B_i contains the first i vertices that algorithm Aux colors in blue and R be the set of vertices which are red after the termination of the algorithm. There exists a chain $(A_1, ..., A_p)$ produced by the greedy algorithm used for maximizing the function $f(A) = g(A) + |A|$, such that*

- *If* $|R| < n - p$, *then*

$$B_i = A_i, \text{ for each } i = 1, ..., p$$

and since $B_p = A_p$ *we have* $g_{Aux}(p, G) = g(A_p)$

- *If* $|R| = n - p$, *then* $g_{Aux}(p, G) = g(A_p) = n - p$.

Let $H = \{\widehat{G}_i \in \widehat{G} \mid \widehat{G}_i \in S_{0,2}\}$ with $q = |H|$ and $H' = \{\widehat{G}_i \in \widehat{G} \mid \widehat{G}_i \in S_{0,3}\}$ with $q' = |H'|$. Wlog we assume that $\widehat{G} = \{\widehat{G}_1, ..., \widehat{G}_{t-q-q'}\} \cup H \cup H'$. Let $n_i = |V(\widehat{G}_i)|$, for each $i \in [1, t]$. For $i \in [1, t - q - q']$, consider the following numbers:

$$x_i = \left\{ \begin{array}{ll} 1 & \text{if } l_i \in \{0, 1\} \\ \left\lfloor \frac{l_i+1}{2} \right\rfloor & \text{if } k_i \geq 1, l_i \geq 2 \\ \left\lfloor \frac{l_i}{2} \right\rfloor & \text{if } k_i = 0, l_i \geq 4 \end{array} \right\}$$

Now, observe that the following hold, regarding $g_{Aux}(x_i, \widehat{G}_i)$:

1. $l_i = 0$
   $g_{Aux}(1, \widehat{G}_i) = k_i \geq 2$ and $g_{Aux}(1, \widehat{G}_i) = (k_i + 1) - 1 = n_i - 1$

2. $l_i = 1$
   $g_{Aux}(1, \widehat{G}_i) = k_i + 1 \geq 2$ and $g_{Aux}(1, \widehat{G}_i) \geq \frac{2}{3}((k_i + 3) - 1) = \frac{2}{3}(n_i - 1)$

3. $k_i \geq 1, l_i \geq 2$
   $g_{Aux}(\left\lfloor \frac{l_i+1}{2} \right\rfloor, \widehat{G}_i) = k_i + l_i \geq 2\left\lfloor \frac{l_i+1}{2} \right\rfloor$. It's $\frac{k_i+l_i}{k_i+2l_i+1-\left\lfloor \frac{l_i+1}{2} \right\rfloor} \geq \frac{k_i+l_i}{1.5(k_i+l_i)+0.5} \geq \frac{3}{5} \Rightarrow g_{Aux}(\left\lfloor \frac{l_i+1}{2} \right\rfloor, \widehat{G}_i) \geq \frac{3}{5}(n_i - \left\lfloor \frac{l_i+1}{2} \right\rfloor)$.

4. $k_i = 0, l_i \geq 4$
   $g_{Aux}(\left\lfloor \frac{l_i}{2} \right\rfloor, \widehat{G}_i) = l_i$. If l_i is even then $\frac{l_i}{2l_i+1-\left\lfloor \frac{l_i}{2} \right\rfloor} = \frac{l_i}{1.5l_i+1} \geq \frac{4}{7} \Rightarrow g_{Aux}(\left\lfloor \frac{l_i}{2} \right\rfloor, \widehat{G}_i) \geq \frac{4}{7}(n_i - \left\lfloor \frac{l_i}{2} \right\rfloor)$. While, if l_i is odd then $\frac{l_i}{2l_i+1-\left\lfloor \frac{l_i}{2} \right\rfloor} = \frac{l_i}{1.5(l_i+1)} \geq \frac{5}{9} \Rightarrow g_{Aux}(\left\lfloor \frac{l_i}{2} \right\rfloor, \widehat{G}_i) \geq \frac{5}{9}(n_i - \left\lfloor \frac{l_i}{2} \right\rfloor)$.

Lemma 3.5.2. *For* $x = \sum_{i=1}^{t-q-q'} x_i$: *(a)* $g_{Aux}(p, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i) \geq 2p$, *for* $p \leq x$ *and (b)* $g_{Aux}(p, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i) \geq \frac{5}{9}((n - |V(H)| - |V(H')|) - p)$, *for* $p \geq x$.

Proof. *(a)* It is clear that $g_{Aux}(p, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i) \geq 2p$, for $p \leq t - q - q'$.
For $t - q - q' < p \leq x$:

$$g_{Aux}(p, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i) = \sum_{i=1}^{t-q-q'} (k_i + l_i) = \sum_{i=1}^{t-q-q'} g_{Aux}(x_i, \widehat{G}_i) \geq \sum_{i=1}^{t-q-q'} 2x_i = 2x \geq 2p$$

(b)

$$g_{Aux}(x, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i) = \sum_{i=1}^{t-q-q'} (k_i + l_i) = \sum_{i=1}^{t-q-q'} g_{Aux}(x_i, \widehat{G}_i) \geq \sum_{i=1}^{t-q-q'} \frac{5}{9}(n_i - x_i)$$

$$\Rightarrow g_{Aux}(x, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i) \geq \frac{5}{9}((n - |V(H)| - |V(H')|) - x)$$

Thus, $g_{Aux}(p, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i) \geq \frac{5}{9}((n - |V(H)| - |V(H')|) - p)$, for $p \geq x$. □

Lemma 3.5.3. *For $y = q + \left\lfloor \frac{q}{12} \right\rfloor$: (a) $g_{Aux}(p, H) \geq \frac{25}{13}p$, for $p \leq y$ and (b) $g_{Aux}(p, H) \geq \frac{25}{47}(|V(H)| - p)$, for $p \geq y + 1$.*

Proof. It's $g_{Aux}((1+\alpha)q, H) = (2+\alpha)q$, for $\alpha \leq 1$. For $\alpha = \frac{\left\lfloor \frac{q}{12} \right\rfloor}{q}$ it holds that $\frac{g_{Aux}((1+\alpha)q,H)}{(1+\alpha)q} = \frac{(2+\alpha)q}{(1+\alpha)q} \geq \frac{25}{13}$ and $\frac{g_{Aux}((1+\alpha)q+1,H)}{|V(H)|-((1+\alpha)q+1)} = \frac{(2+\alpha)q+1}{5q-(1+\alpha)q-1} = \frac{(2+\alpha)q+1}{(4-\alpha)q-1} > \frac{2q+\frac{q}{12}}{4q-\frac{q}{12}} = \frac{25}{47}$. Thus for $y = q + \left\lfloor \frac{q}{12} \right\rfloor$, it's $g_{Aux}(p, H) \geq \frac{25}{13}p$, for $p \leq y$ and $g_{Aux}(p, H) \geq \frac{25}{47}(|V(H)| - p)$, for $p \geq y + 1$. □

Lemma 3.5.4. *For $q > 0, q' = 0$: (a) $g_{Aux}(p, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i \cup H) \geq \frac{25}{13}p$, for $p \leq x + y$ and (b) $g_{Aux}(p, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i \cup H) \geq \frac{25}{47}((n - |V(H')|) - p)$, for $p \geq x + y + 1$.*

Proof. It is clear that $g_{Aux}(p, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i \cup H) \geq \frac{25}{13}p$, for $p \leq (t - q - q') + y$. For $(t - q - q') + y < p \leq x + y$:

$$g_{Aux}(p, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i \cup H) \geq \sum_{i=1}^{t-q-q'} (k_i + l_i) + (2+\alpha)q \geq 2x + \frac{25}{13}y \geq \frac{25}{13}(x+y) \geq \frac{25}{13}p$$

Furthermore,

$$g_{Aux}(x + y + 1, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i \cup H) \geq \sum_{i=1}^{t-q-q'} (k_i + l_i) + ((2+\alpha)q + 1)$$

$$\geq \frac{5}{9}((n - |V(H)| - |V(H')|) - x) + \frac{25}{47}(|V(H)| - (y+1)) \geq \frac{25}{47}((n - |V(H')|) - (x+y+1))$$

$$\Rightarrow g_{Aux}(p, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i \cup H) \geq \frac{25}{47}((n - |V(H')|) - p), \text{ for } p \geq x + y + 1$$

□

Lemma 3.5.5. *For $y' = q' + \left\lfloor \frac{q'}{2} \right\rfloor$: (a) $g_{Aux}(p, H') \geq 2p$, for $p \leq y'$ and (b) $g_{Aux}(p, H') \geq \frac{6}{11}(|V(H')| - p)$, for $p \geq y' + 1$.*

Proof. For $\beta \leq \frac{1}{2}$ it's $g_{Aux}((1+\beta)q', H') = 3q' \geq 2(1+\beta)q'$ and for $\beta = \frac{\left\lfloor \frac{q'}{2} \right\rfloor}{q'}$ it's $\frac{g_{Aux}((1+\beta)q'+1,H')}{|V(H')|-((1+\beta)q'+1)} = \frac{3q'}{7q'-(1+\beta)q'-1} = \frac{3q'}{(6-\beta)q'-1} > \frac{3q'}{6q'-\frac{q'}{2}} = \frac{6}{11}$. Thus for $y' = q' + \left\lfloor \frac{q'}{2} \right\rfloor$ it is $g_{Aux}(p, H') \geq 2p$, for $p \leq y'$ and $g_{Aux}(p, H') \geq \frac{6}{11}(|V(H')| - p)$, for $p \geq y' + 1$. □

Lemma 3.5.6. *For $q = 0, q' > 0$: (a) $g_{Aux}(p, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i \cup H') \geq 2p$, for $p \leq x + y'$ and (b) $g_{Aux}(p, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i \cup H') \geq \frac{6}{11}((n - |V(H)|) - p)$, for $p \geq x + y' + 1$.*

Proof. It is clear that $g_{Aux}(p, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i \cup H') \geq 2p$, for $p \leq (t-q-q') + q'$. For $t - q < p \leq x + y'$:

$$g_{Aux}(p, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i \cup H') = \sum_{i=1}^{t-q-q'} (k_i + l_i) + 3q' \geq 2x + 2y' \geq 2p$$

Furthermore,

$$g_{Aux}(x + y' + 1, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i \cup H') = \sum_{i=1}^{t-q-q'} (k_i + l_i) + 3q'$$

$$\geq \frac{5}{9}((n-|V(H)|-|V(H')|)-x)+\frac{6}{11}(|V(H')|-(y'+1)) \geq \frac{6}{11}((n-|V(H)|)-(x+y'+1))$$

$$\Rightarrow g_{Aux}(p, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i \cup H') \geq \frac{6}{11}((n - |V(H)|) - p), \text{ for } p \geq x + y' + 1$$

□

Lemma 3.5.7. *For $q > 0, q' \notin \{1, 3\}$: (a) $g_{Aux}(p, \widehat{G}) \geq \frac{25}{13}p$, for $p \leq x + y + y'$ and (b) $g_{Aux}(p, \widehat{G}) \geq \frac{25}{47}(n - p)$, for $p \geq x + y + y' + 1$.*

