Approximation Algorithms for Independence Systems

Theophile François THIERY

August 21, 2023
Version: Double Spacing Submission

Queen Mary University of London

School of Mathematical Sciences

Combinatorics Group

Approximation Algorithms for Independence Systems

Theophile François THIERY

1. Reviewer Dr. Viresh Patel
School of Mathematical Sciences
Queen Mary University of London

2. Reviewer Dr. Piotr Krysta
Department of Computer Science
University of Liverpool

Supervisors Dr. Justin Ward and Dr. Mark Jerrum

August 21, 2023

Theophile François THIERY
Approximation Algorithms for Independence Systems
, August 21, 2023
Reviewers: Dr. Viresh Patel and Dr. Piotr Krysta
Supervisors: Dr. Justin Ward and Dr. Mark Jerrum

Queen Mary University of London
Combinatorics Group

School of Mathematical Sciences
Mile End Road
E1 4NS and London

Declaration

I, Theophile F. Thiery, confirm that the research included within this thesis is my own work or that where it has been carried out in collaboration with, or supported by others, that this is duly acknowledged below and my contribution indicated. Previously published material is also acknowledged below. I attest that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge break any UK law, infringe any third party's copyright or other Intellectual Property Right, or contain any confidential material.

I accept that the College has the right to use plagiarism detection software to check the electronic version of the thesis. I confirm that this thesis has not been previously submitted for the award of a degree by this or any other university. The copyright of this thesis rests with the author and no quotation from it or information derived from it may be published without the prior written consent of the author.

Details of collaboration and publications: The work in Chapter 2 and 5 was conducted in collaboration with Dr. Justin Ward. The work in Chapter 4 was conducted in collaboration with Dr. Chien-Chung Huang and Dr. Justin Ward. The results were published as follows:

- Chapter 2 is based on [TW23]: "An Improved Approximation for Maximum Weighted k-Set Packing", co-authored with Justin Ward. It has appeared in *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023*.
- Chapter 4 is based on [HTW20]: "Improved Multi-Pass Streaming Algorithms for Submodular Maximization with Matroid Constraints", co-authored with Chien-Chung Huang and Justin Ward. It has appeared in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2020, August 17-19, 2020, Virtual Conference*.
- Chapter 5 is based on [TW22]: "Two-Sided Weak Submodularity for Matroid Constrained Optimization and Regression", co-authored with Justin Ward. It has appeared in *Conference on Learning Theory, COLT, 2-5 July 2022, London, UK*.

London, August 21, 2023

Theophile François THIERY

Abstract

In this thesis, we study three maximization problems over independence systems.

- Chapter 2 – Weighted k-Set Packing is a fundamental combinatorial optimization problem that captures matching problems in graphs and hypergraphs. For over 20 years Berman's algorithm stood as the state-of-the-art approximation algorithm for this problem, until Neuwohner's recent improvements. Our focus is on the value $k = 3$ which is well motivated from theory and practice, and for which improvements are arguably the hardest. We largely improve upon her approximation, by giving an algorithm that yields state-of-the-art results. Our techniques are simple and naturally expand upon Berman's analysis. Our analysis holds for any value of k with greater improvements over Berman's result as k grows.
- Chapter 3 – We continue the study of the weighted k-set packing problem. Building on Chapter 2, we reach the tightest approximation factor possible for $k = 3$, and $k \geq 7$ using our techniques. As a consequence, we improve over all the results in Chapter 2. In particular, we obtain $\sqrt{3}$, and $\frac{k}{2}$-approximation for $k = 3$ and $k \geq 7$ respectively. Our result for $k \geq 7$ is in fact analogous to that of Hurkens and Schrijver who obtained the same approximation factor for the unweighted problem.
- Chapter 4 – We present improved multipass streaming algorithms for maximizing monotone and arbitrary submodular functions over independence systems. Our result demonstrates that the simple local-search algorithm for maximizing a monotone submodular function can be efficiently simulated using a few passes over the dataset. Our results improve the number of passes needed compared to the state-of-the-art.
- Chapter 5 – We conclude the thesis by presenting improved approximation algorithms for Sparse Least-Square Estimation, Bayesian A-optimal Design, and Column Subset Selection over a matroid constraint. At the heart of this chapter is the demonstration of a new property that considered applications satisfy. We call it: *β-weak submodularity*. We leverage this property to derive new algorithms with strengthened guarantees. The notion of β-weak submodularity is of independent interest and we believe that it will have further use in machine learning and statistics.

Acknowledgement

First and foremost, I would like to thank Justin for making my Ph.D. journey so enjoyable. It was! You have been a great advisor from a personal and professional standpoint, providing constant support, great laughs, and guidance. Working with you was an immense pleasure. You always took the time to sit through our meetings and listen to strange ideas, complaints, and more. Above else, you trusted me. Thank you!

I thank my thesis examiners, Piotr and Viresh, for their time and for providing useful suggestions and observations. I also thank Mark and Felix for agreeing to be my annual review supervisors, and for always keeping their door open.

During my time at QMUL, I spent a month abroad in Bremen. I would like to thank Nicole for her guidance, sound advice, and for hosting me. Although things didn't go as planned, I hope we can continue to work together. Thanks to Alex, Jens, Felix, and Mohit for taking the time to integrate me into their group.

I would like to thank my companion and friend in this Ph.D. adventure, Louis. It would definitely not have been the same without you. I have really enjoyed our time together. I also thank the *Argag* group, Asier, David, Gerardo, Konrad, for our games of Hanabi, our food adventures, and for being yourselves. I will miss spending time with you more than London. Finally, I thank the Maths' Ph.D. cohort for providing camaraderie and empathy.[1]

Je voudrais remercier mes parents pour m'avoir toujours laissé la liberté d'entreprendre ce qu'il me plaisait et pour être là même dans les moments plus difficiles. Je remercie aussi mes frères, Gaspard et Ulysse, dont je suis extrêmement fier, de tolérer mes visites de dernières minutes à Paris. Je vous aime. Un grand merci à mes amis de la *Ludique en ligne* pour ces moments partagés autour d'un écran. J'espère qu'on trouvera le temps de se réunir. Merci à Cédric, Moritz, et Maurice. Je suis heureux d'être votre ami. Merci à mes amis de toujours, Vincent et Shao. Même si nos chemins sont bien différents, je n'en serai pas là sans vous.

Par dessus tout, je veux remercier Diem-Ha pour avoir la chance d'être à tes côtés, pour être compréhensive, pour apporter de la stabilité, de la légèreté et me rappeler qu'il y a un moment pour tout.

[1] *Tulipe* by Sophie Guerrive. Permission of the author was given.

Contents

1 General Introduction

In this thesis, we consider some classical combinatorial optimization problems, which we will refer to as *maximization problems over independence systems*. We are given a set function $f : 2^X \to \mathbb{R}_{\geq 0}$ over a ground set of elements and aim to find a solution S of maximum value subject to some combinatorial constraint $\mathcal{I}$. More precisely, we solve the following abstract problem

$$\arg\max f(S) \text{ such that } S \in \mathcal{I}, \tag{1.1}$$

The problems studied in Chapter 2 to Chapter 5 can be cast as (1.1), as detailed next.

1.1 Weighted p-Set Packing

The first and arguably the most important contribution of this thesis regards the *weighted p-Set Packing* problem.

The p-set packing problem is commonly used as a generalization of matching problems in graphs. Matching problems are a fundamental area of research in graph theory and combinatorial optimization, with numerous applications in computer science, and operations research. Due to the wide range of practical applications that they encompass, they play a crucial role in the design and analysis of algorithms. For instance, AdWords, an online advertising platform developed by Google, uses matching algorithms to match ads with relevant search queries. More generally, they have been used in the development of recommendation systems, online dating platforms, and social networks.

In the standard *graph matching* problem, we are given a graph $G = (V, E)$, where V denotes the *vertex set* and E is the *edge set*. The set $E \subseteq \{(u, v) \colon u, v \in V \text{ and } u \neq v\}$ contains unordered pairs of vertices. Two vertices that share an edge are *adjacent* or *neighbors*. As an example, one can think of G as a social network, where V is the set of users with an edge between two users if they know each other. The objective is to find the most *valuable* subset of edges $M \subseteq E$ that do not share any common endpoints. Such a subset of edges is called a *matching*. The value of a matching is measured using a *set function* which is a discrete function $f : 2^E \to \mathbb{R}_{\geq 0}$ that for any subset of edges $S \subseteq E$ outputs its value $f(S)$. The most natural class of set functions that graph matching problems consider are *weighted functions*, also called *linear* or *modular functions*.

Definition 1.1.1 (Weight Function)**.** A set function $f : 2^E \to \mathbb{R}_{\geq 0}$ is a *weighted function* if for any set $S \subseteq E$ the value of S is equal to the sum of the values of the elements contained in that set. Thus, it satisfies

$$f(S) \triangleq \sum_{e \in S} f(e).$$

If all the elements have the same weight we say that the function is *unweighted*.

The p-set packing problem is an extremely simple generalization of Graph Matching, in which each edge now contain up to p distinct vertices. More formally, we are given a *p-hypergraph* $H = (V, E)$ with vertex set V, and hyperedge set E, such that each hyperedge $e \in E$ contains at most p vertices. A p-hypergraph is *uniform* if each hyperedge has precisely p vertices. The weighted p-set packing problem should now be clear.

Definition 1.1.2 (Weighted p-Set Packing)**.** Given p-hypergraph $H = (V, E)$ and a weighted set function $f\colon 2^E \to \mathbb{R}_{\geq 0}$, the goal is to find a maximum weight sub-collection $M \subseteq E$ such that the hyperedges in M are pairwise non-intersecting.

Thus, the graph matching problem is in fact a special case of the 2-set packing problem in which the underlying graph is uniform. Another important special case of the above problem is the *3-dimensional matching* problem. It belongs to Karp's list of *21 NP-complete Problems* [Kar72], which became a historically significant list of problems for evaluating the performance of algorithms and approximation techniques. 3-Dimensional Matching is a special instance of 3-Set Packing in which the hypergraph is *3-partite*. A *p-partite* hypergraph is a p-uniform hypergraph in which the vertices can be partitioned into p disjoint parts and each hyperedge contains exactly one vertex from each part. A *bipartite graph* is a 2-partite hypergraph.

A problem of immediate generality is the weighted independent set in $(p+1)$-claw free graph. Given a graph $G = (V, E)$, an *independent set* is a subset of vertices $U \subseteq V$ such that no two vertices in U are adjacent in G. A *d-claw* is a graph that consists of a central vertex v called the *center* and d adjacent vertices to v that form an independent set. The d neighbors of v are called the *talons*. A $(p+1)$-claw free graph is a graph that doesn't contain a $(p+1)$-claw as an *induced* subgraph. Given a subset of vertices $S \subseteq V$ of G, the induced subgraph on S, which we denote $G[S]$, is the graph whose vertex set is S and the edge set retains **all** edges from G between those vertices. Given a p-hypergraph $H = (V, E)$, we can construct a $(p+1)$-claw free graph G with a correspondence between the independent sets in G and the matchings in H. Indeed, by contracting each hyperedge to a single node and drawing an edge between two vertices if the corresponding hyperedges intersect, we obtain a $(p+1)$-claw free graph. It is easy to verify that G is $(p+1)$-claw free, and that finding the maximum weight matching in H corresponds to a maximum weight independent set in G. The construction is shown in Figure 1.1

As a consequence of the widely believed computational complexity assumption that P $\neq$ NP, there is no exact algorithm to solve the p-set packing problem that runs in polynomial time in

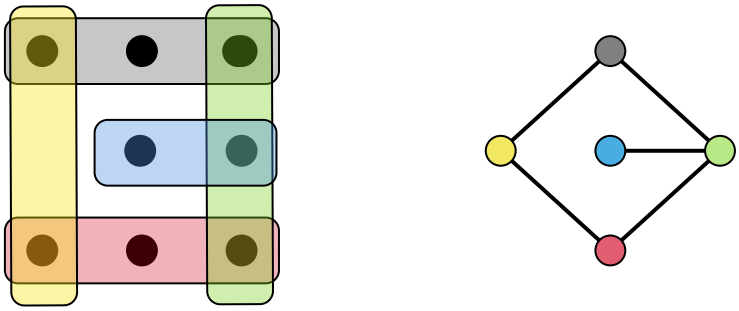


Fig. 1.1: Turning a 3-hypergraph (left) into a 4-claw free graph (right). Observe that the green, grey, blue, and red vertices form a 3-claw with the green vertex as the center.

the size of the input for $p \geq 3$[HSS06]. This motivates the design of approximation algorithms that obtain approximate solutions close to the optimal one in polynomial time.

Definition 1.1.3 (Informal)**.** Consider a maximization problem Π as in (1.1). An algorithm ALG for Π is an *α-approximation algorithm* for $\alpha \geq 1$ if for any input instance $I \in \Pi$, Algorithm ALG runs in *polynomial time* in the instance size and

$$\text{ALG}(I) \leq \text{OPT}(I) \leq \alpha \cdot \text{ALG}(I),$$

where $\text{OPT}(I), \text{ALG}(I)$ is the value of the problem's objective function evaluated on the optimal solution and the algorithm solution to input I, respectively. We say that α is the *approximation factor* of algorithm ALG.

An algorithm with $\alpha = 1$ is an exact approximation algorithm, and thus always finds an optimal solution. Among the problems that do not admit an exact approximation algorithm, some admit approximation algorithms up to any desirable accuracy degree. In short, there are problems, such that for any $\varepsilon > 0$, there exists a $(1+\varepsilon)$-approximation algorithm. Such algorithms are known as *polynomial time approximation schemes* (PTAS). On the other hand, there are problems for which there is a value $\bar{\alpha}$ such that no α-approximation algorithm can exist for $\alpha \leq \bar{\alpha}$, assuming that P $\neq$ NP. We refer to the class APX as problems for which an α-approximation algorithm exists, where α is a constant. The classes satisfy PTAS $\subset$ APX, and if P $\neq$ NP then the inclusion is strict.

Unweighted p-Set Packing falls into the class of APX-Hard problems. Hazan et al. showed that the p-set packing problem is hard to approximate within a factor $\Omega\left(\frac{p}{\log(p)}\right)$ for $p \geq 3$ [HSS06]. For small values of p, the unweighted p-partite hypergraph matching is NP-hard to approximate beyond a factor $98/97$, $54/53$, $30/29$ and $23/22$ for $p = 3, 4, 5$ and 6, respectively [BK03a]. In comparison, Graph Matching is solvable exactly in polynomial time as shown in the seminal works of Kuhn [Kuh55] and Edmonds [Edm65] in the bipartite and non-bipartite case, respectively. From an approximation perspective, there is a gap in our understanding between the weighted and the unweighted p-Set Packing problem. After a series of improvements [HS89; Hal95; CGM13; SW13], Cygan [Cyg13] obtained a $\frac{p+1+\varepsilon}{3}$-approximation algorithm for any $\varepsilon > 0$ for the unweighted problem. On the other hand, Neuwohner gave

new approximation algorithms for the weighted problem, which were the first to break the $\frac{p+1}{2}$ barrier. After a series of papers [Neu21; Neu22; Neu23], her work culminates today in an approximation factor equal to $\min\{0.5(p+1) - 0.0002, 0.4986(p+1) + 0.0208\}$ for $p \geq 4$ and 1.99999998 for $p = 3$.

We refer the reader to Chapter 2 for a detailed literature review of the p-set packing problem and improvements over these factors.

1.1.1 Overview of the contributions

In Chapter 2 and 3, we study the weighted p-set packing problem. All state-of-the-art algorithms for the p-set packing problem use local-search methods, where the algorithm attempts to improve the solution quality by making small changes to the current solution. Two crucial parameters dictate the guarantee of the final solution. First, the magnitude of the *small* changes dictates both the approximation guarantee and the running time of the algorithm. Making greater modification to the solution can be time-consuming but usually yields improved guarantees. Secondly, deciding whether a change *improves* the solution quality imposes a measure to be able to compare two solutions. While it is standard to use the objective function of the problem to compare solutions' quality, other measures that favor certain algorithmic behaviors can be preferred.

In Chapter 2, we give an algorithm attaining an approximation factor of 1.761 for weighted 3-set packing, improving on the recent best result of $2 - \frac{1}{63'700'992}$ due to Neuwohner [Neu21]. Our algorithm is based on the local-search procedure of Berman that attempts to improve the sum of squared weights rather than the problem's objective. Berman's algorithm attains an approximation factor of $\frac{p+1}{2}$ [Ber00] using exchanges of size p. Using larger exchanges of size $O(p^3)$, we provide a relatively simple analysis to obtain an approximation factor of 1.811 when $p = 3$. We then show that the tools we develop can be adapted to attain an approximation factor of 1.761 using exchanges of size $O(p^{O(1/\varepsilon)})$. This results in an algorithm with running time equal $O(n^{p^{O(1/\varepsilon)}})$. Although our primary focus is on the case $p = 3$ due its connection with the 3-dimensional matching problem, our approach in fact gives slightly stronger improvements on the factor $\frac{p+1}{2}$ for all $p > 3$. In particular, as p increases the approximation factor asymptotically converges to $\frac{p}{2}$.

Expanding our work, we give an algorithm with approximation factor $\sqrt{3}$ for $p = 3$, and $\frac{p}{2}$ for all $p \geq 7$ in Chapter 3. Our method builds on Chapter 2, but requires extending the analysis significantly. Our guarantees are tight with respect to the algorithm we consider. For $p = 3$, our local-search algorithm with bounded exchange size matches the performance guarantees of the same local-search algorithm with *unbounded exchange size*. Therefore, this result reaches and sets a new barrier for approximation algorithm to improve upon. Improving over the factor $\sqrt{3}$ demands novel optimization methods which we leave as an open question. For $p \geq 7$, our approximation factor improves over Neuwohner's result and Chapter 2 [Neu22]. Neuwohner obtained an asymptotic ratio equal to $\frac{p}{2}$ using exchanges of size $O(\log(n))$, whereas we use exchanges of size $O(p^{O(1/\varepsilon)})$ in Chapter 2. In contrast to both analyses, we prove that the factor $\frac{p}{2}$ is attainable in the non-asymptotic regime, i.e. for all

$p \geq 7$. To prove this result we need to consider exchanges of size at most $O((p/\varepsilon)^{O(1/\varepsilon)})$. This results in an algorithm that runs in time $O(n^{O(p/\varepsilon)^{O(1/\varepsilon)}})$. This result is again the best possible with respect to the algorithm that we consider. In fact, Neuwohner [Neu22] proves that even with exchanges of size $O(\log(n))$, it is impossible to get past this factor simply by running the squared weighted local-search. Our analysis also gives improvements over Chapter 2 for $p = 4, 5, 6$ and holds for the more general problem of finding a maximum weight independent set in a $(p+1)$-claw free graph.

Both chapters are based on joint work with Justin Ward. Some results in the first chapter appear in [TW23].

1.2 Streaming Algorithms subject to Independence Systems

Combinatorial optimization phrases problems in economics, and operations research in a mathematical language amenable to optimization, and defines combinatorial objects that capture properties displayed by real-world applications. Due to the growth of computational resources that goes in pair with an increase in datasets size, designed algorithms must now incorporate new requirements, so they can extract information efficiently. In this section, we introduce the problem of maximizing *submodular functions* over *independence system* in the *streaming* setting. Maximizing a submodular function over an independence system encompasses the p-set packing problem and represents a broad class of combinatorial problems present in many real-world situations. It offers a versatile framework capturing specialized applications, where matching constraints and/or linear objectives aren't suited. Our interest is mostly directed towards *fast* approximation algorithms. In fact, we study algorithms that process the dataset on the fly. We are interested in the trade-off between the approximation guarantee and the number of passes through the dataset.

1.2.1 Independence Systems

We begin this section by presenting *independence* systems. Introduced by Jenkyns [Jen75] in his Ph.D. thesis, an independence system on a ground set X is a combinatorial constraint that shapes the set of feasible solutions $\mathcal{I}$.

Definition 1.2.1 (Independence System)**.** An *independence system* is a pair $(X, \mathcal{I})$ where $X \triangleq \{e_1, \ldots, e_n\}$ is an arbitrary finite set of size n and $\mathcal{I} \subseteq 2^X$ is a collection of subsets of X such that

$$A \subseteq B \in \mathcal{I} \implies A \in \mathcal{I}.$$

The variables $e_1, \ldots, e_n$ are the *elements* of the ground set. In concrete applications, the set X is the dataset, and $\mathcal{I}$ is the set of feasible solutions. A solution $S \in \mathcal{I}$ is said to be *independent*.

Independence systems require the set of feasible solutions to be closed under inclusion, so any subset of a feasible solution remains feasible. This motivates the definition of *bases*.

Definition 1.2.2 (Base of an independence system)**.** A *base* B of an independence system $(X, \mathcal{I})$ is a maximal (inclusion-wise) independent set in $\mathcal{I}$. The *rank* of an independence system is the maximum size of a base. We denote it by $\text{rank}(X)$.

Independence systems are, unfortunately, too general. Most concrete applications in fact satisfy stronger properties than downward closeness. By imposing additional constraints independence systems can be decomposed further into smaller classes to capture key properties that applications exhibit. So far, many interesting classes have been introduced and in this thesis, we will consider the following classes: Matroid, p-Matroid-Intersection, p-Hypergraph Matching, and p-Matchoid. We refer the reader to [War12b] for an overview of broader classes than those considered here, including weak/strong p-exchange, p-parity, p-extendible, p-system. We will use this sans-serif font to denote the different classes of independence systems. Axiomatized before the notion of independence systems, Matroid is perhaps the most central class in this thesis. It is independence system class equipped with an *augmentation property*.

Definition 1.2.3 (Matroid)**.** A matroid $\mathcal{M} = (X, \mathcal{I})$ is an independence system such that for every $A, B \in \mathcal{I}$ with $|A| < |B|$, there exists an element $e \in B - A$ such that $A + e \in \mathcal{I}$.

Here and throughout the thesis, we define $A + B$ and $A - B$ as $A \cup B$ and $A \setminus B$, to be the union and the removal of set B with respect to the set A, respectively. For simplicity, we also write $A + e$ instead of $A + \{e\}$. Additionally, given a set $A \subseteq X$, we let $|A|$ be the number of elements in A. Matroids generalize the notion of linear independence in vector spaces. In fact, standard combinatorial results prove that all bases must have the same size (equal to the rank) if $\mathcal{M}$ is a matroid; this coincides with the fact that all bases of a vector space must have the same cardinality [Sch+03]. An important special case is when $\mathcal{I} \triangleq \{S \subseteq X \colon |S| \leq k\}$ for $k \in \mathbb{N}$ is the set of all subsets of size at most k. We say that $\mathcal{M}$ is a *uniform matroid*. Similarly, a *partition matroid* is when $\mathcal{I} \triangleq \{S \subseteq X \colon |S \cap X_i| \leq k_i,\ \forall i = 1, \ldots, \ell\}$ where X is partitioned in disjoint sets $X_1 \sqcup X_2 \sqcup \ldots \sqcup X_\ell$ and $k_i \in \mathbb{N}$. We will use $\sqcup$ to denote the union of disjoint sets in this thesis. Generalizing matroids, we introduce p-Matroid-Intersection.

Definition 1.2.4 (p-Matroid-Intersection)**.** Given p matroids $\{(X, \mathcal{I}_i)\}_{i=1}^{p}$, each defined on the same ground set X, a p-matroid intersection is an independence system $\mathcal{M} = (X, \mathcal{I})$ such that $\mathcal{I} \triangleq \bigcap_{i=1}^{p} \mathcal{I}_i$.

Therefore, given a p-matroid intersection $\mathcal{M} = (X, \mathcal{I})$, a set $A \in \mathcal{I}$ is independent if it is independent in each of the p matroids. Various combinatorial objects can be encoded as a p-matroid intersection including bipartite matchings, and Hamiltonian paths. The next paragraph will describe the reduction of the first problem carefully. A constraint of immediate generality is p-Matchoid, which was defined for $p = 2$ by Edmonds [Edm71] and studied by Jenkyns [Jen75]. One can intuitively think of a p-matchoid as a collection of matroids in which each element "participates" in at most p of the matroid constraints.

Definition 1.2.5 (p-Matchoid)**.** Given a collection of matroids $\{(X_i, \mathcal{I}_i)\}_{i=1}^m$, a *$p$-matchoid* $\mathcal{M} = (X, \mathcal{I})$ is an independence system defined over $X = \bigcup_{i=1}^m X_i$ such that every element $e \in X$ appears in the ground set of at most p of these matroids and $\mathcal{I} \triangleq \{S \subseteq X \colon S \cap X_i \in \mathcal{I}_i, \forall i = 1, \ldots, m\}$.

Since $\mathcal{I}$ does not necessarily have a succinct representation, we assume access to an *independence oracle*. For a given set S, querying the oracle will answer whether S is independent, i.e., $S \in \mathcal{I}$.

Matching constraints as Independence Systems

The two latter constraints might seem overwhelming. However, we show that p-partite hypergraph matchings and p-hypergraph matchings can be expressed as a p-matroid intersection and a p-matchoid, respectively.

Given a p-partite hypergraph $H = (V, E)$ with partition $V = V_1 \sqcup \ldots \sqcup V_p$, we define a collection of matroids $\{\mathcal{M}_j = (E, \mathcal{I}_j)\}_{j=1}^p$. The ground set of each matroid is the set of hyperedges. For each $j = 1, \ldots, p$, we define the set $\mathcal{I}_j \triangleq \{S \subseteq E \colon |S \cap \delta(v)| \leq 1, \text{ for all } v \in V_j\}$, where $\delta(v)$ is the set of hyperedges that contain v. The constraint defining $\mathcal{I}_j$ implies that each vertex in the j^{th} partition is *incident* to at most 1 hyperedge in the current solution. Therefore, an independent set in $\mathcal{I} \triangleq \bigcap_{j=1}^p \mathcal{I}_j$ is a p-partite hypergraph matching.

More generally, given a p-hypergraph $H = (V, E)$, we can define a collection of matroids $\{\mathcal{M}_v = (\delta(v), \mathcal{I}_v)\}_{v \in V}$, and $\mathcal{I}_v \triangleq \{S \subseteq \delta(v) \colon |S| \leq 1\}$. Since each edge contains at most p vertices, each edge participates in at most p matroids. The collection $\{\mathcal{M}_v\}_{v \in V}$ defines a p-matchoid whose objects are matchings in H.

An identical construction shows that a p-partite b-matching and a p-hypergraph b-matching can be expressed as p-matroid and a p-matchoid, respectively. A b-matching is a subset of hyperedges such that the number of hyperedges containing a given vertex v is at most b_v, where b_v is some given integer, and $b = (b_v)_{v \in V}$. A hypergraph matching is a hypergraph b-matching, where $b_v = 1$ for all $v \in V$.

1.2.2 Submodular Functions

In the second part of the thesis, we will be mostly interested in a relaxation of weighted functions, known as *submodular functions*. Briefly, submodular functions have a fundamental role in combinatorial optimization due to their property of diminishing returns, which makes them useful in a wide range of fields, including machine learning, social network analysis, and economics [LB10; KKT03].

Definition 1.2.6 (Submodular Function)**.** A set function $f : 2^X \to \mathbb{R}_{\geq 0}$ is *submodular* if for all sets $A, B \subseteq X$ the following inequality holds:

$$f(A) + f(B) \geq f(A \cup B) + f(A \cap B).$$

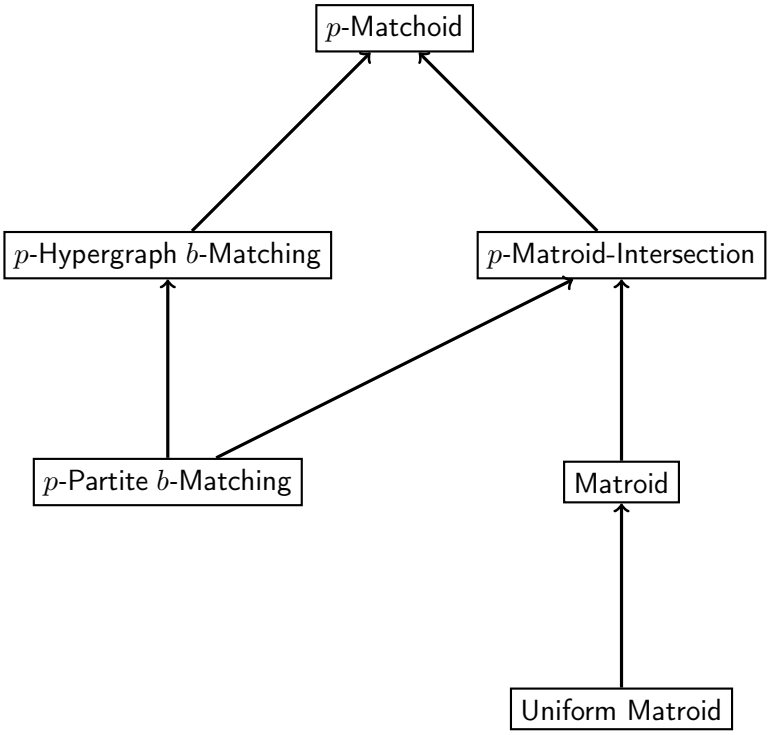


Fig. 1.2: Hierarchical visualization of independence system classes considered in this thesis. There is an arrow $A \to B$ if the class A is included in the class B.

For any set function, we will write $f(e \mid A) \triangleq f(A + e) - f(A)$ to denote the *marginal increase* in f when adding and element $e \in X$ to a set A. More generally, we will write $f(B \mid A) \triangleq f(A \cup B) - f(A)$ for any set $A, B \subseteq X$. Fischer, Nemhauser, and Wolsey [NWF78; FNW78] show the following equivalent statements.

Proposition 1.2.7 (Alternative characterizations)**.** *Given a set function $f : 2^X \to \mathbb{R}_{\geq 0}$, the following statements are equivalent:*

- *f is a submodular function.*
- *For all $A \subseteq B \subseteq X$ and all $e \in X \setminus A$, then $f(e \mid A) \geq f(e \mid B)$.*
- *For all $A, B \subseteq X$, the following holds: $\sum_{e \in B \setminus A} f(e \mid A) \geq f(A \cup B) - f(A)$.*
- *For all $A, B \subseteq X$, the following holds: $\sum_{e \in B \setminus A} f(e \mid A \cup B - e) \leq f(B \cup A) - f(A)$.*

The second bullet point is perhaps the most intuitive definition of submodular functions. It tells that the marginal contribution of an element decreases as the size of the underlying set increases. We observe that submodular functions are not necessarily increasing. We say that an increasing submodular function is *monotone*.

Definition 1.2.8 (Monotone Set Function)**.** A set function $f : 2^X \to \mathbb{R}_{\geq 0}$ is *monotone* if for all sets $A \subseteq B \subseteq X$, we have $f(A) \leq f(B)$.

Since f does not necessarily have a succinct representation, we assume access to a *value oracle*. For a given set S, querying the oracle will return the value of the set $f(S)$.

1.2.3 Streaming Algorithms

As stated in Definition 1.1.3, approximation algorithms for an independence system $(X, \mathcal{I})$ do not have any constraint concerning the accessibility of the ground set X. We call an algorithm: *offline*, if it assumes that X is available at all times. Given the growth of modern datasets, we focus on *streaming* algorithms, which process the dataset on the fly, i.e. element by element. They are designed to operate with limited memory and often produce approximate answers based on a sketch of the data stream.

For the next definition, let $X_\sigma \triangleq (x_{\sigma(1)}, x_{\sigma(2)}, \ldots, x_{\sigma(n)})$ be a permutation of the ground set, where σ is a permutation of the set $\{1, 2, \ldots, n\}$, and n is the size of X. For any $t \geq 1$, let $X_\sigma^{(t)} \triangleq (x_{\sigma(1)}, x_{\sigma(2)} \ldots, x_{\sigma(t)})$ be the first t elements of the ground set w.r.t σ.

Definition 1.2.9 (Streaming Algorithm for Independence Systems)**.** Given a maximization problem Π in the form (1.1) whose instances $I = ((X, \mathcal{I}), f) \in \Pi$ are independence systems on a ground set of size n, a *streaming algorithm* ALG is a procedure that, for any permutation σ, defines a sequence of sets $\{M_\sigma^{(t)}\}_{t=1}^n$, with $M_\sigma^{(t)} \subseteq X_\sigma^{(t)}$ such that for all t:

- Given $M_\sigma^{(t)} \cup \{x_{\sigma(t+1)}\}$ as an input to ALG, it finds in polynomial time a subset $U_\sigma^{(t)}$ that satisfies $M_\sigma^{(t+1)} \triangleq (M_\sigma^{(t)} \cup \{x_{\sigma(t+1)}\}) \setminus U_\sigma^{(t)}$,
- $\left|M_\sigma^{(t)}\right| \leq o(n)$.

It is an α-approximation algorithm if for any instance $I = ((X, \mathcal{I}), f) \in \Pi$ and permutation σ:

$$\text{ALG}(M_\sigma^{(n)}) \leq \text{OPT}(I) \leq \alpha \cdot \text{ALG}(M_\sigma^{(n)}),$$

where $\text{OPT}(I)$ is the value of the problem's objective function evaluated on the optimal solution, and $\text{ALG}(M_\sigma^{(n)})$ is the value of the solution output by ALG on the input $M_\sigma^{(n)}$.

Remark 1.2.10. We point out some subtleties in Definition 1.2.9. In the second bullet point, we only require that the streaming algorithm isn't able to store the entire ground set. The exact memory requirement for streaming algorithms is not uniform across the field of theoretical computer science. Strictly speaking, streaming algorithms enforce a memory size equal to $O(\log(n))$. The definition that we give refers to *semi-streaming* algorithms. For the problems that we consider in this thesis, the $o(n)$ term is replaced by $O(k \,\text{POLYLOG}(k))$ where k is the rank of the independence system. The reason we don't consider algorithms with memory $O(\text{POLYLOG}(k))$ is that they can't store enough information to achieve any approximation guarantee [Fei+05] (ex: for storing a maximum matching of a graph).

Given that the approximation factor is computed with respect to any ordering σ of the ground set, which can be chosen in an adversarial way, streaming algorithms perform worse than their offline counterparts. Streaming algorithms are thus given a sequence of elements $\{x_{\sigma(t)}\}_{t=1}^n$

presented one at a time. At each time step, their memory $M_\sigma^{(t)}$ is bounded. Given the set of elements stored in memory and a newly presented element, they must decide whether to include this element in the memory and potentially discard some set of elements $U_\sigma^{(t)}$. The decision to discard elements is irrevocable. At the end of the stream, the algorithm outputs a solution from $M_\sigma^{(n)}$.