Proof. If $q' \notin \{1, 3\}$ then $\frac{g_{Aux}((1+\beta)q', H')}{|V(H')|-(1+\beta)q'} = \frac{3}{6-\beta} \geq \frac{3}{6-0.4} = \frac{15}{28} \Rightarrow g_{Aux}(y', H') \geq \frac{15}{28}(|V(H')| - y')$ which implies that

$$g_{Aux}(p, H') \geq \frac{15}{28}(|V(H')| - p), \text{ for } p \geq y'$$

For $p \leq (t - q - q') + y + q'$, it is $g_{Aux}(p, \widehat{G}) \geq \frac{25}{13}p$.
For $t - q + y < p \leq x + y + y'$, it is

$$g_{Aux}(p, \widehat{G}) \geq \sum_{i=1}^{t-q-q'} (k_i + l_i) + (2 + \alpha)q + 3q' \geq 2x + \frac{25}{13}y + 2y'$$

$$\geq \frac{25}{13}(x + y + y') \geq \frac{25}{13}p$$

Furthermore,

$$g_{Aux}(x+y+y'+1,\widehat{G}) \geq \sum_{i=1}^{t-q-q'}(k_i+l_i)+(2+\alpha)q+1+3q'$$

$$\geq \frac{5}{9}((n-|V(H)|-|V(H')|)-x)+\frac{25}{47}(|V(H)|-(y+1))+\frac{15}{28}(|V(H')|-y')$$

$$\geq \frac{25}{47}(n-(x+y+y'+1)) \Rightarrow g_{Aux}(p,\widehat{G}) \geq \frac{25}{47}(n-p), \text{ for } p \geq x+y+y'+1$$

□

Lemma 3.5.8. *For $q > 0, q' \in \{1,3\}$: (a) $g_{Aux}(p,\widehat{G}) \geq \frac{25}{13}p$, for $p \leq x+y+y'+2$ and (b) $g_{Aux}(p,\widehat{G}) \geq \frac{25}{47}(n-p)$, for $p \geq x+y+y'+3$.*

Proof. For $q' \in \{1,3\}$, let $\widehat{G}_j \in H$. From lemma 3.5.4, we know that for $y = (q-1)+\left\lfloor \frac{q-1}{12} \right\rfloor$ it's $g_{Aux}(p, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i \cup (H \backslash \widehat{G}_j)) \geq \frac{25}{13}p$, for $p \leq x+y$ and $g_{Aux}(p, \bigcup_{i=1}^{t-q-q'} \widehat{G}_i \cup (H \backslash \widehat{G}_j)) \geq \frac{25}{47}((n-|V(\widehat{G}_j)|-|V(H')|)-p)$, for $p \geq x+y+1$. Furthermore, $g_{Aux}(p, H' \cup \widehat{G}_j) \geq 2p$, for $p \leq y'+2$ and $g_{Aux}(p, H' \cup \widehat{G}_j) \geq \frac{3}{5}(|V(H')|+|V(\widehat{G}_j)|-p)$, for $p \geq y'+2$. So, $g_{Aux}(p,\widehat{G}) \geq \frac{25}{13}p$, for $p \leq x+y+(y'+2)$ and $g_{Aux}(p,\widehat{G}) \geq \frac{25}{47}(n-p)$, for $p \geq x+(y+1)+(y'+2)$. □

Observation 3.5.2. *By lemmas 3.5.2 to 3.5.8 it is made obvious that for the graph $\widehat{G}$ there exists an integer d and numbers $\widehat{\theta}$, $\widehat{\sigma}$ such that*

$$g_{Aux}(p,\widehat{G}) \geq \widehat{\theta} \cdot p \text{ for } p \leq d, \text{ where } \widehat{\theta} = \frac{25}{13}$$

$$g_{Aux}(p,\widehat{G}) \geq \widehat{\sigma} \cdot (n-p) \text{ for } p > d, \text{ where } \widehat{\sigma} = \frac{25}{47}$$

Observation 3.5.3. *The parameter α used in lemma 3.5.3 can be written as $\alpha = \frac{\lfloor c \cdot q \rfloor}{q}$ for some quantity c. Throughout the proof we have considered $c = \frac{1}{12}$. As already shown in the proof, it holds $\frac{g_{Aux}((1+\alpha)q,H)}{(1+\alpha)q} = \frac{(2+\alpha)q}{(1+\alpha)q} \geq \frac{2+c}{1+c}$ and $\frac{g_{Aux}((1+\alpha)q+1,H)}{|V(H)|-((1+\alpha)q+1)} = \frac{(2+\alpha)q+1}{5q-(1+\alpha)q-1} = \frac{(2+\alpha)q+1}{(4-\alpha)q-1} > \frac{2q+c \cdot q}{4q-c \cdot q} = \frac{2+c}{4-c}$. If we take,*

$$\frac{\frac{2+c}{1+c}(e-1)}{1+\frac{2+c}{1+c}e} = \frac{2+c}{4-c}$$

, it gives $c = 1-\frac{5}{2e}$. Thus, if we substitute $c_{old} = \frac{1}{12}$ for $c_{new} = 1-\frac{5}{2e}$ in lemmas 3.5.4, 3.5.7 and 3.5.8, which means to substitute the value $\frac{2+c_{old}}{1+c_{old}} (= \frac{25}{13})$ for $\frac{2+c_{new}}{1+c_{new}}$ and the value $\frac{2+c_{old}}{4-c_{old}} (= \frac{25}{47})$ for $\frac{2+c_{new}}{4-c_{new}}$, it easily follows that for the numbers $\widehat{\theta}, \widehat{\sigma}$ discussed in Observation 3.5.2 it holds : $\widehat{\theta} = \frac{2+c_{new}}{1+c_{new}}$, $\widehat{\sigma} = \frac{2+c_{new}}{4-c_{new}}$.

Let again $f(A) = g(A) + |A|$, be the objective function of p-Max 1-hop Domination.

Theorem 3.5.1. *There exists a $\frac{6e-5}{6e+5}$-approximation polynomial time algorithm for the problem of maximizing the function $g : 2^V \to \mathbb{R}^+$ of the p-Max 1-hop Domination problem, under cardinality constraints (of the form $|A| = p$) for connected graphs.*

Proof. Let G be a connected graph (of at least 3 vertices) and $\widehat{G}$ be its decomposed graph as discussed throughout the previous sections. Let $(A_1, ..., A_p)$, $(\widehat{A}_1, ..., \widehat{A}_p)$ be the chains produced by the greedy algorithm when it is applied on instances $\mathcal{I}_{(p,1)}$ and $\widehat{\mathcal{I}}_{(p,1)}$ respectively. Using Observations 3.5.1 to 3.5.3 we have

$$\widehat{g}(\widehat{A}_p) \geq \widehat{\theta} \cdot p \text{ for } p \leq d$$

$$\widehat{g}(\widehat{A}_p) \geq \widehat{\sigma}(n-p) \text{ for } p > d$$

Thus,

$$\max\{g(A_p), g(\widehat{A}_p)\} \geq \min\{\frac{\widehat{\theta}(e-1)}{1+\widehat{\theta}e}, \widehat{\sigma}\} g(A_p^*) = \frac{6e-5}{6e+5} g(A_p^*)$$

□

Corollary 3.5.1.1. *Theorem 3.5.1 also holds for any graph G, all connected components of which have at least three vertices, since lemma 3.5.1 can be applied to such a graph G.*

3.6 Generalization

We will assume that whenever we refer to a decomposition we mean decomposition D2 but the same generalized results can be obtained if decomposition D1 is applied instead.

Now, let G be any graph, not necessarily connected. Let also G_1 be the union of all its connected components, each having at least $k+2$ vertices and G_2 be the union of all the remaining connected components (each having of course, at most $k+1$ vertices). Let f_G, f_{G_1}, f_{G_2} denote the functions we want to maximize in each instance where the given graphs are respectively G, G_1, G_2 and the number of dominators in each instance is p, p_1, p_2 respectively, with $p_1 + p_2 = p$. Let $(A_1^G, ..., A_p^G), (A_1^{G_1}, ..., A_{p_1}^{G_1}), (A_1^{G_2}, ..., A_{p_2}^{G_2})$ be the chains produced by the greedy algorithm for each instance and $A_p^{G*}, A_{p_1}^{G_1*}, A_{p_2}^{G_2*}$ be the respective optimal solutions. We assume that same ties in different instances are broken in the same way. By the definition of the greedy algorithm it is self-proven that

$$f_G(A_p^G) \geq f_{G_1}(A_{p_1}^{G_1}) + f_{G_2}(A_{p_2}^{G_2})$$

which implies that

$$g_G(A_p^G) \geq g_{G_1}(A_{p_1}^{G_1}) + g_{G_2}(A_{p_2}^{G_2})$$

Another useful observation is that we can decompose each tree of the spanning forest of the graph G_1 in the same way we decompose the spanning tree of any connected graph (of at least $k+2$ vertices). Let $\widehat{G}$ be the union of the decomposed spanning forest $\widehat{G}_1$ (of G_1) and the graph G_2. Similarly, we define the functions $\widehat{f}_G, \widehat{f}_{G_1}$ and the chains $(\widehat{A}_1^G, ..., \widehat{A}_p^G), (\widehat{A}_1^{G_1}, ..., \widehat{A}_p^{G_1})$. We also have,

$$\widehat{g}_G(\widehat{A}_p^G) \geq \widehat{g}_{G_1}(\widehat{A}_{p_1}^{G_1}) + g_{G_2}(A_{p_2}^{G_2})$$

Hence,

$$\max\{g_G(A_p^G), \widehat{g}_G(\widehat{A}_p^G)\} \geq \max\{g_{G_1}(A_{p_1}^{G_1}), \widehat{g}_{G_1}(\widehat{A}_{p_1}^{G_1})\} + g_{G_2}(A_{p_2}^{G_2})$$

Let ρ, $0 < \rho \leq 1$ be such that for any graph G_1 (i.e. a graph, every connected component of which has at least $k+2$ vertices), and the corresponding decomposed graph $\widehat{G}_1$, it holds

$$\max\{g_{G_1}(A_{p_1}^{G_1}), \widehat{g}_{G_1}(\widehat{A}_{p_1}^{G_1})\} \geq \rho \cdot g_{G_1}(A_{p_1}^{G_1 *})$$

Furthermore, we can easily show that $g_{G_2}(A_{p_2}^{G_2}) = g_{G_2}(A_{p_2}^{G_2 *})$; since every connected component of G_2 has at most $k+1$ vertices it follows that any vertex can cover all of the rest of the vertices (within the connected component). Thus,

$$\max\{g_G(A_p^G), \widehat{g}_G(\widehat{A}_p^G)\} \geq \rho \cdot (g_{G_1}(A_{p_1}^{G_1 *}) + g_{G_2}(A_{p_2}^{G_2 *}))$$

If we take $p_1 = |A_p^{G*} \cap V(G_1)|$ and $p_2 = p - p_1$, we get

$$\max\{g_G(A_p^G), g_G(\widehat{A}_p^G)\} \geq \max\{g_G(A_p^G), \widehat{g}_G(\widehat{A}_p^G)\} \geq \rho \cdot g_G(A_p^{G*})$$

3.7 Notes on the weighted version of Maximum Coverage Problem

For the weighted version of MCP where each element e_j has a weight $w(e_j) \geq 0$, the objective is to maximize the function

$$f(\mathcal{S}) = \overline{w}(\bigcup_{S_i \in \mathcal{S}} S_i) = \sum_{e_j \in \bigcup_{S_i \in \mathcal{S}} S_i} w(e_j)$$

, over all sets of subsets $\mathcal{S} \subseteq \{S_1, ..., S_m\}$ with $|\mathcal{S}| = p$.

Once again we assume that the instances we study have the same property (**A**) : we can choose one element v_i out of every set S_i so that no two elements v_i, v_j are the same (for $i \neq j$). The union of these elements forms

the set V and the function f as we have already proven can be equivalently defined over 2^V as :

$$f(A) = \overline{w}(Coverage(A)) = \overline{w}(\bigcup_{v \in A} S_{s(v)})$$

It's easy to observe that

$$f(A) = g(A) + \overline{w}(A), A \subseteq V$$

We will create an infinite number of instances for which an approximate solution (produced by the greedy algorithm) for the problem of maximizing f if used to approximate g can lead to arbitrarily poor results. For example, let $G = (V, E)$, with $V = \{v_1, ..., v_n\}$ be a star, with v_1 being its center and $v_2, ..., v_n$ the leaves. The weights are : $w(v_1) = \alpha \cdot (n-1) \cdot W$, and $w(v_2) = ... = w(v_n) = W$, with $\alpha, W > 0$. It holds,

$$f(v_1) = (\alpha + 1)(n-1)W, g(v_1) = (n-1)W$$

$$f(v_i) = (\alpha(n-1) + 1)W, g(v_i) = \alpha(n-1)W, i > 1$$

Observe that $\frac{g(v_1)}{g(v_i)} = \frac{1}{\alpha} \to 0$, as $\alpha \to \infty$.

Let $A \subseteq B \subseteq V$ and $v \in V \backslash B$. Using the property of submodularity of f we have,

$$f(A \cup \{v\}) - f(A) \geq f(B \cup \{v\}) - f(B) \Rightarrow$$

$$g(A \cup \{v\}) + \overline{w}(A \cup \{v\}) - (g(A) + \overline{w}(A)) \geq g(B \cup \{v\}) + \overline{w}(B \cup \{v\}) - (g(B) + \overline{w}(B))$$

, from which we get

$$g(A \cup \{v\}) - g(A) \geq g(B \cup \{v\}) - g(B), \text{for each } A \subseteq B \subseteq V \text{ and } v \in V \backslash B$$

Thus g is submodular, but g is in general nonmonotone. For the problem of approximately maximizing a general nonnegative nonmonotone submodular function subject to cardinality contraints the reader is encouraged to refer to [Buc+14].

Chapter 4

(In)approximability results for p-Max k-hop Domination and variants

4.1 Problem definition

Definition 4.1.1 (p-Max k-hop Domination)**.** *Given an undirected graph $G = (V, E)$ and integers p, k, find a set D of p vertices of G so that the total number of vertices that can be reached from some vertex in D in at most k hops (i.e. traversing at most k edges) is maximized. We refer to the vertices of D as the dominators.*

We also define some notions and functions which will be used in this section.

Definition 4.1.2. *A vertex $v \in D$ (i.e. a dominator) is said to k-dominate a vertex u if $d(v, u) \leq k$, where $d(v, u)$ is the distance between v and u in a given graph $G = (V, E)$.*

Definition 4.1.3. *Given a graph $G = (V, E)$ and a set $D \subseteq V$ we define*

$$Coverage_k(D) = \bigcup_{v \in D} \mathcal{N}_k[v], \text{ where } \mathcal{N}_k[v] = \Big| \bigcup_{u \in V : d(v,u) \leq k} u \Big|$$

For simplicity we will write $Coverage_k(v)$ instead of $Coverage_k(\{v\})$.

4.2 Hardness of p-Max k-hop Domination

The reader will come across the notion of reduction many times, therefore we provide an informal definition so that it will be clear whenever it is implicitly used (see [AB09] for the formal definition of *reducibility*).

Definition 4.2.1. *A polynomial time reduction from a problem A to a problem B, is a pair (R, S) of (polynomial time) algorithms where*

- *R takes as input an instance of problem A and transforms it into an instance of problem B.*
- *S takes as input a solution for problem B and outputs a solution for problem A.*

If such an algorithm exists then we say that the problem A is reducible to B.

It's easy to see that if A is reducible to B, then solving B is at least as hard as solving A. Note, that all reductions in the current work are polynomial time reductions.

We now proceed with the proof of the hardness of p-Max k-hop Domination.

Definition 4.2.2 (k-hop dominating set [BM14])**.** *Given an undirected graph $G = (V, E)$, a subset $D \subseteq V$ of its vertex set is a k-hop dominating set iff for every vertex $v \in V \backslash D$ there exists some vertex $u \in D$ such that $d(u, v) \leq k$.*

If D is a k-hop dominating set then it is easy to see that $Coverage_k(D) = V(G)$.

Definition 4.2.3 (The decision problem P'_k [BM14])**.** *Given an undirected graph $G = (V, E)$, is there a k-hop dominating set of size at most r ?*

Definition 4.2.4 (The optimization problem P_k [BM14])**.** *Given an undirected graph $G = (V, E)$, find a k-hop dominating set of minimum size.*

Definition 4.2.5. *A vertex cover of a graph $G = (V, E)$ is a subset V' of its vertex set V, such that for every edge $\{u, v\} \in E$, at least one of its endpoints is included in V'.*

Definition 4.2.6 (Vertex Cover Problem)**.** *Given an undirected graph $G = (V, E)$, does G contain a Vertex Cover of size at most r ?*

The problem P_k of finding a k-hop dominating set of minimum size is shown to be computationally hard [BM14]. To show this, they prove that P'_k is **NP**-complete by establishing a (polynomial time) reduction of the Vertex Cover Problem (a well known **NP**-complete problem) to the decision problem P'_k.

Proposition 4.2.1. p-Max k-hop Domination *is computationally hard.*

Proof. Let A be a polynomial time algorithm which behaves as follows: It takes as input an undirected graph $G = (V, E)$ and an integer $r \leq |V(G)|$. If for some $p = 1, ..., r$ the optimal solution of the p-Max k-hop Domination problem is equal to $V(G)$ then its output is "YES", otherwise its output is "NO". If p-Max k-hop Domination can be solved in polynomial time, then so does A, but algorithm A also solves the decision problem P'_k which is shown to be **NP**-complete. □

4.3 On p-Max K-hop Domination for fixed K

We have already shown in a previous chapter that p-Max k-hop Domination is a special case of MCP;simply, for a given graph G take $\mathcal{S} = \{\mathcal{N}_k[u] : u \in V(G)\}$ and consider the variant of MCP, in which exactly p sets must be chosen. Thus, it can be approximated within $(1 - \frac{1}{e})$ [Hoc96]. Theorem 3.1.2 [Fei98] states that no approximation algorithm with a constant approximation ratio better that $(1 - \frac{1}{e})$ exists for (weighted) MCP, unless $\mathbf{P} = \mathbf{NP}$, even if the weights of elements are restricted to 1. It can be shown [MO11] that p-Max 1-hop Domination[1] has the same bound on inapproximability under $\mathbf{P} \neq \mathbf{NP}$, using a gap-preserving reduction from MCP, which means that the ratio $(1 - \frac{1}{e})$ is the best we can hope for. We will show that p-Max 2-hop Domination has the same bound on inapproximability, using a gap-preserving reduction from p-Max 1-hop Domination, and then generalize this result for p-Max K-hop Domination[2] using a gap-preserving reduction from p-Max $(K-1)$-hop Domination, for any fixed $K \geq 2$.

4.3.1 Hardness of Approximation

Let Π be an optimization problem where the objective is to maximize or minimize a specific function f. Let also $OPT(\mathcal{I})$ denote the optimal value of f (maximum or minimum) for any instance $\mathcal{I}$ of Π.

Definition 4.3.1 (c-gap Π)**.** *For a maximum optimization problem* Π *the goal is to distinguish between two cases for* $OPT(\mathcal{I})$*:*

1. *"YES" instance* : $OPT(\mathcal{I}) \geq x$
2. *"NO" instance* $OPT(\mathcal{I}) < c \cdot x; c < 1$

Respectively, for a minimum optimization problem Π *the goal is to distinguish between the two cases:*

1. *"YES" instance* : $OPT(\mathcal{I}) \leq x$
2. *"NO" instance* : $OPT(\mathcal{I}) > c \cdot x; c > 1$

The c-gap Π problem is a promise problem which means that not all instances are possible. Hence we assume that we are promised that the input $\mathcal{I}$ of the c-gap Π problem is either a "YES" or a "NO" instance. Also note that c is not necessarily a constant. Generally, x is a function of the instance, that is $x = x(\mathcal{I})$ an c is a function of the size $|\mathcal{I}|$ of the instance, that is $c = c(|\mathcal{I}|)$.

[1] The p-Max 1-hop Domination problem is originally defined in [MO11], where it is called k-VERTEX MAXIMUM DOMINATION (k-MaxVD), with parameter k denoting the number of "dominators" (which in our definition is denoted by p).

[2] We assume that the number of hops is part of the input (instance) of the p-Max k-hop Domination problem. Whenever we want to refer to a variant of the problem where the number of hops is some fixed integer, then we write it with capital K.

Lemma 4.3.1. *If the c-gap* Π *problem is hard, then c-approximating* Π *is hard as well.*

Proof. Let Π be a maximum optimization problem (the other case is symmetrical). Assume that there exists a polynomial time c-approximation algorithm C for the problem Π. It is true that

$$c \cdot OPT(\mathcal{I}) \leq C(\mathcal{I}) \leq OPT(\mathcal{I})$$

Consider an algorithm D which behaves as follows: If $C(\mathcal{I}) \geq c \cdot x$, then $D(\mathcal{I}) =$ "YES", while if $C(\mathcal{I}) < c \cdot x$, then $D(\mathcal{I}) =$ "NO". Now observe that if $OPT(\mathcal{I}) \geq x$, then $C(\mathcal{I}) \geq c \cdot OPT \geq c \cdot x \Rightarrow D(\mathcal{I}) =$"YES". While if $OPT(\mathcal{I}) < c \cdot x$, then $C(\mathcal{I}) \leq OPT(\mathcal{I}) < c \cdot x \Rightarrow D(\mathcal{I}) =$ NO. It is made obvious that D solves the c-gap Π problem, but c-gap Π is hard, thus c-approximating Π is hard as well. □

The previous lemma is exceptionally useful since if we know that the c-gap version of a certain optimization problem is hard, then we deduce that it's also hard to c-approximate it. But the question that arises is how to prove that a c-gap problem is hard ? One possible answer is by reduction from a known to be hard c_1-gap Π_1 problem to a c_2-gap Π_2 problem.