For a reader familiar with *online* algorithms, there are a couple of differences that distinguish streaming algorithms from online algorithms. In both settings, the elements arrive on the fly. However, instead of maintaining a small memory footprint, online algorithms maintain a valid solution. When an element arrives, an online algorithm decides to either put it in the solution forever or discard it, in which case it will never be part of the solution. Both decisions are irrevocable. The decision at time t is computed with respect to entire past history $X_\sigma^{(t-1)}$ and the new element $x_{\sigma(t)}$. Sometimes, we allow unbounded computational power. On the other hand, streaming algorithms output the final solution *at the end of the stream* and discard forever elements that become useless. The decisions of the streaming algorithm at a time step t are solely based on its current memory $M_\sigma^{(t-1)}$ and the new element $x_{\sigma(t)}$.

Multipass algorithms are streaming algorithms that perform several passes over the dataset.

Definition 1.2.11 (m-pass algorithm)**.** Given a maximization problem Π as in (1.1) and an independence system instance $I = ((X, \mathcal{I}), f) \in \Pi$, a *$m$-pass streaming algorithm* ALG is a streaming algorithm on the input sequence of elements $(X_{\sigma_1}, X_{\sigma_2}, \ldots, X_{\sigma_m})$ where $\sigma_1, \ldots, \sigma_m$ are arbitrary permutations of X.

There is a slight subtlety in that elements in the memory after processing X_{σ_1} reappear in X_{σ_2} and thus must be duplicated. In general, we can simply assume that elements in the current memory $M_{\sigma_q}^{(n)}$ after processing X_{σ_q} are not reintroduced in $X_{\sigma_{q+1}}$.

1.2.4 Overview of the contributions

In Chapter 4, we give improved multipass streaming algorithms for the problem of maximizing a monotone or arbitrary non-negative submodular function subject to a general p-matchoid constraint.

For monotone submodular functions, our algorithm attains a guarantee of $p + 1 + \varepsilon$ using $O(p/\varepsilon)$-passes and requires storing only $O(k)$ elements, where k is the rank of the p-matchoid. This immediately gives an $O(1/\varepsilon)$-pass $(2 + \varepsilon)$-approximation algorithm for monotone submodular maximization in a matroid and $(3 + \varepsilon)$-approximation for monotone submodular matching.

To put it into perspective, the best approximation algorithms for maximizing a monotone submodular function over Matroid, p-Matroid-Intersection and p-Matchoid have approximation ratio equal to $\frac{e}{e-1}$, $p + \varepsilon$ and $p + 1$, respectively [Cal+11; FW14; LSV10]. While our results do not match the state-of-the-art (except for p-Matchoid), the approximation ratio that our algorithm achieves is in fact equal to that of the standard local-search algorithm. Local-search

has an approximation factor equal to $p+1$ for maximizing a monotone submodular function subject to a p-matchoid constraint.

The local-search procedure improves the current solution by finding small exchanges and requires access to the entire dataset at all times. Our result demonstrates that this assumption is not necessary. Our algorithm is effectively a streaming local-search procedure that simulates its offline counterpart. It shows that $O(p/\varepsilon)$-passes over the ground set is sufficient to obtain guarantees that are at most $1+\varepsilon$ times worse than the local-search algorithm.

Our techniques build on the work of Chakrabarti et al. and Chekuri et al. [CK15; CK15]. Subject to a p-matchoid constraint, they design a single pass streaming algorithm with an approximation factor equal to $4p$. Chakrabarti et al. improves this ratio to $p+1+\varepsilon$ in $O(\frac{p^4\log(p)}{\varepsilon^3})$-passes. We adapt the algorithm of Chekuri et al. [CGQ15] to the multipass setting. Using a clever parametrization of each pass, we obtain a rapid convergence in $O(p/\varepsilon)$-passes.

We extend our techniques to obtain the first multipass streaming algorithm for general, non-negative submodular functions subject to a p-matchoid constraint with a number of passes independent of the size of the ground set and k. We show that a randomized $O(p/\varepsilon)$-pass algorithm storing $O(p^3k\log(k)/\varepsilon^3)$ elements gives a $(p+1+\bar{\gamma}_{\text{off}}+\varepsilon)$-approximation, where $\bar{\gamma}_{\text{off}}$ is the guarantee of the best-known offline algorithm for the same problem. The chapter finishes with the design of the first multipass streaming algorithm for maximizing a *regularized* monotone submodular function under a uniform matroid.

This chapter is based on joint work with Chien-Chung Huang and Justin Ward, appearing in [HTW20].

1.3 Independence Systems in Machine Learning

In Chapter 5, we focus on important problems in machine learning that have a connection with submodular functions and independence systems. More precisely, we focus on some subset selection problems subject to a matroid constraint.

Subset selection problems are ubiquitous in statistics and machine-learning as they provide interpretability of high-dimensional models via the selection of a few features of interest. In most subset selection problems, we are given an independence system $(\mathcal{X},\mathcal{I})$ where $\mathcal{X}$ is the entire set of features. Since $\mathcal{X}$ can be large, we require selecting a few variables from $\mathcal{X}$ to explain some quantity of interest. Intuitively, it might be helpful for the reader to imagine a socio-economical study, where we are interested in finding the main factors that influence a certain decision or behavior.

In many applications, the independent set $\mathcal{I}$ considered is a uniform matroid. Thus, we require selecting at most k features from $\mathcal{X}$, where k is given as an input. However, in some applications, observations are mutually exclusive, or it might be desirable to spread

observations amongst multiple different classes. Uniform matroids do not capture such combinatorial constraints which requires using general matroids such as partition matroids.

The task of finding representative features is intimately related to that of a *summarizing* procedure. We want to select variables that collectively express as much information as possible. Given that submodular functions have applications in document summarization [LB10; Bai+15], it is not surprising that subset selection problems display submodular-like properties. This connection was observed by Das and Kempe [DK08; DK11]. Focusing on the *sparse least-square estimator problem*, they showed that under certain assumptions on $\mathcal{X}$ the least-square objective function was submodular [DK08]. They further reinforce this connection by showing that the objective function is *weakly submodular* [DK11].

Definition 1.3.1 (γ-Weakly Submodular Functions)**.** Given $\gamma > 0$, a monotone set function $f : 2^X \to \mathbb{R}_{\geq 0}$ is γ-*weakly submodular*[1] if for all sets $A \subseteq B \subseteq X$ the following inequality holds:

$$\gamma \cdot (f(B) - f(A)) \leq \sum_{e \in B \setminus A} f(e \mid A)\,. \tag{1.2}$$

The parameter γ is called the *(lower) submodularity ratio of* f. The definition relaxes the third bullet point of Proposition 1.2.7. Observe that, a monotone set function f is submodular if and only if it has a submodularity ratio of at least 1. Since then, there have been other approaches that consider other variants of weakly submodular functions [BZC18; Kuh+18; Qia+18]. Unlike Proposition 1.2.7, alternative characterizations are not equivalent. In Section 5.1.2, we give a detailed comparison between the different definitions.

For the least-square regression problem, Das and Kempe showed that the deviation from submodularity γ is controlled by a spectral parameter that depends on the covariance matrix between observations in $\mathcal{X}$ [DK11]. More importantly, they connect the weak submodularity property to the efficiency of the greedy algorithm. They prove that the greedy algorithm, which in each iteration adds the element with the largest marginal contribution, has a guarantee equal to $\frac{e^\gamma}{e^\gamma - 1}$ for maximizing a γ-weakly submodular function subject to a uniform matroid. While this result is optimal [Har+19] under the assumption that a subexponential number of queries are used, the problem of maximizing a weakly submodular function under a matroid constraint isn't settled. Chen et al. [CFK18] give a $(1 + 1/\gamma)^2$-approximation algorithm that remains the state-of-the-art.

The downside of Definition 1.3.1 is that it only bounds the value of a set when elements are added. It tells nothing about the decrease in value of that set when elements are removed. Thus, Definition 1.3.1 *enforces* algorithms to use a greedy-type of strategy to bound their performance guarantees while state-of-the-art algorithms for maximizing submodular functions use local-search. Thus, it is natural to ask whether subset selection problems satisfy stronger forms of submodularity that can be leveraged to obtain improved guarantees. We answer this question positively.

[1] We note that the definition given here, which is also used in [Bia+17; CFK18; Ele+17; Har+19; SY20], is slightly adapted from the original definition given in [DK11].

1.3.1 Overview of the contributions

In Chapter 5 we consider 3 widely studied subset selection problems subject to a matroid constraint. We consider: *Sparse Least-Square Regression*, *Bayesian A-Optimal Design*, and *Column Subset Selection* [DK08; DK11; Har+19; KSG08; Alt+16; Far+15].

Our first contribution is a refinement of the definition of weak submodularity by Das and Kempe. We introduce the notion of an upper submodularity ratio β, that considers the effect of the removal of elements. Effectively, our definition is analogous to that of Das and Kempe and relaxes the fourth bullet point of Proposition 1.2.7. We demonstrate that the upper submodularity ratio is bounded by spectral quantities linked to the input data for each application that we consider. Surprisingly, the attained spectral bounds imply that $\beta \leq \frac{1}{\gamma}$.

More generally, we reduce all the above problems to the question of maximizing a set function f with lower and upper submodularity ratio γ and β over a matroid constraint. It captures the subset selection problems that we study by setting $\beta = 1/\gamma$. Using our refined definition, we derive new, strengthened approximation guarantees. Improving the analysis of Chen et al.'s algorithm [CFK18], we obtain an enhanced guarantee equal to $1 + \gamma^{-2}$ when $\beta = 1/\gamma$. As the deviation of the set function from submodularity reduces, the approximation guarantee converges to $1/2$.

Our last contribution is the design of a novel approximation algorithm that achieves an optimal asymptotic approximation factor equal to $\frac{e}{e-1}$ as both submodularity ratios tend to 1. In other words, the performance of our algorithm increases as the function is closer to being submodular, i.e. when γ, β tends to 1. It achieves an optimal approximation under the assumption that P $\neq$ NP as $\beta = \gamma = 1$. Our algorithm proceeds in a local-search fashion demonstrating the versatility of our refinement. It is inspired by the work of Filmus and Ward [FW14] and is the first algorithm that asymptotically matches this factor.

This chapter is based on joint work with Justin Ward, appearing in [TW22].

Appendix

1.A State-of-the-art results

For completeness, we detail the state-of-the-art results for maximizing a weighted and monotone submodular function over the set of independence systems that we consider in this thesis. For compactness, we give the results in the form of a table (see Table 1.1 and 1.2). As notations, we use $-$ for a given cell to indicate that the best approximation/hardness result can be derived from another appropriate cell of the table by copying the result.

Although this thesis focuses mostly on maximizing monotone set functions, we point out that maximizing an arbitrary submodular function, even, subject to a uniform matroid is poorly understood. Buchbinder et al. [BF19] give a 2.5975-approximation algorithm whereas the best hardness result is 2.0366 by Gharan and Vondrák [GV11]. Improving either of these factors is an extremely important question in this area.

Offline	Unweighted		Weighted		Refs
	APX	Hardness	APX	Hardness	
Matroid	1	1	1	1	[Sch+03]
Matching	1	1	1	1	[Edm65]
p-Hypergraph	$\frac{p+1}{3}$	$\Omega\left(\frac{p}{\log(p)}\right)$	τ_p	$\Omega\left(\frac{p}{\log(p)}\right)$	[Cyg13; HSS06; TW23; Neu23]
p-intersection	$\frac{p}{2}$	$-$	$p-1$	$-$	[LSV10; LSV13]
p-matchoid	$\frac{p}{2}$	$-$	p	$-$	[LSV13; KH78]

Tab. 1.1: State-of-the-art approximation factors for maximizing linear objective functions over various independence systems. Here τ_p follows from Table 3.1 for $p \leq 361$ and $\tau_p = 0.4986(p+1) + 0.0208$ for $p \geq 361$.

1.B Basic Results from linear algebra, calculus, and more

We close this appendix with basic results from linear algebra, analysis, and matroid theory. Theorems can be found in graduate textbooks (see [RW05, Section A.3], [AS16], [Sch+03], [Ste04]). We start with the formula to compute the inverse of a block matrix.

Offline	Monotone Submodular		Refs
	APX	Hardness	
Matroid	$\frac{e}{e-1}$	$\frac{e}{e-1}$	[Cal+11; FW14; Fei98]
Bip. Matching	2	–	[LSV10]
p-Hypergraph	$\min\left\{p; \frac{p+3}{2}\right\}$	$\Omega\left(\frac{p}{\log(p)}\right)$	[War12a; HSS06]
p-intersection	p	–	[LSV10]
p-matchoid	$p+1$	–	[CVZ14]

Tab. 1.2: State-of-the-art approximation factor for maximizing a monotone submodular objective function over various independence systems.

Lemma 1.B.1 (Block Matrix Inverse). *Let B, A, U, V be matrices of conformable size. Then,*

$$\begin{pmatrix} B & U \\ V & A \end{pmatrix}^{-1} = \begin{pmatrix} B^{-1} + B^{-1}USVB^{-1} & -B^{-1}US \\ -SVB^{-1} & S \end{pmatrix}.$$

where $S = (A - VB^{-1}U)^{-1}$ is the Schur complement of B.

A somewhat related theorem is *Sherman-Morrisson-Woodbury formula* that computes the inverse of a matrix A after being updated by the matrix UCV.

Lemma 1.B.2 (Sherman-Morrisson-Woodbury formula). *Let A, U, C, V be matrices of conformable sizes. Then,*

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U\left(C^{-1} + VA^{-1}U\right)^{-1}VA^{-1}.$$

We will also use Cauchy-Schwarz, Young's inequality [Ste04] and Chernoff bound [AS16, Theorem A.1.16].

Theorem 1.B.3 (Cauchy-Schwarz). *Given real numbers $a_1, \ldots, a_n$, and $b_1, \ldots, b_n$, the following inequality is true:*

$$(a_1b_1 + \ldots + a_nb_n)^2 \leq \left(a_1^2 + \ldots + a_n^2\right)\left(b_1^2 + \ldots + b_n^2\right).$$

Theorem 1.B.4 (Young's inequality). *Given two non-negative real numbers $a, b \in \mathbb{R}_{\geq 0}$ and two conjugate real numbers $p, q > 1$ such that $\frac{1}{p} + \frac{1}{q} = 1$, then*

$$ab \leq \frac{a^p}{p} + \frac{b^q}{q}.$$

Lemma 1.B.5 (Chernoff Bound). *Let X_i, $1 \leq i \leq n$ be mutually independent random variables with $\mathbb{E}[X_i] = 0$ and $|X_i| \leq 1$ for all i. Set $S = X_1 + \cdots + X_n$. Then for any a,*

$$\Pr[S > a] < e^{-a^2/2n}.$$

The final result of this section is related to matroids. Proposition 1.B.6 is a fundamental theorem in matroid theory. It defines a bijection between the elements of any two bases such that elements in bijection are interchangeable, i.e., swapping them preserves the independence of the bases.

Proposition 1.B.6 ([Sch+03])**.** *Let $\mathcal{M} = (X, \mathcal{I})$ be a matroid. Then for any pair of bases A, B of $\mathcal{M}$, there exists a bijection $\pi : A \to B$ so that $A - a + \pi(a) \in \mathcal{I}$ for all $a \in A$.*

2 Improved Approximation for Weighted k-Set Packing

A portion of this chapter is part of a publication which appeared in SODA'23 [TW23]. Nonetheless, the presentation of the results is specific to the thesis. In particular, Section 2.5.1 differs from [TW23] which enables us to obtain strengthened results.

2.1 Introduction

In this chapter, we consider the weighted k-set packing problem. Given a weighted collection of n sets, each containing at most k elements from some universe $\mathcal{U}$, the goal is to return a collection of disjoint sets of maximum total weight. The weighted k-set packing problem generalizes many practical and theoretical problems. When $k = 2$, it encompasses the maximum weight matching problem. For $k = 3$, it generalizes the 3-dimensional matching problem, figuring in Karp's list of "21st NP-complete problems" [Kar72] which involves finding a maximum matching in a 3-partite hypergraph. In accordance with the title of this thesis, the combinatorial constraint underlying the k-set packing problem is a k-matchoid 1.1. While there is an exact algorithm for the maximum weight matching problem [Edm65], the 3-dimensional matching problem is NP-hard even in the unweighted case [Kar72]. For low values of k, unweighted k-dimensional matching is in fact NP-hard even to approximate beyond a factor of $98/97$, $54/53$, $30/29$ and $23/22$ for $k = 3, 4, 5$, and 6, respectively [BK03a; HSS06], and $\Omega(k/\ln(k))$ for general k [HSS06].

In contrast, the best approximation algorithm for the unweighted problem is a $\frac{k+1+\varepsilon}{3}$-approximation due to Cygan [Cyg13] with subsequent improvements by Fürer and Yu [FY14] to the running time dependence on ε. It is instructive to observe that all of the best known algorithms in the unweighted regime use local-search procedures, which repeatedly improve a solution S by adding some small number of sets not currently in S and removing intersecting sets from S. If each such swap attempts to add only one set at a time, then this leads to a k-approximation. Hurkens and Schrijver [HS89] showed that for any $\varepsilon > 0$, an algorithm performing swaps of size $O(\varepsilon^{-1})$ gives a $\frac{k+\varepsilon}{2}$-approximation, and subsequent improvements to $\frac{k+1+\varepsilon}{3}$ [Cyg13; FY14] have been obtained by increasing the swap size further to $\Omega(\log(n))$.

Surprisingly, in the case of *weighted* k-set packing, using swaps of size $O(\varepsilon^{-1})$ with respect to the original weight function leads to an approximation factor of only $k - 1 + \varepsilon$ [AH98]. However, Berman [Ber00] showed that by *squaring* the weight of each set and using swaps of size k to find a local optimum of the resulting instance results in a $\frac{k+1+\varepsilon}{2}$-approximation with respect to the original weight function (where here the ε is due to a further rescaling procedure to ensure the algorithm terminates in polynomial time).

Berman's algorithm in fact applies to the more general problem of finding a *maximum weight independent set of vertices in a* $(k+1)$*-claw free graph*. Recall from Section 1.1 that a d-claw is an induced subgraph of G comprising a single vertex (called the *center* of the claw) adjacent to a set of d pairwise non-adjacent vertices (called the *talons* of the claw). A graph is then $(k+1)$-claw free if it contains no induced $(k+1)$-claw. By creating a graph containing a vertex for each set in a k-set packing instance and an edge between sets that are non-disjoint, we can convert the (weighted) set packing problem to a (weighted) independent set problem, and if each set has size at most k, then the maximum size of a claw in the resulting graph is also k (see Figure 1.1). We call the graph G obtained in this way the *conflict graph* for the underlying set packing instance. For simplicity, we will henceforth consider the general problem of finding a maximum weight independent set in some vertex weighted $(k+1)$-claw free graph and adopt the associated vocabulary.

In this vocabulary, Berman's local search algorithm squares the weight of all vertices of the graph and then considers a restricted set of "claw swaps." For each vertex a in some current solution A, the algorithm searches for a claw of G centered at a. It adds the talons of this claw to A and discards any conflicting vertices from A as long as this increases the total (now squared) weight of A. The key difficulty in the analysis of the algorithm is in translating local optimality with respect to the squared weighting function w^2 into a guarantee in terms of the original weight function w. To accomplish this, Berman employs a 2-round charging argument, whereby vertices in the optimal solution distribute their weight among neighboring vertices in the locally optimal solution A produced by the algorithm.

For over 20 years, Berman's algorithm has remained the state-of-the-art approximation result for both Weighted k-Set Packing and Maximum Weight Independent Set in $(k+1)$-Claw Free Graphs. In a recent breakthrough result, Neuwohner [Neu21] broke the barrier of $\frac{k+1}{2}$ and obtained a slightly improved approximation ratio equal to $\frac{k+1+\varepsilon}{2} - \frac{1}{63{,}700{,}992}$ by squaring the weights and then considering larger exchanges than in Berman's algorithm. The key observation behind her analysis is that the charging argument employed by Berman is only tight when the weights of vertices in A and O are nearly identical, where here and throughout the text O refers to the optimal solution. Neuwohner's analysis leverages this observation to create a more complex charging scheme that considers several different classes of vertices. She then argues that in any solution A that is locally optimal under swaps of size $O(k^2)$, there must exist some set of vertices with weight constituting a significant fraction of the weight A, which receive less than $\frac{k+1}{2}$ times their weight under the new charging scheme. In a follow-up paper, Neuwohner [Neu22] showed that any local-search algorithm that works by improving some power w^α of the weights cannot improve on the factor $\frac{k}{2}$ even using swaps of size $O(\log n)$. However, Neuwohner manages to attain the factor $\frac{k}{2}$ asymptotically using swaps of size $O(\log n)$. She proves that for any $\delta > 0$, there is a k_δ such that for any $k \geq k_\delta$, considering swaps of size $O(\log n)$ with the squared weighting has approximation ratio $\frac{k+\delta}{2}$. The threshold is equal to $k_\delta = \frac{200{,}000}{\delta^3}$. As the rate of convergence to $\frac{k}{2}$ is relatively slow, for $k = 3$, where the potential for improvement in the ratio is the largest, the best factor remains $\frac{k+1+\varepsilon}{2} - \frac{1}{63{,}700{,}992}$. In further work, Neuwohner [Neu23] has recently shown that the barrier of $\frac{k}{2}$ can in fact be surpassed by running the unweighted local search algorithm

on appropriate sub-instances of a given instance. The techniques she employs require that $k \geq 4$. When $k = 4$, she obtains an improvement of 0.002 over the factor of $\frac{4+1}{2}$. As with previous results, the improvement over the factor $\frac{k+1}{2}$ grows with k to 0.0115 when $k = 13$, and $0.4986(k+1) + 0.0208$ for all $k \geq 14$.

Running Time: We emphasize that the algorithms considered here have a running time that is roughly $O(n^s)$ where s is swap size. Thus, the algorithms by Berman and Neuwohner [Ber00; Neu21; Neu22; Neu23] are polynomial time algorithm if k is constant. Additionally, known algorithms that use swap of size $O(\log(n))$ [Cyg13; Neu22; Neu23] can be turned into polynomial time algorithms by means of *color coding* which doesn't apply to the problem of finding a maximum independent set in a $(k+1)$-claw free graph. We point out that [BK03b] is state-of-the-art approximation when k is a function of n.

Our Results

Given this stream of recent progress in the asymptotic approximability of the weighted k-set packing for large k, it is natural to ask whether it is possible to obtain significant improvements in approximation specifically in the case of small k. In this chapter, we answer this question affirmatively, by giving two new approximation guarantees for the weighted k-set packing problem by using a variant of Berman's squared-weight local search with larger exchanges. We first present a relatively simple analysis showing that exchanges adding up to $k^2(k-1)+k$ sets is sufficient to obtain a factor 1.811 for Weighted 3-Set Packing, improving on the factor $\frac{k+1}{2} = 2$ by 0.189. We then show that by refining our basic analysis, it is possible to attain a 1.761-approximation using swaps of size $k^{O(1/\varepsilon)}$. Our results imply better improvements for $k > 3$, and we show that our algorithms' guarantees improve asymptotically to $(k+\frac{1}{2})/2$ and $k/2$, respectively, as k grows. The latter result matches Neuwohner's asymptotic result [Neu22] using a smaller swap size equal to $k^{O(1/\varepsilon)}$ instead of $O(\log(n))$. We summarize our results in the following theorem[1].

Theorem

A squared-weight local search algorithm performing exchanges of size $k^{O(1/\varepsilon)}$ is a polynomial time $\frac{k+1-\tau_k}{2}$-approximation for the weighted k-set packing problem, where $\tau_k \geq \tau_3 = 0.239$ and $\lim_{k\to\infty} \tau_k = 1$. The same algorithm with exchanges of size $k^2(k-1)+k$, is a $\frac{k+1-\tau'_k}{2}$-approximation with $\tau'_k \geq \tau'_3 = 0.189$ and $\lim_{k\to\infty} \tau'_k = 1/2$.

The exact statement of the above theorem is given in Theorem 2.4.9 and Theorem 2.5.5. Further specific values for our approximation guarantee, as well as the improvement $\tau_k/2$, $\tau'_k/2$ that we make over $\frac{k+1}{2}$ are given in Table 2.1. The precise value of τ_k depends on considering and balancing the worst of several quantities. To provide a brief overview, here we have simply listed the final results that follow from our techniques. After performing our main analysis, we provide and prove a more detailed version of the above theorem that explains how the numerical quantities in Table 2.1 are obtained.

[1] The results presented in [TW23] are weaker than the one presented here. There, in the best case, we obtained an asymptotic behavior equal to $\lim_{k\to\infty} \tau_k = 2/3$

While our results are also based on considering larger exchanges in the squared-weight local search algorithm introduced by Berman [Ber00], we adopt a different approach than that employed by Neuwohner [Neu22]. In Section 2.3, we give a compact proof of Berman's guarantee that avoids an explicit charging argument. This allows us to make explicit the slack present in the technical inequalities used to relate w^2 to w using local optimality. For each vertex a in a locally optimal solution A, we consider two different types of slack. The first, which we denote by Δ_a, captures the tightness of the claw swap centered at a (i.e. how much the total squared weight of A would decrease after performing the claw-swap centered at a). The second term, which we denote as Ψ_a, measures the slack in the remaining argument due to the deviation of the weight of the talons and of the neighbors of the talons from the weight of a. More precisely, Ψ_a captures the slack in two technical inequalities applied to vertex weights: $xy \leq \frac{1}{2}x^2 + \frac{1}{2}y^2$ and $\sum_i z_i^2 \leq (\max_i z_i) \sum_i z_i$, where all $z_i > 0$. Both of these inequalities are tight only when $x = y$ and all z_i are equal, respectively.

Our simpler analysis then works by considering exchanges of size $O(k^3)$ and bounding the sum of Δ_a and Ψ_a away from 0 in two cases. In the first case, suppose that a vertex $a \in A$ has some vertex b of similar weight that would be removed by the claw swap centered at a. Then, we show that the swaps centered at a and b cannot both be tight, since otherwise the swap which brings the sets of talons of a and b together would be improving. Hence, for any vertex a with a "close" vertex b of this sort, $\Delta_a + \Delta_b$ must be bounded away from 0, where the exact amount depends on the similarity between a's weight and b's weight. In order to exploit this in our analysis, we construct an auxiliary graph containing such "close" vertices, which we use to group individual claw swaps into larger exchanges involving $O(k^2)$ claws. We then show that the total slack we gain across all such large exchanges is a significant fraction of the weight of all the vertices of A whose claws participate in the exchange. For the remaining vertices a that have no "close" vertex b, we show that Ψ_a is large. To gain some further intuition in this case, one can consider the example shown in Figure 2.1, which is the worst case when applying Berman's algorithm to a single, isolated claw. Here the locality gap is only $\sqrt{3} < \frac{3+1}{2}$, which we show can be attributed to the slack Ψ_a. We show that even when a claw is not strictly isolated, as long as all of the other vertices in its neighborhood have significantly smaller weight than that of its center vertex, there is still a relatively large amount of slack Ψ_a that can be exploited.

By balancing these two cases, we are able to save over Berman's charging scheme for *all* vertices in A, rather than only a subset of constant weight. This is enough to obtain a 1.811-approximation for Weighted 3-Set Packing, which we present in Section 2.4. In Section 2.5 we show that by considering swaps of size $k^{O(1/\varepsilon)}$, it is possible to handle separately a key bottleneck case in our analysis and thus improve the ratio further to 1.761 (when $k = 3$). Note that when $k = 3$, the small example in Figure 2.1 shows that we cannot attain an approximation factor smaller than $\sqrt{3} \approx 1.732 > 3/2$. Intuitively, our improvements increase because the gap between $\frac{k+1}{2}$ and the bound of $\sqrt{k}$ for an isolated single claw (as shown for $k = 3$ in Figure 2.1) grows larger as k increases.

Swap Size:	$k^2(k-1)+k$		$k^{O(1/\varepsilon)}$	
k	$\tau'_k/2$	APX	$\tau_k/2$	APX
3	0.189	1.811	0.239	1.761
4	0.210	2.290	0.302	2.199
5	0.219	2.781	0.337	2.663
6	0.225	3.275	0.361	3.139
7	0.229	3.771	0.378	3.622
8	0.232	4.268	0.392	4.108
9	0.234	4.766	0.401	4.598
10	0.236	5.264	0.411	5.089

Tab. 2.1: Approximation ratio for different values of k and our improvements over $\frac{k+1}{2}$. In the last column, we removed an additional $O(\varepsilon)$ term to the approximation.

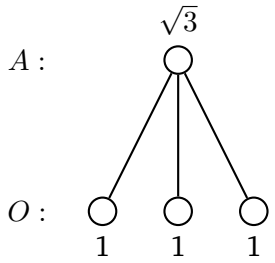


Fig. 2.1: An isolated bad example for the weight-squared local search.

Further related work

Nearly all algorithmic results for both the k-set packing problem and the maximum independent set problem in $(k+1)$-claw free graphs are based on variants of local search and greedy algorithms. In the unweighted setting, a simple local-search attempting to swap at most 2 vertices into the current solution yields a $\frac{k+1+\varepsilon}{2}$-approximation. Hurkens and Schrijver [HS89] showed that by considering swaps that add $O(\varepsilon^{-1})$ vertices gives a $\frac{k+\varepsilon}{2}$-approximation. They also show that their analysis is tight, in the sense that any local-search which swaps a constant number of vertices has approximation factor at least equal to $\frac{k+\varepsilon}{2}$ [HS89]. In contrast, Halldórson [Hal95] proved that a pure local search algorithm performing non-constant size swaps $\Omega(\log n)$ achieves a $\frac{k+2+\varepsilon}{3}$-approximation. This analysis was refined by Cygan et al. [CGM13] to obtain a ratio equal to $\frac{k+1+\varepsilon}{3}$. Due to the large swap sizes, the previous two results yield quasi-polynomial time algorithms. Sviridenko and Ward [SW13] and Cygan [Cyg13] designed polynomial-time local search algorithms with approximation factors of $\frac{k+2+\varepsilon}{3}$ and $\frac{k+1+\varepsilon}{3}$, respectively, by using techniques from fixed-parameter tractability. Fürer and Yu [FY14] gave a $\frac{k+1+\varepsilon}{3}$ approximation algorithm with improved dependence on ε and also gave an instance with locality gap $\frac{k+1}{3}$ for any algorithm using swaps of size $O(n^{1/5})$. All algorithms considering swaps of size $O(\log n)$ rely on the underlying structure specific to the k-set packing problem to find swaps in polynomial time, and thus do not generalize to the maximum independent set problem in $(k+1)$-claw free graphs.

In the weighted setting, Arkin and Hassin showed that the standard weighted local-search algorithm performing swaps of size $O(\varepsilon^{-1})$ yields only a $k-1+\varepsilon$ approximation [AH98]. Chandra and Halldórson [CH01] showed that the associated locality gap could be circumvented by combining a greedy algorithm followed by a local-improvement strategy that always selects the best improvement at each stage, yielding a $\frac{2(k+1)+\varepsilon}{3}$-approximation. As we have already noted, Berman [Ber00] obtained a $\frac{k+1}{2}$ approximation by considering a local search guided by the squared weights and swaps of size k. For smaller swaps of size 2, Berman and Krysta [BK03b] showed that a local search guided by w^α, for an appropriately chosen

$1 < \alpha < 2$ has an approximation factor of $0.667k$, $0.651k$, and $0.646k$ for $k = 3$, $k = 4$, and $k > 4$, respectively.

The k-set packing problem has also been studied via linear programming hierarchies. In this context, Chan and Lau [CL12] give an LP-rounding algorithm with approximation ratio $k - 1 + \frac{1}{k}$ for k-set packing and $k - 1$ for k-dimensional matching. They also show that even after the linear program is strengthened by a linear number of rounds of the Sherali-Adams lifting procedure, its integrality gap remains at least $k - 2$. In contrast, they show that by including a polynomial number of extra constraints, the integrality gap can be reduced to $\frac{k+1}{2}$. Singh and Talwar [ST10] showed that the same integrality gap of $\frac{k+1}{2}$ can be achieved by applying $O(k^2)$ rounds of Chvátal-Gomory cuts to natural LP for the k-set packing problem.

2.2 Preliminaries

In this section, we fix the notations used throughout Chapter 2 and 3. We consider the general setting in which we are given a vertex-weighted $(k+1)$-claw free graph $G = (V, E)$ and seek an independent set of maximum weight. For each $v \in V$, we let $w_v \in \mathbb{R}_+$ denote the given weight of v and for any $A \subseteq V$ we let $w(A)$ denote the total weight $\sum_{v \in A} w_v$ of all vertices in A.

For any two subsets A, B of vertices in V we define the *neighbourhood of A in B*, written $N(A, B)$, as $N(A, B) \triangleq \{b \in B \colon (a, b) \in E(G) \text{ for some } a \in A\} \cup (A \cap B)$. To simplify notation, we will write $N(o, A)$ instead of $N(\{o\}, A)$ for a vertex $o \in V$, and additionally use the shorthand $A - a$ for $A \setminus \{a\}$. Because G is $(k+1)$-claw free, the neighborhood $N(v, V)$ of any $v \in V$ contains at most k pairwise non-adjacent vertices. In particular, if A is an independent set of vertices, then $|N(v, A)| \leq k$ for all $v \in V$ and $N(v, A) = \{v\}$ for all $v \in A$.

The general local search procedure that we analyze is shown in Algorithm 1. The procedure maintains a current solution S, which we initialize using the standard greedy algorithm. We let $s \geq 1$ be a parameter governing the size of the exchanges performed by the algorithm. The algorithm repeatedly searches for an independent set of at most sk vertices $C \subseteq V \setminus S$ with total *squared* weights larger than the total squared weight of the conflicting vertices $N(C, S)$ in S. Whenever such a set is found, the algorithm adds C to S and removes $N(C, S)$ from S. Formally, for any $A \subseteq V$, we let $w^2(A) \triangleq \sum_{v \in A} w_v^2$. Then, Algorithm 1 exchanges a set $C \subseteq V \setminus S$ for $N(C, S)$ only if $w^2(C) > w^2(N(C, S))$. We can implement the search for each improvement in time $O(n^{sk})$ via simple enumeration. By using a pre-processing procedure to rescale and round the input weights, it can be ensured that the algorithm converges to an a local optimum in polynomial time while suffering a slight loss of approximation [Ber00]. In fact, because this results in only a small, polynomial dependence on this loss factor, a simple partial enumeration procedure can be used to remove the loss entirely, as we show in Section 2.5.3.

Algorithm 1: Squared Weight Local Search with s-Exchanges

$S \leftarrow$ the output of the standard greedy algorithm applied to G and w;
repeat
  $S' \leftarrow S$;
  foreach $C \subseteq V \setminus S$ of containing at most sk vertices **do**
    if C is an independent set and $w^2(C) > w^2(N(C,S))$ **then**
      $S' \leftarrow S \cup C \setminus N(C,S)$;
      break;
until $S = S'$;
return S

Thus, in all of our remaining analysis, we will suppose that the algorithm has terminated and produced a locally optimal solution A for our instance. We let O denote the optimal solution of this same instance. Note that both A and O are independent sets of G, and since G contains no $(k+1)$-claw, the maximum degree in the subgraph of G induced by $A \cup O$ is at most k.