Definition 4.3.2 (Gap-preserving reduction)**.** *Let* Π_1, Π_2 *be maximum (or minimum) optimization problems. Let* F *be a function that transforms an instance* $\mathcal{I}_1$ *of* Π_1 *into an instance* $\mathcal{I}_2$ *of* Π_2 *such that the following are satisfied,*

For maximization problems :

1. *If* $OPT_{\Pi_1}(\mathcal{I}_1) \geq x_1$*, then* $OPT_{\Pi_2}(\mathcal{I}_2) \geq x_2$
2. *If* $OPT_{\Pi_1}(\mathcal{I}_1) \leq c_1 \cdot x_1$*, then* $OPT_{\Pi_2}(\mathcal{I}_2) \leq c_2 \cdot x_2; c_1, c_2 < 1$

For minimization problems :

1. *If* $OPT_{\Pi_1}(\mathcal{I}_1) \leq x_1$*, then* $OPT_{\Pi_2}(\mathcal{I}_2) \leq x_2$
2. *If* $OPT_{\Pi_1}(\mathcal{I}_1) \geq c_1 \cdot x_1$*, then* $OPT_{\Pi_2}(\mathcal{I}_2) \geq c_2 \cdot x_2; c_1, c_2 > 1$

Again we can consider $x_1 = x_1(\mathcal{I}_1), c_1 = c_1(|\mathcal{I}_1|)$ *and similarly for* x_2, c_2*. If additionally the reduction from instance* $\mathcal{I}_1$ *to* $\mathcal{I}_2$ *can be done in polynomial time, then the set of functions* (x_1, c_1, x_2, c_2) *is a gap-preserving reduction from* c_1*-gap* Π_1 *to* c_2*-gap* Π_2*.*

Another answer to our central question regarding the hardness of approximation is what it is known as *gap-introducing reduction* which along with gap-preserving reduction and the PCP theorem establish a powerful mechanism for obtaining inapproximability results [Vaz03].

4.3.2 An overview of Theorem 3.1.2

A study of the proof of Theorem 3.1.2 [Fei98], leads to a very useful result according to which the $(1 - \frac{1}{e} + \varepsilon)$-gap version of MCP is **NP**-hard, for any $\varepsilon > 0$. We will try to highlight the points within the proof where this result it is implied.

Definition 4.3.3 (MAX SAT)**.** *Given a CNF formula ϕ (i.e. ϕ is the conjuction of clauses each of which is a disjunction of literals[3]), and weights w_c associated with each clause c, find an assignment of the variables so that the total weight of the satisfied clauses is maximized.*

A variant of MAX SAT is MAX tSAT where each clause has exactly t literals and in turn, MAX tSAT-B is a variant of the latter problem where an instance of MAX tSAT-B is a tCNF-B formula ϕ meaning that each clause of ϕ has exactly t literals and each variable occurs at most B times.

In the work of [Fei98] we come across the following proposition,

Proposition 4.3.1 ([Fei98])**.** *For some $\epsilon > 0$, it is* **NP***-hard to distinguish between satisfiable 3CNF-5 formulas, and 3CNF-5 formulas in which at most a $(1 - \epsilon)$-fraction of the clauses can be satisfied simultaneously.*

The previous proposition equivalently states that for some $\epsilon > 0$ the $(1 - \epsilon)$-gap Max 3SAT-5 problem which is the $(1 - \epsilon)$-gap version of the optimization problem MAX 3SAT-5 is **NP**-hard. A reduction is performed from an instance of MAX 3SAT-5 which is a 3CNF-5 formula ϕ to an instance $\mathcal{I}_{MCP}$ for MCP with a ground set of N points and p the number of sets to choose, such that

- If ϕ is satisfiable, then all N points can be covered by p sets.
- If only a $(1 - \epsilon)$-fraction of the original formula are simultaneously satisfied, then p sets can cover at most $(1 - \frac{1}{e} + \gamma(\kappa)))N$ points, for some function γ of a parameter κ with respect to the instance $\mathcal{I}_{MCP}$ with $\gamma(\kappa) \to 0$, as $\kappa \to \infty$.

Observe that it is implicitly stated that a reduction can be established from $(1-\epsilon)$-gap MAX 3SAT-5 to $(1 - \frac{1}{e} + \gamma(\kappa))$-gap MCP. Thus, we deduce that $(1 - \frac{1}{e} + \varepsilon)$-gap MCP is **NP**-hard (for any $\varepsilon > 0$).

4.3.3 Inapproximability bound of p-Max K-hop Domination

Let $G = (V, E)$ be a graph, where $V = \{v_1, ..., v_n\}$. We will show how to construct the graph G_R which will be used to establish the desired gap-preserving reduction.

[3]Recall that a literal is a boolean variable in either positive form (x) or negated form ($\neg x$).

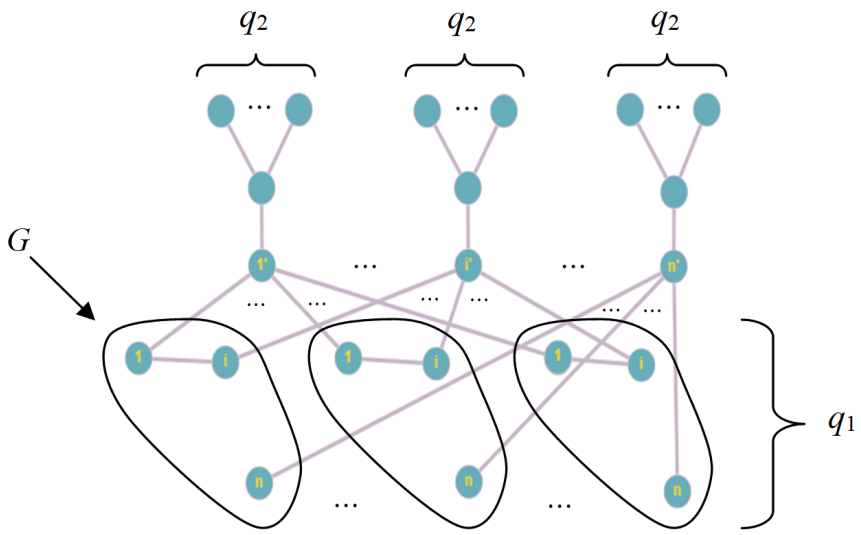


Figure 4.1: The graph G_R

First we define the set of vertices $V(G_R)$ which consists of the following sets of vertices :

- $V^{(1)} = \{v_1^{(1)}, ..., v_n^{(1)}\}$
- $V^{(2)} = \{v_1^{(2)}, ..., v_n^{(2)}\}$

  ...
- $V^{(q_1)} = \{v_1^{(q_1)}, ..., v_n^{(q_1)}\}$
- $V_1 = \{u_0^1, ..., u_{q_2}^1\}$
- $V_2 = \{u_0^2, ..., u_{q_2}^2\}$

  ...
- $V_n = \{u_0^n, ..., u_{q_2}^n\}$
- $V' = \{v'_1, ..., v'_n\}$

The set of edges $E(G_R)$ is defined as the union of the following sets of edges:

- $E^{(1)} = \big\{\{v_i^{(1)}, v_j^{(1)}\} \mid \{v_i, v_j\} \in E)\big\}$
- $E^{(2)} = \big\{\{v_i^{(2)}, v_j^{(2)}\} \mid \{v_i, v_j\} \in E)\big\}$

  ...
- $E^{(q_1)} = \big\{\{v_i^{(q_1)}, v_j^{(q_1)}\} \mid \{v_i, v_j\} \in E)\big\}$

- $E_1 = \big\{\{u_0^1, u_1^1\}, \{u_0^1, u_2^1\}, ..., \{u_0^1, u_{q_2}^1\}\big\}$
- $E_2 = \big\{\{u_0^2, u_1^2\}, \{u_0^2, u_2^2\}, ..., \{u_0^2, u_{q_2}^2\}\big\}$

  ...
- $E_n = \big\{\{u_0^n, u_1^n\}, \{u_0^n, u_2^n\}, ..., \{u_0^n, u_{q_2}^n\}\big\}$
- $E' = \Big(\bigcup_{i=1}^{n} \big\{\{v_i', v_i^{(1)}\}, \{v_i', v_i^{(2)}\}, ..., \{v_i', v_i^{(q_1)}\}\big\}\Big) \bigcup \big\{\{v_1', u_0^1\}, \{v_2', u_0^2\}, ..., \{v_n', u_0^n\}\big\}$

Observe that each graph $(V^{(i)}, E^{(i)})$ is isomorphic to G which is to say that it is a "copy" of G (hence there are q_1 copies of G) and each graph (V_i, E_i) is a star of $q_2 + 1$ vertices with u_0^i being its center (see Figure 4.1). The integers q_1, q_2 are much larger than the length of input but bounded by a polynomial of the length of input, hence the reduction (from G to G_R) can be done in polynomial time.

Let $OPT_{(p,1)}(G)$ (resp. $OPT_{(p,2)}(G_R)$) denote the maximum number of vertices that a set of p vertices of G (resp. G_R) can 1-dominate (resp. 2-dominate). Using the notation we introduced, we state and prove the following lemma:

Lemma 4.3.2. *It holds*

(*i*) *If* $OPT_{(p,1)}(G) = \mathbf{max}$, *then* $OPT_{(p,2)}(G_R) \geq p(q_2 + 2) + q_1 \cdot \mathbf{max}$

(*ii*) *If* $OPT_{(p,1)}(G) \leq (1 - \frac{1}{e})\mathbf{max}$, *then*

$$OPT_{(p,2)}(G_R) \leq (1 - \frac{1}{e} + \frac{p(q_2+2)}{e(p(q_2+2) + q_1 \cdot \mathbf{max})})(p(q_2+2) + q_1 \cdot \mathbf{max})$$

Proof. (*i*) Assume that $OPT_{(p,1)}(G)$ is derived from the optimal set of vertices $\{v_{i_1}, ..., v_{i_p}\}$. The number of vertices of G_R that the set $\{v'_{i_1}, ..., v'_{i_p}\}$ 2-dominates, is $p(q_2 + 1) + p + \sum_{l=1}^{q_1} \big|\bigcup_{j=1}^{p} \mathcal{N}[v_{i_j}^{(l)}]\big| = p(q_2 + 2) + q_1 \cdot \mathbf{max}$, therefore $OPT_{(p,2)}(G_R) \geq p(q_2 + 2) + q_1 \cdot \mathbf{max}$.

(*ii*) First, we will show that $OPT_{(p,2)}(G_R)$ can be derived from an optimal set of vertices that is a subset of V' (observe that $p \leq |V'| = n$).
We have,

$$Coverage_2(u_j^i) \subseteq Coverage_2(u_0^i) \subseteq Coverage_2(v_i'), \text{ for each } i, j$$

Furthermore,

$$\big|Coverage_2(v_i') - Coverage_2(v_i^{(j)})\big| \geq q_2$$

, while

$$\left|Coverage_2(v_i^{(j)}) - Coverage_2(v_i')\right| < 2n, \text{ for each } i, j$$

By considering $q_2 \geq 2n$ we know that there must be an optimal set that is a subset of V' while the derived optimal solution is $OPT_{(p,2)}(G_R) = p(q_2+2) + q_1 \cdot OPT_{(p,1)}(G)$. Thus, if $OPT_{(p,1)}(G) \leq (1-\frac{1}{e})\mathbf{max}$ then

$$OPT_{(p,2)}(G_R) \leq (1-\frac{1}{e}+\frac{p(q_2+2)}{e(p(q_2+2)+q_1\cdot\mathbf{max})})(p(q_2+2)+q_1\cdot\mathbf{max})$$

□

Theorem 4.3.1. *There is no polynomial time approximation algorithm with any constant factor better than* $(1-\frac{1}{e})$ *for* p-Max 2-hop Domination *unless* $\mathbf{P} = \mathbf{NP}$.

Proof. In [MO11] it is implicitly stated that the $(1-\frac{1}{e}+\varepsilon)$-gap version of p-Max 1-hop Domination is computationally hard, for any $\varepsilon > 0$, which is shown by establishing a gap-preserving reduction from MCP (see section 4.3.2). If we take $\varepsilon \to 0^+$, then by establishing the reduction described in lemma 4.3.2 and taking q_1 large enough, it follows that the $(1-\frac{1}{e}+\varepsilon')$-gap version of p-Max 2-hop Domination is hard as well, for any $\varepsilon' > 0$. □

We assume now that we remove every edge $\{v_i', u_0^i\}$ and replace it with a path of $K-1$ edges which includes both v_i', u_0^i, hence in the new graph —we call it G_R^K— it will be $d(v_i', u_0^i) = K-1$, for each $i = 1, ..., n$.