In order to define a set of claw swaps, Berman [Ber00] makes a mapping $\pi\colon O \to A$ by $\pi(o) = \arg\max\{w_x : x \in N(o,A)\}$, breaking ties in an arbitrary, consistent manner. Note that $\pi(o)$ is the neighbour of o in A of maximum weight. Using π, we define a collection of sets $\mathcal{C} = \{C_a\}_{a\in A}$, where $C_a \triangleq \{o : \pi(o) = a\}$. Then, each vertex $o \in O$ appears in exactly one set $C_a \in \mathcal{C}$. We observe each set C_a forms the talons of a claw of G centered at vertex $a \in A$. Thus $|C_a| \leq k$ for all $a \in A$. Moreover for each $a \in A$, we have $w_a \geq w_v$ for all $v \in N(C_a, A)$.

For each $a \in A$, we define $N_a^+ \triangleq \{a\} \cup \bigcup_{o\in C_a} N(o, A-a)$. Note that if $C_a \neq \emptyset$ then $N_a^+ = N(C_a, A)$ and if $C_a = \emptyset$ then $N_a^+ = \{a\}$. For each $a \in A$, we consider in our analysis a local operation that adds C_a to A and removes N_a^+ from A. We call each such operation a 1-*exchange*, since it involves the talons of one claw C_a. Local optimality with respect to these 1-exchanges then implies that for any $a \in A$,

$$w^2(C_a) \leq w^2(N_a^+) \leq w_a^2 + \sum_{o\in C_a} w^2(N(o, A-a)), \tag{2.1}$$

where the final inequality follows since $a \in N(o,A)$ for all $o \in C_a$. Note that for empty claws with $C_a = \emptyset$, the above inequality follows immediately from $N_a^+ = \{a\}$. In this case, observe that the corresponding 1 exchange simply removes a from the solution A.

2.3 A simple proof of Berman's algorithm

We review the argument from the analysis of Berman [Ber00], which shows that the absence of improving 1-exchanges for w^2 implies that $w(O) \leq \frac{k+1}{2} w(A)$. Berman's proof uses a 2-stage charging argument and shows that each vertex in the current solution A receives less than

$(k+1)/2$ times its weight. Here we present a (arguably) simpler proof without charging argument, in which we make explicit the slack in several inequalities that are key in the analysis of [Ber00]. For each $a \in A$, and $o \in C_a$, we define the following quantities to measure this slack:

$$\begin{aligned}
\psi_{a,o} &\triangleq (w_o - w_a)^2 + w_a w(N(o, A-a)) - w^2(N(o, A-a)), \\
\Psi_a &\triangleq \sum_{o \in C_a} \psi_{a,o}, \\
\Delta_a &\triangleq w^2(N_a^+) - w^2(C_a).
\end{aligned}$$

It is important for the reader to become familiar with these notations as they will crucially be used in Chapter 2 and 3. Consider first Ψ_a and note that for each $a \in A$ and $o \in C_a$, $(w_o - w_a)^2 \geq 0$ and by construction of C_a, we have $w_v \leq w_a$ for all $v \in N(o, A)$. Thus, $w^2(N(o, A-a)) = \sum_{v \in N(o,A-a)} w_v^2 \leq w_a \sum_{v \in N(o,A-a)} w_v = w_a w(N(o, A-a))$ and so $\psi_{a,o} \geq 0$ for all $o \in C_a$. It then follows that $\Psi_a \geq 0$ for all $a \in A$. Next, note that since $|C_a| \leq k$ for each a, local optimality with respect 1-exchanges (2.1) implies that $\Delta_a \geq 0$ for all $a \in A$. We now show that the values Ψ_a and Δ_a can indeed be treated as slack in the analysis of Berman's algorithm:

Lemma 2.3.1. *Suppose A is locally optimal with respect to 1-exchanges. Then,*

$$2w(O) \leq w(A) + \sum_{o \in O} w(N(o, A)) - \sum_{a \in A} \left[\frac{\Delta_a}{w_a} + \frac{\Psi_a}{w_a} \right].$$

Proof of Lemma 2.3.1. Fix a single claw C_a and $o \in C_a$. Then,

$$\begin{aligned}
2w_o w_a &= w_o^2 + w_a^2 - (w_o - w_a)^2 \\
&= w_o^2 + w_a^2 - (w_o - w_a)^2 - w^2(N(o, A-a)) + w^2(N(o, A-a)) \\
&\quad - w_a w(N(o, A-a)) + w_a w(N(o, A-a)) \\
&= w_o^2 + w_a^2 - w^2(N(o, A-a)) + w_a w(N(o, A-a)) - \psi_{a,o}.
\end{aligned} \tag{2.2}$$

Equation (2.2) holds for every $o \in C_a$. Summing over all $o \in C_a$ then gives:

$$\begin{aligned}
2w_a w(C_a) &= |C_a| w_a^2 + w^2(C_a) - \sum_{o \in C_a} w^2(N(o, A-a)) + w_a \sum_{o \in C_a} w(N(o, A-a)) - \Psi_a \\
&\leq (|C_a| + 1) w_a^2 + w^2(C_a) - w^2(N_a^+) + w_a \sum_{o \in C_a} w(N(o, A-a)) - \Psi_a \\
&= (|C_a| + 1) w_a^2 - \Delta_a + w_a \sum_{o \in C_a} w(N(o, A-a)) - \Psi_a \\
&= w_a^2 - \Delta_a + w_a \sum_{o \in C_a} w(N(o, A)) - \Psi_a,
\end{aligned}$$

where the inequality follows from the second inequality in (2.1), and the final equation from the fact that $a \in N(o, A)$ for all $o \in C_a$ by construction, and so $w_a^2 + w_a w(N(o, A - a)) = w_a w(N(o, A))$ for each $o \in C_a$. Dividing both sides by w_a gives

$$2w(C_a) \leq w_a + \sum_{o \in C_a} w(N(o, A)) - \left[\frac{\Delta_a}{w_a} + \frac{\Psi_a}{w_a} \right], \tag{2.3}$$

which holds for each $a \in A$. Summing (2.3) over all $a \in A$ and recalling that each $o \in O$ appears in exactly one set $C_a \in \mathcal{C}$ then completes the proof. □

As an immediate corollary, we recover the standard approximation result of Berman [Ber00].

Corollary 2.3.2. *For any A that is locally optimal with respect to 1-exchanges,*

$$w(O) \leq \frac{k+1}{2} w(A).$$

Proof of Corollary 2.3.2. As we have noted above, we have $\Psi_a \geq 0$ for all $a \in A$ and since A is locally optimal with respect to 1-exchanges, $\Delta_a \geq 0$ for all $a \in A$. Thus, Lemma 2.3.1 implies that

$$2w(O) \leq w(A) + \sum_{o \in O} w(N(o, A)).$$

Now, we note that since O is an independent set and G is $(k+1)$-claw free, each $a \in A$ appears in $N(o, A)$ for at most k distinct $o \in O$. Thus, $\sum_{o \in O} w(N(o, A)) \leq k w(A)$. Using this in the inequality above and dividing through by 2 then completes the proof. □

A remarkable fact about the proof of Lemma 2.3.1 is that it can be easily modified to powers other than 2. We believe that this fact could be helpful to understand the surprising power of the squared weighting for the k-set packing problem. Unfortunately, we haven't been able to exploit it yet.

Lemma 2.3.3. *Suppose that A is locally optimal with respect to 1-exchanges and w^α, with $\alpha > 1$. Then,*

$$\alpha w(C_a) \leq w_a + (\alpha - 2) |C_a| w_a + \sum_{o \in C_a} w(N(o, A)) - \left[\frac{\Delta_a^{(\alpha)} + \Psi_a^{(\alpha)}}{w_a^{\alpha - 1}} \right],$$

where $\Delta_a^{(\alpha)}, \Psi_a^{(\alpha)} \geq 0$ and

- $\Delta_a^{(\alpha)} \triangleq w^\alpha(N_a^+) - w^\alpha(C_a)$,
- $\Psi_a^{(\alpha)} \triangleq \sum_{o \in C_a} \left[(w_o^\alpha + (\alpha - 1) w_a^\alpha - \alpha w_o w_a^{\alpha - 1}) + w_a^{\alpha - 1} w(N(o, A - a)) - w^\alpha(N(o, A - a)) \right]$.

Proof of Lemma 2.3.3. The proof is identical to that of Lemma 2.3.1. Fixing a single claw C_a and $o \in C_a$, we begin with the term $\alpha w_o w_a^{\alpha-1}$ to which we add and subtract $w_o^\alpha + (\alpha-1)\, w_a^\alpha$.

$$\alpha w_o w_a^{\alpha-1} = w_o^\alpha + (\alpha-1)\, w_a^\alpha - \left((w_o^\alpha + (\alpha-1)\, w_a^\alpha) - \alpha w_o w_a^{\alpha-1} \right)$$

Letting $\psi_{a,o}^{(\alpha)} \triangleq (w_o^\alpha + (\alpha-1)\, w_a^\alpha) - \alpha w_o w_a^{\alpha-1} + w_a^{\alpha-1} w(N(o, A-a)) - w^\alpha(N(o, A-a))$, and adding and subtracting $w^\alpha(N(o, A-a))$ and $w_a^{\alpha-1} w(N(o, A-a))$, we get

$$\alpha w_o w_a^{\alpha-1} = w_o^\alpha + (\alpha-1)\, w_a^\alpha - w^\alpha(N(o, A-a)) + w_a^{\alpha-1} w(N(o, A-a)) - \psi_{a,o}^{(\alpha)}.$$

Summing over all $o \in C_a$, we obtain the following equation

$$\begin{aligned} \alpha w(C_a) w_a^{\alpha-1} = w^\alpha(C_a) + |C_a|(\alpha-1)\, w_a^\alpha + \sum_{o \in C_a} \left[w_a^{\alpha-1} w(N(o, A-a)) - w^\alpha(N(o, A-a)) \right] \\ - \Psi_a^{(\alpha)}. \end{aligned}$$

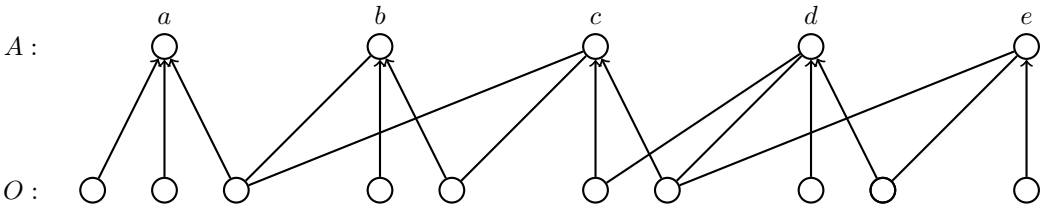
Substituting the first second inequality in Equation (2.1) which now holds with respect to the power α, we have

$$\begin{aligned} \alpha w(C_a) w_a^{\alpha-1} &\leq w^\alpha(C_a) + (|C_a|(\alpha-1)+1)\, w_a^\alpha - w^\alpha(N_a^+) + \sum_{o \in C_a} w_a^{\alpha-1} w(N(o, A-a)) - \Psi_a^{(\alpha)}, \\ &= w_a^\alpha + |C_a|(\alpha-1)\, w_a^\alpha + \sum_{o \in C_a} w_a^{\alpha-1} w(N(o, A-a)) - \Delta_a^{(\alpha)} - \Psi_a^{(\alpha)}, \\ &= w_a^\alpha + |C_a|(\alpha-2)\, w_a^\alpha + \sum_{o \in C_a} w_a^{\alpha-1} w(N(o, A)) - \Delta_a^{(\alpha)} - \Psi_a^{(\alpha)}. \end{aligned}$$

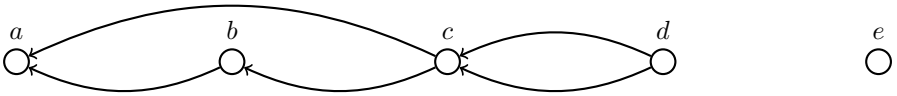
Dividing through by $w^{\alpha-1}$ yields the desired result. The positivity of $\Delta_a^{(\alpha)}$ is by the absence of improving claw-swaps. The positivity of $\Psi_a^{(\alpha)}$ follows from Young's inequality (Theorem 1.B.4, with conjugated exponent equal to $p = \alpha$ and $q = \alpha/(\alpha-1)$). □

2.4 An Improved Algorithm Using Larger Exchanges

We now show that when A is locally optimal with respect to larger exchanges, we can obtain a better approximation ratio. Our proof will proceed by obtaining a lower bound on the total slack Δ_a and Ψ_a for all vertices in Lemma 2.3.1. Before going further we give some high-level intuition for our approach. From the proof of Corollary 2.3.2, we see that the approximation ratio of Algorithm 1 is close to $\frac{k+1}{2}$, only when both Ψ_a and Δ_a are close to 0. For a given vertex $a \in A$, Δ_a measures the *tightness* of the 1-exchanges centered at a, in the sense that having Δ_a equal to 0 means that the 1-exchanges centered at a satisfies $w^2(C_a) = w^2(N_a^+)$. Suppose that there are two vertices a, b such that $b \in N_a^+ \backslash \{a\}$, and consider an exchange which attempts to add $C_a \cup C_b$ and removes $N_a^+ \cup N_b^+$. If this larger exchange is non-improving, we will show that we cannot have both $\Delta_a = 0$ and $\Delta_b = 0$. Intuitively, this follows since b is counted once in $N_a^+ \cup N_b^+$ but once in *both* N_a^+ and N_b^+. Assuming that b has a large weight compared to a yields a substantial improvement. On the



(a) Conflict graph



(b) Exchange graph H_ε

Fig. 2.2: In this picture, we show the exchange graph $H_{1/4}$ (Figure 3.2), coming from the conflict graph $G[A \cup O]$ in Figure 2.2a. We assume that $w_a = w_b = w_c = 1$, $w_d = 4/5$, and $w_e = 1/2$. In Figure 2.2a, we label the edge from each vertex of o to $\pi(o)$ with an arrow and assume that ties are broken by ordering vertices by label.

other hand, if all the vertices in $N_a^+ \backslash \{a\}$ have low weight compared to a, then we show that we can bound the slack term Ψ_a away from 0 well.

Our general approach will consider a set of s-exchanges bringing the talons of $s > 1$ claws into A simultaneously. In order to define it, we make use of the following auxiliary graph.

Definition 2.4.1 (Exchange Graph H_ε)**.** Fix $0 \leq \varepsilon \leq 1$. We define the *exchange graph* H_ε to be a directed graph with $V(H_\varepsilon) = A$ and arcs $(a, b) \in E(H_\varepsilon)$ from a to b with $b \neq a$ for every $o \in C_b$ such that: $a \in N(o, A - b)$ and $w_a \geq (1 - \varepsilon)w_b$.

In the later analysis, we use that endpoints of a given arc $(a, b) \in H_\varepsilon$ can't both have tight claw-swaps, i.e., $\Delta_a = \Delta_b = 0$. Note that for any arc $(a, b) \in E(H_\varepsilon)$, we have $(1 - \varepsilon)w_b \leq w_a \leq w_b$. Additionally, observe that the exchange graph may contain parallel arcs (we will later remove this assumption). In Figure 2.2 we show an example of a graph G and the corresponding exchange graph H_ε. Note that the first condition of Definition 2.4.1 implies that in H_ε contains an arc (x, y) or (y, x) only if $x \in A$ and $y \in A$ are joined by a path of length 2 in $G[A \cup O]$. Since the maximum degree in $G[A \cup O]$ is k, there are at most $k(k-1)$ paths of length 2 ending at any vertex $x \in A$, and so the maximum degree of any vertex $x \in V(H_\varepsilon)$ is $k(k-1)$.

We will refer to vertices of degree 0 in H_ε as *isolated vertices* and let I denote the set of all isolated vertices. We call the remaining vertices $D \triangleq A \setminus I$ *non-isolated vertices*. We consider each type of vertex separately, and show that the total value of the slack term is large in both cases.

2.4.1 Removing parallel arcs, triangles and more

We will further assume that the conflict graph $G[A \cup O]$ doesn't contain cycles of small length. This assumption is not necessary but will greatly simplify later discussions. We show how to remove this assumption in Section 2.5.4. More precisely, we assume Reduction 1.

Reduction 1. Assume that the conflict graph doesn't contain a cycle of length $4m$ where m is a constant.

A consequence of Reduction 1 with $m = 1$ is the absence of 4-cycles in the conflict graph, which in particular excludes the presence of parallel arcs. Similarly, the absence of 6-cycles excludes the presence of triangles in the exchange graph, ... etc. The main use of the reduction is to partition the exchange graph into vertex disjoint *trees*, instead of arbitrary graphs. Lemma 2.4.2 summarizes it. The proof is present in Section 2.5.4.

Lemma 2.4.2. *Assuming that Reduction 1 holds for some constant m, then any connected induced subgraph $F \subseteq H$ of the exchange graph H with longest path of length at most $2m-2$ is a tree.*

2.4.2 Bounding the slack for non-isolated vertices

The goal of this section is to prove Lemma 2.4.4. It bounds away the slack for *non-isolated* vertices by considering an *s-exchange* in which $s > 1$ claws C_a are added together to A. Recall that D is exactly the set of non-isolated vertices in H_ε and $I = A \setminus D$.

Definition 2.4.3 (Locally optimal)**.** A tree T is said to be *locally optimal* if $w^2(N_T^+) - w^2(C_T) \geq 0$, where $N_T^+ \triangleq \bigcup_{v \in T} N_v^+$ and $C_T \triangleq \bigcup_{v \in T} C_v$.

Using exchanges of size $s \leq 1 + k(k-1)$, which will imply the local optimality of trees of size $|V(T)| \leq s$ we prove the following lemma.

Lemma 2.4.4. *Let $0 \leq \varepsilon \leq 1/2$ and suppose that A is locally optimal under s-exchanges for $s \leq 1 + k(k-1)$. Then,*

$$\sum_{a \in D} \left[\frac{\Delta_a + \Psi_a}{w_a} \right] \geq \frac{1-\varepsilon}{2-\varepsilon} w(D) + \sum_{a \in D} \sum_{o \in C_a} \varepsilon w(N(o, I)).$$

Lemma 2.4.4 implies that we save almost half of the total weight of the vertices in D, i.e., $(1-\varepsilon)/(2-\varepsilon)$. This contrasts with the proof of Berman's algorithm where we had only $\sum_{a \in A} \frac{\Delta_a + \Psi_a}{w_a} \geq 0$. Intuitively, the term $\varepsilon w(N(o, I))$ shows that an ε-fraction of the weight of each *isolated* neighbor of a *non-isolated* vertex is saved.

High-level Intuition: Proving this Lemma requires proving Lemma 2.4.5, Lemma 2.4.6 and Lemma 2.4.7. Lemma 2.4.5 formalizes our intuition that the claw-swaps centered at endpoints of an arc $(b, a) \in H_\varepsilon$ can't be tight (assuming the absence of large improving exchange). Thus, $\frac{\Delta_a}{w_a} + \frac{\Delta_b}{w_b} \geq \frac{w_b^2}{w_a}$. In Lemma 2.4.6 we observe that Ψ_a captures the difference of weight between w_b and w_a which further augments the slack from w_b^2/w_a to w_b. Lemma 2.4.7 finalizes the proof by sending the slack w_b to both a and b.

Constructing exchanges: We start by constructing an appropriate set of s-exchanges for $s = 1 + k(k-1)$. To do this, we partition the vertices of D as follows. Let T initially

be a collection of arcs from H_ε constituting an arbitrary undirected spanning tree in each connected component of H_ε (note that here we will ignore the direction of each arc). As long as T contains an undirected path of length at least three, we remove one of the middle arcs (i.e. an arc incident on 2 vertices of degree at least 2) of this path from T. Observe that each such alteration decreases the number of arcs in T, and so this procedure terminates. At the end of the procedure, our final set of arcs T is a collection of disjoint trees, each containing no path of length 3. This implies that each connected component of T must be a star. Moreover, at the end of the procedure all vertices of D have degree at least one in T, since we never remove an arc incident on a vertex of degree less than two. When the process terminates, it follows that T is a disjoint collection of stars $T_1, \ldots, T_\ell$ contained in H_ε, with each vertex of D appearing in exactly 1 star. Since the maximum degree of a vertex in H_ε is at most $k(k-1)$, we have $|V(T_i)| \leq 1 + k(k-1)$ for all $i = 1, \ldots, \ell$ and $|C_a| \leq k$ for each $a \in V(T_i)$. Thus, each such swap (that altogether brings the set of talons centered at all vertices in $V(T_i)$ for some $i = 1, \ldots, \ell$) is an s-exchange adding an independent set of at most sk vertices to A and so will be considered by Algorithm 1 when $s = 1 + k(k-1)$.[2]

Lemma 2.4.5 is essential in the forthcoming proofs. Given a tree T that is locally optimal w.r.t w^2, it measures the effect of large exchanges. It proves that the gain is related to the weights of the endpoints of the arcs in the tree. The gain is greater when the weights of endpoints do not differ too much and thus motivates the definition of the set D.

Lemma 2.4.5. *Let T be a tree in the exchange graph H that is locally optimal with respect to w^2. Then,*

$$\sum_{v \in V(T)} \frac{\Delta_v}{w_v} \geq \sum_{(u,v) \in E(T)} \frac{w_u^2}{w_v}.$$

Proof of Lemma 2.4.5. Recall that for each vertex v we define $\Delta_v \triangleq w^2(N_v^+) - w^2(C_v)$. For any subset $X \subseteq A$, we similarly define $\Delta_X \triangleq w^2\left(\bigcup_{v \in X} N_v^+\right) - w^2\left(\bigcup_{v \in X} C_v\right)$. We first prove by induction on the size of $V(T)$ that:

$$0 \leq \Delta_{V(T)} \leq \left(\sum_{v \in V(T)} \frac{\Delta_v}{w_v} - \sum_{(a,b) \in E(T)} \frac{w_a^2}{w_b} \right) w_{\hat{v}}, \tag{2.4}$$

where $\hat{v} = \arg\max_{v \in V(T)} w_v$. For the case in which $|V(T)| = 1$, we must have $V(T) = \{\hat{v}\}$, and $E(T) = \emptyset$. Thus,

$$\left(\sum_{v \in V(T)} \frac{\Delta_v}{w_v} - \sum_{(a,b) \in E(T)} \frac{w_a^2}{w_b} \right) w_{\hat{v}} = \frac{\Delta_{\hat{v}}}{w_{\hat{v}}} \cdot w_{\hat{v}} = \Delta_{\hat{v}} \geq 0,$$

[2] We briefly note that each claw except for the claw C_v associated with central vertex of a star T_i shares an element of O with C_v. Thus one can in fact reduce the size of exchanges required by our algorithm from $k^2(k-1)+k$ to $k(k-1)^2+k$. To avoid introducing further details, we have used a simpler bound throughout.

as required, where the final inequality follows from local optimality with respect to 1-exchanges. Suppose now that (2.4) holds for all trees T with $|V(T)| \leq t < s$ and consider some tree T with $|V(T)| = t + 1$. As above, let $\hat{v}$ be a vertex of $V(T)$ with maximum weight and now let $T_1, \ldots, T_c$ be the connected components of $T[V(T) - \hat{v}]$ obtained by removing $\hat{v}$. Then, each T_i is a tree with $|V(T_i)| \leq t$ and the arcs incident to $\hat{v}$ in T are of the form $(t_1, \hat{v}), \ldots, (t_c, \hat{v})$, with $t_i \in V(T_i)$ for each $i = 1, \ldots, c$, (where the orientation of each arc follows from the fact that $w_{\hat{v}}$ is the largest weight in $V(T)$). Further let $\hat{v}_i = \arg\max_{v \in V(T_i)} w_v$. Then, local optimality with respect to s-exchanges implies that:

$$
\begin{aligned}
0 \leq \Delta_{V(T)} &= w^2\left(\bigcup_{v \in V(T)} N_v^+\right) - w^2\left(\bigcup_{v \in V(T)} C_v\right) \\
&\leq \sum_{i=1}^{c}\left[w^2\left(\bigcup_{v \in V(T_i)} N_v^+\right) - w^2\left(\bigcup_{v \in V(T_i)} C_v\right)\right] + w^2(N_{\hat{v}}^+) - w^2(C_{\hat{v}}) - \sum_{i=1}^{c} w_{t_i}^2 \\
&= \sum_{i=1}^{c}\left[\Delta_{T_i}\right] + \Delta_{\hat{v}} - \sum_{i=1}^{c} w_{t_i}^2 \\
&\leq \sum_{i=1}^{c}\left[\left(\sum_{v \in V(T_i)} \frac{\Delta_v}{w_v} - \sum_{(a,b) \in E(T_i)} \frac{w_a^2}{w_b}\right) w_{\hat{v}_i}\right] + \Delta_{\hat{v}} - \sum_{i=1}^{c} w_{t_i}^2 \\
&\leq \sum_{i=1}^{c}\left[\left(\sum_{v \in V(T_i)} \frac{\Delta_v}{w_v} - \sum_{(a,b) \in E(T_i)} \frac{w_a^2}{w_b}\right) w_{\hat{v}}\right] + \frac{\Delta_{\hat{v}}}{w_{\hat{v}}} \cdot w_{\hat{v}} - \sum_{i=1}^{c} \frac{w_{t_i}^2}{w_{\hat{v}}} \cdot w_{\hat{v}} \\
&= \left(\sum_{v \in V(T)} \frac{\Delta_v}{w_v} - \sum_{(a,b) \in E(T)} \frac{w_a^2}{w_b}\right) w_{\hat{v}}.
\end{aligned}
$$

Here, the second inequality follows from the fact that for each arc $e_i = (t_i, \hat{v})$ between T_i and $\hat{v}$, we have $t_i \in N_{\hat{v}}^+$ and $t_i \in N_{t_i}^+$. Thus, $w_{t_i}^2$ is counted in both $\bigcup_{v \in V(T_i)} w^2(N_v^+)$ and $w^2(N_{\hat{v}}^+)$ but only once in $\bigcup_{v \in V(T)} N_v^+$. Moreover, each element of O appears in at most 1 of the sets C_v and so $w^2\left(\bigcup_{v \in V(T)} C_v\right) = w^2(C_{\hat{v}}) + \sum_{i=1}^{c} w^2\left(\bigcup_{v \in V(T_i)} C_v\right)$. The third inequality follows from the second inequality of the induction hypothesis (2.4). The fourth inequality follows again from the first inequality of the induction hypothesis (2.4) and $w_{\hat{v}} \geq w_{v_i}$ for all i. This completes the induction step for the proof of (2.4). Rearranging (2.4), for any $V(T)$ with $|V(T)| \leq s$ we have:

$$
\sum_{v \in V(T)} \frac{\Delta_v}{w_v} \geq \sum_{(a,b) \in E(T)} \frac{w_a^2}{w_b}. \tag{2.5}
$$

□

Lemma 2.4.6 further refines Lemma 2.4.5. Given a tree $V(T) \subseteq D$, by adding the term Ψ for each vertex it proves that the gain can be improved. Compared to Lemma 2.4.5, the saving on edges of the tree now only depends on the weight of the smallest endpoint.

Lemma 2.4.6. *Let $\varepsilon \geq 0$ and suppose that T is a locally optimal tree in H_ε with $|V(T)| \leq s$. Let $B \subseteq A$ be any set of vertices such that H_ε contains no arc (u,v) or (v,u) between any $u \in B$ and any $v \in V(T)$. Then,*

$$\sum_{v \in V(T)} \left[\frac{\Delta_v + \Psi_v}{w_v} \right] \geq \sum_{(a,b) \in E(T)} w_a + \sum_{v \in V(T)} \sum_{o \in C_v} \varepsilon w(N(o,B)).$$

Proof of Lemma 2.4.6. Recall that for all $v \in V(T)$, we have

$$\begin{aligned}
\Psi_v &\triangleq \sum_{o \in C_v} (w_o - w_v)^2 + w_v w(N(o, A - v)) - w^2(N(o, A - v)) \\
&\geq \sum_{o \in C_v} \left[w_v w(N(o, T - v)) - w^2(N(o, T - v)) + w_v w(N(o, A \setminus T)) - w^2(N(o, A \setminus T)) \right] \\
&\geq \sum_{o \in C_v} \left[w_v w(N(o, T - v)) - w^2(N(o, T - v)) + w_v w(N(o, B)) - w^2(N(o, B)) \right] \\
&\geq \sum_{o \in C_v} \left[w_v w(N(o, T - v)) - w^2(N(o, T - v)) + \varepsilon w_v w(N(o, B)) \right].
\end{aligned}$$

The second line follows by distinguishing the neighbors of v into those that belong to T (where $v \in T$) and those that don't (where $v \in A \setminus T$). The third line is by positivity of the term: $w_v w(N(o, A \setminus T)) - w^2(N(o, A \setminus T))$, since for any $o \in C_v$ we have that $w_a \leq w_v$ for all $a \in N(o, A - v)$. Finally, the last line is by definition of the set B. In particular, for any $a \in N(o, B) \subseteq N(o, A - v)$ for some $o \in C_v$, we must have $w_a < (1 - \varepsilon) w_v$ since otherwise an arc (a, v) would be present in H_ε. Thus, for all $a \in N(o, B)$, $w_a w_v - w_a^2 \geq w_v w_a - (1 - \varepsilon) w_v w_a = \varepsilon w_v w_a$.

On the other hand, we observe that

$$\sum_{o \in C_v} \left[w_v w(N(o, T - v)) - w^2(N(o, T - v)) \right] \geq \sum_{u : (u,v) \in E(T)} w_v w_u - w_u^2.$$

Dividing through by w_v and summing over the vertices in the tree, we obtain that:

$$\sum_{v \in V(T)} \frac{\Psi_v}{w_v} \geq \sum_{(u,v) \in E(T)} \left[w_u - \frac{w_u^2}{w_v} \right] + \varepsilon \sum_{v \in V(T)} \sum_{o \in C_v} w(N(o, B)).$$

The claimed result then follows by combining the previous equation and Lemma 2.4.5. □

In Lemma 2.4.6, the slack from the s-exchanges is expressed using the arcs of the tree T. We show that this can in turn be bounded with respect to the total weight of all vertices of T by spreading the slack uniformly across the vertices.

Lemma 2.4.7. *Given a directed tree T of size $t > 2$ such that for any $(b,a) \in E(T)$ then $w_b \geq (1-\varepsilon)w_a$. Then, for any $1/2 \geq \varepsilon > 0$, we have*

$$\sum_{(a,b)\in E(T)} w_a \geq \frac{(t-1)(1-\varepsilon)}{t-\varepsilon} w(T). \tag{2.6}$$

A weaker estimate holds for any $\varepsilon > 0$. Let $\chi_\varepsilon^{(t)} \triangleq \frac{(t-1)}{t}(1-\varepsilon)$, then under the same conditions and any $\varepsilon \in (0,1)$ the following holds:

$$\sum_{(a,b)\in E(T)} w_a \geq \chi_\varepsilon^{(t)} w(T). \tag{2.7}$$

Proof of Lemma 2.4.7. We begin with the simpler proof of Equation (2.7). Let r be the vertex of minimum weight in T and set r as the root. Forgetting about the orientation of the arcs, we create a mapping $\sigma : E \to V$ that maps each arc to the children node contained in this arc. In this way, every vertex gets assigned an arc except the root. Using that for each arc $(a,b) \in E(T)$, we have that $w_a \geq (1-\varepsilon)w_b$ and summing over all arcs, we get

$$\sum_{(a,b)\in E(T)} w_a \geq (1-\varepsilon) \sum_{(a,b)\in E(T)} w_{\sigma((a,b))} = (1-\varepsilon)\left[\sum_{v\in T} w_v - w_r\right].$$

Since w_r has minimum weight, we have that $w_r \leq \frac{1}{t}\sum_{v\in T} w_v$ which yields the desired inequality.

We now prove Equation (2.6). Let r be a vertex of T with minimum weight, and fix some edge $(r,x) \in E(T)$. For each remaining arc $(a,b) \in E(T) - (r,x)$, consider the unique *undirected* path from r ending with (a,b), and let $v \in \{a,b\}$ be the vertex at the end of this path. Note that every vertex of $V(T) \setminus \{r,x\}$ serves as v for exactly one edge $(a,b) \in E(T) - (r,x)$. Moreover, if $v = a$, then $w_a = w_v$, and if $v = b$, then $w_a \geq (1-\varepsilon)w_b = (1-\varepsilon)w_v$. Let $z \triangleq \sum_{(a,b)\in E(T)-(r,x)} w_a$. Then, by the above discussion, $z \geq (1-\varepsilon)\sum_{v\in V(T)\setminus\{r,x\}} w_v$. Consider the function:

$$f(z) = \frac{w_r + z}{w_r + w_x + (1-\varepsilon)^{-1}z} \leq \frac{\sum_{(a,b)\in E(T)} w_a}{\sum_{v\in V(T)} w_v}.$$

To complete the proof it suffices to show that $f(z) \geq \frac{(t-1)(1-\varepsilon)}{t-\varepsilon}$. Observe that since $(r,x) \in E(T)$, $(1-\varepsilon)w_x \leq w_r \leq w_x$, and so $\frac{w_r}{w_r+w_x} \leq \frac{1}{2}$. Thus, for $\varepsilon \leq 1/2$, $f(z)$ is a non-decreasing function of z. Moreover, since w_r is the smallest weight of any vertex in T, $z \geq |E(T) - (r,x)|w_r = (t-2)w_r$. Thus,

$$\begin{aligned} f(z) &\geq \frac{w_r + (t-2)w_r}{w_r + w_x + (1-\varepsilon)^{-1}(t-2)w_r} \geq \frac{(t-1)w_r}{w_r + (1-\varepsilon)^{-1}w_r + (1-\varepsilon)^{-1}(t-2)w_r} \\ &= \frac{(t-1)(1-\varepsilon)}{t-\varepsilon}, \end{aligned}$$

where the second inequality follows again from $w_r \geq (1-\varepsilon)w_x$. □

The proof of Lemma 2.4.4 directly follows by combining Lemma 2.4.6 and 2.4.7. We restate it here for simplicity,

Lemma 2.4.4. *Let $0 \leq \varepsilon \leq 1/2$ and suppose that A is locally optimal under s-exchanges for $s \leq 1 + k(k-1)$. Then,*

$$\sum_{a \in D} \left[\frac{\Delta_a + \Psi_a}{w_a} \right] \geq \frac{1-\varepsilon}{2-\varepsilon} w(D) + \sum_{a \in D} \sum_{o \in C_a} \varepsilon w(N(o, I)).$$

Proof of Lemma 2.4.4. Recall that the collection of stars $T_1, \ldots, T_\ell$ has the property that each $a \in D$ appears in exactly one T_i and $2 \leq |V(T_i)| \leq k(k-1)+1 \leq s$. For each T_i, since A is locally optimal with respect to s-exchanges, Lemma 2.4.6 (with $B = I$) and Lemma 2.4.7, respectively, imply

$$\begin{aligned} \sum_{a \in V(T_i)} \left[\frac{\Delta_a}{w_a} + \frac{\Psi_a}{w_a} \right] &\geq \sum_{(a,b) \in E(T_i)} w_a + \sum_{a \in V(T_i)} \sum_{o \in C_a} \varepsilon w(N(o, I)) \\ &\geq \frac{1-\varepsilon}{2-\varepsilon} w(T_i) + \sum_{a \in V(T_i)} \sum_{o \in C_a} \varepsilon w(N(o, I)). \end{aligned}$$

Summing the resulting inequalities for each $i = 1, \ldots, \ell$ then gives the stated result. Note that the final inequality is tight only when $|V(T_i)| = 2$ for all T_i. □

2.4.3 Bounding the slack for isolated claws

Throughout the remainder of this chapter and the next one, it is helpful to consider the following quantity. For any $t \geq 0$ and $\delta \in (0, 1)$, we set the parameter:

$$\rho_{t,\delta} \triangleq t\delta - \frac{\delta}{1-\delta}. \tag{2.8}$$

In particular, for all $0 \leq t \leq k$, $\rho_{t,\delta} - t\delta = \rho_{k,\delta} - k\delta$. The parameter $\rho_{k,\delta}$ will be the amount of slack received by isolated vertices.

We now consider those claws C_a where $a \in I$ is an isolated vertex. Observe that for all such claws, we must have $w_v \leq (1-\varepsilon)w_a$ for every $v \in \bigcup_{o \in C_a} N(o, A-a)$, since otherwise there would be an edge (v, a) in H_ε. It follows that for all $a \in I$, and $o \in C_a$,

$$w^2(N(o, A-a)) \leq (1-\varepsilon) w_a w(N(o, A-a)). \tag{2.9}$$

In Lemma 2.4.8, we derive a bound for the slack of isolated claws.

Lemma 2.4.8. *Suppose that A is locally optimal with respect to 1-exchanges. Let $\delta \triangleq 1 - \sqrt{1-\varepsilon} \geq 0$. Then, for any $a \in I$,*

$$\frac{\Psi_a}{w_a} \geq (\rho_{k,\delta} - k\delta)\, w_a + \delta \sum_{o \in C_a} w(N(o, A)).$$

The crucial observation that Lemma 2.4.8 captures is that the slack $\rho_{k,\delta}$ for isolated vertices is non-trivial. The proof later shows that the term $-k\delta$ cancels out. While this observation may seem obvious, we point out a subtle difficulty in handling it. Suppose that we are given $a, b, c \in A$ and assume that a is isolated, and $b \in N_a^+$ and $a \in N_c^+$. By assumption, we have that $\Psi_c / w_c \geq \varepsilon w_a$. Intuitively, it means that we save an ε-fraction of a's weight for each such c. The issue is that the number of such c is equal to $|N(a, O \backslash C_a)|$ (using Reduction 1) and can be arbitrarily small. To handle this situation, we use that b's weight is greatly smaller than a's weight. Thus, vertex a will keep a fraction of the overall slack equal to $\rho_{k,\delta} w_a$, leaving a smaller saving equal to δw_b for vertices in $b \in N_a^+ \backslash \{a\}$, where $\delta \leq \varepsilon$. The parameter δ can be thought of as a way of dividing the slack from Ψ_a into one portion that pays for a and another that will pay for the neighbors.

Proof of Lemma 2.4.8. Fix $a \in I$. Since $a \in I$, Equation (2.9) implies that

$$\psi_{a,o} = (w_a - w_o)^2 + w_a w(N(o, A-a)) - w^2(N(o, A-a)) \geq (w_a - w_o)^2 + \varepsilon w_a w(N(o, A-a))$$

for every $o \in C_a$ and so

$$\Psi_a \geq \sum_{o \in C_a} (w_a - w_o)^2 + \varepsilon w_a \sum_{o \in C_a} w(N(o, A-a)).$$

Define $\alpha_o \triangleq w_o / w_a$ and $\beta \triangleq \sum_{o \in C_a} w(N(o, A-a))/w_a$. Then we can reformulate the above inequality as

$$\begin{aligned} \Psi_a \geq \sum_{o \in C_a} (w_a - \alpha_o w_a)^2 + \varepsilon \beta w_a^2 &= w_a^2 \left(\sum_{o \in C_a} (1-\alpha_o)^2 + \varepsilon\beta \right) \\ &= w_a^2 \left(\sum_{o \in C_a} (1-\alpha_o)^2 + (\delta - \delta^2)\beta \right) + w_a^2 \delta\beta. \end{aligned} \tag{2.10}$$

The second equation follows from the identity $\varepsilon = 2\delta - \delta^2$, which follows from our choice of δ. We now lower bound the bracketed expression on the right. Since A is locally optimal with respect to 1-exchanges, Equation (2.1) and (2.9) imply that

$$\sum_{o \in C_a} w_o^2 = w^2(C_a) \leq w^2(N_a^+) \leq w_a^2 + \sum_{o \in C_a} w^2(N(o, A-a)) \leq w_a^2 + \sum_{o \in C_a} w_a (1-\varepsilon) w(N(o, A-a)).$$

Reformulating this inequality in terms of the values α_o and β, gives us the following constraint:

$$\sum_{o \in C_a} \alpha_o^2 w_a^2 \leq w_a^2 + (1-\varepsilon) w_a^2 \beta = w_a^2 + (1-\delta)^2 w_a^2 \beta.$$

Dividing through by w_a^2 and then rearranging, we obtain $\beta \geq \frac{\left(\sum_{o \in C_a} \alpha_o^2\right) - 1}{(1-\delta)^2}$. Then,

$$\begin{aligned}
\sum_{o \in C_a} (1-\alpha_o)^2 + (\delta - \delta^2)\beta &\geq \sum_{o \in C_a} (1-\alpha_o)^2 + \frac{\delta}{1-\delta}\left(\left(\sum_{o \in C_a} \alpha_o^2\right) - 1\right) \\
&= |C_a| - 2\sum_{o \in C_a} \alpha_o + \sum_{o \in C_a} \alpha_o^2 + \frac{\delta}{1-\delta}\left(\left(\sum_{o \in C_a} \alpha_o^2\right) - 1\right) \\
&= |C_a| - 2\sum_{o \in C_a} \alpha_o + \frac{1}{1-\delta}\sum_{o \in C_a} \alpha_o^2 - \frac{\delta}{1-\delta} \\
&\geq |C_a| - 2\sum_{o \in C_a} \alpha_o + \frac{1}{1-\delta} \cdot \frac{1}{|C_a|}\left(\sum_{o \in C_a} \alpha_o\right)^2 - \frac{\delta}{1-\delta}, \qquad (2.11)
\end{aligned}$$

where the last inequality follows from Cauchy-Schwarz (Theorem 1.B.3). We can express the above lower bound as $f(x)$ where $f(x) = \frac{1}{|C_a|(1-\delta)}x^2 - 2x + |C_a| - \frac{\delta}{1-\delta}$ and $x = \sum_{o \in C_a} \alpha_o$. Then, note that $\frac{d^2 f}{dx^2} = \frac{2}{|C_a|(1-\delta)} > 0$, so f is convex in x, and when $x = |C_a|(1-\delta)$, we have $\frac{df}{dx} = 0$. Thus,

$$\begin{aligned}
&|C_a| - 2\sum_{o \in C_a} \alpha_o + \frac{1}{1-\delta} \cdot \frac{1}{|C_a|}\left(\sum_{o \in C_a} \alpha_o\right)^2 - \frac{\delta}{1-\delta} \\
&\qquad = f\left(\sum_{o \in C_a} \alpha_o\right) \geq f\left(|C_a|(1-\delta)\right) = |C_a|\,\delta - \frac{\delta}{1-\delta} = \rho_{|C_a|,\delta}. \qquad (2.12)
\end{aligned}$$

Combining the inequalities (2.10), (2.11), and (2.12) we finally have

$$\frac{\Psi_a}{w_a} \geq \frac{\rho_{|C_a|,\delta} w_a^2 + \delta\beta w_a^2}{w_a} = \rho_{|C_a|,\delta} w_a + \delta \sum_{o \in C_a} w(N(o, A-a)).$$

The last step of the proof is by observing that $\sum_{o \in C_a} w(N(o, A-a)) = \sum_{o \in C_a} w(N(o, A)) - |C_a|$, and that $\rho_{t,\delta} - t\delta = \rho_{k,\delta} - k\delta$ for all $0 \leq t \leq k$, c.f. Equation (2.8). □

2.4.4 Combining the Bounds

We now combine the bounds on the slack for isolated and non-isolated claws given by Lemmas 2.4.7 and 2.4.8 with Lemma 2.3.1 to obtain a guarantee for Algorithm 1. Theorem 2.4.9 proves that the approximation factor is a trade-off between the slack that isolated and non-isolated claws receive.

Theorem 2.4.9. *For any $0 \leq \varepsilon \leq 1/2$ and $\delta = 1 - \sqrt{1-\varepsilon}$, if A is locally optimal under s-exchanges for $s \geq 1 + k(k-1)$ then,*

$$2w(O) \leq \left[k + 1 - \min\left\{\frac{1-\varepsilon}{2-\varepsilon}, \rho_{k,\delta}\right\}\right] w(A).$$

Proof of Theorem 2.4.9. Observe that $\delta \leq \varepsilon$. Then, by Lemma 2.4.4,

$$\sum_{a \in D}\left[\frac{\Delta_a + \Psi_a}{w_a}\right] \geq \frac{1-\varepsilon}{2-\varepsilon} w(D) + \sum_{a \in D}\sum_{o \in C_a} \varepsilon w(N(o,I)) \geq \frac{1-\varepsilon}{2-\varepsilon} w(D) + \sum_{a \in D}\sum_{o \in C_a} \delta w(N(o,I))$$

By Lemma 2.4.8, and since $\Delta_a \geq 0$ for all vertices, we have

$$\sum_{a \in I}\left[\frac{\Delta_a + \Psi_a}{w_a}\right] \geq (\rho_{k,\delta} - k\delta)\, w(I) + \delta \sum_{a \in I}\sum_{o \in C_a} w(N(o,A)).$$

Combining the 2 bounds above and recalling that the sets D, I partition A, and that every $o \in O$ appears in C_a for exactly one $a \in A$, we obtain

$$\begin{aligned}\sum_{a \in A}\left[\frac{\Delta_a + \Psi_a}{w_a}\right] &\geq \frac{1-\varepsilon}{2-\varepsilon} w(D) + (\rho_{k,\delta} - k\delta)\, w(I) + \sum_{a \in A}\sum_{o \in C_a} \delta w(N(o,I)) \\ &= \frac{1-\varepsilon}{2-\varepsilon} w(D) + (\rho_{k,\delta} - k\delta) w(I) + \sum_{o \in O} \delta w(N(o,I)).\end{aligned}$$

Substituting the above expression in Berman's final guarantee (Lemma 2.3.1) we then have:

$$\begin{aligned}2w(O) &\leq w(A) + \sum_{o \in O} w(N(o,A)) - \frac{1-\varepsilon}{2-\varepsilon} w(D) - (\rho_{k,\delta} - k\delta) w(I) - \sum_{o \in O} \delta w(N(o,I)) \\ &= w(A) + \sum_{o \in O} w(N(o,D)) + (1-\delta) \sum_{o \in O} w(N(o,I)) - \frac{1-\varepsilon}{2-\varepsilon} w(D) - (\rho_{k,\delta} - k\delta) w(I) \\ &\leq w(A) + k w(D) + k(1-\delta) w(I) - \frac{1-\varepsilon}{2-\varepsilon} w(D) - (\rho_{k,\delta} - k\delta) w(I) \\ &= (k+1) w(A) - \frac{1-\varepsilon}{2-\varepsilon} w(D) - \rho_k w(I) \leq (k+1) w(A) - \min\left\{\frac{1-\varepsilon}{2-\varepsilon}, \rho_{k,\delta}\right\} w(A).\end{aligned}$$

The second inequality is by $(k+1)$-claw freeness of the conflict graph. Thus, each vertex in the current solution conflicts with at most k vertices of O. □

We conclude this section with Theorem 2.4.10 that states the exact approximation factor.

Theorem 2.4.10. *Algorithm 1 with exchanges of size $s \leq k(k-1)+1$ has approximation ratio $\frac{k+1-\tau_k}{2}$ where $\tau_k = \min_{\varepsilon \in (0,1/2)}\left\{\frac{1-\varepsilon}{2-\varepsilon}, k\delta - \frac{\delta}{1-\delta}\right\}$ where $\delta = 1 - \sqrt{1-\varepsilon}$. For $k = 3$, it yields an approximation factor of 1.811. As k increases, τ_k increases and $\lim_{k\to\infty} \tau_k = 1/2$.*

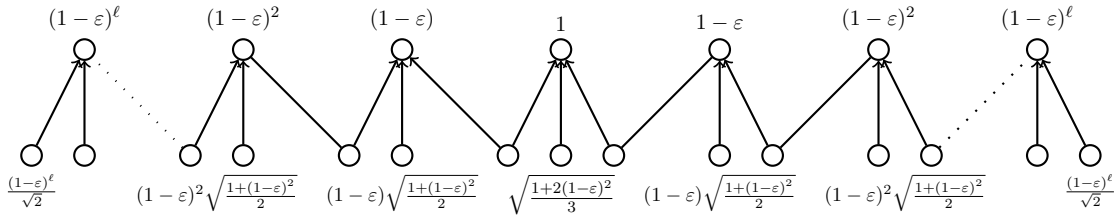


Fig. 2.3: Almost tight example for our analysis, where the vertices at the top are the vertices in the current solution, and vertices at the bottom are the vertices in the optimal solution. The values written are for individual vertices.

Proof of Theorem 2.4.10. The approximation factor is a straightforward consequence of Theorem 2.4.9. For $k = 3$, setting $\varepsilon = 0.3918$, we have $2 - \frac{1}{2} \cdot \frac{1-\varepsilon}{2-\varepsilon} \leq 1.81092$, whereas $2 - \frac{1}{2} \cdot \rho_{3,1-\sqrt{1-\varepsilon}} \leq 1.81095$. The increase in τ_k directly follows from the definition of $\rho_{k,\delta}$ which is an increasing function of k for a fixed value of δ.

To prove the asymptotic convergence for large k, we can simply set $\varepsilon = 1 - \left(\frac{k-1}{k}\right)^2$ and $\delta = 1 - \sqrt{1-\varepsilon} = 1/k$. On the one hand, we observe that: $\frac{1-\varepsilon}{2-\varepsilon} \leq 1/2$, whereas

$$\rho_{k,1/k} = k \cdot \frac{1}{k} - \frac{\frac{1}{k}}{1-\frac{1}{k}} = 1 - \frac{1}{k-1} \to_{k\to\infty} 1.$$

Thus, for large values of k, the first term $\frac{1-\varepsilon}{2-\varepsilon}$ is the bottleneck and the approximation ratio converges to $\frac{k+1-1/2}{2}$. □

2.4.5 A matching lower bound

Here we give a small example to show that novel ideas have to be incorporated in order to improve our analysis from Section 2.4. This analysis leads to a factor of 1.81 when $k = 3$, by balancing the improvement $\frac{1-\varepsilon}{2-\varepsilon}$ obtained for non-isolated vertices with the improvement $\rho_{3,1-\sqrt{1-\varepsilon}}$ obtained for isolated vertices. This leads to a value $\varepsilon \approx 0.3918$ (see Table 2.2).

The example shown in Figure 2.3 provides an almost tight example of our analysis, up to an error of 0.02 in the approximation. The example consists of a central vertex with 3 vertices of O mapped to it by π. We connect this central vertex by two paths of vertices all of which are connected to 2 vertices of O in the mapping π. The weights of the vertices are set so each vertex in A is isolated in $H_{\varepsilon'}$ for some ε' infinitesimally larger than ε. Thus, in our analysis, we will consider each claw as a single swap. The weights of the vertices in OPT are fixed so that $\Delta_a = 0$ for all $a \in A$. Note that our example is *not* a tight example for Algorithm 1 since there is an improving 2-exchange. However, as we will show this example implies that to make further progress we need to either consider larger swaps involving isolated vertices, or improve on the bound $\frac{1-\varepsilon}{2-\varepsilon}$ for non-isolated vertices, allowing us to increase ε in our final analysis. The approximation ratio of Figure 2.3 is equal to:

$$\frac{w(O)}{w(A)} = \frac{3\sqrt{\frac{1+2(1-\varepsilon)^2}{3}} + 4\sqrt{\frac{1+(1-\varepsilon)^2}{2}}\sum_{i=1}^{\ell-1}(1-\varepsilon)^i + 4\frac{(1-\varepsilon)^\ell}{\sqrt{2}}}{1+2\sum_{i=1}^{\ell}(1-\varepsilon)^i},$$
$$\underset{\ell\to\infty}{\to} \frac{3\sqrt{\frac{1+2(1-\varepsilon)^2}{3}} + 4\sqrt{\frac{1+(1-\varepsilon)^2}{2}}(\varepsilon^{-1}-1)}{2\varepsilon^{-1}-1}.$$

For $\varepsilon = 0.3918$, the value of the previous ratio is equal to $\simeq 1.80857$. In contrast, for the same value of ε, the bound obtained for non-isolated vertices is equal to $2 - \frac{1-\varepsilon}{2(2-\varepsilon)} = 1.81091$.

Figure 2.3 demonstrates that minor modifications of our current analysis cannot beat a factor of 1.8. This example captures the tension that the variable ε faces. On the one hand, the approximation factor of Figure 2.3 decreases as ε increases. But, as ε increases the bound for the exchange, i.e., $\frac{1-\varepsilon}{2-\varepsilon}$, decreases. This suggests that to surpass the 1.8 factor, we must either improve our bound for non-isolated vertices, or extend our techniques to combine isolated vertices into multiple swaps. We adopt the first approach in the next section, and the latter approach in Chapter 3

2.5 Further improving the bound

An important bottleneck in our analysis of non-isolated claws occurs when $|V(T_i)| = 2$ for all T_i. In this case, each star is simply an isolated edge. Lemma 2.4.7 implies that if we could ensure a partition of the H_ε into larger connected components of size at least $|V(T_i)| \geq \ell$ for all T_i, then we could improve the gain of $\frac{1-\varepsilon}{2-\varepsilon}$ we obtain for non-isolated claws to $\frac{\ell-1}{\ell}(1-\varepsilon) \to_{\ell\to\infty} (1-\varepsilon)$. However, this is not possible when H_ε contains *maximal* connected components of size 2. Formally, we define an isolated component as follows:

Definition 2.5.1. An *isolated component* of size t is a maximal connected component in H_ε of size t, where the connectivity is taken forgetting about the orientation of the arcs.

Observe that an isolated component of size 1 is an *isolated vertex* and agrees with the definition from Section 2.4. An isolated component C has the property that arcs leaving or entering C have the ratio of the weight of the endpoints bounded away from 1.

Proposition 2.5.2. *Let C be an isolated component of size t. Given $a \in C$ and $b, c \notin C$ such that $b \in N_a^+$ and $a \in N_c^+$, we have that $w_b < (1-\varepsilon)w_a$ and $w_a < (1-\varepsilon)w_c$.*

We will combine the basic techniques from the previous section to show that the total slack received by vertices inside such component is at least as large as the slack received by isolated vertices. Using larger swaps, we will ensure that the remaining non-isolated vertices of H_ε can be partitioned into trees T_i of size at least ℓ. Lemma 2.5.3 computes the slack in an isolated component.

Lemma 2.5.3. *Let T be an isolated component of size t, where t is some constant. Then,*

$$\sum_{v\in T}\left[\frac{\Delta_v + \Psi_v}{w_v}\right] \geq \left(\rho_{k,\delta} - k\delta + \frac{t-1}{t}(1-\delta)^5\right) w(T) + \delta \sum_{v\in T}\sum_{o\in C_v} w(N(o,A)).$$

Observe that the slack increases as the size of the connected component increases and matches that of isolated vertices for $t = 1$. We advise the reader to jump directly to Section 2.5.1 for an application of the lemma. For the proof of Lemma 2.5.3, we introduce two variables which we denote $\theta_{a,o}$ and Θ_a, respectively. For each $a \in A$, and $o \in C_a$, we let

$$\theta_{a,o} \triangleq w_a w(N(o, A-a)) - w^2(N(o, A-a)), \qquad \text{and} \qquad \Theta_a \triangleq \sum_{o \in C_a} \theta_{a,o}.$$

The variable Θ_a is the slack induced by the difference of weights between endpoints of arcs that have a as head. We will use these notations in Chapter 3 as well. Therefore, $\Psi_a \triangleq \sum_{o \in C_a} \left[(w_o - w_a)^2 + \theta_{a,o}\right] = \sum_{o \in C_a} (w_o - w_a)^2 + \Theta_a$. Moreover, for a subset $Y \subseteq A$, we let $\theta_{a,o}(Y) \triangleq w_a w(N(o, Y-a)) - w^2(N(o, Y-a))$, and $\Theta_v(Y) \triangleq \sum_{o \in C_v} \theta_{v,o}(Y)$.

Proof of Lemma 2.5.3. Lemma 2.5.3 is an extension of Lemma 2.4.8. The proof follows the same ideas. We are given an isolated component T of size t, which we may assume to be a tree by Lemma 2.4.2/Reduction 1. By Proposition 2.5.2, we know that for any $v \in T$ and $u \in N_v^+ \setminus T$, we have that $\frac{w_u}{w_v} \leq 1 - \varepsilon$. For each vertex, we decompose $\theta_{v,o}(A) \triangleq \theta_{v,o}(T) + \theta_{v,o}(A \setminus T)$ into the slack induced by arcs inside the tree and arcs entering the tree that have v as head. The proof splits $\theta_{v,o}(A \setminus T)$ into two parts. Some amount is kept at v, whereas some amount is distributed to $N_v^+ \setminus T$. More precisely, we let $\beta_{v,o} = w(N(o, A \setminus T))$ for $o \in C_v$, and define $\beta \triangleq \sum_{v \in T} \sum_{o \in C_v} \beta_{v,o}$. From Proposition 2.5.2, we get that:

$$\begin{aligned} \frac{\theta_{v,o}(A \setminus T)}{w_v} &= w(N(o, A \setminus T)) - \frac{w^2(N(o, A \setminus T))}{w_v}, \\ &\geq \varepsilon w(N(o, A \setminus T)) \\ &= \delta(1-\delta)\beta_{v,o} + \delta\beta_{v,o}, \end{aligned}$$

where the third line follows from the definition of $\varepsilon = 2\delta - \delta^2$. Summing over the all $o \in C_v$ and $v \in T$, we get that:

$$\sum_{v \in T} \sum_{o \in C_v} \frac{\theta_{v,o}(A \setminus T)}{w_v} \geq \delta(1-\delta)\beta + \delta\beta. \tag{2.13}$$

Focusing on the quantity $\theta_{v,o}(T)$, we get:

$$\sum_{v \in T} \sum_{o \in C_v} \frac{\theta_{v,o}(T)}{w_v} = \sum_{v \in T} \sum_{o \in C_v} \left[w(N(o, T-v)) - \frac{w^2(N(o, T-v))}{w_v} \right] \geq \sum_{(u,v) \in E(T)} \left[w_u - \frac{w_u^2}{w_v} \right]. \tag{2.14}$$

The inequality is by positivity of each squared bracket term since each vertex in $N(o, T-v)$ has weight smaller than v. In the remaining of the computation, we want to bound $\sum_{v \in T} \Psi_v / w_v$. This is done as in Theorem 2.4.8. In particular, since T is an isolated component of size t,

we obtain a lower estimate of the weight of its neighborhood (i.e., β) given that T is locally optimal with respect to w^2. Thus,

$$w^2(N_v^+ \setminus T) \leq (1-\varepsilon) w_v w(N_v^+ \setminus T) \leq (1-\delta)^2 w_v \sum_{o \in C_v} \beta_{v,o}.$$

On the other hand, $w^2(N_v^+ \cap T) = w_v^2 + \sum_{u:(u,v)\in E(T)} w_u^2$, where the equality holds by Lemma 2.4.2, since the induced subgraph $T \subseteq H$ is a tree. Substituting both expressions, we obtain

$$\begin{aligned}
\sum_{v \in T} \frac{\Delta_v}{w_v} &= \sum_{v \in T} \frac{w^2(N_v^+) - w^2(C_v)}{w_v} \\
&= \sum_{v \in T} \frac{w^2(N_v^+ \cap T)}{w_v} + \frac{w^2(N_v^+ \setminus T)}{w_v} - \frac{w^2(C_v)}{w_v} \\
&\leq (1-\delta)^2 \beta + \sum_{(u,v)\in E(T)} \frac{w_u^2}{w_v} + \sum_{v \in T} \left[w_v - \frac{w^2(C_v)}{w_v} \right].
\end{aligned}$$

Lemma 2.4.5 states that $\sum_{v\in T} \frac{\Delta_v}{w_v} \geq \sum_{(u,v)\in E(T)} \frac{w_u^2}{w_v}$. By applying it on the left-hand side, we get that

$$\beta \geq \frac{1}{(1-\delta)^2} \sum_{v \in T} \left[\frac{w^2(C_v)}{w_v} - w_v \right]. \tag{2.15}$$

The previous computation gives us a lower bound on β which will be helpful to bound Ψ_v away from 0 for vertices in the tree. We focus on the amount of slack induced by arcs that have one endpoint of the tree as head and the tail outside T. We introduce the variables $\alpha_{v,o} \triangleq \frac{w_o}{w_v}$ that are normalized weights for the set of talons, and let $\alpha_v = \sum_{o\in C_v} \alpha_{v,o}$. Applying Equation (2.13) and (2.15) we have

$$\begin{aligned}
&\sum_{v \in T} \sum_{o \in C_v} \left[\frac{1}{w_v} (w_v - w_o)^2 + \frac{\theta_{v,o}(A \setminus T)}{w_v} \right] \\
&= \sum_{v \in T} \left(\frac{w^2(C_v)}{w_v} - 2w(C_v) + |C_v|\, w_v \right) + \delta(1-\delta)\beta + \delta\beta \\
&\geq \sum_{v \in T} \left(\frac{w^2(C_v)}{w_v} - 2w(C_v) + |C_v|\, w_v \right) + \frac{\delta}{1-\delta} \sum_{v \in T} \left[\frac{w^2(C_v)}{w_v} - w_v \right] + \delta\beta \\
&= \sum_{v \in T} w_v \left(\frac{1}{1-\delta} \sum_{o \in C_v} \alpha_{v,o}^2 - 2 \sum_{o \in C_v} \alpha_{v,o} + |C_v| - \frac{\delta}{1-\delta} \right) + \delta\beta \\
&\geq \sum_{v \in T} w_v \left(\frac{1}{(1-\delta)\,|C_v|} \alpha_v^2 - 2\alpha_v + |C_v| - \frac{\delta}{1-\delta} \right) + \delta\beta.
\end{aligned}$$

The last inequality follows from Cauchy-Schwarz (Theorem 1.B.3). We bound each bracketed expression for each vertex in the tree. Each bracketed expression is a function of α_v that

attains it minimum at $\alpha_v^* = (1-\delta)\,|C_v|$. Substituting each α_v by the corresponding minimum value implies that

$$\sum_{v\in T}\sum_{o\in C_v}\left[\frac{1}{w_v}(w_v - w_o)^2 + \frac{\theta_{v,o}(A\setminus T)}{w_v}\right] \geq \sum_{v\in T}\rho_{|C_v|,\delta}w_v + \delta\beta \tag{2.16}$$

Next, we unwind the definition of β. We use the following observation in the following computations: Given a vertex $u \in N_v^+ \setminus T$, the number of vertices $v \in T$ such that $u \in N_v^+$ is exactly 1. Otherwise, it induces a cycle of length at most $2t+2$ in the conflict graph $G[A \cup O]$ which is a contradiction to Reduction 1. Thus,

$$\beta = \sum_{v\in T}\sum_{o\in C_v} w(N(o, A\setminus T)) = \sum_{v\in T}\sum_{o\in C_v} w(N(o,A)) - |C_v|\,w_v - \sum_{(u,v)\in E(T)} w_u.$$

Substituting it into Equation (2.16), and using that $\rho_{m,\delta} - m\delta = \rho_{k,\delta} - k\delta$, for any $0 \leq m \leq k$, we get

$$\begin{aligned}&\sum_{v\in T}\sum_{o\in C_v}\left[\frac{1}{w_v}(w_v - w_o)^2 + \frac{\theta_{v,o}(A\setminus T)}{w_v}\right]\\ \geq &\sum_{v\in T}(\rho_{k,\delta} - k\delta)\,w_v - \delta\sum_{(u,v)\in E(T)} w_u + \delta\sum_{v\in T}\sum_{o\in C_v} w(N(o,A)).\end{aligned} \tag{2.17}$$

Finally, we consider the slack induced by the Δ's and the arcs inside the tree equal to: $\sum_{v\in T}\left[\frac{\Delta_v + \Theta_v(T)}{w_v}\right]$. By Lemma 2.4.5 and Equation (2.14), we get that

$$\sum_{v\in T}\left[\frac{\Delta_v + \Theta_v(T)}{w_v}\right] \geq \sum_{(u,v)\in E(T)}\frac{w_u^2}{w_v} + \sum_{(u,v)\in E(T)}\left[w_u - \frac{w_u^2}{w_v}\right] = \sum_{(u,v)\in E(T)} w_u. \tag{2.18}$$

Adding both Equation (2.17) and (2.18), we get that

$$\sum_{v\in T}\left[\frac{\Delta_v + \Psi_v}{w_v}\right] \geq (\rho_{k,\delta} - k\delta)\,w(T) + (1-\delta)\sum_{(u,v)\in E(T)} w_u + \delta\sum_{v\in T}\sum_{o\in C_v} w(N(o,A)).$$

The result then follows by applying Lemma 2.4.7 to the second term and using that $(1-\varepsilon) = (1-\delta)^2$. □

2.5.1 Large connected components

In the previous section, we computed the slack for maximal connected components in H_ε of size t. For an *isolated* component of size up to $\ell - 1$, we will apply Lemma 2.5.3. We now denote by I the set of vertices that belong to some isolated component of size up to $\ell - 1$. Let $D \triangleq A \setminus I$ be the set of vertices that belong to maximal connected component of size at least ℓ. We partition D into vertex disjoint trees of size at least ℓ that are locally optimal w.r.t w^2:

Constructing exchanges: Take the graph $G[D] = (V(D), E(D)) \subseteq H_\varepsilon$. We proceed as follows: if there is a path of length $2\ell - 1$, then we remove the ℓ^{th} edge of this path. We continue this procedure in the graph with an edge remove. The procedure terminates when no path of length $2\ell - 1$ is present. Observe that the removal of an edge potentially creates two connected components with at least ℓ connected vertices on each side. Since there is a finite number of edges, the procedure must terminate and results in a collection of vertex disjoint connected components of size at least ℓ. We prove that the number of vertices in each connected component is a tree of bounded size. Let T be a set obtained using this procedure. Then, the longest path in T has size $2\ell - 2$. By Lemma 2.4.2 which follows from Reduction 1 applied with $m = \ell$, the set T must be a tree. Each vertex in T has degree at most k^2, thus $|T| \leq \sum_{n=0}^{2\ell-2} k^{2n}$. In Algorithm 1, we will consider exchanges of size $s = k^{2\ell} \geq \sum_{n=0}^{2\ell-2} k^{2n}$. Applying the above procedure, we partition D into bounded size vertex disjoint trees with at least ℓ vertices that are locally optimal with respect to s-exchanges.

A straightforward corollary of Lemma 2.4.6 together with Lemma 2.4.7 yields the following result.

Lemma 2.5.4. *Let T be a connected component of size at least ℓ in H_ε that is locally optimal. Let $B \subseteq A$ be any set of vertices such that H_ε contains no arc (u, v) or (v, u) between any $u \in B$ and any $v \in T$*

$$\sum_{v \in T} \frac{\Delta_v + \Psi_v}{w_v} \geq \chi_\varepsilon^{(\ell)} w(T) + \varepsilon \sum_{v \in T} \sum_{o \in C_v} w(N(o, B)),$$

where we recall that $\chi_\varepsilon^{(\ell)} \triangleq \frac{\ell-1}{\ell}(1 - \varepsilon)$.

The main difference with Lemma 2.4.4 is the improvement in the factor $\frac{1-\varepsilon}{2-\varepsilon}$ which is now $\frac{\ell-1}{\ell}(1 - \varepsilon)$ and asymptotically approaches $(1 - \varepsilon)$ as ℓ tends to ∞. Together Lemma 2.5.3 and 2.5.4 are sufficient to obtain an improvement for all values of k when considering larger swaps. Theorem 2.5.5 proves that the approximation ratio is a trade-off between the slack that isolated vertices and large connected components receive.