Let $OPT_{(p,K)}(G_R^K)$ denote the maximum number of vertices that a set of p vertices of G_R^K can K-dominate.

Lemma 4.3.3. *It holds,*

(*i*) *If* $OPT_{(p,K-1)}(G) = \mathbf{max}$, *then* $OPT_{(p,K)}(G_R^K) \geq p(q_2+K)+q_1\cdot\mathbf{max}$

(*ii*) *If* $OPT_{(p,K-1)}(G) \leq (1-\frac{1}{e})\mathbf{max}$, *then*

$$OPT_{(p,K)}(G_R^K) \leq (1-\frac{1}{e}+\frac{p(q_2+K)+e(K-2)(n-p)}{e(p(q_2+K)+q_1\cdot\mathbf{max})})(p(q_2+K)+q_1\cdot\mathbf{max})$$

Proof. (*i*) Assume that $OPT_{(p,K-1)}(G)$ is derived from the optimal set of vertices $\{v_{i_1}, ..., v_{i_p}\}$. The number of vertices of G_R^K that the set $\{v'_{i_1}, ..., v'_{i_p}\}$ K-dominates, is $p(q_2+K)+q_1\cdot\mathbf{max}+\alpha(K-2)(n-p)$, $0 \leq \alpha \leq 1$, therefore $OPT_{(p,K)}(G_R^K) \geq p(q_2+K)+q_1\cdot\mathbf{max}$.

(*ii*) Let u be a vertex of any path from v'_i to the leaves of the star. It's easy to see that $Coverage_K(u) \subseteq Coverage_K(v'_i)$, for each i. Furthermore,

$$\left|Coverage_K(v'_i) - Coverage_K(v_i^{(j)})\right| \geq q_2$$

, while

$$\left|Coverage_K(v_i^{(j)}) - Coverage_K(v'_i)\right| < 2n, \text{ for each } i, j$$

By considering $q_2 \geq 2n$, we know that there must be an optimal set of p vertices of G_R^K which is a subset of V' and the derived optimal solution is $OPT_{(p,K)} = p(q_2 + K) + \alpha(K-2)(n-p) + q_1 \cdot M$, where $0 \leq \alpha \leq 1$ and M is the number of vertices of a graph $G^{(i)}$ that the optimal set K-dominates. If $OPT_{(p,K-1)}(G) \leq (1 - \frac{1}{e})\mathbf{max}$ then, $OPT_{(p,K)}(G_R^K) \leq p(q_2 + K) + (K-2)(n-p) + q_1(1 - \frac{1}{e})\mathbf{max} \Rightarrow$

$$OPT_{(p,K)}(G_R^K) \leq (1 - \frac{1}{e} + \frac{p(q_2+K) + e(K-2)(n-p)}{e(p(q_2+K) + q_1 \cdot \mathbf{max})})(p(q_2+K) + q_1 \cdot \mathbf{max})$$

□

Theorem 4.3.2. *There is no polynomial time approximation algorithm with any constant factor better than* $(1 - \frac{1}{e})$ *for* p-Max K-hop Domination *unless* $\mathbf{P} = \mathbf{NP}$, *for any constant integer* $K \geq 1$.

Proof. Let $\mathcal{R}_K$ be a polynomial time algorithm which on input G it outputs the graph G_R^K, for $K \geq 2$. Let $q_1^{(K)}, q_2^{(K)}$ be the corresponding integers. We have $|V(G_R^K)| = (q_1^{(K)} + q_2^{(K)} + K)n$, where $n = |V(G)|$. Now consider the following graph: $\mathcal{R}_j(\mathcal{R}_{j-1}(...\mathcal{R}_2(G)...))$, which we denote as $\mathcal{G}_j, 2 \leq j \leq K$. It easily follows that

$$|V(\mathcal{G}_j)| = (q_1^{(j)} + q_2^{(j)} + j)|V(\mathcal{G}_{j-1})|$$

Furthermore, we know that $q_1^{(j)}$ is a polynomial of $|V(\mathcal{G}_{j-1})|$, while $q_2^{(j)}$ is a linear polynomial of $|V(\mathcal{G}_{j-1})|$. Hence,

$$|V(\mathcal{G}_j)| = \Theta(|V(\mathcal{G}_{j-1})|^{c_{j-1}}), \text{ for some constant } c_{j-1}$$

Ultimately, there exists constant $c = \prod_{i=1}^{K-1} c_i$ (which does not depend on n) such that

$$|V(\mathcal{G}_K)| = \Theta(n^c)$$

Using lemma 4.3.3, a gap-preserving reduction can be established in polynomial time from p-Max 1-hop Domination to p-Max K-hop Domination for any fixed $K \geq 2$. Thus, the $(1 - \frac{1}{e} + \varepsilon)$-gap p-Max K-hop Domination problem is computationally hard and consequently no $(1 - \frac{1}{e} + \varepsilon)$-approximation polynomial time algorithm exists for p-Max K-hop Domination for any $\varepsilon > 0$, unless $\mathbf{P} = \mathbf{NP}$. □

Finally, regarding the problem p-Max k-hop Domination where the number of hops is not fixed (i.e. is part of the input), assume that there exists a polynomial time $(1 - \frac{1}{e} + \varepsilon)$-approximation algorithm for some $\varepsilon > 0$, then the same algorithm can be used to $(1 - \frac{1}{e} + \varepsilon)$-approximate each problem p-Max K-hop Domination for every fixed $K \geq 2$, in polynomial time.

Corollary 4.3.2.1. *There is no polynomial time approximation algorithm with any constant factor better than* $(1 - \frac{1}{e})$ *for* p-Max k-hop Domination *unless* $\mathbf{P} = \mathbf{NP}$.

Chapter 5

The problem of graph externalities

5.1 Introduction

An instance $\mathcal{I}$ of the problem of graph externalities OPT-EXT, (introduced in [Fot+20]) is a triplet (G, O, val), where $G = (V, E)$ is an undirected graph, O is a collection of $m = |O|$ valued objects and $val : O \to \mathbb{R}^+$ is a function that gives the valuation $val(o)$ of every object $o \in O$. Any allocation of *every* object to *discrete* vertices of the graph is considered to be a feasible solution for the problem. An allocation is described by a function $\pi : V(G) \to O \cup \{\perp\}$. Since the number of objects m might be less than the number n of vertices, we write $\pi(v) = \perp$, if under the allocation π no object is allocated to vertex v. In fact, for every instance of the graph, $m \leq n = |V|$ must hold.

Definition 5.1.1. *Under an allocation π, the externality of a vertex $v \in V$, denoted as $ext_\pi(v)$ is:*

- $ext_\pi(v) = 0$, *if* $\pi(v) = \perp$
- $ext_\pi(v) = val(\pi(u)) - val(\pi(v))$, $u := argmax_{w \in \mathcal{N}[v]:\pi(w) \neq \perp}\{val(w)\}$

The exact notion is that the externality of v, conditioned that $\pi(v) \neq \perp$, is derived from a neighbor of v, and v derives externality from the vertex u that has the object with the highest valuation among the objects that the vertices of the neighborhood of v have. If no such neighbor u exists it means that either no object has been allocated to any of the neighbors of v or v has zero degree or $val(\pi(v))) > val(\pi(w))$, for all $w \in \mathcal{N}(v)$ with $\pi(w) \neq \perp$. In all three aforementioned cases, the externality of v is 0.

The objective of OPT-EXT is to maximize the *graph externality*, which is to maximize the function

$$Ext_\pi(G) = \sum_{v \in V} ext_\pi(v)$$

, over all possible (feasible) allocations π.

5.2 The OPT-EXT problem with two valuations

If the valuation of each object is restricted to be either 0 or 1, then the problem of graph externalities is denoted as OPT-EXT(0,1). Note that OPT-EXT(0,1) is equivalent to every variant of the problem OPT-EXT where the valuation of each object is chosen between only two valuations, namely val_1, val_2. It can be shown that OPT-EXT(0,1) is computationally hard [Fot+20] using the **NP**-completeness of the Dominating Set problem (which recall that it is denoted as P_1' in section 4.2).

Let $G = (V, E)$ be a graph and O be a collection of m objects, with $m < n = |V(G)|$. We consider a new collection O' which extends the collection O by adding $n - m$ objects and a new function $val' : O' \to \{0, 1\}$ which extends val by assigning to each object $o \in O' \backslash O$ zero value. A helpful proposition, proved in their work, is the following:

Proposition 5.2.1 ([Fot+20])**.** *A ρ-approximation for (G, O', val') implies a ρ-approximation for (G, O, val). Formally, if for the instance (G, O', val') and for an allocation π' we have $Ext_{\pi'}(G) \geq \rho \cdot Ext_{\pi'^*}(G)$, then we can create an allocation π for the instance (G, O, val) such that $Ext_{\pi}(G) \geq \rho \cdot Ext_{\pi^*}(G)$, where π^*, π'^* are the corresponding optimal allocations for the two instances.*

To get the aforementioned allocation π from π' we remove $|O'| - |O|$ objects with zero value from the vertices, starting from those which derive zero externality. Note that it doesn't matter if we remove an object of O and keep an object of O'. Thus, it is made obvious that it suffices to only study those instances (G, O, val) for which the number $|O|$ of the objects is equal to the number $|V(G)|$ of the vertices.

5.2.1 Approximating OPT-EXT(0,1)

First we give the definition of the problem AUX as given in [Fot+20],

Definition 5.2.1. *Given a graph $G = (V, E)$ and two integers b, r satisfying $b + r \leq n$, separate the vertices in three sets B, R, W of blue, red and white vertices respectively so that:*

1. $|B| \leq b$
2. $|R| \leq r$
3. $\forall v \in R, \exists u \in B$ *s.t.* $u \in \mathcal{N}(v)$

and the value $|B| + |R|$ is maximized.

Observation 5.2.1. *If $b + r = n$, then we can create an instance $\mathcal{I} = (b, \mathcal{S})$ for* MCP*, where $\mathcal{S} = \{\mathcal{N}[u] \mid u \in V(G)\}$ and reduce* AUX *to* MCP*.*

An optimal collection $\mathcal{S}^*$ of at most b sets for $\mathcal{I}$, can be converted to an optimal coloring B^*, R^*, W^* of G for AUX *such that $f(\mathcal{S}^*) = |B^*| + |R^*|$. Furthermore, we have shown that since f is monotone, if $|S^*| < b$ then we can include random $b - |\mathcal{S}^*|$ unpicked sets and get an optimal set $\mathcal{S}^{*\prime}$ with $|\mathcal{S}^{*\prime}| = b$ and thus every optimal solution for* AUX *with less than b blue vertices can be transformed into an optimal solution for* AUX *with exactly b blue vertices.*

In fact, every optimal solution for AUX with less than b blue vertices can be transformed into an optimal solution with exactly b blue vertices for any $b + r$ $(\leq n)$ [Fot+20].