Theorem 2.5.5. *Set $\varepsilon = 2\delta - \delta^2$. If A is locally optimal under s-exchanges for $s \geq k^{2\ell}$ then,*

$$2w(O) \leq \left[k + 1 - \min\left\{\chi_\varepsilon^{(\ell)}, \rho_{k,\delta}\right\}\right] w(A).$$

Proof of Theorem 2.5.5. By definition of ε, we have that $\delta \in (0, 1)$ and $\delta \leq \varepsilon$. We partitioned the set of vertices into "non-isolated components" of size at least ℓ, and isolated connected components of size $t = 1, \ldots, \ell - 1$. Recall that I is the set of vertices which belong to an

isolated component of size at most $\ell - 1$. Given an isolated component T of size t, we apply Lemma 2.5.3 and obtain that:

$$\sum_{v \in T} \left[\frac{\Delta_v + \Psi_v}{w_v} \right] \geq \left(\rho_{k,\delta} - k\delta + \frac{t-1}{t}(1-\delta)^5 \right) w(T) + \delta \sum_{v \in T} \sum_{o \in C_v} w(N(o, A))$$
$$\geq (\rho_{k,\delta} - k\delta)\, w(T) + \delta \sum_{v \in T} \sum_{o \in C_v} w(N(o, I)).$$

The inequality uses that $\frac{t-1}{t}(1-\delta)^5$ is an increasing function of $t \geq 1$ and that $I \subseteq A$. Now, given a connected component $T \subseteq D$ from the decomposition of size at least ℓ, we apply Lemma 2.5.4 with $B = I$ and get that

$$\sum_{v \in T} \frac{\Delta_v + \Psi_v}{w_v} \geq \chi_\varepsilon^{(\ell)} w(T) + \varepsilon \sum_{v \in T} \sum_{o \in C_v} w(N(o, I))$$
$$\geq \chi_\varepsilon^{(\ell)} w(T) + \delta \sum_{v \in T} \sum_{o \in C_v} w(N(o, I)),$$

where the second inequality uses that $\delta \leq \varepsilon$. Summing over all isolated components and all connected components of size at least ℓ which altogether partition A, we have that

$$\sum_{v \in A} \left[\frac{\Delta_v + \Psi_v}{w_v} \right] \geq (\rho_{k,\delta} - k\delta)\, w(I) + \chi_\varepsilon^{(\ell)} w(D) + \delta \sum_{v \in A} \sum_{o \in C_v} w(N(o, I))$$
$$= (\rho_{k,\delta} - k\delta)\, w(I) + \chi_\varepsilon^{(\ell)} w(D) + \delta \sum_{o \in O} w(N(o, I))$$

The second equation is because $\{C_v\}_{v \in A}$ partition the optimal solution. Using this bound in Lemma 2.3.1 we finally obtain:

$$2w(O) \leq w(A) + \sum_{o \in O} w(N(o, A)) - (\rho_{k,\delta} - k\delta)\, w(I) - \chi_\varepsilon^{(\ell)} w(D) - \delta \sum_{o \in O} w(N(o, I))$$
$$= w(A) + \sum_{o \in O} w(N(o, D)) + (1-\delta) \sum_{o \in O} w(N(o, I))$$
$$- \chi_\varepsilon^{(\ell)} w(D) - (\rho_{k,\delta} - k\delta)\, w(I).$$

Using the fact that for any $B \subseteq A$, each $a \in B$ appears in $N(o, B)$ for at most k distinct values of o, we get that: $\sum_{o \in O} w(N(o, D)) \leq k w(D)$ and $\sum_{o \in O} w(N(o, I)) \leq k w(I)$. Replacing the above bounds in the previous computation, we obtain the desired result

$$2w(O) \leq w(A) + k w(D) + k(1-\delta) w(I) - \chi_\varepsilon^{(\ell)} w(D) - (\rho_{k,\delta} - k\delta)\, w(I)$$
$$= w(A) + k w(A) - \chi_\varepsilon^{(\ell)} w(D) - \rho_{k,\delta} w(I)$$
$$\leq (k+1) w(A) - \min\left\{ \chi_\varepsilon^{(\ell)},\, \rho_{k,\delta} \right\} w(A). \qquad \square$$

We conclude this section with Theorem 2.5.6 that states the exact approximation for Algorithm 1 using large exchanges.

Theorem 2.5.6. *Algorithm 1 with exchanges of size $s = k^{2\ell}$ has approximation ratio $\frac{k+1-\tau_k}{2}$ where $\tau_k = \min_{\varepsilon\in(0,1)}\left\{\chi_\varepsilon^{(\ell)}, k\delta - \frac{\delta}{1-\delta}\right\}$, where $\varepsilon = 2\delta - \delta^2$. For $k = 3$, we obtain an approximation factor of $1.761 + \epsilon'$, for $\ell = O(1/\epsilon')$ with $\epsilon' > 0$. As k increases, τ_k increases and $\lim_{k\to\infty} \tau_k = 1 - \epsilon'$.*

Proof of Theorem 2.5.6. The first part of the theorem follows from Theorem 2.5.5 using that $\varepsilon = 2\delta - \delta^2$. For $k = 3$, setting $\varepsilon = 0.5208$, we have $\chi_\varepsilon^{(\ell)} \triangleq 2 - \frac{1}{2}\cdot\frac{\ell-1}{\ell}(1-\varepsilon) = 2 - \frac{1}{2}\cdot(1-O(\epsilon'))(1-\varepsilon) = 2 - 0.2396(1-O(\epsilon')) = 1.7604 + O(\epsilon')$. The proof then follows with an appropriate rescaling of ϵ'. The second term in the bracketed expression is equal to $\rho_{3,1-\sqrt{1-0.5208}} \leq 0.4787$, where $\delta \triangleq 1 - \sqrt{1-\varepsilon}$. In this case, the approximation factor is at most 1.7607.

To prove the asymptotic convergence for large k, we can simply set $\delta = 1/k$. We observe that $\chi_\varepsilon^{(\ell)} \triangleq \frac{\ell-1}{\ell}(1-\delta)^2 = (1-O(\epsilon'))\left(1-\frac{1}{k}\right)^2$ whose value tends to $1 - O(\epsilon')$ as k tends to infinity. On the other hand,

$$\rho_{k,1/k} = k\cdot\frac{1}{k} - \frac{\frac{1}{k}}{1-\frac{1}{k}} = 1 - \frac{1}{k-1} \to_{k\to\infty} 1.$$

Thus, for large values of k and an appropriate rescaling of ε', the approximation ratio converges to $\frac{k+1-(1-\epsilon')}{2} = \frac{k+\epsilon'}{2}$. □

2.5.2 Numerical results for small values

In the previous sections, we have shown how to translate local optimality with respect s-exchanges into guarantees depending on a given parameter ε. Here, we give concrete guarantees for various values of k. The asymptotic behavior is quantified in Theorem 2.4.10 and 2.5.6 respectively. The exact numbers for the value of ε and the improvement over the factor $\frac{k+1}{2}$ are displayed in Table 2.2.

Swap Size:	$k^2(k-1)+1$			$k^{O(1/\epsilon')}$		
k	$\tau_k/2$	APX	ε	$\tau_k/2$	APX	ε
3	0.189	1.811	0.3918	0.239	$1.761 + \epsilon'$	0.5208
4	0.210	2.290	0.2753	0.302	$2.199 + \epsilon'$	0.3955
5	0.219	2.781	0.2144	0.337	$2.663 + \epsilon'$	0.3249
6	0.225	3.275	0.1759	0.361	$3.139 + \epsilon'$	0.2771
7	0.229	3.771	0.1494	0.378	$3.622 + \epsilon'$	0.2421
8	0.232	4.268	0.1298	0.392	$4.108 + \epsilon'$	0.2152
9	0.234	4.766	0.1148	0.401	$4.598 + \epsilon'$	0.1939
10	0.236	5.264	0.1029	0.411	$5.089 + \epsilon'$	0.1764

Tab. 2.2: Optimal settings for ε and approximation ratio for different values of k. Here, $\tau_k/2$, measures the improvement over $\frac{k+1}{2}$. We recall that $\ell = O(1/\epsilon')$ controls the size of the swaps we consider.

2.5.3 Bounding on the number of swaps performed by Algorithm 1

In all of our preceding analysis, we have relied only on local optimality of the set A produced by Algorithm 1, without considering the time required to converge to such a local optimum. Here, we show that the weight-scaling argument used by Berman [Ber00], together with one round of partial enumeration, can be combined with our results to obtain a polynomial time algorithm. We first briefly review the general weight-scaling approach used in [Ber00].

Suppose that any A that is locally optimal with respect to the improvements considered by Algorithm 1 for a weight function w satisfies $\alpha w(A) \geq w(O)$ for some approximation factor $\alpha \geq 1$. Let $G = (V, E)$ be a given $(k+1)$-claw free graph with weights w_v for $v \in V$, and let $O \subseteq V$ be an independent set of G with maximum weight. We run the standard greedy algorithm to construct a solution S_0 and then set $d \triangleq \frac{n}{\epsilon w(S_0)}$. We then define a new instance of the problem using the weight function $\tilde{w}_v \triangleq \lfloor dw_v \rfloor$ for all $v \in V$ and apply Algorithm 1 to this new instance, starting from the solution S_0. Then, for all sets S maintained by Algorithm 1, we have $\tilde{w}(S) \leq dw(S) \leq dw(O)$ and since the weights $\tilde{w}_v$ are integral, the algorithm can thus make at most

$$\tilde{w}^2(O) - \tilde{w}^2(S_0) \leq \tilde{w}^2(O) \leq k\tilde{w}^2(S_0) \leq k\tilde{w}(S_0)^2 \leq k\left(dw(S_0)\right)^2 = kn^2\epsilon^{-2}$$

improvements before arriving at a locally optimal set A. For the second inequality, note that whenever $w_a \leq w_b$, $\tilde{w}_a^2 \leq \tilde{w}_b^2$ as well, and so any greedy solution for weight function w is also greedy solution for weight function $\tilde{w}^2$. The inequality then follows since the greedy algorithm has an approximation factor of at most k for the maximum weighted independent set problem in $(k+1)$-claw free graphs.

Let A be the locally optimal solution produced by applying Algorithm 1 to G with weight function $\tilde{w}$. Then, $\alpha\tilde{w}(A) \geq \tilde{w}(O)$ and so

$$\alpha dw(A) \geq \alpha\tilde{w}(A) \geq \tilde{w}(O) \geq dw(O) - |O|,$$

which in turn implies

$$\alpha w(A) \geq w(O) - \frac{\epsilon w(S_0)}{n}|O| \geq w(O) - \epsilon w(S_0) \geq w(O) - \epsilon w(O).$$

Altogether, then applying Algorithm 1 to $\tilde{w}$ gives us an approximation factor of $\alpha/(1-\epsilon)$ by using at most $kn^2\epsilon^{-2}$ improvements.

We now show that in fact this loss of ϵ can be removed entirely. For each $v \in V$, we construct a residual instance $G'(V', E') = G[V \setminus N(v, V)]$. We then run the above local search routine on G' with $\epsilon = (\alpha - 1)n^{-1} = \Omega(n^{-1})$ and return the best solution obtained across all n instances. Note that for any independent set I in G', $I \cup \{v\}$ is an independent set in G. Let $\hat{v} = \arg\max_{a \in O} w_a$ be the heaviest vertex in the optimal solution and consider the residual

instance in which $v = \hat{v}$. Let A' be the solution produced by our algorithm on this instance and let $O' \triangleq O - \hat{v}$. Then, $A = A' \cup \{\hat{v}\}$ is an independent set in G and

$$\alpha w(A) = \alpha w_{\hat{v}} + \alpha w(A') \geq \alpha w_{\hat{v}} + w(O') - \epsilon w(O') = w(O) + (\alpha - 1) w_{\hat{v}} - \frac{\alpha - 1}{n} w(O') \geq w(O),$$

where the last inequality follows from $w_{\hat{v}} = \max_{v \in O} w_v \geq \frac{1}{|O|} w(O) \geq \frac{1}{n} w(O) \geq \frac{1}{n} w(O')$. Altogether then, considering the best of all n solutions produced by the algorithm gives us a solution of weight at least $w(A)$ and so we obtain a factor α approximation. Moreover, the final algorithm performs at most $n^3 k \epsilon^{-2} = O(n^4 k)$ improvements across all n iterations of the algorithm.

2.5.4 Removing small cycles

Using Reduction 1 we assume that no cycle of constant length in the conflict graph exist. The reduction is due to Arkin and Hassin [AH98]. In particular, they transform an arbitrary locally optimal instance with respect to w into a solution that doesn't contain small cycle. They perform *crossing* operations that are described in the proof. Reduction 1 is a consequence of Lemma 2.5.7 which tells that it is sufficient to consider instances whose conflict graph has a large girth.

Lemma 2.5.7 (Lemma 7 [AH98])**.** *Let γ be a given constant. Consider a locally optimal solution A with respect to s-exchanges such that the conflict graph $G[A \cup O]$ satisfies $\frac{w(O)}{w(A)} = \alpha$ for some α. Then, there exists an instance $G[A' \cup O']$ such that $\frac{w(O')}{w(A')} = \alpha$. Additionally, the conflict graph $G[A' \cup O']$ has girth at least γ, and A' is locally optimal with respect to s-exchanges.*

Proof of Lemma 2.5.7. We show how to transform the original instance into one whose conflict graph has girth increased by an additive factor of 2. Consider the conflict graph $G[A \cup O]$ with node weights w, we perform *crossing operations*. Given an edge $e = (a, o)$, the crossing of e can be described in the following way: we create a copy $G[A' \cup O']$ of $G[A \cup O]$ with *identical weights*. Thus, $w_v = w_{v'}$ if v' is the copy of v. Then, we replace the edges e and its copy e' by the edges (a, o') and (a', o). Clearly, this operation ensures the $(k+1)$-claw freeness of the graph $G[A \cup A' \cup O \cup O']$ after the crossing. Secondly, the solution $A \cup A'$ is still locally optimal with respect to w^2. Consider a potential s-exchange $R \subseteq O \cup O'$ in the duplicated instance before the crossing. Clearly, in this instance this swap is non-improving. Once the crossing performed the neighborhood of the vertices in R doesn't change apart for the vertices o, o'. The crossing has only enlarged the neighborhood of R in a consistent way since $w_a = w_{a'}$.

To remove a cycle C of minimal length, we select an arbitrary edge $e = (a, o) \in C$ from it and perform a crossing operation. The only thing to show is that the crossing did not create a new cycle of minimal length which is not a duplicate of a minimal length cycle in the original graph. Suppose by contradiction that it is the case. Then, this cycle must use the edges $(o, a'), (o', a)$. Otherwise, it is mapped to a cycle in the original graph before the duplication. Let the original graph be G and the copy be G'. Denote this cycle by $C'' = S_1(o, a')S_1'(o', a)$, where S_1 is the path from a to o contained in G and S_1' the path from a' to o' contained in G'.

Then, if C'' has minimal length then either $S_1(o,a)$ or $S'_1(o,a)$ can be mapped to a cycle of length smaller than C in the original graph, a contradiction. Hence, all the cycles of minimal length after the crossing operation are copies of some minimal length cycle in the original graph.

A slight issue with this operation is the potential duplication of other minimal length cycles. The procedure is then the following. Enumerate all the minimal length cycles $\{C_i\}$ in the original graph G. Denote by $C_{i,j}$ the jth copy of C_i. To ensure efficient removal of all minimal length cycles, we remove all the $C_{i,j}$'s at the ith step. In particular, suppose that we want to cancel C_i and all its copies. Then, we identify an edge $e \in C_i$ and its copy $e_j \in C_{i,j}$ for all j. Then, we duplicate the instance and perform the crossing operation of e and e_j's altogether. Cross e with e' and e_j with e'_j. This procedure again doesn't create new small length cycles other than duplicates of an original cycle C_k and is still locally optimal. When all crossing operations are done, we end up with a conflict graph of girth increased by 2. Thus, by repeating this procedure, we may assume that the girth of the conflict graph is at least some given constant γ. □

As a corollary of Lemma 2.5.7 (applied with $\gamma = 4m$), we suppose the following reduction

Reduction 1. Assume that the conflict graph doesn't contain a cycle of length $4m$ where m is a constant.

2.5.5 Technical lemmas to build the exchanges

Using that Reduction 1 holds, we obtain structural properties about the exchange graph H. They allow us to assume that the large exchanges that we consider are in fact vertex disjoint *trees*. In this first lemma, we argue that, if the graph that we consider doesn't contain long path, then it must be a tree. We recall that a path $P = v_1 \dots v_t$ between t vertices has size $t-1$.

Lemma 2.5.8. *Suppose Reduction 1 holds for a constant $4m+2$. Then, any connected induced subgraph $F \subseteq H$ with longest path in F of length at most $2m$ is a tree.*

Proof of Lemma 2.5.8. By contradiction, we assume that there is cycle $\Sigma = v_1 v_2 \dots v_t v_1$ of length $t \leq 2m+1$. By definition of H, the edge (v_i, v_{i+1}) exists if either $v_{i+1} \in N^+_{v_i} - v_i$ or $v_i \in N^+_{v_{i+1}} - v_{i+1}$. Without loss of generality, we assume that the first case happens. Thus, there exists one vertex of $o_i \in C_{v_i}$ such that $v_{i+1} \in N(o_i, A - v_i)$. Applying the previous argument for each edge (v_i, v_{i+1}) yields a cycle $\Sigma' = v_1 o_1 v_2 o_2 v_3 \dots v_t o_t v_1$ in G of length $2(t) \leq 4m+2$, which is a contradiction. □

Lemma 2.5.9 is a technical lemma about the structure of long paths in trees.

Lemma 2.5.9. *Let T be a tree with longest path of length equal to j for some $j \geq 1$. Then, all longest paths $P = v_1 \dots v_{j+1}$ of length j in T must cross at a $v_{\lceil (j+2)/2 \rceil}$.*

Proof of Lemma 2.5.9. Let $P = v_1 \dots v_{j+1}$, and $P' = v'_1 \dots v'_{j+1}$ be two vertex disjoint longest paths. Consider the path $P_1 = v_1 \dots v'_{j+1}$ joining v_1 to v'_{j+1}. Because T is a tree the path P_1 exists and is unique. In particular $P_1 = \bar{P}Q\bar{P}'$ where $\bar{P} \subseteq P$ is a sub-path of P that contains v_1. Similarly, $\bar{P}' \subseteq P'$ is a sub-path of P' containing v'_{j+1}. The path Q starts with some vertex $v_i \in P$ for some i and $v'_l \in P'$ for some l. Since P and P' are vertex disjoint we have $v_i \neq v'_l$, thus one of the following paths must have a greater length than j: $\left\{P_1; \bar{P}Qv'_l \dots v'_1; v_{j+1} \dots v_i Q\bar{P}'; v_{j+1} \dots v_i Q v'_l \dots v'_1\right\}$. Hence, two longest paths must intersect.

Given that P and P' intersect, let $I \triangleq \{i \colon v_i \in P \cap P'\}$ be the set of indices of the vertices in the intersection of the two paths. Since T is a tree, given three integers $i \leq p \leq l$ such that i and l belong to I, then $p \in I$ otherwise it would create a cycle. Let i^* and l^* be the two integers such that $[i^*, l^*] \cap \mathbb{N} = I$.

Suppose by contradiction that $v_{\lceil (j+2)/2 \rceil}$ does not belong to the intersection of P and P'. Then, either both integers i^* and l^* are strictly smaller than $\lceil (j+2)/2 \rceil$ or both of them are strictly bigger. In the second case, either $v_1 \dots v_{i^*} \dots v'_{j+1}$ or $v_1 \dots v_{i^*} \dots v'_1$ is a path strictly longer than j. The first case is by an identical observation where we argue that either $v_j \dots v_{l^*} \dots v'_{j+1}$ or $v_j \dots v_{l^*} \dots v'_1$ has greater length. □

Lemma 2.5.10 gives an upper bound on the number of vertices that a tree with bounded degree and no long path can have. The bound obtained in this lemma is not sharp.

Lemma 2.5.10. *Let T be a tree that doesn't contain a path of length m and each vertex $v \in T$ has degree at most $d \geq 1$. Then, $|V(T)| \leq \sum_{s=0}^{m-1} d^s$.*

Proof of Lemma 2.5.10. We prove the lemma by induction of the length of the longest path in T. If $m = 2$, then the longest path has size at most 1. Thus, T is either an edge or a single vertex. Thus, we have $|V(T)| \leq 2 \leq 1 + d = \sum_{s=0}^{1} d^s$.

Let T be a tree with longest path size equal to j. Take a path $P = v_1 \dots v_{j+1}$ be a path of length j. By Lemma 2.5.9, all maximum longest paths must cross at $v_{\lceil (j+2)/2 \rceil}$. Thus removing, $v_{\lceil (j+2)/2 \rceil}$ from T creates at most d disjoint trees each, where each path in the subtrees has maximum length at most $\leq j - 1$. Applying the induction hypothesis we get that $|V(T)| \leq 1 + d \cdot (\#\text{vertices in subtrees}) \leq 1 + d \cdot \sum_{s=0}^{j-2} d^s = \sum_{s=0}^{j-1} d^s$. □

Finally, Lemma 2.5.11 partitions any large connected component in few vertex disjoint trees of bounded size.

Lemma 2.5.11. *Assume that Reduction 1 holds for some constant $4m$. Let C be some connected component of size greater or equal to m, then there is a decomposition of C into vertex disjoint trees T such that $|V(T)| \in \left[m, \sum_{s=0}^{2m-2} k^{2s}\right]$. Moreover, the length of the longest path is at most $2m - 2$.*

Proof of Lemma 2.5.11. Given $C \subseteq H$, we proceed as follows: for each path of length $2m-1$ (using $2m-1$ edges) delete the $\lceil \frac{m}{2} \rceil$ edge. Continue until no more path of length $2m-1$ can be found. The procedure terminates as there is a finite number of edges. Then, each formed connected component, say T, has the length of their longest path at most $2m-2$ by construction. By Lemma 2.5.8, the induced subgraph on T is a tree. It remains to bound the size of T. Every time an edge is deleted, which potentially creates two disjoint connected components, each side of the path contains m vertices[3]. Thus, for every connected component T obtained when the procedure terminates we have $|V(T)| \geq m$. The upper bound on the size of T follows from Lemma 2.5.10 since each connected component has the length of their longest path at most $2m-2$ and the degree of each vertex in H is at most k^2. □

[3] A path of length $2m-1$ has $2m-1$ edges and hence has $2m$ vertices

3 A $\sqrt{3}$-approximation for Weighted 3-Set Packing

The work in this Chapter is specific to this thesis and builds on Chapter 2.

3.1 Recap from Chapter 2

We continue the study of the maximum weight independent set problem in a $(k+1)$-claw free graph that generalizes the weighted k-set packing problem. Given a vertex weighted $(k+1)$-claw free graph, we seek to find an independent set of maximum weight. More precisely, we build on our previous work in Chapter 2 to design an improved approximation algorithm that greatly improves over our previous results. We recommend the reader to read Chapter 2 before going further in our analysis. In fact, we borrow all the notations introduced in the previous chapter and largely expand upon the previous proofs. We will for instance refer to the parameters $\Delta_a, \Psi_a, \Theta_a, \theta_a$ and $\rho_{k,\delta}$.

Our main result in this chapter is to give a $\sqrt{3}$-approximation for Maximum Weight Independent Set in 4-claw free graphs. It implies a $\sqrt{3}$-approximation for the weighted 3-dimensional matching problem, and improves over Theorem 2.5.6 and thus over [Neu21]. Since our analysis is based on extending Berman's algorithm with a greater swap size, our result is tight for $k = 3$. Indeed, the locality gap with unbounded swap size of Berman's algorithm is $\sqrt{k}$ as shown in Figure 2.1. Perhaps surprisingly, our proof shows that Figure 2.1 is in fact the *only* instance that yields a ratio of $\sqrt{3}$. All other structures in the conflict graph yield an improvement over the factor $\sqrt{3}$.

Neuwohner [Neu22] shows that Berman's algorithm restricted to swaps of size at most $O(\log(n))$ (where n is the number of vertices) cannot give a better approximation factor than $\frac{k}{2}$. For all $k \geq 7$, our second result is to match this result using smaller swaps of size $O((k/\varepsilon)^{O(1/\varepsilon)})$ independent of n. For $k \geq 7$ and any $\varepsilon > 0$, we obtain a $\frac{k+\varepsilon}{2}$-approximation algorithm. Not only is our result tight, but it improves over [Neu22] who designed an algorithm with asymptotic approximation factor equal to $\frac{k}{2}$. Here, asymptotic means that for any $\varepsilon > 0$, there is a k_ε such that for all $k \geq k_\varepsilon$, there is an approximation algorithm with guarantee at least $\frac{k+\varepsilon}{2}$. The proof requires $k_\varepsilon \geq \frac{200'000}{\varepsilon^3}$ and uses swaps of size $O(\log(n))$. Our proof naturally extends to the regime $k = 4, 5, 6$, and we obtain state-of-the-art results that almost match the factor $\frac{k}{2}$. We believe that our proof can be modified to obtain a ratio of $\frac{k+\varepsilon}{2}$ for all $k \geq 4$ and $\varepsilon > 0$. The exact statement of the theorem below is found in Section 3.7.1.

Theorem

For $k \geq 3$, Berman's algorithm (Algorithm 1) with s-exchanges, with $s = O\big((k/\varepsilon)^{O(1/\varepsilon)}\big)$ has approximation equal to $\frac{k+1-\tau_k}{2}$, where $\tau_3 = 4 - 2\sqrt{3}$, $\tau_4 = 0.8204$, $\tau_5 = 0.9282$, $\tau_6 = 0.9836$, and $\tau_k = 1 - \varepsilon$ for any $k \geq 7$, and $\varepsilon > 0$.

Bottleneck case in Chapter 2

We start by recalling the main obstacle to improve over Theorem 2.5.6. There, the approximation ratio is a trade-off between two sub-instances that may appear in the exchange graph H_ε. The tight cases are either the large connected components of size at least ℓ, or the isolated vertices in the exchange graph. All others cases induce a greater slack. In the first case, the slack is equal to $\chi^{(\ell)} \triangleq \frac{\ell-1}{\ell}(1-\delta)^2$, whereas in the second it is equal to $\rho_{k,\delta} \triangleq k\delta - \frac{\delta}{1-\delta}$. The final approximation ratio requires balancing these two quantities.

The crucial observation is that we did not use the absence of large improving swaps for isolated vertices. In the proof, we have solely used that the ratio between the weights of the endpoints of arcs in H that contain at least one isolated vertex is bounded away from 1. Using that isolated vertices also belong to large non-improving swaps we will improve the approximation factor.

Using large swaps for isolated vertices

Our main objective when extending our proof is the following: In order to get arbitrarily close to a $\frac{k}{2}$ approximation, we would like to set $\chi^{(\ell)}$ arbitrarily close to 1, which, as ℓ tends to infinity, requires setting δ to a tiny constant with a desirable accuracy. However, as δ tends to 0, the parameter $\rho_{k,\delta}$ tends to 0.

To compensate the loss induced from setting δ to a lower value, we refine the exchange graph by adding new arcs with a greater difference of weight between the corresponding endpoints. We look at the effect of large exchanges on formerly isolated vertices. More precisely, let H_ε be the exchange graph defined as in Definition 2.4.1. Let's call the novel graph H' after the inclusion of new arcs. Essentially, we want to refine the former decomposition of A into vertex disjoint trees to incorporate the arcs $H' \setminus H_\varepsilon$. Importantly, we preserve the decomposition that was obtained in H_ε. As invariant, we make sure that vertices receiving an appropriate amount of slack in H_ε still receive the same amount after the refinement.

The main effect is on isolated vertices in H_ε. Given an isolated vertex v in H_ε (c.f. Chapter 2), either v is no longer isolated in H' or v is still isolated in H'. In the first case, we can use the existence of arcs in $H' \setminus H_\varepsilon$ that connect v to bound the parameter Δ_v due to the absence of large improving swaps. In the second case, the vertex v is *very isolated* in the sense that the weight of the neighbors of v are in fact much further away from w_v than expected. Thus, the parameter $\rho_{k,\delta}$ can be strengthened to $\rho_{k,\delta'}$ where $\delta' \geq \delta$. This procedure enables us to reduce δ. However, refining the exchange graph once isn't enough to reduce the approximation factor to $\frac{k}{2}$. Thus, we will refine the exchange graph up to L times.

The overall argument is the following: We create an iterative refinement of the exchange graph $\{H_i\}_{i=1}^{L+1}$ with $L+1$ layers, where $H_1 \triangleq H_\varepsilon$ (c.f. Definition 2.4.1). Similarly to Definition 2.4.1, each layer is defined with a prescribed threshold ε_i, where $\varepsilon_{i+1} \triangleq 2\varepsilon_i - \varepsilon_i^2 \geq \varepsilon_i$. In the exchange graph $H_{\leq i} \triangleq \bigcup_{j=1}^{i} H_j$, we have a partition of $A = (A \setminus I_i) \sqcup I_i$, where I_i is the set of isolated components of size less than ℓ. For simplicity, we can think of I_i as isolated vertices in $H_{\leq i}$, as small maximal connected components are handled similarly. To $H_{\leq i}$, we add arcs from H_{i+1}. We refine the partition $I_i \triangleq D_{i+1} \sqcup P_{i+1} \sqcup I_{i+1}$. Given a vertex $v \in I_i$, if v is still isolated in $H_{\leq i+1}$, then $v \in I_{i+1}$. The vertices in I_{i+1} will receive an amount of slack of at least ρ_{k,ε_i}. Otherwise, $v \in D_{i+1} \sqcup P_{i+1}$, meaning that v belongs to a large connected component in $H_{\leq i+1}$. We distinguish two cases: either $v \in D_{i+1}$ in which case v belongs to a large connected component contained solely in $H_{\leq i+1}[I_i]$, or $v \in P_{i+1}$ where we will need to expand some large exchange in $A \setminus I_i$ by adding the connected component containing v to it. The set P_{i+1} is thus named *pendant* vertices. The vertices in D_{i+1}, P_{i+1} will receive an amount of slack equal to $\chi_{i-1}^{(\ell)} \triangleq \frac{\ell-1}{\ell}(1-\varepsilon_{i-1})^5 + \rho_{k,\varepsilon_{i-1}}$, and $\nu_{i-1} \simeq (1-\varepsilon_{i-1})^6 + \rho_{k,\varepsilon_{i-1}}$ respectively.

3.2 Definitions, notations and structural properties

We continue to expand the results of the previous section and keep the same notations (c.f. $\Delta_a, \Psi_a, \Theta_a, \theta_a$ and $\rho_{k,\delta}$.). We fix A to be the locally optimal solution of Algorithm 1, and denote O the optimal solution. Let $L \in \mathbb{N}$ be some integer fixed throughout the remainder of this chapter which defines the number of layers of the exchange graph. Additionally, we define the following sequence of variables $\varepsilon_{-1} \leq \varepsilon_0 \leq \varepsilon_1 \leq \varepsilon_2 \leq \ldots \leq \varepsilon_L$, where $\varepsilon_1 \in (0,1)$ is a value fixed in advance, and $\varepsilon_{-1} = 0$, $\varepsilon_0 \triangleq 1 - \sqrt{1-\varepsilon_1}$ and $\varepsilon_i \triangleq 2\varepsilon_{i-1} - \varepsilon_{i-1}^2$ for $i = 2, \ldots, L$. Observe that for $i \geq 1$, we have

$$(1-\varepsilon_i) = (1 - 2\varepsilon_{i-1} + \varepsilon_{i-1}^2) = (1-\varepsilon_{i-1})^2 = \ldots = (1-\varepsilon_1)^{2^{i-1}}.$$

The variables $\varepsilon_0, \varepsilon_1$ play the role of δ, ε in Chapter 2 respectively.

Recall, the following mapping $\pi \colon O \to A$ by $\pi(o) = \arg\max\{w_x : x \in N(o, A)\}$ which defines the collection of disjoint *claws* that partitions the optimal solution $\mathcal{C} = \{C_a\}_{a \in A}$, where $C_a \triangleq \{o : \pi(o) = a\}$ is the *claw centered at* a.

Definition 3.2.1 (Exchange graph H_i)**.** We define the exchange graph $H_i = (A, E)$ at layer $i = 1, \ldots, L+1$, where the ground set A are the vertices in the current solution returned by Algorithm 1, and there is an arc $(b, a) \in E(H_i)$ from b to a for every $o \in C_a$ such that:

- $b \in N(o, A - a)$,
- $\frac{w_b}{w_a} \in \begin{cases} (1-\varepsilon_1, 1-\varepsilon_{-1}] & \text{if } i = 1, \\ (1-\varepsilon_i, 1-\varepsilon_{i-1}] & \text{if } i = 2, \ldots, L, \\ [0, 1-\varepsilon_L] & \text{if } i = L+1. \end{cases}$

Remark 3.2.2. There is no arc that belongs to both H_i and H_j for $i \neq j$.

Moreover, we define $H_{\leq i} \triangleq \bigcup_{j=1}^{i} H_j$ to be the union of the exchange graph for the first i layers and let $H \triangleq \bigcup_{j=1}^{L+1} H_j$ to be the graph that contains the entire set of arcs. Similarly, for a sequence of sets $S_1, S_2, \ldots, S_t$, we define $S_{\leq i} \triangleq \bigcup_{j=1}^{i} S_j$. Let $\ell \in \mathbb{N}$ to be a fixed integer throughout the remainder of the proof. It defines the largest size of the exchange which we will consider in H_1. This parameter is identical to the one used in Chapter 2 when handling large connected components (Lemma 2.5.4). In the analysis, we will distinguish three types of vertices per layer.

Definition 3.2.3 (D_i, P_i, I_i)**.** We create L layers of the exchange graph H that partitions the vertex set A into different classes. The *connectivity* of a graph is taken irrespective of the orientation of the arcs.

- At layer 1, we let $A \triangleq D_1 \sqcup I_1$, where D_1 is the set of *non-isolated* vertices, i.e., they belong to a connected component that has size at least ℓ in H_1. The rest of the vertices in $A \setminus D_1$ forms I_1. Each vertex in I_1 belongs to a maximal connected component of size at most $\ell - 1$.
- For layer $2 \leq j \leq L$, we refine I_{j-1} such that $I_{j-1} \triangleq D_j \sqcup P_j \sqcup I_j$. The set D_j is the set of vertices that belong to a connected component of size at least ℓ in $H_{\leq j}[I_{j-1}]$, which is the induced subgraph of $H_{\leq j}$ restricted to I_{j-1}. The set P_j is the set of *pendant* vertices. It is the set of vertices that belong to a maximal connected component of size strictly less than ℓ in $H_{\leq j}[I_{j-1}]$, but belong to a maximal connected component of greater size in $H_{\leq j}$. Finally, we have $I_j = I_{j-1} \setminus (P_j \sqcup D_j)$ which corresponds to the vertex set of the set of connected components of size strictly less than ℓ in $H_{\leq j}$.

Definition 3.2.4. It is convenient to define $S_i \triangleq P_i \sqcup D_i$ for $i = 1, \ldots, L+1$ as the set of non-isolated vertices at layer i where $S_1 = D_1$ and $P_1 = \emptyset$, and $S_{L+1} = I_L$

Remark 3.2.5. For every $j = 1, \ldots, L$ the sets D_j, P_j, I_j are well-defined.

Proof of Remark 3.2.5. For any $j \geq 1$, we look at the graph $H_{\leq j}[I_{j-1}]$, where here $I_0 = A$. It is composed of multiple *maximal* connected components. Take a maximal connected component say $C \subseteq H_{\leq j}[I_{j-1}]$. Then, either it has size greater or equal to ℓ in which case all the vertices in C belong to D_j or C has size strictly less than ℓ. In the second case, we distinguish between whether C is still a maximal connected component in the graph $H_{\leq j}$ or if C is part of a larger connected component in $H_{\leq j}$. In the former case, all the vertices in C belong to I_j, whereas in the latter case they belong to P_j. □

3.2.1 Exchanges

Like in Chapter 2, our analysis of the w^2 local-search works by decomposing our current solution A into vertex disjoint trees. Each built tree will be non-improving with respect to w^2. We describe the set of exchanges here, but before we proceed further, we give a high-level intuition of the construction of the swaps. The construction is by induction on the number of layers L (we will not be using H_{L+1}).

3.2.2 High-level construction of the set of exchanges

We start at layer 1 and compute a decomposition of $A \triangleq D_1 \sqcup I_1$ into vertex disjoint trees contained in H_1. In the first layer, the respective decomposition of D_1 and I_1 is in fact the same as the one performed in Section 2.5.1 Chapter 2 where $D_1 = D$ and $I_1 = I$. Then, we contract each tree in a single node, which we *label* by the number of vertices present in the original tree. Contracted vertices resulting from the contraction of greater or equal to ℓ vertices are set as *roots*. We add edges from H_2 to the graph with contracted nodes, where there is an arc (T, T') between two contracted trees if there exists two vertices $u \in T, v \in T'$ such that $(u, v) \in E(H_2)$ (excluding self-loops). We look at the vertices in $I_1 \triangleq D_2 \sqcup P_2 \sqcup I_2$ that we decompose into vertex disjoint trees in $H_2 \cup H_1$ (possibly containing contracted vertices). Each such tree will exclusively contain vertices in D_2 or P_2 or I_2. The partitioning of D_2 is identical to the one in Section 2.5.1 but performed in the contracted graph. The trees spanning the vertices in P_2 will be attached to roots. Again, we contract the trees in a single node, and label them by the number of vertices of the ground set A that they contain. The contracted trees which result from the contraction of at least ℓ vertices from A are set as *roots*. In layer 2, the trees containing vertices of $D_{\leq 2}$ and P_2 are *roots* for the next iteration. In layer i, we assume that we have a decomposition of $(A \setminus I_{i-1}) \sqcup I_{i-1}$ into vertex disjoint trees. We further decompose I_{i-1} into vertex disjoint trees using edges from H_i. The set of trees in $H_{\leq i}[I_{i-1}]$ containing at least ℓ vertices forms D_i. The trees obtained from the decomposition of $A \setminus I_{i-1}$ are expanded by attaching the trees spanning P_i. We contract each tree in a single node, and continue the process by adding arcs from H_{i+1}. Figure 3.1 and 3.2 highlight a decomposition up to the second layer.

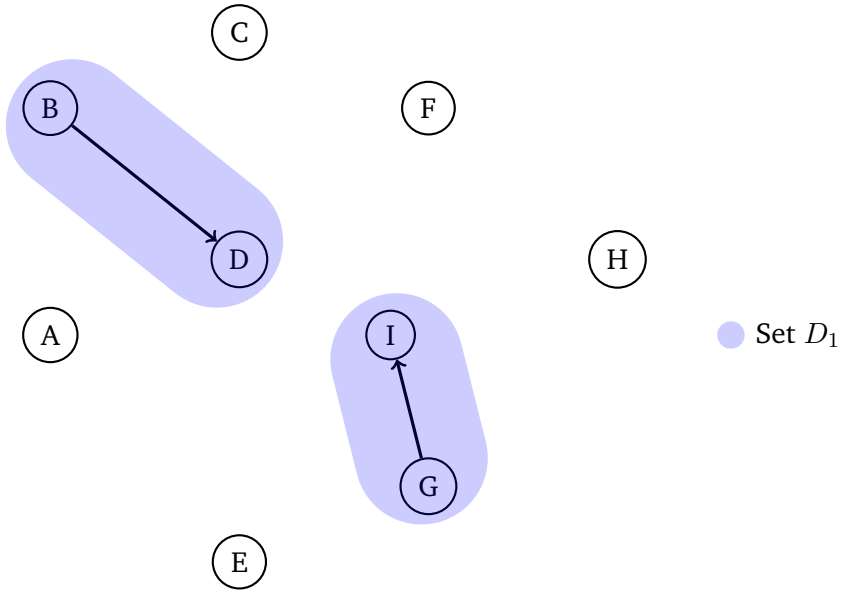


Fig. 3.1: Exchange graph at level 1 with $\ell = 2$. An arc $(b, a) \in H_1$ is present if $b \in N_a^+$, and $\frac{w_b}{w_a}$ is between $1 - \varepsilon_1$ and 1. The nodes B, D, I , G are in D_1 and A, C, E, F, H are in I_1

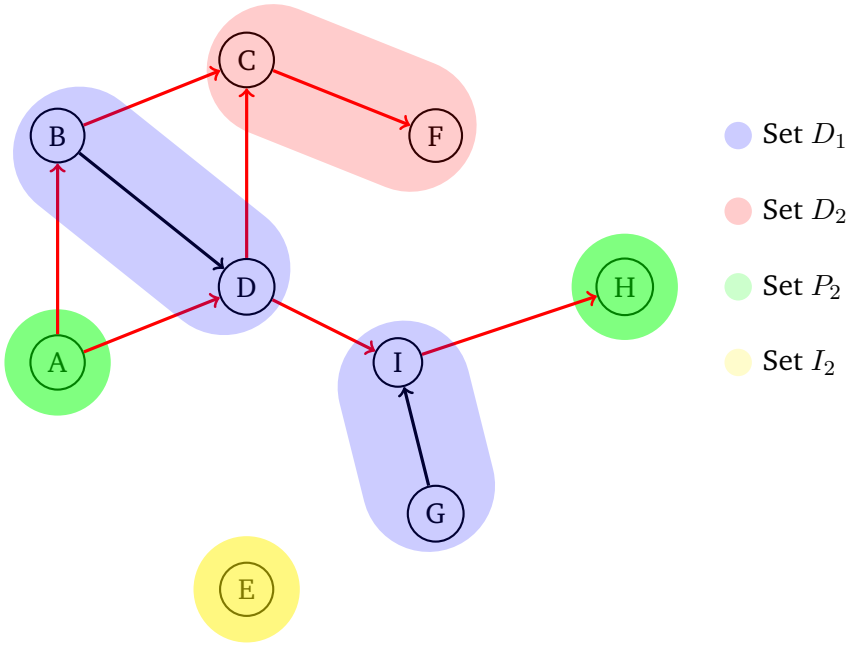


Fig. 3.2: Exchange graph $H_{\leq 2}$ at level 2 with $\ell = 2$. The red arcs are in H_2. The decomposition of A in vertex disjoint trees is made of 4 trees: {B,D,A}, {I, G, H}, {C, F} and {E}.

3.2.3 Formal Decomposition

Using Reduction 1, we assume throughout this section that the conflict graph $G[A \cup O]$ doesn't contain cycles of length $4(\ell^2 + \ell L + 2)$, where L, ℓ are constants independent of n and k fixed in advance. In particular, by Lemma 2.5.8 any connected induced subgraph $F \subseteq H$ that doesn't contain a path of length $2(\ell^2 + \ell L + 1)$ is a tree. Since $H_{\leq i} \subseteq H$, any connected induced subgraph $H_{\leq i}[F]$ without paths of length $2(\ell^2 + \ell L + 1)$ is also tree. We start by partitioning H_1 into vertex disjoint trees.

Lemma 3.2.6. *The set D_1 can be partitioned into vertex disjoint trees such that each tree $T \subseteq H_1$ and $|T| \in \left[\ell, \sum_{s=0}^{2\ell-2} k^{2s}\right]$. Additionally, there is a decomposition of I_1 into vertex disjoint trees $T \subseteq H_1$ of size up to $\ell - 1$. In each tree, the length of the longest path is at most $2\ell - 2$.*

Proof of Lemma 3.2.6. By definition, the set I_1 corresponds to maximal connected components of size up to $\ell - 1$ in H_1. Since the longest path in this component has length at most $\ell - 2$, we apply Lemma 2.5.8 which shows that it must be a tree in H. For vertices in D_1, we apply Lemma 2.5.11 with $m = \ell$. □

Lemma 3.2.6 is the base case of our decomposition. In Lemma 3.2.7, we expand the construction to the next layers. It gives a formal description of the set of exchanges that we consider.

Lemma 3.2.7. *For any $i = 1, \ldots, L$, there is a decomposition of $A \setminus I_i$ into vertex disjoint trees $\mathcal{T}$ with the following properties:*

- *Each tree can be written as* $\mathcal{T} \triangleq T_j \cup \left(\bigcup_{p=j+1}^{i} \bigcup_{m=1}^{m_p} T_{p,m}\right)$*, where* $T_j \subseteq D_j$ *for some* $1 \leq j \leq i$ *and each* $T_{p,m} \subseteq P_p$*, and* m_p *is a bounded integer independent of* n *and* k*. The sets* T_j *and* $T_{p,m}$ *are trees. The longest path in* $\mathcal{T}$ *has length at most* $2\ell^2 + 2\ell i$*.*
- *Given a vertex* $a \in I_{p-1} \cap T_{p,m}$ *for some* p, m*, let* C *be the maximal connected component containing* a *in* $H_{\leq p-1}[I_{p-1}]$*, then* $C \subseteq T_{p,m}$*. Similarly, for* $j \geq 2$ *and* $a \in I_{j-1} \cap T_j$*, the maximal connected component* C *containing* a *in* $H_{\leq j-1}[I_{j-1}]$ *satisfies* $C \subseteq T_j$*.*
- *A tree* $T_{p,m} \subseteq P_p$ *is connected to* $T_j \cup \left(\bigcup_{q=j+1}^{p-1} \bigcup_{m=1}^{m_q} T_{q,m}\right)$ *using an arc from* H_p*.*
- *The size of* $\mathcal{T}$ *is at most* $|\mathcal{T}| \leq (k^2\ell)^{i-1} \cdot \sum_{s=0}^{2\ell-2} (k\ell)^{2s}$ *and* $|T_j| \geq \ell$*.*

The main interest of the following Lemma is that it allows to partition the ground set in vertex disjoint trees of bounded size. It ensures that Algorithm 1 runs in polynomial time. The second bullet point is slightly more mysterious. It will be crucially used in Lemma 3.2.10 and 3.2.11. The advantage is that all arcs that are incoming or leaving a subtree $T_{p,m} \in \mathcal{T}$ ($T_j \in \mathcal{T}$) belong to $H_{\geq p}$ ($H_{\geq j}$, respectively).

Proof of Lemma 3.2.7. The proof follows by induction on i. The base case $i = 1$, holds by applying Lemma 3.2.6. Assume by induction that the above properties hold until layer $i - 1$. The entirety of the proof concentrates on the set $D_i \sqcup P_i \subseteq I_{i-1}$ after the addition of the arcs from H_i to the exchange graph $H_{\leq i-1}$. The goal is to expand $\mathcal{T}$ and incorporate the sets D_i and P_i to the decomposition.

We proceed as follows: by the induction hypothesis, we consider the current decomposition of $A \setminus I_{i-1}$ into trees $\{\mathcal{T}\}$ and the maximal connected components in $H_{\leq i-1}[I_{i-1}]$. We contract each connected component and each tree in a single node. In this graph, we add arcs between two contracted nodes if there exists two vertices contained in these contracted nodes which share an arc in H_i. Let G' be the contracted graph. We focus on G' restricted to the nodes arising from the contraction of the maximal connected components partitioning I_{i-1}, which we denote by $G'\big|_{I_{i-1}}$. Consider a maximal connected component C in $G'\big|_{I_{i-1}}$. Then, either

- Case D_i: C contains ℓ or more vertices in the *uncontracted* graph.
- Case P_i: C contains $\ell - 1$ or less vertices in the *uncontracted* graph and it is not maximal in G'.
- Case I_i: C contains $\ell - 1$ or less vertices in the *uncontracted* graph and it is maximal in G'.

In the first case, the following procedure yields a decomposition of D_i with the desirable properties. For each path of length exactly $2\ell - 1$ in C, delete the ℓ-the edge. Thus, each connected component formed by the above heuristic has longest path of length at most $2\ell - 2$, and contains at least ℓ vertices because both sides of the divided path have length at least ℓ. Each component formed by the procedure is a tree and is added to the collection $\{\mathcal{T}\}$. When the procedure terminates, the longest path in each formed component has length at

most the length of the longest path in the G' plus the length of the longest path in each contracted vertex. Here, we note that contracted vertices are maximal connected components in $H_{\leq i-1}[I_{i-1}]$ which by definition have size up to $\ell - 1$ so their longest path has length at most $\ell - 1$. Therefore, the partitioning of C creates connected components each of which has longest path length at most $2\ell - 2 + (2\ell - 1)(\ell - 1) = (2\ell - 1)\ell - 1 \leq 2\ell^2$.

By Lemma 2.5.8, assuming $G[A \cup O]$ doesn't contain cycles of length $4\ell^2 + 4$ we have that each constructed connected component is a tree. The degree of a vertex in G' is at most $k^2(\ell - 1)$ since each contracted node contains at most $\ell - 1$ vertices each with degree at most k^2 in H. By Lemma 2.5.10, the maximum size of a tree that doesn't a contain path of length $2\ell - 1$ and with degree at most $k^2(\ell - 1)$ is at most $\sum_{s=0}^{2\ell-2}(k^2(\ell - 1))^s$. Since each contracted vertex contains at most $\ell - 1$ vertices, the size of the constructed subgraph in H is at most $(\ell - 1) \cdot \sum_{s=0}^{2\ell-2}(k^2(\ell - 1))^s \leq \sum_{s=0}^{2\ell-2}(k\ell)^{2s}$. Finally, the second bullet point is verified by construction of G' and the path splitting argument since the nodes in G' are exactly the maximal connected components in $H_{\leq i-1}[I_{i-1}]$.

In the second case, we consider C that we attach to some tree $\mathcal{T} \subseteq A \setminus I_{i-1}$ with an arc from H_i. In the decomposition, we replace $\mathcal{T}$ by $\mathcal{T} \sqcup C$. The existence of an edge between C and $\mathcal{T}$ is by definition of P_i. We perform this operation for all such C and uncontract G'. By induction, the size of a tree $\mathcal{T}$ contained in $A \setminus I_{i-1}$ is at most $(k^2\ell)^{(i-1)-1} \cdot \sum_{s=0}^{2\ell-2}(k\ell)^{2s}$. Each vertex in $\mathcal{T}$ has degree at most k^2 in H. To each vertex in $\mathcal{T}$, we may attach a tree of size at most $|C| \leq \ell - 1 \leq \ell$. Thus, each such expanded tree $\mathcal{T}' \subseteq (A \setminus I_{i-1}) \cup P_i$ has size at most: $\left((k^2\ell)^{(i-1)-1} \cdot \sum_{s=0}^{2\ell-2}(k\ell)^{2s}\right) \cdot k^2\ell = (k^2\ell)^{i-1} \cdot \sum_{s=0}^{2\ell-2}(k\ell)^{2s}$. It remains to prove that the above construction is a tree. The length of the longest path in $\mathcal{T}'$ is at most the length of the longest path in $\mathcal{T}$ plus twice the longest path in a connected component attached to it. Hence, it is at most $2\ell^2 + 2\ell(i-1) + 2(\ell - 2) \leq 2\ell^2 + 2\ell i$. Since the graph $G[A \cup O]$ doesn't contain cycle of length $4(\ell^2 + \ell i + 1)$, we apply Lemma 2.5.8 and get that the extended graph is a tree. The second bullet point follows again from the definition of G'. □

From Lemma 3.2.7, we obtain a decomposition of the ground set of the exchange graph into vertex disjoint trees.

Corollary 3.2.8. *There is a decomposition of A into vertex disjoint trees $\{\mathcal{T}\}$ with the following properties:*

- *Each tree can be written as $\mathcal{T} \triangleq T_j \cup \left(\bigcup_{p=j+1}^{L} \bigcup_{m=1}^{m_p} T_{p,m}\right)$, where $T_j \subseteq D_j$ for some $j \in [1, L+1]$, where $D_{L+1} = I_L$, each $T_{p,m} \subseteq P_p$, and m_p is a bounded integer independent of n and k.*
- *The induced subgraph $H[\mathcal{T}]$ on $\mathcal{T}$ is a tree.*
- *Given a vertex $a \in I_{p-1} \cap T_{p,m}$ for some p, m, let C be the maximal connected component containing a in $H_{\leq p-1}[I_{p-1}]$, then $C \subseteq T_{p,m}$. Similarly, for $j \geq 2$ and $a \in I_{j-1} \cap T_j$, the maximal connected component C containing a in $H_{\leq j-1}[I_{j-1}]$ satisfies $C \subseteq T_j$.*
- *A tree $T_{p,m} \subseteq P_p$ is connected to $T_j \cup \left(\bigcup_{q=j+1}^{p-1} \bigcup_{m=1}^{m_q} T_{q,m}\right)$ using an arc from H_p.*

- *The size of $\mathcal{T}$ is at most $|\mathcal{T}| \leq (k^2\ell)^{L-1} \cdot \sum_{s=0}^{2\ell-2}(k\ell)^{2s}$. Furthermore, $|T_j| \geq \ell$ for $j = 1, \ldots, L$.*

Proof of Corollary 3.2.8. Lemma 3.2.7 gives a decomposition of the ground set $A \setminus I_L$ into vertex disjoint trees $\{\mathcal{T}\}$ with desirable properties. The second bullet point follows from Lemma 2.5.8 since the length of the longest path is bounded by a function of ℓ and L which are constants independent of n, and k. It remains to show that I_L can be partitioned as well. Recall that I_L is the set of maximal connected components of size at most $\ell - 1$ in the graph $H_{\leq L}$. Since they have size at most $\ell - 1$ vertices, Lemma 2.5.8 implies that they must be trees. Adding them to $\{\mathcal{T}\}$ satisfies all the properties. The third bullet point follows from the same graph contraction argument as in Lemma 3.2.7. □

Given Corollary 3.2.8, we introduce the following definitions

Definition 3.2.9 (Root/Pendant Tree, Bridge Arc)**.** Given a tree $\mathcal{T} \triangleq T_j \cup \left(\bigcup_{p=j+1}^{L} \bigcup_{m=1}^{m_p} T_{p,m}\right)$ as in Corollary 3.2.8 for $j \in [1, L+1]$, we say that T_j is the *root tree* and that $T_{p,m}$ are *pendant trees*. Additionally, the arc that satisfies the fourth bullet point is called the *bridge arc*.

3.2.4 Numerical properties of the decomposition

Via the decomposition from Corollary 3.2.8 we obtain an important property which will be used throughout the proof. It is similar to Property 2.5.2, but generalized to multiple layers.

Lemma 3.2.10. *For $i \geq 1$, let $a \in I_i$ and let $C \subseteq I_i$ be the maximal connected component containing a in $H_{\leq i}$. For $b \in N_a^+ \setminus C$ and $c \notin C$ such that $a \in N_c^+ - c$, we have:*

$$\frac{w_b}{w_a} \leq 1 - \varepsilon_i \qquad \text{and} \qquad \frac{w_a}{w_c} \leq 1 - \varepsilon_i.$$

Proof of Proposition 3.2.10. By definition of I_i, the connected component $C \subseteq I_i$ is isolated in the graph $H_{\leq i}$ and has size up to $\ell - 1$. Thus, for every potential arc (b, a) or (a, c) as in the proposition with $b, c \notin C$, we have that $(b, a), (a, c) \notin H_{\leq i}$ which implies that the ratio between the weight of the tail and the weight of the head is bounded by $1 - \varepsilon_i$. □

Lemma 3.2.11. *Let T be a pendant or root tree such that $T \subseteq S_j$. Let $a \in T$, then for every adjacent vertex $b \in (N_a^+ \setminus T) \cap S_i$ of a not in the tree, we have:*

$$\frac{w_b}{w_a} \leq \begin{cases} 1 & \text{if } i = j = 1, \\ 1 - \varepsilon_{\max\{i,j\}-1} & \text{else.} \end{cases}$$

Proof of Lemma 3.2.11. There is nothing to prove if $i = j = 1$. For $j \geq 2$, let $C \subseteq I_{j-1}$ be the maximal connected component in $H_{\leq j-1}[I_{j-1}]$ that contains a. For $i \geq 2$, let $C' \subseteq I_{i-1}$ be the maximal connected component in $H_{\leq i-1}[I_{i-1}]$ that contains b. Since b doesn't belong to T, we use the second property of Lemma 3.2.7 for $i \geq 2$ which implies that $b \notin C$. Similarly, we

have that $a \notin C'$ for $j \geq 2$. Applying Lemma 3.2.10 to the set of indices i, j that are greater than 2 yields the desired result. □

3.3 Efficient charging argument

Let's recall the analysis in Chapter 2. There, we distinguished between two sets of vertices D_1 and I_1. For vertices in D_1 we used the effect of large exchanges so that vertices in D_1 have their slack term Δ_v/w_v bounded away from 0. For vertices in I_1 we used that the weight of their neighbors differ largely from their weight. In particular, by balancing $\varepsilon = \varepsilon_1$ and $\delta = \varepsilon_0$, we proved that each vertex in I_1 receives $(1-\delta)$ times their weight from each adjacent vertex in H. Extending the proof to handle multiple layers is more delicate. We demonstrate that each vertex in S_{i+1} for $i \geq 1$ receives $1 - \varepsilon_{i-1}$ times their weight from adjacent vertices in H.

Recall the definition of $\theta_{a,o}$ and Θ_a. For $a \in A$ and $o \in C_a$, we let $\theta_{a,o} \triangleq w_a w(N(o, A - a)) - w^2(N(o, A - a))$, and let $\Theta_a \triangleq \sum_{o \in C_a} \theta_{a,o}$. Moreover, for a subset $Y \subseteq A$, we let $\theta_{a,o}(Y) \triangleq w_a w(N(o, Y - a)) - w^2(N(o, Y - a))$ to be the restriction of $\theta_{a,o}$ to adjacent vertices in Y. Similarly, we let $\Theta_a(Y) \triangleq \sum_{o \in C_a} \theta_{a,o}(Y)$. In short, Θ_a roughly captures the difference of weights between the endpoints of the arcs that are pointing towards a. Lemma 3.3.1 deals with $\theta_{a,o}(A \setminus Y)$. It shows the effect of the layering by having a greater bound for vertices in higher layers. This will be helpful in future computations to ensure that each vertex receives an appropriate amount of slack.

Lemma 3.3.1. *Let $T \subseteq S_j$ with $j \geq 1$ be a pendant or root tree (Definition 3.2.9), let X be a set of vertices containing $T \subseteq X$. Denote by $X^+ \triangleq \{x \in X : x \in N_v^+ - v \text{ for some } v \in T\}$ the set of vertices in X in the neighborhood of T. For any $a \in T$ and $o \in C_a$, we have*

$$\frac{\theta_{a,o}(A \setminus X)}{w_a} \geq \mathbf{1}_{[j \geq 2]} \cdot \varepsilon_{j-1} w(N(o, S_{\leq j} \setminus X^+)) + \sum_{i=j+1}^{L+1} \varepsilon_{i-1} w(N(o, S_i \setminus X^+)),$$

where $\mathbf{1}_{[j \geq 2]}$ is the characteristic vector of the event $j \geq 2$ and $S_{L+1} = I_L$.

Observe that the lower bound on $\theta_{a,o}(A \setminus X)$ depends exclusively on X^+. This is expected as Θ_a captures the slack induced by the arcs pointing towards a.

Proof of Lemma 3.3.1. Since $T \subseteq X$, we have $\theta_{a,o}(A \setminus X) = w_a w(N(o, A \setminus X)) - w^2(N(o, A \setminus X))$. Given $a \in T$ and $o \in C_a$, observe that the set $N(o, A \setminus X) = N(o, A \setminus X^+)$ since $X \setminus X^+$ is exactly the sets of vertices of X that do not appear in the neighborhood of T. Thus, we

have $\theta_{a,o}(A \setminus X) = \theta_{a,o}(A \setminus X^+)$ for all $a \in T$ and $o \in C_a$. Moreover, since the sets $\{S_i\}_{i=1}^{L+1}$ form a partition of A we can write

$$\begin{aligned}\theta_{a,o}(A \setminus X) &= w_a w(N(o, A \setminus X^+)) - w^2(N(o, A \setminus X^+)) \\ &= \sum_{i=1}^{L+1} \left[w_a w(N(o, S_i \setminus X^+)) - w^2(N(o, S_i \setminus X^+)) \right].\end{aligned}$$

We apply Lemma 3.2.11 to every adjacent vertex of a in $A \setminus X^+ \subseteq A \setminus T$. It implies that

$$\begin{aligned}\sum_{i=1}^{L+1} w^2(N(o, S_i \setminus X^+)) &\leq w_a \sum_{i=1}^{L+1} (1 - \mathbf{1}_{[i \geq 2 \vee j \geq 2]} \varepsilon_{\max\{i,j\}-1}) w(N(o, S_i \setminus X^+)) \\ &\leq w_a (1 - \mathbf{1}_{[j \geq 2]} \varepsilon_{j-1}) w(N(o, S_{\leq j} \setminus X^+)) \\ &\quad + w_a \sum_{i=j+1}^{L+1} (1 - \varepsilon_{i-1}) w(N(o, S_i \setminus X^+)).\end{aligned}$$

Substituting the previous bound into $\theta_{a,o}(A \setminus X^+)$ and dividing by w_a yields the desired result. □

3.4 Slack for Large Trees

Our proof focuses on bounding the slack for large trees defined in Corollary 3.2.8. More precisely, we let

$$\mathcal{T} \triangleq T_{j+1} \cup \bigcup_{p=j+2}^{L} \bigcup_{m=1}^{m_p} T_{p,m}, \tag{3.1}$$

be a tree as in Corollary 3.2.8 where $T_{j+1} \subseteq D_{j+1}$ and $T_{p,m} \subseteq P_p$ for each $p = j+2, \ldots, L$, and $j = 0, \ldots, L$, where we define $D_{L+1} = I_L$. Recall that $T_{p,m} \subseteq \mathcal{T}$ is a pendant tree which is attached to $T_{j+1} \cup \bigcup_{q=j+2}^{p-1} \bigcup_{r=1}^{r_q} T_{q,r}$ using a bridge arc, denoted $e_{p,m}$.

In the rest of Section 3.4, we capture the slack present in several key quantities. The lemmas are stated in a general form. In Section 3.6, we employ them to analyze the slack that vertices in root trees and pendant trees receive.

3.4.1 Slack from Large Exchanges

We start by analyzing the slack induced by the absence of improving exchanges in $\mathcal{T}$.

Definition 3.4.1 (Locally optimal)**.** A tree T is said to be *locally optimal* if $w^2(N_T^+) - w^2(C_T) \geq 0$, where $N_T^+ \triangleq \bigcup_{v \in T} N_v^+$ and $C_T \triangleq \bigcup_{v \in T} C_v$.

In the rest of the analysis, we assume that Algorithm 1 terminates and that each $\mathcal{T}$ is locally optimal. The exact value of s given as input to Algorithm 1 will be given later. Since

Algorithm 1 checks all possible subsets of size at most sk, all subsets of $\mathcal{T}$ are also locally optimal. In particular, each root tree and pendant tree is locally optimal.

Assuming that $\mathcal{T}$ is locally optimal, we use Lemma 2.4.5 to derive the gain by performing large exchanges. For simplicity, we define $\Gamma_E \triangleq \sum_{(u,v)\in E} \frac{w_u^2}{w_v}$ for a subset of edges E. Lemma 3.4.2 shows that the amount of slack available is bounded a function of the arcs in $\mathcal{T}$ which is exactly $\Gamma_{E(\mathcal{T})}$.

Lemma 3.4.2. *Given a locally optimal tree $\mathcal{T}$ as in Equation (3.1). Let $E'_{p,m} = E(T_{p,m}) + e_{p,m}$ be the set of edges in the pendant tree $T_{p,m}$ with the bridge edge $e_{p,m}$. Then,*

$$\sum_{v\in\mathcal{T}} \frac{\Delta_v}{w_v} \geq \Gamma_{E(T_{j+1})} + \sum_{p,m} \Gamma_{E'_{p,m}}.$$

Proof of Lemma 3.4.2. The proof is a straightforward application of Lemma 2.4.5 applied to $\mathcal{T}$. Indeed,

$$\sum_{v\in\mathcal{T}} \frac{\Delta_v}{w_v} \geq \sum_{(u,v)\in E(\mathcal{T})} \frac{w_u^2}{w_v} = \Gamma_{E(\mathcal{T})} = \Gamma_{E(T_{j+1})} + \sum_{p,m} \Gamma_{E'_{p,m}}. \qquad \square$$

Lemma 3.4.2 simply partitions the slack induced by arcs of $\mathcal{T}$ into those that belong to the root tree and to the pendant trees respectively. Bridge arcs are assigned to pendant trees. Importantly, $\Gamma_{E'_{p,m}}$ contains as many terms as number of vertices $|T_{p,m}|$. We will use this fact for pendant trees to map each arc to a vertex in Section 3.6.

3.4.2 Exterior Slack

In the next lemma, we bound the quantity $\sum_{a\in T} \Theta_a(A \setminus T)$ away from 0 for some tree T. Recall that, $\Theta_a(A \setminus T)$ roughly captures the difference of weights between endpoints of arcs from $A \setminus T$ to T. If the difference is large, it means that the neighborhood $N_T^+ \setminus T$ has a small weight compared to T, which we leverage in the proof to bound the weight of the set of talons of each claw centered at $v \in T$. The lemma is stated in a slightly more general form by considering an additional set $Y \subseteq N_T^+ \setminus T$.

Lemma 3.4.3. *Let $j \in [1, L]$ and let $T \subseteq S_{j+1}$ be a root or pendant tree in $\mathcal{T}$ (see Equation 3.1) where $D_{L+1} = I_L$. Consider a subset $Y \subseteq S_{\leq j+1} \setminus T$ disjoint from T and denote by $F \triangleq \{(y, v) \in H : y \in Y, v \in T\}$ the set arcs from Y to T. Then,*

$$\sum_{v\in T} \sum_{o\in C_v} \frac{(w_v - w_o)^2 + \theta_{v,o}(A \setminus (T \cup Y))}{w_v} \geq \left(\rho_{k,\varepsilon_{j-1}} - k\varepsilon_{j-1}\right) w(T) - \frac{\varepsilon_{j-1}}{1-\varepsilon_{j-1}} \sum_{(y,x)\in F} \frac{w_y^2}{w_x} + \xi,$$

where

$$\xi \geq \sum_{v\in T} \sum_{o\in C_v} \sum_{i=1}^{L} \varepsilon_{i-1} w(N(o, S_{i+1})) - \varepsilon_{j-1} \sum_{(u,v)\in E(T)} w_u - \varepsilon_{j-1} \sum_{(y,v)\in F} w_y.$$

To simplify the explanation, we set $Y = \emptyset$ and $F = \emptyset$. In this case, we see that every vertex $v \in T$ receives a slack of $\rho_{k,\varepsilon_{j-1}}$ which coincides with the fact that $v \in I_j$. We also notice that for $o \in C_v$, each vertex in $N(o, S_{i+1})$ receives ε_{i-1} times its weight from v. The subtlety is the slight loss equal to $\varepsilon_{j-1} \sum_{(u,v)\in E(T)} w_u$.

The next lemma is analogous to Lemma 3.4.3 above, except that it deals with the case $j = 0$. It is an identical statement where we set $Y = \emptyset$. It is stated in this form because we will not need to consider the set Y when dealing with root trees in S_1.