Let (G, O, val) be an instance of OPT-EXT(0,1) with $m = n$ and let $p = \{o \in O \mid val(o) = 1\}$. If we take $b = p$ and $r = n - p$, then every feasible solution *sol* for AUX it can be used to create an allocation π for OPT-EXT(0,1) and vice versa, such that the number of the red vertices derived from *sol* is equal to $Ext_\pi(G)$. Consider now the instance $\mathcal{I}$ of MCP as described in Observation 5.2.1. Observe that for the instance $\mathcal{I}$, the objective function f can be written as $f(A) = g(A) + |A|$, $A \subseteq V$. It's unavoidable to see the connection between the two pairs of problems, that of AUX and OPT-EXT(0,1) and that of the maximization problems with respective objective functions f and g. The connection is clear now and it is shown below.

$$\max_{A\subseteq V:|A|\leq p} f(A) \iff \max_{\substack{|B|\leq p,|R|\leq n-p \\ \forall v\in R,\exists u\in B:u\in\mathcal{N}(v)}} |B| + |R|$$

$$\downarrow$$

$$\max_{A\subseteq V:|A|=p} f(A) \iff \max_{\substack{|B|=p,|R|\leq n-p \\ \forall v\in R,\exists u\in B:u\in\mathcal{N}(v)}} |B| + |R|$$

$$\downarrow$$

$$\max_{A\subseteq V:|A|=p} g(A) \iff \max_{\text{feasible } \pi} Ext_\pi(G)$$

Figure 5.1: An illustration that shows how the problems (*i*) MCP (*ii*) AUX and (*iii*) OPT-EXT(0,1) are related.

Theorem 5.2.1. *There exists a $\frac{6e-5}{6e+5}$-approximation polynomial time algorithm for* OPT-EXT(0,1).

Proof. Let $sol(p), \widehat{sol}(p)$ be the solutions produced by *Aux* when applied on any graph G and its decomposed spanning forest $\widehat{G}$ (using decomposition

D2) respectively and OPT be the number of red vertices derived from an optimal coloring of G. Recall that for the case of the decomposed graph, ties are not broken arbitrarily but based on a priority rule which we describe in section 3.5, where algorithm *Aux* (Algorithm 4) is given too. The analysis we developed in the Generalization section (section 3.6) in combination with corollary 3.5.1.1, gives

$$\max\{|sol(p)| - p, |\widehat{sol}(p)| - p\} \geq \frac{6e-5}{6e+5} OPT$$

What remains is to prove that we can find a solution for AUX for the graph G for which the number of red vertices is at least $|\widehat{sol}(p)| - p$. Now consider the sets of blue and red vertices $\widehat{B}, \widehat{R}$ with respect to the solution $\widehat{sol}(p)$. We color in blue each vertex $v \in \widehat{B}$ of the graph G and then we randomly color in red the white vertices which are neighbors of blue vertices. Observe that since $\widehat{G}$ is a subgraph of G, the total number of the red vertices in G with respect to the coloring we've just described is at least $|\widehat{R}|$, which completes the proof. □

Recall that the graph $\widehat{G}$ is produced by applying Decomposition D2 on the spanning forest of G. If instead of applying Decomposition D2, we apply Decomposition D1, then it's not hard to see that the approximation ratio becomes $\frac{e-1}{e+1}$ which matches the approximation ratio obtained in [Fot+20]. Both of the proposed algorithms which achieve the approximation ratio $\frac{e-1}{e+1}$ (i) apply algorithm *Aux* on the initial graph G and (ii) compute a feasible solution for another instance which is an auxiliary graph, which in the case of [Fot+20] is formed by the edges of a maximum matching of G and their endpoints. The latter solution is transformed into a (feasible) solution for the initial instance (i.e. for G).

5.2.2 A *PTAS*-like scheme for a variant of OPT-EXT(0,1)

Definition 5.2.2 ([Vaz03]). *Let Π be an* **NP***-hard optimization problem, with objective function f_Π. We will say that algorithm $\mathcal{A}$ is an approximation scheme for Π if on input $(\mathcal{I}, \varepsilon)$, where $\mathcal{I}$ is an instance of Π and $\varepsilon > 0$ is an error parameter, it outputs a solution s such that:*

1. *$f_\Pi(\mathcal{I}, s) \leq (1+\varepsilon)OPT$, if Π is a minimization problem.*
2. *$f_\Pi(\mathcal{I}, s) \geq (1-\varepsilon)OPT$, if Π is a maximization problem*

Definition 5.2.3 ([Vaz03]). *Algorithm $\mathcal{A}$ will be said to be a polynomial time approximation scheme (PTAS) if for each fixed $\varepsilon > 0$, its running time is bounded by a polynomial in the size of instance $\mathcal{I}$.*

Definition 5.2.4 ([Vaz03]). *If the running time of algorithm $\mathcal{A}$ is bounded by a polynomial in the size of instance $\mathcal{I}$ and $\frac{1}{\varepsilon}$, then $\mathcal{A}$ will be said to be a fully polynomial time approximation scheme (FPTAS)*

To begin with, we consider an instance $\mathcal{I} = (p, \{S_1, ..., S_m\})$ of MCP. We introduce the notion of *frequency* which, for a specific element, is the number of sets that contain it. If F is a lower-bound on the frequencies of the elements i.e. any element appears in at least F sets, then we have the following theorem:

Theorem 5.2.2 ([SF13]). *The greedy algorithm for* MCP*, is a polynomial-time* $(1 - e^{-\frac{F \cdot p}{m}})$*-approximation algorithm with frequency lower-bounded by* F *on instances with* m *sets where we can pick up to* p *sets.*

Since the approximation ratio of $(1 - \frac{1}{e})$ still applies, we get the following corollary.

Corollary 5.2.2.1 ([SF13]). *The greedy algorithm for* MCP *gives approximation guarantee of* $(1 - e^{-\max\{\frac{F \cdot p}{m}, 1\}})$.

Now, let G be a graph which we see as an instance of p-Max 1-hop Domination and $\widehat{G} = \{\widehat{G}_1, ..., \widehat{G}_t\}$ be the decomposed graph, for the construction of which, we applied Decomposition D1 on a spanning forest of G. Recall that p-Max 1-hop Domination is a special case of MCP. We have already shown how to transform an instance of p-Max 1-hop Domination into an instance $\mathcal{I}$ of MCP, with ground set V and objective function f of the form $f(A) = g(A) + |A|, A \subseteq V$. Similarly, let $\widehat{\mathcal{I}}$ be an instance of MCP with respect to $\widehat{G}$, with objective function $\widehat{f}$ of the form $\widehat{f}(A) = \widehat{g}(A) + |A|, A \subseteq V$. Let again $(A_1, ..., A_p)$ (resp. $(\widehat{A}_1, ..., \widehat{A}_p)$) be the chain produced by the greedy algorithm when applied on $\mathcal{I}$ (resp. on $\widehat{\mathcal{I}}$) and A_p^* (resp. $\widehat{A}_p^*$) an optimal solution (of p elements) for $\mathcal{I}$ (resp. for $\widehat{\mathcal{I}}$). First we prove the following lemma.

Lemma 5.2.1. *It* G *is connected then*

$$\max\{g(A_p), \widehat{g}(\widehat{A}_p)\} \geq \frac{e}{e+1}(1 - e^{-\frac{F \cdot p}{m}})g(A_p^*)$$

Proof. We have

$$f(A_p) \geq (1 - e^{-\max\{\frac{F \cdot p}{m}, 1\}})f(A_p^*)$$

$$\Rightarrow g(A_p) + p \geq (1 - e^{-\max\{\frac{F \cdot p}{m}, 1\}})(g(A_p^*) + p)$$

$$\Rightarrow g(A_p) \geq (1 - e^{-\max\{\frac{F \cdot p}{m}, 1\}})g(A_p^*) - pe^{-\max(\frac{F \cdot p}{m}, 1)}$$

, from which we get

$$g(A_p) \geq (1 - e^{-\frac{F \cdot p}{m}})g(A_p^*) - \frac{p}{e}$$

Furthermore, recall that in section 3.4.1 we showed that

$$\widehat{g}(\widehat{A}_p) \geq p, p \leq t$$

$$\widehat{g}(\widehat{A}_p) = n - p, p \geq t$$

Finally we get,

$$\max\{g(A_p), \widehat{g}(\widehat{A}_p)\} \geq \frac{e}{e+1}(1 - e^{-\frac{F \cdot p}{m}})g(A_p^*)$$

□

The analysis we developed in the Generalization section (see section 3.6) in combination with lemma 5.2.1 gives the proof for the next lemma.

Lemma 5.2.2. *For any graph G, it holds*

$$\max\{g(A_p), g(\widehat{A}_p)\} \geq \frac{e}{e+1}(1 - e^{-\frac{F \cdot p}{m}})g(A_p^*)$$

Similar to the definition of α-MCP given in [SF13], we define the variant α-(OPT-EXT(0,1)) of OPT-EXT(0,1) as follows.

Definition 5.2.5 ([SF13])**.** *For each $\alpha, 0 < \alpha \leq 1$, let* α-MCP *be a variant of* MCP *for instances that satisfy the following conditions: If F is a lower bound on the frequencies of the elements and there are m sets, then $\frac{F}{m} \geq \alpha$.*

Definition 5.2.6. *For each $\alpha, 0 < \alpha \leq 1$, let* α-(OPT-EXT(0,1)) *be a variant of* OPT-EXT(0,1) *for instances (i.e. graphs G) that satisfy the following conditions: If $F - 1$ is a lower bound on the degrees of the vertices i.e. $F = \delta(G) + 1$, where $\delta(G) = \min_{u \in V(G)}\{deg(u)\}$, then $\frac{F}{|V(G)|} \geq \alpha$.*

In [SF13], it is shown that the variant α-MCP accepts a *PTAS*. We attempt to obtain a similar scheme for α-(OPT-EXT(0,1)) which is not a *PTAS*, but is close to the notion of a *PTAS* and this result is presented in the following theorem.

Theorem 5.2.3. *For each $\alpha, 0 < \alpha \leq 1$, there is a β-approximation polynomial time algorithm for α-(OPT-EXT(0,1)), for any fixed $\beta < \frac{e}{e+1}$.*

Proof. Using familiar notations, we can easily compute a solution for AUX[1] where the number of the red vertices is $g(A_p)$ (resp. $g(\widehat{A}_p)$) by coloring in blue each vertex $v \in V(G) \cap A_p$ (resp. $v \in V(G) \cap \widehat{A}_p$) and in red each white neighbor of the respective blue vertices. By taking the maximum between these two solutions and making use of lemma 5.2.2, an approximation ratio of $\frac{e}{e+1}(1 - e^{-\frac{F \cdot p}{m}})$ is guaranteed for α-(OPT-EXT(0,1)), where $F = \delta(G) + 1$ and $m = |V(G)|$.
Let β ($< \frac{e}{e+1}$) be the desired approximation ratio. If $p \geq -\frac{m}{F}\ln(1 - \beta(1 + \frac{1}{e}))$, then if we run the greedy algorithm on instances $\mathcal{I}$, $\widehat{\mathcal{I}}$ and then compute

[1]Note that we actually refer to the special case of AUX where $b + r = n$.

the respective solutions for AUX as previously described, we obtain approximation ratio β. Otherwise, observe that p is bounded by a constant, since $\frac{m}{F} \leq \frac{1}{\alpha}$. Furthermore, there are less than $m(m-1)(m-2)...(m-p+1) < m^p$ feasible solutions, which means that we can compute the optimal solution in polynomial time, by enumerating all feasible solutions. □

5.3 The OPT-EXT problem with general valuations

Regarding the OPT-EXT problem with general valuations (note that we do not assume now that $n = m$), it can be shown that in graphs with maximum degree 2 it is possible to solve the problem optimally in polynomial time and that for a family of graphs called *caterpillar trees*[2], there exists an 0.5-approximation polynomial time algorithm [Fot+20]. There are also experimental results provided for the performance of a general greedy algorithm for the problem of OPT-EXT. Nevertheless, there is still no proof provided (to the best of our knowledge), regarding the approximation ratio achieved by any algorithm that approximates the general problem OPT-EXT. For this reason, we provide a rather naturally considered algorithm and a proof for the approximation ratio it achieves.