Lemma 3.4.4. *Let $T \subseteq S_1$ be a root tree in $\mathcal{T}$ (see Equation 3.1). Then,*

$$\sum_{v\in T}\sum_{o\in C_v} \frac{(w_v - w_o)^2 + \theta_{v,o}(A \setminus T)}{w_v} \geq \sum_{v\in T}\sum_{o\in C_v}\sum_{i=1}^{L} \varepsilon_{i-1} w(N(o, S_{i+1})).$$

Proof of Lemma 3.4.3. Let $j \geq 1$, and let $v \in T \subseteq S_{j+1}$ be a vertex in the tree. Consider a neighbor $u \in (N_v^+ \setminus T) \cap S_{i+1}$ that doesn't belong to the tree. By Lemma 3.2.11, we have that:

$$\frac{w_u}{w_v} \leq 1 - \varepsilon_{\max\{i,j\}}.$$

Let $Y^+ \triangleq \{y \in Y : (y, v) \in F\}$ be the set of vertices from Y that have an arc to some vertex in T. Additionally, let $X = T \cup Y$ and $X^+ = T \cup Y^+$. We express $\theta_{v,o}(A \setminus X)$ as a function of $\eta_{v,o}$ and $\beta_{v,o}$. Intuitively, given $v \in T$, an amount $\sum_{o\in C_v} \eta_{v,o}$ is distributed to the neighbors of v outside X^+, whereas an amount $\sum_{o\in C_v} \beta_{v,o}$ is kept at vertex v. Formally for each vertex $v \in T$ and $o \in C_v$ we let:

- $\eta_{v,o} \triangleq \varepsilon_{j-1} w(N(o, S_{\leq j+1} \setminus X^+)) + \sum_{i=j+1}^{L} \varepsilon_{i-1} w(N(o, S_{i+1} \setminus X^+))$
- $\beta_{v,o} \triangleq \left[(1 - \varepsilon_{j-1}) w(N(o, S_{\leq j+1} \setminus X^+)) + \sum_{i=j+1}^{L} (1 - \varepsilon_{i-1}) w(N(o, S_{i+1} \setminus X^+))\right]$

We also define $\eta \triangleq \sum_{v\in T}\sum_{o\in C_v} \eta_{v,o}$ and $\beta \triangleq \sum_{v\in T}\sum_{o\in C_v} \beta_{v,o}$. Observe that both sums are over vertices in $T = X \setminus Y$. The variables $\eta_{y,o}, \beta_{y,o}$ for $y \in Y$ are not defined. For $v \in T$, Lemma 3.3.1 implies

$$\begin{aligned}
\frac{\theta_{v,o}(A\setminus X)}{w_v} &\geq \varepsilon_j w(N(o, S_{\leq j+1}\setminus X^+)) + \sum_{i=j+1}^{L} \varepsilon_i w(N(o, S_{i+1}\setminus X^+)) \\
&= (2\varepsilon_{j-1} - \varepsilon_{j-1}^2) w(N(o, S_{\leq j+1}\setminus X^+)) + \sum_{i=j+1}^{L} (2\varepsilon_{i-1} - \varepsilon_{i-1}^2) w(N(o, S_{i+1}\setminus X^+)) \\
&= \varepsilon_{j-1}(1-\varepsilon_{j-1}) w(N(o, S_{\leq j+1}\setminus X^+)) + \sum_{i=j+1}^{L} \varepsilon_{i-1}(1-\varepsilon_{i-1}) w(N(o, S_{i+1}\setminus X^+)) + \eta_{v,o} \\
&\geq \varepsilon_{j-1}\beta_{v,o} + \eta_{v,o},
\end{aligned}$$

where in the second line we used the definition of $\varepsilon_i = 2\varepsilon_{i-1} - \varepsilon_{i-1}^2$ for $i = 1, \dots, L$. The final inequality follows from the fact that $\varepsilon_{i-1} \geq \varepsilon_{j-1}$ for every $i = j+1, \dots, L$. Summing over all the vertices in T and the vertices of O in the claw centered at them, we have

$$\sum_{v \in T} \sum_{o \in C_v} \frac{\theta_{v,o}(A \setminus X)}{w_v} \geq \varepsilon_{j-1}\beta + \eta. \tag{3.2}$$

Next, we give a lower bound on β. It is obtained using that the tree T is locally optimal (Definition 3.4.1), and using Lemma 2.4.5. Since the tree $T \subseteq S_{j+1}$ we can apply Lemma 3.2.11 to the neighbors $u \in (N_v^+ \setminus X^+) \cap S_{i+1} \subseteq (N_v^+ \setminus T) \cap S_{i+1}$ outside X^+. Because $\{S_i\}_{i=1}^{L+1}$ forms a partition of A, we have

$$\begin{aligned} w^2(N_v^+ \setminus X^+) &= w^2\big((N_v^+ \cap S_{\leq j+1}) \setminus X^+\big) + \sum_{i=j+1}^{L} w^2\big((N_v^+ \cap S_{i+1}) \setminus X^+\big) \\ &\leq w_v \left[(1-\varepsilon_j) w\big((N_v^+ \cap S_{\leq j+1}) \setminus X^+\big) + \sum_{i=j+1}^{L} (1-\varepsilon_i) w\big((N_v^+ \cap S_{i+1}) \setminus X^+\big) \right]. \end{aligned}$$

By definition of the sequence of ε, we know that $1 - \varepsilon_j = (1-\varepsilon_{j-1})^2$ for $j = 1, \dots, L$ and that $1 - \varepsilon_i = (1-\varepsilon_{i-1})^2 \leq (1-\varepsilon_{i-1})(1-\varepsilon_{j-1})$ for $i \geq j \geq 1$. Applying these two observations and factoring by $(1-\varepsilon_{j-1})$, we obtain

$$\begin{aligned} \frac{w^2(N_v^+ \setminus X^+)}{w_v} &\leq (1-\varepsilon_{j-1}) \left[(1-\varepsilon_{j-1}) w\big((N_v^+ \cap S_{\leq j+1}) \setminus X^+\big) + \sum_{i=j+1}^{L} (1-\varepsilon_{i-1}) w\big((N_v^+ \cap S_{i+1}) \setminus X^+\big) \right] \\ &\leq (1-\varepsilon_{j-1}) \sum_{o \in C_v} \beta_{v,o}. \end{aligned}$$

In the last inequality we have simply applied a union bound over the neighbors $N_v^+ \setminus X^+$ since $N_v^+ \setminus X^+$ is the union of the sets $N(o, A \setminus X^+)$ for all $o \in C_v$. On the other hand, we have $w^2(N_v^+ \cap X^+) = w_v^2 + \sum_{u:(u,v) \in E(T)} w_u^2 + \sum_{u:(u,v) \in F} w_u^2$ for all $v \in T$ where the equality holds since X^+ is a tree. Indeed, the length of the longest path in the induced subgraph $H[X^+]$ is at most the length of the longest path in T plus 2 (since $Y^+ \subseteq N_T^+ \setminus T$). Thus, the length of the longest path in X^+ is bounded by a constant. Using Lemma 2.4.2, we conclude that the induced subgraph $H[X^+]$ must be a tree. Substituting both expressions, we obtain

$$\begin{aligned} \sum_{v \in T} \frac{\Delta_v}{w_v} &= \sum_{v \in T} \frac{w^2(N_v^+) - w^2(C_v)}{w_v} \\ &= \sum_{v \in T} \frac{w^2(N_v^+ \setminus X^+)}{w_v} + \frac{w^2(N_v^+ \cap X^+)}{w_v} - \frac{w^2(C_v)}{w_v} \\ &\leq (1-\varepsilon_{j-1})\beta + \sum_{(u,v) \in E(T)} \frac{w_u^2}{w_v} + \sum_{v \in T} \left[w_v - \frac{w^2(C_v)}{w_v} \right] + \sum_{(u,v) \in F} \frac{w_u^2}{w_v}. \end{aligned}$$

Since T is locally optimal, Lemma 2.4.5 shows that $\sum_{v\in T}\frac{\Delta_v}{w_v}\geq\sum_{(u,v)\in E(T)}\frac{w_u^2}{w_v}$. Substituting this expression to bound the left-hand side of the previous equation yields the following lower estimate on β,

$$\beta\geq\frac{1}{1-\varepsilon_{j-1}}\sum_{v\in T}\left[\frac{w^2(C_v)}{w_v}-w_v\right]-\frac{1}{1-\varepsilon_{j-1}}\sum_{(u,v)\in F}\frac{w_u^2}{w_v}. \tag{3.3}$$

The next step of the proof consists of using Equation (3.3) and Equation (3.2) to bound the left-hand side of Lemma 3.4.3. But before doing so we introduce the variables $\alpha_{v,o}\triangleq w_o/w_v$ for $v\in T, o\in C_v$, and $\alpha_v\triangleq\sum_{o\in C_v}\alpha_{v,o}$. Substituting Equation (3.2) and (3.3) into the following computation we get that,

$$\begin{aligned}
&\sum_{v\in T}\frac{1}{w_v}\sum_{o\in C_v}(w_o-w_v)^2+\sum_{v\in T}\sum_{o\in C_v}\frac{\theta_{v,o}(A\setminus X)}{w_v}\\
\geq&\sum_{v\in T}\left(\frac{w^2(C_v)}{w_v}-2w(C_v)+|C_v|\,w_v\right)+\varepsilon_{j-1}\beta+\eta\\
=&\sum_{v\in T}w_v\left(\frac{1}{1-\varepsilon_{j-1}}\sum_{o\in C_v}\alpha_{v,o}^2-2\sum_{o\in C_v}\alpha_{v,o}+|C_v|-\frac{\varepsilon_{j-1}}{1-\varepsilon_{j-1}}\right)-\frac{\varepsilon_{j-1}}{1-\varepsilon_{j-1}}\sum_{(u,v)\in F}\left[\frac{w_u^2}{w_v}\right]+\eta\\
\geq&\sum_{v\in T}w_v\left(\frac{\alpha_v^2}{(1-\varepsilon_{j-1})\,|C_v|}-2\alpha_v+|C_v|-\frac{\varepsilon_{j-1}}{1-\varepsilon_{j-1}}\right)-\frac{\varepsilon_{j-1}}{1-\varepsilon_{j-1}}\sum_{(u,v)\in F}\left[\frac{w_u^2}{w_v}\right]+\eta.
\end{aligned}$$

The last inequality follows from Cauchy-Schwarz (Theorem 1.B.3). Giving a lower bound on the previous expression requires minimizing each round bracketed expression. Each of them is a function of α_v, which attains its minimum at $\alpha_v^*=(1-\varepsilon_{j-1})\,|C_v|$. Replacing each α_v by the minimum value yields,

$$\sum_{v\in T}\sum_{o\in C_v}\frac{(w_o-w_v)^2+\theta_{v,o}(A\setminus X)}{w_v}\geq\sum_{v\in T}\rho_{|C_v|,\varepsilon_{j-1}}w_v-\frac{\varepsilon_{j-1}}{1-\varepsilon_{j-1}}\sum_{(u,v)\in F}\left[\frac{w_u^2}{w_v}\right]+\eta. \tag{3.4}$$

It remains to expand η. We fix $o\in C_v$ for some $v\in T$. By assumption the set X^+ lies in $S_{\leq j+1}$, and the sets S_i are disjoint. Thus, $X^+\cap S_{i+1}=\emptyset$ for all $i\geq j$ and hence,

$$\varepsilon_{j-1}w(N(o,S_{\leq j+1}\cap X^+))+\sum_{i=j+1}^{L}\varepsilon_{i-1}w(N(o,S_{i+1}\cap X^+))=\varepsilon_{j-1}\big(w(N(o,T))+w(N(o,Y^+))\big), \tag{3.5}$$

where the equality is by disjointness of T and Y^+. Summing the first term on the right-hand side over all $v\in T$ and $o\in C_v$, we get that

$$\sum_{v\in T}\sum_{o\in C_v}w(N(o,T))=\sum_{v\in T}|C_v|\,w_v+\sum_{(u,v)\in E(H[T])}w_u=\sum_{v\in T}|C_v|\,w_v+\sum_{(u,v)\in E(T)}w_u. \tag{3.6}$$

The set $E(H[T])$ is the edge set of the induced subgraph of the exchange graph on T, which by the second bullet point of Corollary 3.2.8 is exactly equal to $E(H[T]) = E(T)$. Similarly, summing the second term over all $v \in T$ and $o \in C_v$, we get that

$$\sum_{v \in T} \sum_{o \in C_v} w(N(o, Y^+)) = \sum_{(y,v) \in F} w_y. \tag{3.7}$$

By combining Equation (3.5), (3.6) and (3.7) we obtain the following

$$\begin{aligned}\zeta &\triangleq \sum_{v \in T} \sum_{o \in C_v} \left[\varepsilon_{j-1} w(N(o, S_{\leq j+1} \cap X^+)) + \sum_{i=j+1}^{L} \varepsilon_{i-1} w(N(o, S_{i+1} \cap X^+)) \right] \\ &= \varepsilon_{j-1} \left(\sum_{v \in T} |C_v| \, w_v + \sum_{(u,v) \in E(T)} w_u + \sum_{(y,v) \in F} w_y \right).\end{aligned}$$

Observe that the weight of $N(o, S_{i+1} \cap X^+)$ for $v \in T$ and $o \in C_v$ is captured by ζ, and that η captures the weight of the complementary set $N(o, S_{i+1} \setminus X^+)$. Therefore,

$$\begin{aligned}\eta &= \eta + \zeta - \zeta \\ &= \sum_{v \in T} \sum_{o \in C_v} \left[\varepsilon_{j-1} w(N(o, S_{\leq j+1})) + \sum_{i=j+1}^{L} \varepsilon_{i-1} w(N(o, S_{i+1})) \right] - \zeta \\ &\geq \sum_{v \in T} \sum_{o \in C_v} \sum_{i=1}^{L} \varepsilon_{i-1} w(N(o, S_{i+1})) - \zeta \\ &= \sum_{v \in T} \sum_{o \in C_v} \sum_{i=1}^{L} \varepsilon_{i-1} w(N(o, S_{i+1})) - \varepsilon_{j-1} \left(\sum_{v \in T} |C_v| \, w_v + \sum_{(u,v) \in E(T)} w_u + \sum_{(y,v) \in F} w_y \right).\end{aligned}$$

The inequality is by the monotonicity of the sequence of $\varepsilon_i' s$. After substituting the above bound on η in Equation (3.4), Lemma 3.4.6 then follows by recalling that $\rho_{t,\varepsilon} - t\varepsilon = \rho_{k,\varepsilon} - k\varepsilon$ for any $0 \leq t \leq k$ and $\varepsilon \geq 0$. □

Proof of Lemma 3.4.4. Let $T \subseteq S_1$ be a root tree. By Lemma 3.3.1 and since $T \subseteq S_1$ the following holds for any $v \in T$ and $o \in C_v$,

$$\frac{\theta_{v,o}(A \setminus T)}{w_v} \geq \sum_{i=2}^{L+1} \varepsilon_{i-1} w(N(o, S_i \setminus T)) = \sum_{i=2}^{L+1} \varepsilon_{i-1} w(N(o, S_i)) \geq \sum_{i=1}^{L} \varepsilon_{i-1} w(N(o, S_{i+1})),$$

where in the second equality we used that $T \cap S_i = \emptyset$ for any $i = 2, \ldots, L+1$. The last inequality is by the identity $\varepsilon_i \geq \varepsilon_{i-1}$ for all i. Summing over all $v \in T$ and $o \in C_v$ and using that $(w_v - w_o)^2 \geq 0$, we get the desired inequality. □

3.4.3 Interior Slack

In the previous section, we were given a tree T, and we measured the slack arising from arcs that are incoming into T. In this section, we analyze the slack arising from arcs between vertices in T. More precisely, we consider the quantity $\Theta_a(T)$ for some tree T and $a \in T$. This section complements Section 3.4.2 which focused on bounding $\Theta_a(A \setminus T)$. We prove a slightly more general lemma than described above by considering $\Theta_a(T \cup Y)$ for some set Y and $a \in T$.

Lemma 3.4.5. *Let T, Y be two sets such that $Y \subseteq A \setminus T$. Denote by $F \triangleq \{(y,v) \in H \colon y \in Y, v \in T\}$ the set arcs from Y to T and $\bar{F} \triangleq \{(v,y) \in H \colon v \in T, y \in Y\}$ the set of arcs from T to Y. Then,*

$$\Gamma_{E(T)\cup F\cup \bar{F}} + \sum_{v\in T} \frac{\Theta_v(T\cup Y)}{w_v} \geq \sum_{(u,v)\in E(T)} w_u + \sum_{(y,v)\in F} w_y + \sum_{(x,y)\in \bar{F}} \frac{w_x^2}{w_y}.$$

Proof of Lemma 3.4.5. For simplicity, let $X = T \cup Y$. We expand the parameter $\Theta_v(X)$.

$$\begin{aligned}
\sum_{v\in T} \frac{\Theta_v(X)}{w_v} &\triangleq \sum_{v\in T} \sum_{o\in C_v} \left[w(N(o, X-v)) - \frac{w^2(N(o,X-v))}{w_v} \right] \\
&= \sum_{(u,v)\in E(H[T])\cup F} \left[w_u - \frac{w_u^2}{w_v} \right] \\
&= \sum_{(u,v)\in E(T)\cup F} \left[w_u - \frac{w_u^2}{w_v} \right],
\end{aligned}$$

where $E(H[T])$ is the edge set of the induced subgraph of the exchange graph on T. The last equality follows from the second bullet point of Corollary 3.2.8. We combine the above computation with $\Gamma_{E(T)\cup F\cup \bar{F}}$ to obtain the desired result. Recall that $\Gamma_{E(T)\cup F\cup \bar{F}} = \sum_{(u,v)\in E(T)\cup F\cup \bar{F}} \frac{w_u^2}{w_v}$. Thus,

$$\begin{aligned}
\Gamma_{E(T)\cup F\cup \bar{F}} + \sum_{v\in T} \frac{\Theta_v(X)}{w_v} &= \sum_{(u,v)\in E(T)\cup F\cup \bar{F}} \frac{w_u^2}{w_v} + \sum_{(u,v)\in E(T)\cup F} \left[w_u - \frac{w_u^2}{w_v} \right] \\
&= \sum_{(u,v)\in E(T)\cup F} w_u + \sum_{(u,v)\in \bar{F}} \frac{w_u^2}{w_v}.
\end{aligned}$$

□

3.4.4 Final Expression of the Slack

We conclude Section 3.4 by deriving a general estimate for the slack induced by vertices in a locally optimal tree T. The result follows by combining the results from the previous two subsections.

Lemma 3.4.6. *Let $j \in [1, L]$ and let $T \subseteq S_{j+1}$ be a root or pendant tree in $\mathcal{T}$ (see Equation 3.1) where $D_{L+1} = I_L$. Consider a subset $Y \subseteq S_{\leq j+1} \setminus T$ and denote by $F \triangleq \{(y,t) \in H \colon y \in Y, t \in$*

$T\}$ *the set arcs from* Y *to* T *and* $\bar{F} \triangleq \{(v,y) \in H \colon v \in T, y \in Y\}$ *the set of arcs from* T *to* Y. *Then,*

$$\Gamma_{E(T)\cup F\cup\bar{F}} + \sum_{v\in T} \frac{\Psi_v}{w_v} \geq \left(\rho_{k,\varepsilon_{j-1}} - k\varepsilon_{j-1}\right) w(T) + (1-\varepsilon_{j-1}) \sum_{(u,v)\in E(T)} w_u + \sum_{v\in T} \sum_{o\in C_v} \sum_{i=1}^{L} w(N(o,S_{i+1})) + \sum_{(u,v)\in F} w_u \left(1 - \varepsilon_{j-1} - \frac{w_u}{w_v} \frac{\varepsilon_{j-1}}{1-\varepsilon_{j-1}}\right) + \sum_{(u,v)\in\bar{F}} \frac{w_u^2}{w_v}.$$

Proof of Lemma 3.4.6. The proof combines Lemma 3.4.3 and Lemma 3.4.5. For simplicity, let $\text{LHS} \triangleq \Gamma_{E(T)\cup F\cup\bar{F}} + \sum_{v\in T} \frac{\Psi_v}{w_v}$ be the left-hand side of the equation in Lemma 3.4.6 and let $X \triangleq T \cup Y$. By definition of Ψ and Γ, we recall that

$$\text{LHS} = \left[\Gamma_{E(T)\cup F\cup\bar{F}} + \sum_{v\in T} \frac{\Theta_v(X)}{w_v}\right] + \left[\sum_{v\in T} \sum_{o\in C_v} \frac{(w_v - w_o)^2 + \theta_{v,o}(A \setminus X)}{w_v}\right].$$

We apply Lemma 3.4.5 to the first bracketed term and Lemma 3.4.3 to the second term. Thus,

$$\begin{aligned}\text{LHS} \geq \sum_{(u,v)\in E(T)\cup F} w_u + \sum_{(u,v)\in\bar{F}} \frac{w_u^2}{w_v} + \left(\rho_{k,\varepsilon_{j-1}} - k\varepsilon_{j-1}\right) w(T) - \frac{\varepsilon_{j-1}}{1-\varepsilon_{j-1}} \sum_{(y,x)\in F} \frac{w_y^2}{w_x} \\ + \sum_{v\in T} \sum_{o\in C_v} \sum_{i=1}^{L} \varepsilon_{i-1} w(N(o,S_{i+1})) - \varepsilon_{j-1} \sum_{(u,v)\in E(T)\cup F} w_u,\end{aligned}$$

The inequality then follows by grouping the sums over the same sets. □

In the next lemma, we provide a complement to Lemma 3.4.6 when $j = 0$.

Lemma 3.4.7. *Let* $T \subseteq S_1$ *be a root tree in* $\mathcal{T}$ *(see Equation 3.1). Then,*

$$\Gamma_{E(T)} + \sum_{v\in T} \frac{\Psi_v}{w_v} \geq \sum_{(u,v)\in E(T)} w_u + \sum_{v\in T} \sum_{o\in C_v} \sum_{i=1}^{L} \varepsilon_{i-1} w(N(o,S_{i+1})).$$

Proof of Lemma 3.4.7. The proof combines Lemma 3.4.4 and Lemma 3.4.5. By definition of Ψ and Γ, we recall that

$$\Gamma_{E(T)} + \sum_{v\in T} \frac{\Psi_v}{w_v} = \left[\Gamma_{E(T)} + \sum_{v\in T} \frac{\Theta_v(T)}{w_v}\right] + \left[\sum_{v\in T} \sum_{o\in C_v} \frac{(w_v - w_o)^2 + \theta_{v,o}(A \setminus T)}{w_v}\right]$$

We apply Lemma 3.4.5 to the first bracketed term (with $Y = \emptyset$) and Lemma 3.4.4 to the second term to obtain

$$\Gamma_{E(T)} + \sum_{v\in T} \frac{\Psi_v}{w_v} \geq \sum_{(u,v)\in E(T)} w_u + \sum_{v\in T} \sum_{o\in C_v} \sum_{i=1}^{L} \varepsilon_{i-1} w(N(o,S_{i+1})).$$ □

3.5 Root Tree

The goal of this section is to prove that the vertices in a root tree $T \subseteq D_j$ of size t receives an amount of slack equal to $\chi_{j-1}^{(t)} \triangleq \frac{t-1}{t}(1-\varepsilon_{j-1})^5 + \rho_{k,\varepsilon_{j-1}}$ for $j \geq 2$ and $\chi_{-1}^{(t)} \triangleq \frac{t-1}{t}(1-\varepsilon_0)^2$ for $j = 1$. Lemma 3.5.1 deals with the case $j \geq 2$.

Lemma 3.5.1. *Let $j \in [1, L]$ and let $T \triangleq T_{j+1} \subseteq D_{j+1}$ be the root tree in $\mathcal{T}$ (see Equation (3.1)), where $D_{L+1} = I_L$. Suppose that T has size t. Then,*

$$\Gamma_{E(T)} + \sum_{v \in T} \frac{\Psi_v}{w_v} \geq \left(\chi_{j-1}^{(t)} - \varepsilon_{j-1}k\right) w(T) + \sum_{v \in T} \sum_{o \in C_v} \left[\sum_{i=1}^{L} \varepsilon_{i-1} w(N(o, S_{i+1}))\right],$$

where $\chi_{j-1}^{(t)} \triangleq \frac{t-1}{t}(1-\varepsilon_{j-1})^5 + \rho_{k,\varepsilon_{j-1}}$.

In short, Lemma 3.5.1 shows that the improvement over the factor $\frac{k+1}{2}$ for vertices in a root tree is equal to $\chi_{j-1}^{(t)}$. In fact, the term $k\varepsilon_{j-1}$ will cancel out in the final analysis. The parameter $\chi_{j-1}^{(t)}$ is made of two terms: $\frac{t-1}{t}(1-\varepsilon_{j-1})^5$ and $\rho_{k,\varepsilon_{j-1}}$. The first term is due to the absence of improving swaps in a tree of size t. Recall that the ratio of the weight of two endpoints of an arc in T is at least $1-\varepsilon_j$. The second term $\rho_{k,\varepsilon_{j-1}}$ simply follows from the fact $T \subseteq I_{j-1}$ is *isolated* in the previous layer.

Proof of Lemma 3.5.1. The proof of Lemma 3.5.1 is obtained by giving a lower bound on the expression after using Lemma 3.4.6. For simplicity, we denote LHS $\triangleq \Gamma_{E(T)} + \sum_{v \in T} \frac{\Psi_v}{w_v}$. We use Lemma 3.4.6 (with $Y = \emptyset$) which results in the following bound

$$\text{LHS} \geq \left(\rho_{k,\varepsilon_{j-1}} - k\varepsilon_{j-1}\right) w(T) + (1-\varepsilon_{j-1}) \sum_{(u,v) \in E(T)} w_u + \sum_{v \in T} \sum_{o \in C_v} \sum_{i=1}^{L} \varepsilon_{i-1} w(N(o, S_{i+1})).$$

The final step is show that $\sum_{(u,v) \in E(T)} w_u \geq \frac{t-1}{t}(1-\varepsilon_{j+1})w(T)$ which follows from Lemma 2.4.7 (Chapter 2) since $w_u \geq (1-\varepsilon_{j+1})w_v$ for every arc (u,v) in $E(T) \subseteq H_{\leq j+1}$. Therefore, we get

$$\begin{aligned}
\text{LHS} &\geq \left(\frac{t-1}{t}(1-\varepsilon_{j-1})(1-\varepsilon_{j+1}) + \rho_{k,\varepsilon_{j-1}} - k\varepsilon_{j-1}\right) w(T) + \sum_{v \in T} \sum_{o \in C_v} \sum_{i=1}^{L} w(N(o, S_{i+1})) \\
&= \left(\chi_{j-1}^{(t)} - k\varepsilon_{j-1}\right) w(T) + \sum_{v \in T} \sum_{o \in C_v} \sum_{i=1}^{L} w(N(o, S_{i+1})),
\end{aligned}$$

where in the second equation we used that $\varepsilon_{j+1} = (1-\varepsilon_{j-1})^4$ for all $j \geq 1$. □

Lemma 3.5.1 holds for rooted trees $T \subseteq D_{j+1}$ for $j \geq 1$. The next lemma deals with the rooted trees in D_1.

Lemma 3.5.2. *Let $T \triangleq T_1 \subseteq D_1$ be the root tree in $\mathcal{T}$ from Equation (3.1) of size $|T| = t$. Define $\chi_{-1}^{(t)} \triangleq \frac{t-1}{t}(1-\varepsilon_0)^2$. Then,*

$$\Gamma_{E(T)} + \sum_{v \in T} \frac{\Psi_v}{w_v} \geq \chi_{-1}^{(t)} w(T) + \sum_{v \in T} \sum_{o \in C_v} \left[\sum_{i=1}^{L} \varepsilon_{i-1} w(N(o, S_{i+1})) \right],$$

Proof of Lemma 3.5.2. The proof of Lemma 3.5.2 is obtained by giving a lower bound on the expression after using Lemma 3.4.7. For simplicity, we denote LHS $\triangleq \Gamma_{E(T)} + \sum_{v \in T} \frac{\Psi_v}{w_v}$. We use Lemma 3.4.7 which gives the following bound

$$\text{LHS} \geq \sum_{(u,v) \in E(T)} w_u + \sum_{v \in T} \sum_{o \in C_v} \sum_{i=1}^{L} w(N(o, S_{i+1})). \tag{3.8}$$

It remains to bound the first term of the right-hand side. Since $w_u \geq (1-\varepsilon_0)^2 w_v$ for all arcs (u, v) in $E(T) \subseteq H_1$, we can apply Lemma 2.4.7 to the tree T of size t to get that

$$\sum_{(u,v) \in E(T)} w_u \geq \frac{t-1}{t}(1-\varepsilon_0)^2 w(T).$$

The lemma then finishes by substituting the above expression into Equation (3.8). □

3.6 Pendant Tree

In this section we analyze the slack that vertices in pendant trees receive. We consider a pendant tree $T_{p,m} \subseteq P_p$ that belongs to $\mathcal{T}$ (see Equation (3.1)) which is attached to $T_{j+1} \cup \bigcup_{q=j+2}^{p-1} \bigcup_{r=1}^{r_q} T_{q,r}$ using a bridge arc $e_{p,m} = (t, s)$. To ease the notation, we will let $T = T_{p,m}$ be the pendant tree in this section.

Lemma 3.6.1 is equivalent to Lemma 3.5.1 for pendant trees. Brief modifications of Lemma 3.5.1 could be used to handle the pendant trees. However, obtained numerical results would be worse than the one claimed in the introduction especially for low values of k. To obtain the desired results, we adopt a more intricate approach. More precisely, we will use Lemma 3.4.6 with a carefully chosen set Y that includes the endpoint of the bridge arc that lies outside T.

Lemma 3.6.1. *Let $T \subseteq P_{j+1}$ be a pendant tree of some tree $\mathcal{T}$ from Equation (3.1). Denote $E'_T = E(T) + (t, s)$, where $e = (t, s)$ is the bridge arc. Then,*

$$\Gamma_{E'_T} + \sum_{v \in T} \frac{\Psi_v}{w_v} \geq (\nu_{j-1} - k\varepsilon_{j-1})\, w(T) + \sum_{v \in T} \sum_{o \in C_v} \left[\sum_{i=1}^{L} \varepsilon_{i-1} w(N(o, S_{i+1})) \right],$$

where $\nu_{j-1} \triangleq \rho_{k,\varepsilon_{j-1}} + (1-\varepsilon_{j-1})^5 \big(1 - (1-\varepsilon_{j-1})^2 \varepsilon_{j-1}\big)$.

In short, Lemma 3.6.1 shows that the improvement over the factor $\frac{k+1}{2}$ for vertices in a pendant tree in S_{j+1} is at least equal to ν_{j-1}. The proof in fact demonstrates that the lemma is tight when the pendant tree consists of a single vertex.

Proof of Lemma 3.6.1. The proof again crucially rely on Lemma 3.4.6, where we set Y as the endpoint of the bridge arc that is outside T. More formally, we distinguish two cases depending on the orientation of the bridge arc.
Case 1: Suppose that $s \in T$ and $t \in A \setminus T$. The orientation of the arc implies that $w_s \geq w_t$. Let $Y = \{t\}$. We define $F \triangleq \{(t,v) \in H \colon v \in T\}$, and $\bar{F} \triangleq \{(v,t) \in H \colon v \in T\}$ to be set arcs between T and Y. By the second bullet point of Corollary 3.2.8, and since $T \cup Y \subseteq \mathcal{T}$, it implies that the induced subgraph $H[T \cup Y]$ is a tree and thus $F = \{(t,s)\}$ and $\bar{F} = \emptyset$. We apply Lemma 3.4.6 with Y, F, $\bar{F}$ defined previously and get that

$$\Gamma_{E'_T} + \sum_{v \in T} \frac{\Psi_v}{w_v} \geq \left(\rho_{k,\varepsilon_{j-1}} - k\varepsilon_{j-1}\right) w(T) + (1-\varepsilon_{j-1}) \sum_{(u,v) \in E(T)} w_u + \sum_{v \in T} \sum_{o \in C_v} \sum_{i=1}^{L} w(N(o, S_{i+1})) + w_t \left(1 - \varepsilon_{j-1} - \frac{w_t}{w_s} \frac{\varepsilon_{j-1}}{1-\varepsilon_{j-1}}\right).$$

We claim the following two identities which we will prove separately.

Claim 3.6.2. *If $w_t \in \left[(1-\varepsilon_{j-1})^4; (1-\varepsilon_{j-1})^2\right] w_s$, we have that $w_t \left(1 - \varepsilon_{j-1} - \frac{w_t}{w_s} \frac{\varepsilon_{j-1}}{1-\varepsilon_{j-1}}\right) + \rho_{k,\varepsilon_{j-1}} w_s \geq \nu_{j-1} w_s$.*

Claim 3.6.3. *For any tree $T \subseteq S_{\leq j+1}$ and $s \in T$, the following inequality holds $\rho_{k,\varepsilon_{j-1}} w(T-s) + (1-\varepsilon_{j-1}) \sum_{(u,v) \in E(T)} w_u \geq \nu_{j-1} w(T-s)$.*

Therefore, substituting both Claim 3.6.2 (using that the bridge arc lies in H_{j+1} for $j \geq 1$) and Claim 3.6.3 into the previous equation finishes the proof of the first case.
Case 2: Suppose now that $s \in A \setminus T$ and $t \in T$. Again, we let $Y = \{s\}$ and define $F \triangleq \{(s,v) \in H \colon v \in T\}$, and $\bar{F} \triangleq \{(v,s) \in H \colon v \in T\}$. By the second bullet point of Corollary 3.2.8, and since $T \cup Y \subseteq \mathcal{T}$, it implies that the induced subgraph $H[T \cup Y]$ is a tree and thus $F = \emptyset$ and $\bar{F} = \{(t,s)\}$. We apply Lemma 3.4.6 with Y, F, $\bar{F}$ defined previously to obtain that

$$\Gamma_{E'_T} + \sum_{v \in T} \frac{\Psi_v}{w_v} \geq \left(\rho_{k,\varepsilon_{j-1}} - k\varepsilon_{j-1}\right) w(T) + (1-\varepsilon_{j-1}) \sum_{(u,v) \in E(T)} w_u + \sum_{v \in T} \sum_{o \in C_v} \sum_{i=1}^{L} w(N(o, S_{i+1})) + \frac{w_t^2}{w_s}. \tag{3.9}$$

The bridge arc (t,s) lies in H_{j+1}, which implies for $j \geq 1$ that $w_t \geq (1-\varepsilon_{j+1})w_s = (1-\varepsilon_{j-1})^4 w_s$. Thus,

$$\begin{aligned}
&\rho_{k,\varepsilon_{j-1}} w(T) + (1-\varepsilon_{j-1}) \sum_{(u,v)\in E(T)} w_u + \frac{w_t^2}{w_s} \\
\geq &\left[\left(\rho_{k,\varepsilon_{j-1}} + (1-\varepsilon_{j-1})^4\right) w_t\right] + \left[\rho_{k,\varepsilon_{j-1}} w(T-t) + (1-\varepsilon_{j-1}) \sum_{(u,v)\in E(T)} w_u\right] \\
\geq &\nu_{j-1} w(T),
\end{aligned}$$

where the last inequality follows by observing that $\rho_{k,\varepsilon_{j-1}} + (1-\varepsilon_{j-1})^4 \geq \nu_{j-1}$ in the first square bracketed expression and by using Claim 3.6.3 in the second. Replacing the above computation in Equation (3.9) concludes the proof. □

We conclude this section with the proofs of Claim 3.6.2 and Claim 3.6.3.

Proof of Claim 3.6.2. For clarity, we will let $\varepsilon_{j-1} = \varepsilon$, and let $x = \frac{w_t}{w_s}$ be the ratio between the weight of the endpoints of the bridge arc. The parameter x takes value in the interval $[(1-\varepsilon)^4, (1-\varepsilon)^2]$. To prove the claim, it is sufficient to show that $g(x) \triangleq x(1-\varepsilon - x\frac{\varepsilon}{1-\varepsilon})$ is a decreasing function of x for $x \in [(1-\varepsilon)^4, (1-\varepsilon)^2]$. Note that g is a quadratic function with a global maximum at $x^* = \frac{(1-\varepsilon)^2}{2\varepsilon}$ monotically increasing until reaching x^* and then monotically decreasing. In particular, if $\varepsilon \leq 1/2$, then $(1-\varepsilon)^2 \leq x^*$. It implies that when $\varepsilon \leq 1/2$, then minimum is attained at $x = (1-\varepsilon)^4$. On the other hand, if $\varepsilon \geq 1/2$, then the minimum is at either limit of the interval $[(1-\varepsilon)^4, (1-\varepsilon)^2]$. We show that the difference between $g((1-\varepsilon)^2)$ and $g((1-\varepsilon)^4)$ is positive meaning that the minimum is reached when $x = (1-\varepsilon)^4$.

$$\begin{aligned}
g((1-\varepsilon)^2) - g((1-\varepsilon)^4) &= (1-\varepsilon)^2(1-\varepsilon-\varepsilon(1-\varepsilon)) - (1-\varepsilon)^4(1-\varepsilon-\varepsilon(1-\varepsilon)^3) \\
&= (1-\varepsilon)^4 - (1-\varepsilon)^4(1-\varepsilon-\varepsilon(1-\varepsilon)^3) \geq 0,
\end{aligned}$$

where the last inequality follows from the fact that $(1-\varepsilon-\varepsilon(1-\varepsilon)^3) \leq 1$. □

Proof of Claim 3.6.3. To see that the equation holds, take the tree T and set s as the root of the tree. In this proof, we forget about the orientation of the arcs and treat them as edges. We create a mapping $\sigma : E \to T - s$ that maps each edge to a vertex. Given $(u,v) \in E(T)$ such that $u = \text{child}(v)$, we assign $\sigma((u,v)) = u$. In this way, each vertex except s has an edge assigned to it. Therefore,

$$(1-\varepsilon_{j-1}) \sum_{(u,v)\in E(T)} w_u \geq (1-\varepsilon_{j-1})^5 \sum_{(u,v)\in E(T)} w_{\sigma((u,v))} = (1-\varepsilon_{j-1})^5 w(T-s),$$

where in the second inequality, we used that $T \subseteq S_{j+1}$ with $j \geq 1$, so the ratio between the weight of the two endpoints is at least $1 - \varepsilon_{j+1} = (1 - \varepsilon_{j-1})^4$. The proof finishes by noting that $(1 - \varepsilon_{j-1})^5 \geq (1 - \varepsilon_{j-1})^5(1 - \varepsilon_{j-1} - \varepsilon_{j-1}(1 - \varepsilon_{j-1})^2)$, so

$$\rho_{k,\varepsilon_{j-1}} w(T - s) + (1 - \varepsilon_{j-1}) \sum_{(u,v)\in E(T)} w_u \geq \nu_{j-1} w(T - s). \qquad \square$$

3.7 Final Results and Conclusion

We finally prove the main result of this chapter. Theorem 3.7.1 states that the final approximation ratio is equal to $\frac{k+1}{2}$ from which we deduct the slack that each vertex receives.