5.3.1 Approximating OPT-EXT

Let $\mathcal{I} = (G, O, val)$ be an instance of OPT-EXT, where $val(o) \in \{q_1, ..., q_L\}$ for each $o \in O$, with $q_1 < ... < q_L$. Consider an optimal allocation π^* for $\mathcal{I}$. Under π^* we define $\alpha_{i,j}$ to denote the number of vertices $u \in V(G)$ for which it holds that $val(\pi^*(u)) = q_i$ and $ext_{\pi^*}(u) = q_j - q_i$. The total graph externality $Ext_{\pi^*}(G)$ can be written as

$$Ext_{\pi^*}(G) = \sum_{i<j} \alpha_{i,j}(q_j - q_i)$$

Let $X_{i,j} = \{u \in V(G) \mid val(\pi^*(u)) = q_i, ext_{\pi^*}(u) = q_j - q_i\}$, for each i, j. Now, consider the following procedure that produces a coloring of the vertices of the graph G with respect to some integer d:

1. Every vertex of G is white.
2. Color in blue every vertex u for which $val(\pi^*(u)) \in \{q_d, q_{d+1}, ..., q_L\}$.
3. Color in red every vertex $u \in \bigcup_{i,j:i<d\leq j} X_{i,j}$.

[2] A *caterpillar tree* consists of a central path —the *backbone*— and of the *pendant* vertices which have degree 1 neighboring only with vertices of the backbone [Fot+20; HS73].

The total number of red vertices (colored by the hypothetical coloring we just described) is $\sum_{i,j:i<d\leq j} |X_{i,j}| = \sum_{i,j:i<d\leq j} \alpha_{i,j}$. Now, if we consider an instance $\mathcal{I}_d = (G, O, val_d)$ for OPT-EXT(0,1) such that for the function $val_d : O \to \{0,1\}$ it holds that $val_d(o) = 1 \Leftrightarrow val(o) \geq q_d$ and $val_d(o) = 0 \Leftrightarrow val(o) < q_d$, for each $o \in O$, then for an optimal allocation π_d^* for $\mathcal{I}_d$ it holds

$$Ext_{\mathcal{I}_d,\pi_d^*}(G) \geq \sum_{i,j:i<d\leq j} \alpha_{i,j}$$

Let ALG be a ρ-approximation algorithm for OPT-EXT(0,1) and π_d an allocation produced by ALG when applied on instance $\mathcal{I}_d$. If under π_d every object is evaluated using val which is the function of $\mathcal{I}$, then we have,

$$Ext_{\mathcal{I},\pi_d}(G) \geq \rho \sum_{i,j:i<d\leq j} \alpha_{i,j} \cdot (q_d - q_{d-1})$$

The previous inequality is true because there are at least $\rho \sum_{i,j:i<d\leq j} \alpha_{i,j}$ vertices with objects of value at most q_{d-1} which derive externality of vertices with objects of value at least q_d. If we write $q_j - q_i = q_j - q_{j-1} + q_{j-1} - ... - q_{i+1} + q_{i+1} - q_i$, then $Ext_{\pi^*}(G)$ can be rewritten as

$$Ext_{\pi^*}(G) = \sum_{d=2}^{L} \beta_d \cdot (q_d - q_{d-1}), \beta_d = \sum_{i,j:i<d\leq j} \alpha_{i,j}$$

If we take $d^* = argmax_{d\in[2,L]}\{Ext_{\mathcal{I},\pi_d}(G)\}$, then

$$Ext_{\mathcal{I},\pi_{d^*}}(G) \geq \frac{\rho}{L-1} Ext_{\pi^*}(G)$$

The preceded analysis gives in fact the proof of the following theorem.

Theorem 5.3.1. *There exists a $\frac{\rho}{L-1}$-approximation polynomial time algorithm for* OPT-EXT*, where L is the number of the different valuations of the objects and it is assumed that a ρ-approximation polynomial time algorithm exists for* OPT-EXT(0,1)*.*

Chapter 6

Conclusions and future work

- We have shown that it is hard to $(1 - \frac{1}{e} + \varepsilon)$-approximate p-Max K-hop Domination for any fixed $K \geq 1$ and for any $\varepsilon > 0$. We once again reduce this problem to MCP and let $f = g + c$ be the objective corresponding function. Regarding the best possible constant approximation ratio for the problem of maximizing g subject to the cardinality contraints we considered when defining the problem (and recall that subject to which, any set that is optimal for f is optimal for g and vice versa) it easily follows that it's upper-bounded by $(1 - \frac{1}{e})$. We provide an algorithm that approximates the optimal value of g within a ratio of $\frac{e-1}{e+\frac{1}{k+1}}$, for $k \geq 2$, which approaches $(1 - \frac{1}{e})$ as $k \to \infty$. In other words with k getting larger, our algorithm seems to be optimal in the sense of approximability. Nevertheless, any improvements depend significantly on the decomposition of the graph G. One question is whether a different decomposition can lead to better approximation ratio guarantees and another question is whether a better (preferably tight) analysis on Decomposition D2, can be developed for any $k \geq 1$.

- We designed an algorithm that approximates OPT-EXT(0,1) within a ratio of $\frac{6e-5}{6e+5}$, which improves the approximation ratio $\frac{e-1}{e+1}$ achieved by the first algorithm proposed for this problem. We also provided the first proof for an approximation ratio achieved by a rather naturally considered algorithm for the problem OPT-EXT with general valuations. Note that if we upper-bound the number L of different valuations of the objects, then our algorithm is constant-approximation polynomial time for this variant.

- Each p-Max K-hop Domination problem is closely related to a dominating set problem which is typically a minimization problem;in our case the K-hop Dominating set problem. There are many variations of dominating sets in the literature (see, e.g., [Pag+02]). The question is, with respect to a dominating set problem if one can define related maximization problems and attempt to approximate them using tech-

niques similar to the ones we introduced (or cited) in the current work, new ones or other techniques which have not come to our attention.

- As a matter of fact, there are more variants of MCP considered in [SF13] and very interesting results regarding them, which can be used to obtain similar results for analogously defined variants of OPT-EXT(0,1) (as for example the variant α-(OPT-EXT(0,1)). Another interested part is the close connection (highlighted by the authors of [SF13]) of MCP (and its variants) to what it is known as Chamberlin-Courant's [CC83] voting rule and the connection of the latter with resource allocation problems [SFS12] (among others). Under Chamberlin-Courant's voting rule (in the approval-based variant), n voters need to elect p candidates[1] out of a group of m candidates as their representatives, such that the total number of voters who do not approve of any of the p candidates is minimized.

- Finally, the problem of maximizing functions that can be expressed as the difference of a γ-weakly submodular function and a modular function (possibly subject to constraints other than cardinality contraints) seems to have a very promising theoretical and practical consequence and there might be many objective functions of this type, for which it is yet to be shown that can be expressed analogously, which could lead to a new, better understanding of the problem and ultimately to new optimization techniques.

[1] We keep our notation, that is p denotes the number of candidates to elect which in [SF13] it is denoted by K.

Bibliography

[AB09] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.

[AS99] Alexander A. Ageev and Maxim Sviridenko. "An 0.828-approximation Algorithm for the Uncapacitated Facility Location Problem." In: *Discret. Appl. Math.* 93.2-3 (1999), pp. 149–156. DOI: 10.1016/S0166-218X(99)00103-1. URL: https://doi.org/10.1016/S0166-218X(99)00103-1.

[BBM08] Maria-Florina Balcan, Avrim Blum, and Yishay Mansour. "Item pricing for revenue maximization." In: *Proceedings of the 9th ACM Conference on Electronic Commerce*. 2008, pp. 50–59.

[BFS17] Niv Buchbinder, Moran Feldman, and Roy Schwartz. "Comparing apples and oranges: Query trade-off in submodular maximization." In: *Mathematics of Operations Research* 42.2 (2017), pp. 308–329.

[Bia+17] Andrew An Bian, Joachim M. Buhmann, Andreas Krause, and Sebastian Tschiatschek. "Guarantees for Greedy Maximization of Non-Submodular Functions with Applications." In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. ICML'17. Sydney, NSW, Australia: JMLR.org, 2017, pp. 498–507.

[Blu90] Norbert Blum. "A new approach to maximum matching in general graphs." In: *International Colloquium on Automata, Languages, and Programming*. Springer. 1990, pp. 586–597.

[BM14] Partha Basuchowdhuri and Subhashis Majumder. "Finding Influential Nodes in Social Networks Using Minimum k-Hop Dominating Set." In: *Applied Algorithms*. Ed. by Prosenjit Gupta and Christos Zaroliagis. Cham: Springer International Publishing, 2014, pp. 137–151. ISBN: 978-3-319-04126-1.

[BMN14] Walid Ben-Ameur, Ali Ridha Mahjoub, and José Neto. "The maximum cut problem." In: *Paradigms of Combinatorial Optimization: Problems and New Approaches* (2014), pp. 131–172.

[Buc+14] Niv Buchbinder, Moran Feldman, Joseph Naor, and Roy Schwartz. "Submodular maximization with cardinality constraints." In: *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*. SIAM. 2014, pp. 1433–1452.

[Cal+11] Gruia Calinescu, Chandra Chekuri, Martin Pal, and Jan Vondrák. "Maximizing a monotone submodular function subject to a matroid constraint." In: *SIAM Journal on Computing* 40.6 (2011), pp. 1740–1766.

[CC83] John R Chamberlin and Paul N Courant. "Representative deliberations and representative decisions: Proportional representation and the Borda rule." In: *American Political Science Review* 77.3 (1983), pp. 718–733.

[CFK18] Lin Chen, Moran Feldman, and Amin Karbasi. "Weakly submodular maximization beyond cardinality constraints: Does randomization help greedy?" In: *International Conference on Machine Learning*. PMLR. 2018, pp. 804–813.

[CFN77a] Gerard Cornuejols, Marshall Fisher, and George L Nemhauser. "On the uncapacitated location problem." In: *Annals of Discrete Mathematics*. Vol. 1. Elsevier, 1977, pp. 163–177.

[CFN77b] Gerard Cornuejols, Marshall L Fisher, and George L Nemhauser. "Exceptional paper—location of bank accounts to optimize float: An analytic study of exact and approximate algorithms." In: *Management science* 23.8 (1977), pp. 789–810.

[Cha00] Bernard Chazelle. "A minimum spanning tree algorithm with inverse-Ackermann type complexity." In: *Journal of the ACM (JACM)* 47.6 (2000), pp. 1028–1047.

[DK11] Abhimanyu Das and David Kempe. "Submodular meets Spectral: Greedy Algorithms for Subset Selection, Sparse Approximation and Dictionary Selection." In: *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*. Ed. by Lise Getoor and Tobias Scheffer. Omnipress, 2011, pp. 1057–1064. URL: https://icml.cc/2011/papers/542%5C_icmlpaper.pdf.

[DK18] Abhimanyu Das and David Kempe. "Approximate Submodularity and its Applications: Subset Selection, Sparse Approximation and Dictionary Selection." In: *Journal of Machine Learning Research* 19.3 (2018), pp. 1–34. URL: http://jmlr.org/papers/v19/16-534.html.

[DRS09] Shaddin Dughmi, Tim Roughgarden, and Mukund Sundararajan. "Revenue submodularity." In: *Proceedings of the 10th ACM conference on Electronic commerce*. 2009, pp. 243–252.

[Ele+17] Ethan Elenberg, Alexandros G Dimakis, Moran Feldman, and Amin Karbasi. "Streaming Weak Submodularity: Interpreting Neural Networks on the Fly." In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/c182f930a06317057d31c73bb2fedd4f-Paper.pdf.

[Ele+18] Ethan R Elenberg, Rajiv Khanna, Alexandros G Dimakis, and Sahand Negahban. "Restricted strong convexity implies weak submodularity." In: *The Annals of Statistics* 46.6B (2018), pp. 3539–3568.

[Fei98] Uriel Feige. "A Threshold of Ln n for Approximating Set Cover." In: *J. ACM* 45.4 (July 1998), pp. 634–652. ISSN: 0004-5411. DOI: 10.1145/285055.285059. URL: https://doi.org/10.1145/285055.285059.

[FG95] Uriel Feige and Michel Goemans. "Approximating the value of two power proof systems, with applications to max 2sat and max dicut." In: *Proceedings Third Israel Symposium on the Theory of Computing and Systems*. IEEE. 1995, pp. 182–189.

[FMV11] Uriel Feige, Vahab S Mirrokni, and Jan Vondrák. "Maximizing non-monotone submodular functions." In: *SIAM Journal on Computing* 40.4 (2011), pp. 1133–1153.

[Fot+20] Dimitris Fotakis, Laurent Gourvès, Stelios Kasouridis, and Aris Pagourtzis. "Object Allocation and Positive Graph Externalities." In: *ECAI 2020*. IOS Press, 2020, pp. 107–114.

[FW90] Michael L. Fredman and Dan E. Willard. "Trans-dichotomous algorithms for minimum spanning trees and shortest paths." In: *Proceedings [1990] 31st Annual Symposium on Foundations of Computer Science* (1990), 719–725 vol.2.

[Gab76] Harold N Gabow. "An efficient implementation of Edmonds' algorithm for maximum matching on graphs." In: *Journal of the ACM (JACM)* 23.2 (1976), pp. 221–234.

[GH85] Ronald L Graham and Pavol Hell. "On the history of the minimum spanning tree problem." In: *Annals of the History of Computing* 7.1 (1985), pp. 43–57.

[GW95] Michel X Goemans and David P Williamson. "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming." In: *Journal of the ACM (JACM)* 42.6 (1995), pp. 1115–1145.

[Har+19] Chris Harshaw, Moran Feldman, Justin Ward, and Amin Karbasi. "Submodular maximization beyond non-negativity: Guarantees, fast algorithms, and applications." In: *International Conference on Machine Learning*. PMLR. 2019, pp. 2634–2643.

[HM90] Ward Hanson and R Kipp Martin. "Optimal bundle pricing." In: *Management Science* 36.2 (1990), pp. 155–174.

[HMS08] Jason Hartline, Vahab Mirrokni, and Mukund Sundararajan. "Optimal marketing strategies over social networks." In: *Proceedings of the 17th international conference on World Wide Web*. 2008, pp. 189–198.

[Hoc96] Dorit S Hochbaum. "Approximating covering and packing problems: set cover, vertex cover, independent set, and related problems." In: *Approximation algorithms for NP-hard problems*. 1996, pp. 94–143.

[HP98] Dorit S Hochbaum and Anu Pathria. "Analysis of the greedy approach in problems of maximum k-coverage." In: *Naval Research Logistics (NRL)* 45.6 (1998), pp. 615–627.

[HS73] Frank Harary and Allen J. Schwenk. "The number of caterpillars." In: *Discrete Mathematics* 6.4 (1973), pp. 359–365. ISSN: 0012-365X. DOI: https://doi.org/10.1016/0012-365X(73)90067-8. URL: https://www.sciencedirect.com/science/article/pii/0012365X73900678.

[IFF01] Satoru Iwata, Lisa Fleischer, and Satoru Fujishige. "A combinatorial strongly polynomial algorithm for minimizing submodular functions." In: *Journal of the ACM (JACM)* 48.4 (2001), pp. 761–777.

[Iwa08] Satoru Iwata. "Submodular function minimization." In: *Mathematical Programming* 112.1 (2008), pp. 45–64.

[KG05] Andreas Krause and Carlos Guestrin. *Optimal nonmyopic value of information in graphical models: efficient algorithms and theoretical limits*. Carnegie Mellon University. Center for Automated Learning and Discovery, 2005.

[KG14] Andreas Krause and Daniel Golovin. "Submodular function maximization." In: *Tractability* 3 (2014), pp. 71–104.

[KKT03] David Kempe, Jon Kleinberg, and Éva Tardos. "Maximizing the spread of influence through a social network." In: *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2003, pp. 137–146.

[Kra+08] Andreas Krause, Jure Leskovec, Carlos Guestrin, Jeanne VanBriesen, and Christos Faloutsos. "Efficient sensor placement optimization for securing large water distribution networks." In: *Journal of Water Resources Planning and Management* 134.6 (2008), pp. 516–526.

[KS96] David R Karger and Clifford Stein. "A new approach to the minimum cut problem." In: *Journal of the ACM (JACM)* 43.4 (1996), pp. 601–640.

[KSG08] Andreas Krause, Ajit Singh, and Carlos Guestrin. "Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies." In: *Journal of Machine Learning Research* 9.2 (2008).

[KST13] Ariel Kulik, Hadas Shachnai, and Tami Tamir. "Approximations for monotone and nonmonotone submodular maximization with knapsack constraints." In: *Mathematics of Operations Research* 38.4 (2013), pp. 729–739.

[LB11] Hui Lin and Jeff Bilmes. "A class of submodular functions for document summarization." In: *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*. 2011, pp. 510–520.

[Lee+09] Jon Lee, Vahab S Mirrokni, Viswanath Nagarajan, and Maxim Sviridenko. "Non-monotone submodular maximization under matroid and knapsack constraints." In: *Proceedings of the forty-first annual ACM symposium on Theory of computing*. 2009, pp. 323–332.

[Les+07] Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne M. VanBriesen, and Natalie S. Glance. "Cost-effective outbreak detection in networks." In: *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Jose, California, USA, August 12-15, 2007*. Ed. by Pavel Berkhin, Rich Caruana, and Xindong Wu. ACM, 2007, pp. 420–429. DOI: 10.1145/1281192.1281239. URL: https://doi.org/10.1145/1281192.1281239.

[LLN06] Benny Lehmann, Daniel Lehmann, and Noam Nisan. "Combinatorial auctions with decreasing marginal utilities." In: *Games and Economic Behavior* 55.2 (2006), pp. 270–296.

[MO11] Eiji Miyano and Hirotaka Ono. "Maximum domination problem." In: *Proceedings of the Seventeenth Computing on The Australasian Theory Symposium-Volume 119*. 2011, pp. 55–62.

[MV80] Silvio Micali and Vijay V Vazirani. "An O (v| v| c| E|) algoithm for finding maximum matching in general graphs." In: *21st Annual Symposium on Foundations of Computer Science (sfcs 1980)*. IEEE. 1980, pp. 17–27.

[NMN01] Jaroslav Nešetřil, Eva Milková, and Helena Nešetřilová. "Otakar Borůvka on minimum spanning tree problem Translation of both the 1926 papers, comments, history." In: *Discrete Mathematics* 233.1 (2001). Czech and Slovak 2, pp. 3–36. ISSN: 0012-365X. DOI: https://doi.org/10.1016/S0012-365X(00)00224-7. URL: https://www.sciencedirect.com/science/article/pii/S0012365X00002247.

[NWF78] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. "An analysis of approximations for maximizing submodular set functions—I." In: *Mathematical programming* 14.1 (1978), pp. 265–294.

[Pag+02] Aris Pagourtzis, Paolo Penna, Konrad Schlude, Kathleen Steinhöfel, David Scot Taylor, and Peter Widmayer. "Server Placements, Roman Domination and Other Dominating Set Variants." In: *Foundations of Information Technology in the Era of Network and Mobile Computing: IFIP 17th World Computer Congress — TC1 Stream / 2nd IFIP International Conference on Theoretical Computer Science (TCS 2002) August 25–30, 2002, Montréal, Québec, Canada*. Ed. by Ricardo Baeza-Yates, Ugo Montanari, and Nicola Santoro. Boston, MA: Springer US, 2002, pp. 280–291. ISBN: 978-0-387-35608-2. DOI: 10.1007/978-0-387-35608-2_24. URL: https://doi.org/10.1007/978-0-387-35608-2_24.

[PR02] Seth Pettie and Vijaya Ramachandran. "An optimal minimum spanning tree algorithm." In: *Journal of the ACM (JACM)* 49.1 (2002), pp. 16–34.

[Sch00] Alexander Schrijver. "A combinatorial algorithm minimizing submodular functions in strongly polynomial time." In: *Journal of Combinatorial Theory, Series B* 80.2 (2000), pp. 346–355.

[Sch91] Alejandro A Schäffer. "Simple local search problems that are hard to solve." In: *SIAM journal on Computing* 20.1 (1991), pp. 56–87.

[SF13] Piotr Skowron and Piotr Faliszewski. "Approximating the Max-Cover Problem with Bounded Frequencies in FPT Time." In: *CoRR* abs/1309.4405 (2013). arXiv: 1309.4405. URL: http://arxiv.org/abs/1309.4405.

[SFS12] Piotr Skowron, Piotr Faliszewski, and Arkadii Slinko. "Fully proportional representation as resource allocation: Approximability results." In: *arXiv preprint arXiv:1208.1661* (2012).

[SK10] Peter Stobbe and Andreas Krause. "Efficient minimization of decomposable submodular functions." In: *Advances in Neural Information Processing Systems* 23 (2010).

[SVW17] Maxim Sviridenko, Jan Vondrák, and Justin Ward. "Optimal approximation for submodular and supermodular optimization with bounded curvature." In: *Mathematics of Operations Research* 42.4 (2017), pp. 1197–1218.

[SY20] Richard Santiago and Yuichi Yoshida. "Weakly Submodular Function Maximization Using Local Submodularity Ratio." In: *CoRR* abs/2004.14650 (2020). arXiv: 2004.14650. URL: https://arxiv.org/abs/2004.14650.

[Top98] Donald M Topkis. *Supermodularity and complementarity*. Princeton university press, 1998.

[Vaz03] Vijay V. Vazirani. "Hardness of Approximation." In: *Approximation Algorithms*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 306–333. ISBN: 978-3-662-04565-7. DOI: 10.1007/978-3-662-04565-7_29. URL: https://doi.org/10.1007/978-3-662-04565-7_29.

[VBK20] Nate Veldt, Austin R Benson, and Jon Kleinberg. "Hypergraph cuts with general splitting functions." In: *arXiv preprint arXiv:2001.02817* (2020).