Theorem 3.7.1. *Let A be a locally optimal solution of Algorithm 1 with respect to s-exchanges where $s \geq (k^2\ell)^L \sum_{s=0}^{2\ell-2} (k\ell)^{2s}$. Then,*

$$w(O) \leq \frac{1}{2}\left[k + 1 - \min\left\{\{\chi_{i-1}^{(\ell)}\}_{i=0}^{L-1}, \{\nu_{i-1}\}_{i=1}^{L-1}, \rho_{k,\varepsilon_{L-1}}\right\}\right] w(A).$$

Proof of Theorem 3.7.1. Corollary 3.2.8 partitions A into vertex disjoint trees $\mathcal{T}$. We use Lemma 3.5.1 and Lemma 3.6.1 to bound the slack that vertices in $\mathcal{T}$ receive. Given $j \in [0, L]$, the vertices in A are covered by vertex disjoint trees of the form:

$$\mathcal{T} \triangleq T_{j+1} \cup \bigcup_{p=j+2}^{L} \bigcup_{m=1}^{m_p} T_{p,m}, \tag{3.10}$$

where $T_{j+1} \subseteq D_{j+1}$ and $T_{p,m} \subseteq P_p$, respectively. Throughout the proof we let $D_{L+1} = I_L$. Let $E_{p,m} \triangleq E(T_{p,m}) + e_{p,m}$ where $e_{p,m}$ is the bridge edge of the pendant tree $T_{p,m}$. Applying Corollary 3.4.2, we get:

$$\sum_{v\in\mathcal{T}} \left[\frac{\Delta_v}{w_v} + \frac{\Psi_v}{w_v}\right] \geq \left[\Gamma_{E(T_{j+1})} + \sum_{v\in T_{j+1}} \frac{\Psi_v}{w_v}\right] + \sum_{p,m} \left[\Gamma_{E_{p,m}} + \sum_{v\in T_{p,m}} \frac{\Psi_v}{w_v}\right].$$

Assume first that $j \neq L$, in which case by Corollary 3.2.8, the tree T_{j+1} has size at least ℓ. We apply Lemma 3.5.1 (or 3.5.2 if $j = 0$), and Lemma 3.6.1 to the first and second squared bracket expression respectively. We obtain

$$\begin{aligned}\sum_{v\in\mathcal{T}} \left[\frac{\Delta_v}{w_v} + \frac{\Psi_v}{w_v}\right] \geq \left(\chi_{j-1}^{(\ell)} - \varepsilon_{j-1}k\right) w(T_{j+1}) + \sum_{p=j+1}^{L-1} \sum_{m=1}^{m_p} (\nu_{p-1} - \varepsilon_{p-1}k)\, w(T_{p+1,m}) \\ + \sum_{v\in\mathcal{T}} \sum_{o\in C_v} \left[\sum_{i=1}^{L} \varepsilon_{i-1} w(N(o, S_{i+1}))\right],\end{aligned} \tag{3.11}$$

where we used the monotonicity of $\chi_{j-1}^{(t)}$ with respect to t. Thus, $\chi_{j-1}^{(|T_j|)} \geq \chi_{j-1}^{(\ell)}$ since $|T_j| \geq \ell$. In the case $j = L$, the expression of $\mathcal{T}$ simplifies to $\mathcal{T} = T$ for some tree $T \subseteq D_{L+1} = I_L$ of size at most $\ell - 1$. Applying Lemma 3.5.1 and using that $|T| \geq 1$, we get that

$$\begin{aligned}\sum_{v\in T}\left[\frac{\Delta_v}{w_v}+\frac{\Psi_v}{w_v}\right] &\geq \Gamma_{E(T)} + \sum_{v\in T}\frac{\Psi_v}{w_v}\\ &\geq \left(\chi_{L-1}^{(1)} - \varepsilon_{L-1}k\right) w(T) + \sum_{v\in T}\sum_{o\in C_v}\left[\sum_{i=1}^{L}\varepsilon_{i-1}w(N(o,S_{i+1}))\right]\\ &= \left(\rho_{k,\varepsilon_{L-1}} - \varepsilon_{L-1}k\right) w(T) + \sum_{v\in T}\sum_{o\in C_v}\left[\sum_{i=1}^{L}\varepsilon_{i-1}w(N(o,S_{i+1}))\right], \qquad (3.12)\end{aligned}$$

where the second inequality follows from the monotonicity of $\chi_{j-1}^{(t)}$. Since $\{S_i\}_{i=1}^{L+1}$ partitions the vertex set A, where we recall that $S_{L+1} = I_L$, Berman's analysis 2.3.1 shows:

$$\begin{aligned}2w(O) &\leq w(A) + \sum_{o\in O} w(N(o,A)) - \sum_{v\in A}\left[\frac{\Delta_v}{w_v}+\frac{\Psi_v}{w_v}\right]\\ &= w(A) + \sum_{o\in O}\sum_{i=0}^{L} w(N(o,S_{i+1})) - \sum_{v\in A}\left[\frac{\Delta_v}{w_v}+\frac{\Psi_v}{w_v}\right].\end{aligned}$$

By Corollary 3.2.8 the set $\{\mathcal{T}\}$ partitions A into vertex disjoint trees. Additionally, the set of talons $\{C_a\}_{a\in A}$ forms a partition of O. Substituting Equation (3.11) and (3.12) into the previous computation, we get that

$$\begin{aligned}2w(O) \leq\ & w(A) + \sum_{o\in O}\sum_{i=1}^{L}(1-\varepsilon_{i-1})w(N(o,S_{i+1})) + \sum_{o\in O} w(N(o,S_1))\\ & - \chi_{-1}^{(\ell)}w(D_1) - \sum_{i=1}^{L-1}\left(\chi_{i-1}^{(\ell)} - k\varepsilon_{i-1}\right) w(D_{i+1}) - \sum_{i=1}^{L-1}\left(\nu_{i-1} - k\varepsilon_{i-1}\right) w(P_{i+1})\\ & - \left(\rho_{k,\varepsilon_{L-1}} - \varepsilon_{L-1}k\right) w(I_L).\end{aligned}$$

Since the conflict graph $G[A \cup O]$ is $(k+1)$-claw free, each vertex $a \in S_{i+1}$ for some i has at most k neighbors in O. Thus, $\sum_{o\in O} w(N(o,S_{i+1})) \leq kw(S_{i+1})$. Moreover, by definition of S_i we know that $S_i = D_i \sqcup P_i$, and $S_{L+1} = I_L$ and $\bigsqcup_{i=1}^{L+1} S_i = A$. Thus,

$$\begin{aligned}2w(O) &\leq (k+1)w(A) - \sum_{i=0}^{L-1}\chi_{i-1}^{(\ell)}w(D_{i+1}) - \sum_{i=1}^{L-1}\nu_{i-1}w(P_{i+1}) - \rho_{k,\varepsilon_{L-1}}w(I_L)\\ &\leq (k+1)w(A) - \min\left\{\{\chi_{i-1}^{(\ell)}\}_{i=0}^{L-1}, \{\nu_{i-1}\}_{i=1}^{L-1}, \rho_{k,\varepsilon_{L-1}}\right\} w(A),\end{aligned}$$

where the last inequality follows by using that $A = \bigsqcup_{i=1}^{L} D_i \sqcup \bigsqcup_{i=2}^{L} P_i \sqcup I_L$ and by taking the minimum over all the terms. □

3.7.1 Exact and asymptotic approximation ratio

We close this chapter with the exact approximation ratio attained by Algorithm 1 for all values of k. The swap size used by Algorithm 1 is equal to swap size $s = O((k\ell)^{O(\text{POLY}(L,\ell))})$, which is the upper bound on the size of a tree found in Corollary 3.2.8. For $k = 3$, Theorem 3.7.7 will prove a tight approximation factor of $\sqrt{3}$. For $k \geq 7$, Theorem 3.7.6 will prove a tight approximation factor of $\frac{k+\delta}{2}$ for any $\delta \geq 0$. It matches the result of Neuwohner [Neu22] who proves that even with $s = O(\log(n))$ where n is the number of sets, Algorithm 1 has approximation ratio at most $\frac{k}{2}$. For $k = 4, 5, 6$, Theorem 3.7.8 yields an approximation factor of $2.0898, 2.5359, 3.0082$, respectively. Table 3.1 summarizes these theorems.

k	APX	τ_k	ε	ℓ	L
3	$\sqrt{3}$	$2-\sqrt{3}$	$1-1/\sqrt{3}$	4	3
4	2.0898	0.4102	$1/2$	$O(1/\epsilon')$	4
5	2.5359	0.4641	0.253609	$O(1/\epsilon')$	4
6	3.0082	0.4918	0.207662	$O(1/\epsilon')$	6
≥ 7	$\frac{k+\epsilon'}{2}$	$\frac{(1-\epsilon')}{2}$	$1/2$	$2/\epsilon'$	$O(\log_2(\log_2(1/\epsilon')))$

Tab. 3.1: Summary of the results for different values of k. The parameter τ_k, measures the improvement over $\frac{k+1}{2}$, i.e. $\tau_k = \frac{k+1}{2} - \text{APX}$. Here, ℓ bounds the size of components in D_j, L is the number of layers and $\varepsilon = \varepsilon_{L-1}$ is the value of ε_{L-1} that we need to set.

Remark 3.7.2. The values chosen in the Table 3.1 might seem a bit arbitrary for $k = 4, 5, 6$. Numerical experiments indicate that the approximation ratio is bounded away from 3.0081, 2.5358, 2.0844 for $k = 6, 5, 4$ when ℓ and L are arbitrarily large.

The approximation ratio of Algorithm 1 depends on key quantities denoted by $\chi_{j-1}^{(\ell)}$, ν_{j-1} and $\rho_{k,\varepsilon_{L-1}}$. Let's recall the exact values of the parameters:

- $\rho_{k,\varepsilon_{L-1}} \triangleq k\varepsilon_{L-1} - \frac{\varepsilon_{L-1}}{1-\varepsilon_{L-1}}$.
- $\chi_{j-1}^{(\ell)} \triangleq \frac{\ell-1}{\ell}(1-\varepsilon_{j-1})^5 + \rho_{k,\varepsilon_{j-1}}$ for $j = 1, \ldots, L-1$
- $\nu_{j-1} \triangleq \rho_{k,\varepsilon_{j-1}} + (1-\varepsilon_{j-1})^5(1-(1-\varepsilon_{j-1})^2\varepsilon_{j-1})$ for $j = 1, \ldots, L-1$.
- $\chi_{-1}^{(\ell)} \triangleq \frac{\ell-1}{\ell}(1-\varepsilon_0)^2 = \frac{\ell-1}{\ell}(1-\varepsilon_1) = \frac{\ell-1}{\ell}(1-\varepsilon_{L-1})^{\frac{1}{2^{L-1}}}$.

Additionally, the parameters ε_i are set so that $\varepsilon_{-1} = 0$, and $\varepsilon_i = 2\varepsilon_{i-1} - \varepsilon_{i-1}^2$ for $i = 1, \ldots, L$.

Matching Hurkens and Schrijver's result for large values of k

When $k \geq 7$, we show that the parameters $\chi_{j-1}^{(\ell)}$ and ν_{j-1} are greater than 1 for $j = 1, \ldots, L-1$, and as ℓ tends to ∞. Therefore, the vertices receiving these terms as slack already improve over the factor $\frac{k}{2}$. We will use the following three claims that we prove later in this section.

Claim 3.7.3. *For a fixed value of k, the function $\rho_{k,x} \triangleq kx - \frac{x}{1-x}$ is an increasing function in the interval $x \in [0, 1 - \frac{1}{\sqrt{k}}]$.*

Claim 3.7.4. *For $k \geq 6$, the function $\chi_k(x) \triangleq (1-x)^5 + \rho_{k,x}$ is greater than 1 for $x \in [0, 1 - \frac{1}{\sqrt{k}}]$.*

Claim 3.7.5. *For $k \geq 7$, the function $\nu_k(x) \triangleq (1-x)^5(1-(1-x)^2 x) + \rho_{k,x}$ is greater than 1 for $x \in [0, 1 - \frac{1}{\sqrt{k}}]$.*

Equipped with the above claims, we prove Theorem 3.7.6. It improves upon the result of Neuwohner [Neu22] who obtained similar guarantees but only as $k \to \infty$. Given Neuwohner's [Neu23] recent result, it yields an improvement for all values of $k \leq 371$. Indeed, for large values of k, the approximation factor is equal to $0.4986(k+1)+0.0208$ which is asymptotically better than our approximation factor.

Theorem 3.7.6. *Given any $\delta \in (0, 1/2)$, and any $k \geq 7$, Algorithm 1 performing s-exchanges for $s \geq (k^2\ell)^L \sum_{s=0}^{2\ell-2} (k\ell)^{2s}$ with $\ell = 2/\delta$ and $L = O(\log_2(\log_2(1/\delta)))$ has approximation ratio equal to:*

$$w(O) \leq \frac{k+\delta}{2} w(A). \tag{3.13}$$

Proof of Theorem 3.7.6. Let $\epsilon' \triangleq \delta/2 = 1/\ell$. To compute the exact approximation ratio from Theorem 3.7.1, it is sufficient to bound the term:

$$\min\left\{ \{\chi_{i-1}^{(\ell)}\}_{i=0}^{L-1}, \{\nu_{i-1}\}_{i=1}^{L-1}, \rho_{k,\varepsilon_{L-1}} \right\}. \tag{3.14}$$

When $k \geq 7$, this term simplifies. In fact, by Claim 3.7.5, all the terms in the sequence $\{\nu_{i-1}\}_{i=1}^{L-1}$ are greater than 1, whereas $\chi_{-1}^{(\ell)} = \frac{\ell-1}{\ell}(1-\varepsilon_0)^2 \leq 1$. Thus, it is sufficient to compute

$$\min\left\{ \{\chi_{i-1}^{(\ell)}\}_{i=0}^{L-1}, \rho_{k,\varepsilon_{L-1}} \right\}.$$

Using Claim 3.7.4 for $k \geq 6$, and $i = 1, \ldots, L-1$, we get that: $\chi_{i-1}^{(\ell)} \triangleq \frac{\ell-1}{\ell}(1-\varepsilon_{i-1})^5 + \rho_{k,\varepsilon_{i-1}} \geq 1 - \frac{1}{\ell}(1-\varepsilon_{i-1}) \geq 1 - \frac{1}{\ell} = 1 - \epsilon'$, where the last inequality is because $\varepsilon_{i-1} \in [0,1]$. Setting $\varepsilon_{L-1} = 1/2$ for all values of k, we get that $\rho_{k,\varepsilon_{L-1}} \geq \rho_{7,1/2} = 5/2$. Substituting both terms into (3.14), it is sufficient to bound

$$\min\left\{ \chi_{-1}^{(\ell)}, 1-\epsilon' \right\}.$$

Recall that $\chi_{-1}^{(\ell)} \triangleq \frac{\ell-1}{\ell}(1-\varepsilon_0)^2$. There, we will use the fact that the number of layers L is large to reduce ε_0. Recall that $(1-\varepsilon_0)^{2^L} = (1-\varepsilon_{L-1}) = 1/2$ since we set $\varepsilon_{L-1} = 1/2$. Hence, $\varepsilon_0 = 1 - (1/2)^{1/2^L}$. Setting, $L \triangleq 1 - \log_2(\log_2((1-\epsilon')^{-1})) = O(\log_2(\log_2(1/\epsilon')))$ where the second equality holds for $\epsilon' \in (0, 1/2)$, we have that

$$(1-\varepsilon_0)^2 = (1-\varepsilon_{L-1})^{1/2^{L-1}} = (1/2)^{1/2^{1-\log_2(\log_2((1-\epsilon')^{-1}))-1}} = 1-\epsilon'.$$

Therefore, we have that: $\chi_{-1}^{(\ell)} \triangleq \frac{\ell-1}{\ell}(1-\varepsilon_0)^2 = \frac{\ell-1}{\ell}(1-\epsilon') \geq 1 - \epsilon' - \frac{1}{\ell} = 1 - 2\epsilon'$. □

Proof of Claim 3.7.3. The derivative of $\rho_{k,x}$ with respect to x is equal to $\frac{k(x-1)^2-1}{(1-x)^2}$. The derivative is equal to 0 at the point $x^* = 1 - \frac{1}{\sqrt{k}}$ and is greater than 0 for $x \in [0, x^*]$. The second-order derivative with respect to x is equal to $\frac{2}{(x-1)^3}$ and is negative for $x \in [0, 1]$, thus $\rho_{k,x}$ is maximized at x^*. □

Proof of Claim 3.7.4. The derivative of $\chi_k(x)$ with respect to x is equal to $\frac{k(x-1)^2-5(x-1)^6-1}{(1-x)^2}$. We show that the numerator $k(x-1)^2 - 5(x-1)^6 - 1$ is greater than 0. Since $\chi_k(0) = 1$, it will show that the derivative is greater than 0 in the interval $[0, 1 - 1/\sqrt{k}]$, which in turns implies that $\chi_k(x)$ is always at least 1. Observe that $k(x-1)^2 - 5(x-1)^6 = (1-x)^2(k - 5 + 5(1-(1-x)^4)) \geq (1-x)^2(1 + 5(1-(1-x)^4))$, where we use that $k \geq 6$. The polynomial $(1-x)^2(1+5(1-(1-x)^4))-1 = -x(x-2)(5x^4-20x^3+35x^2-30x+9)$ has 4 real roots 0, 2, $\frac{1}{2}\left(2 - \sqrt{\frac{6}{\sqrt{5}} - 2}\right) \simeq 0.586$, and $\frac{1}{2}\left(2 + \sqrt{\frac{6}{\sqrt{5}} - 2}\right) \simeq 1.413$. It is positive between 0 and 0.586, and 1.413 and 2. Therefore, the derivative is greater than 0 for all values of $x \in [0, 0.586]$. For values of $x \in (0.586, 1 - 1/\sqrt{k}]$, we observe that $\rho_{6,0.586} \geq 1$ and $\rho_{k,x}$ is an increasing function of k, and x in the interval $x \in [0, 1 - \frac{1}{\sqrt{k}}]$ (Claim 3.7.3). □

Proof of Claim 3.7.5. Note that for any $x \in [0, 1]$, we have $(1 - x(1-x)^2) \geq (1-x)$ thus, we have that $\nu_k(x) \geq (1-x)^6 + \rho_{k,x}$. The derivative of the right-hand side is equal to $\frac{k(1-x)^2-6(1-x)^7-1}{(1-x)^2}$. We show that the numerator is greater than 0. Since $\nu_k(0) = 1$, it will prove that the derivative is greater than 0 in the interval $[0, 1 - 1/\sqrt{k}]$, which in turns implies that $\nu_k(x)$ is always at least 1. We write $k(1-x)^2-6(1-x)^7-1 = (1-x)^2(k-6+6(1-(1-x)^5))-1 \geq (1-x)^2(1 + 6(1-(1-x)^5)) - 1$, where we used that $k \geq 7$. The polynomial is positive between 0 and 0.62. It implies that $\nu_k(x)$ is increasing in the interval $x \in [0, 0.62]$. For values of $x \in (0.62, 1 - 1/\sqrt{k}]$, we note that $\nu_k(x) \geq \rho_{k,x} \geq \rho_{7,0.62} \geq 1$ for all $x \in (0.62, 1 - 1/\sqrt{k}]$ where we used that $\rho_{k,x}$ is an increasing function of x and k in the described interval (Claim 3.7.3). □

Small values of k

For lower values of k, computing the exact approximation ratio simply requires optimizing over ε, ℓ, and L. The approximation factor increases as L, and ℓ increases. However, unlike Theorem 3.7.6, we will fix L as a small constant.

Theorem 3.7.7. *For $k = 3$, Algorithm 1 running s-exchanges for $s \geq (k^2\ell)^L \sum_{s=0}^{2\ell-2} (k\ell)^{2s}$ with $\ell = 4$ and $L = 3$ has approximation ratio equal to:*

$$w(O) \leq \sqrt{3} w(A). \tag{3.15}$$

Theorem 3.7.7 is tight, in the sense that the w^2 local-search has a local gap of at least $\sqrt{k}$ even with unbounded swap size. To facilitate the computations, we chose to set $\ell = 4$. Setting $\ell = O(\varepsilon^{-1})$ yields the same guarantee but increases the slack for some vertices.

Proof of Theorem 3.7.7. For $k = 3$, we compute each term in the $\min\{\cdot\}$-term of Theorem 3.7.1 independently. Observe that we only need three layers, and $\ell = 4$. We set $\varepsilon_3 = 2/3$, thus $\varepsilon_2 = 1 - \sqrt{1-\varepsilon_3} = 1 - 1/\sqrt{3}$. We also have $\varepsilon_1 = 1 - 1/\sqrt[4]{3}$ and $\varepsilon_0 = 1 - 1/\sqrt[8]{3}$. Start by observing that $\rho_{3,\varepsilon_{L-1}} = \rho_{3,1-1/\sqrt{3}} = 4 - 2\sqrt{3} \geq 0.5358$. Secondly, straightforward computations show that $\nu_0 \geq 0.6764$, $\nu_1 \geq 0.5968$, and $\chi^{(4)}_{-1} \geq 0.5698$ $\chi^{(4)}_0 \geq 0.6151$, $\chi^{(4)}_1 \geq 0.5943$. All these terms are strictly greater than $\rho_{3,1-1/\sqrt{3}}$. Substituting each term in Theorem 3.7.1 yields the desired result. □

Theorem 3.7.8. *Algorithm 1 has approximation ratio* 2.0898, 2.5359, 3.0082 *for* $k = 4, 5, 6$ *respectively*

The reason behind the constant gap between our approximation ratio and the value $k/2$ is that the following functions: $\chi_k(x) \triangleq (1-x)^5 + \rho_{k,x}$ and $\nu_k(x) \triangleq (1-x)^5(1-(1-x)^2x) + \rho_{k,x}$ are not strictly greater than 1. In particular, $\chi_4(0.1439) = 0.867369$, $\chi_5(0.0625) = 0.97003$, and $\nu_4(0.143) = 0.818866$, $\nu_5(0.08) = 0.927497$, $\nu_6(0.03) = 0.983567$.

Proof of Theorem 3.7.8. **Case** $k = 4$**:** We set $\ell = 1/\epsilon'$, $L = 4$, and $\varepsilon_L = 0.75$. Computations show that $\rho_{4,\varepsilon_{L-1}} = 1$, and $\chi^{(\ell)}_{-1} \geq 0.8408$, $\chi^{(\ell)}_0 \geq 0.8898$, $\chi^{(\ell)}_1 \geq 0.8676$, $\chi^{(\ell)}_2 \geq 0.9341$ as $\ell \to \infty$. Alternatively, we could express each term with an additive loss of $O(\epsilon')$. Then, we proceed by computing ν's: $\nu_0 \geq 0.8446$, $\nu_1 \geq 0.8203$, $\nu_2 \geq 0.9082$. The minimum of all these term is ν_1. Thus, the approximation ratio is at least $(5 - \nu_1)/2 = 2.08985$.

Case $k = 5$**:** We set $\ell = 1/\epsilon'$, $L = 4$, and $\varepsilon_L = 0.4429$. Computations show that $\rho_{5,\varepsilon_{L-1}} = 0.9282$, and $\chi^{(\ell)}_{-1} \geq 0.9294$, $\chi^{(\ell)}_0 \geq 0.9751$, $\chi^{(\ell)}_1 \geq 0.9704$, $\chi^{(\ell)}_2 \geq 1.0041$ as $\ell \to \infty$. Alternatively, we could express each term with an additive loss of $O(\epsilon')$. Then, we proceed by computing ν's: $\nu_0 \geq 0.9474$, $\nu_1 \geq 0.9282$, $\nu_2 \geq 0.9552$. The minimum of all these term is $\nu_1 = \rho_{5,\varepsilon_{L-1}}$. Thus, the approximation ratio is at least $(6 - \nu_1)/2 = 2.5359$.

Case $k = 6$**:** We set $\ell = 1/\epsilon'$, $L = 6$, and $\varepsilon_L = 0.3722$. Computations show that $\rho_{6,\varepsilon_{L-1}} = 0.9838$, and $\chi^{(\ell)}_{-1} \geq 0.9855$, $\chi^{(\ell)}_{j-1} \geq 1.0004$ for all $j = 1, \ldots, L-1$ as $\ell \to \infty$. Alternatively, we could lower bound each term with an additive loss of $O(\epsilon')$. Then, we proceed by computing ν's: $\nu_{j-1} \geq 0.9837$ for all $j = 1, \ldots, L-1$ Thus, the approximation ratio is at least $(7 - 0.9837)/2 = 3.0082$. □

3.8 Reaching the local-gap instance

Theorem 3.7.7 already proves that the approximation factor is $\sqrt{3}$. However, it is still not clear that Figure 2.1 is indeed the worst-case instance. Theorem 3.7.7 demonstrates that the worst-case is when the slack that vertices receive is equal to $\rho_{3,1-1/\sqrt{3}} = 4 - 2\sqrt{3}$. It only happens for vertices in I_L. The set I_L is made of maximal connected components of size at most $\ell - 1$ in $H_{\leq L}$. Using Lemma 3.5.1, we observe that the vertices having a slack

term exactly equal to $\rho_{3,1-1/\sqrt{3}}$ are maximal connected component of size precisely 1 (thus, isolated vertices in $H_{\leq L}$). In Lemma 3.8.1, we deal with isolated vertices in $H_{\leq L}$. We show that only vertices $v \in I_L$ such that $|C_v| = 3$ have a slack equal to $\rho_{3,1-1/\sqrt{3}}$. Thus, it matches the instance in Figure 2.1.

Lemma 3.8.1. *Let $\{v\} \subseteq I_L$ be an isolated connected component of size 1 in $H_{\leq L}$. If $|C_v| \geq 1$, then*

$$\frac{\Psi_v}{w_v} \geq \varepsilon_{L-1} \sum_{o \in C_v} w(N(o, A)) + w_v \begin{cases} \rho_{k,\varepsilon_{L-1}} - \varepsilon_{L-1} k & \text{if } \varepsilon_{L-1} \leq 1 - \frac{1}{\sqrt{|C_v|}} \\ \left(\sqrt{|C_v|} - 1\right)^2 - \varepsilon_{L-1} |C_v| & \text{else} \end{cases}$$

If $|C_v| = 0$, then $\Psi_v / w_v \geq 0$.

Proof of Lemma 3.8.1. Throughout the proof let $\varepsilon_L \triangleq \varepsilon$ and $\delta \triangleq \varepsilon_{L-1}$. We start by observing that there is nothing to prove if $|C_v| = 0$, as Ψ_v / w_v is at least 0 by definition of Ψ_v. By Lemma 3.2.11, for any $u \in N_v^+ \setminus \{v\} \cap S_{j+1}$ for some $j \in \mathbb{N}$, we have

$$\frac{w_u}{w_v} \leq 1 - \varepsilon_{\max\{L,j\}} = 1 - \varepsilon.$$

For simplicity, we let $\eta \triangleq \delta \sum_{o \in C_v} w(N(o, A - v))$ and $\beta \triangleq \sum_{o \in C_v} w(N(o, A - v))$. Applying Lemma 3.3.1 with $\{v\} = T$, we get that

$$\sum_{o \in C_v} \frac{\theta_{v,o}}{w_v} \geq \sum_{o \in C_v} \varepsilon w(N(o, A - v)) = (\delta - \delta^2)\beta + \eta.$$

Next, we derive a lower bound on the value of β. For each $o \in C_v$, let $\alpha_o = w_o / w_v$. Using that the claw-swap centered at v is non-improving, we have that:

$$\begin{aligned} w_v^2 \sum_{o \in C_v} \alpha_o^2 &\leq w_v^2 + \sum_{o \in C_v} w^2(N(o, A - v)) \\ &\leq w_v^2 + w_v(1 - \varepsilon) \sum_{o \in C_v} w(N(o, A - v)) \\ &= w_v^2 + w_v(1 - \delta)^2 \beta. \end{aligned}$$

The last equality is since $(1 - \varepsilon) = (1 - \delta)^2$. From the previous computation, we obtain that $\frac{\beta}{w_v} \geq (1 - \delta)^{-2}\left(\sum_{o \in C_v} \alpha_o^2 - 1\right)$. Let $x \triangleq \sum_{o \in C_v} \alpha_o$. Applying Cauchy-Schwarz (Theorem 1.B.3), we get that

$$\frac{\beta}{w_v} \geq \max\left\{0; \frac{1}{(1 - \delta)^2}\left(\frac{1}{|C_v|} x^2 - 1\right)\right\}, \tag{3.16}$$

where the $\max\{\cdot\}$ is by noting that $\beta \triangleq \sum_{o\in C_v} w(N(o, A-v)) \geq 0$. We use the previous computation to lower bound the slack induced by Ψ_v. Indeed, substituting the bound on Θ_v, we get that

$$\frac{\Psi_v}{w_v} \geq \sum_{o\in C_v} \frac{1}{w_v}(w_o - w_v)^2 + \frac{\Theta_v}{w_v} \geq w_v \left[\sum_{o\in C_v} (\alpha_o - 1)^2 + (\delta - \delta^2)\frac{\beta}{w_v}\right] + \eta. \tag{3.17}$$

The next computation will proceed by lower bounding the squared bracket expression subject to Constraint (3.16). Due to Constraint (3.16) we distinguish two cases depending on which of the two terms is greater. Using Cauchy-Schwarz (Theorem 1.B.3), we bound the bracketed expression by

$$\sum_{o\in C_v} (\alpha_o - 1)^2 + (\delta - \delta^2)\frac{\beta}{w_v} \geq \frac{1}{|C_v|}x^2 - 2x + |C_v| + (\delta - \delta^2)\frac{\beta}{w_v} \tag{3.18}$$

We make the following branching depending on whether $x \leq \sqrt{|C_v|}$ or not.
Case 1: We suppose that $x \leq \sqrt{|C_v|}$. Under this condition we substitute $\beta \geq 0$ in Equation (3.18) to get,

$$\frac{1}{|C_v|}x^2 - 2x + |C_v| + (\delta - \delta^2)\frac{\beta}{w_v} \geq \frac{1}{|C_v|}x^2 - 2x + |C_v|\,,$$

The minimum is attained at $x^* = |C_v|$. However, here $x \leq \sqrt{|C_v|}$. Hence, in this case we get that the previous expression is bounded by:

$$\frac{1}{|C_v|}x^2 - 2x + |C_v| + \left(\delta - \delta^2\right)\frac{\beta}{w_v} \geq \frac{1}{|C_v|}\left(\sqrt{|C_v|}\right)^2 - 2\sqrt{|C_v|} + |C_v| = \left(\sqrt{|C_v|} - 1\right)^2.$$

Case 2: We now suppose that $x \geq \sqrt{|C_v|}$. Then, substituting β/w_v by the second term in Constraint (3.16) and simplifying, we get that

$$\frac{1}{|C_v|}x^2 - 2x + |C_v| + (\delta - \delta^2)\frac{\beta}{w_v} \geq \frac{1}{|C_v|\,(1-\delta)}x^2 - 2x + |C_v| - \frac{\delta}{1-\delta}$$

The minimum of the previous expression is attained at $x^* = |C_v|\,(1-\delta)$. Nevertheless, we need to make sure that $x^* \geq \sqrt{|C_v|}$. A simple computation shows that $x^* \geq \sqrt{|C_v|}$ if and only if $\delta \leq 1 - 1/\sqrt{|C_v|}$. Hence, the minimum of the previous objective is attained at:

$$x^* = \begin{cases} |C_v|\,(1-\delta) & \text{if } \delta \leq 1 - \frac{1}{\sqrt{C_v}} \\ \sqrt{|C_v|} & \text{else.} \end{cases}$$

Replacing each case in objective value, we get that

$$\frac{1}{|C_v|\,(1-\delta)}x^2 - 2x + |C_v| - \frac{\delta}{1-\delta} = \begin{cases} \rho_{|C_v|,\delta} & \text{if } \delta \le 1 - \frac{1}{\sqrt{C_v}} \\ \left(\sqrt{|C_v|} - 1\right)^2 & \text{else.} \end{cases} \tag{3.19}$$

The rest of the proof then follows by unwinding the notation η and expanding it in Equation (3.17). Indeed,

$$\eta = \delta \sum_{o \in C_v} w(N(o, A - v)) = \delta \sum_{o \in C_v} w(N(o, A)) - \delta\,|C_v|\,w_v$$

The second equation is because v belongs to $N(o, A)$ for each $o \in C_v$ (hence the term $|C_v|$). By substituting the previous equation and Equation (3.19) in Equation (3.17), we obtain the desired result. □

Using Lemma 3.8.1, we can prove that the only tight example is given by Figure 2.1. Define $B_i \triangleq \{v \in A\colon |C_v| = i\}$. Observe that in the proof of Theorem 3.7.1 we use that $(1 - \varepsilon_{L-1}) \sum_{o \in O} w(N(o, I_L)) \le k(1 - \varepsilon_{L-1})w(I_L)$. Intuitively, each vertex $v \in I_L$ receives $k(1 - \varepsilon_{L-1})$ times its weight. By adding $k\varepsilon_{L-1}$ to the RHS of Lemma 3.8.1, we get the slack that each isolated vertex in $H_{\le L}$ receives. Exact values are written in Remark 3.8.2.

Remark 3.8.2. To obtain a tight $\sqrt{3}$ approximation ratio, we have set $\varepsilon_{L-1} = 1 - \frac{1}{\sqrt{3}}$. Hence, for vertices in $v \in B_i$ the amount of slack that each vertex receives is equal to:

$$\begin{cases} \rho_{3,1-\frac{1}{\sqrt{3}}} = 4 - 2\sqrt{3} \ge 0.5358 & \text{if } i = 3, \\ \left(\sqrt{i} - 1\right)^2 - (k - i)\left(1 - \frac{1}{\sqrt{3}}\right) \ge 0.5942 & \text{if } i = 1, 2 \\ 3\left(1 - \frac{1}{\sqrt{3}}\right) \ge 1.2679 & \text{if } i = 0. \end{cases} \tag{3.20}$$

Substituting $\varepsilon_{L-1} = 1 - \frac{1}{\sqrt{3}}$ in the proof of Lemma 3.8.1 in the case $|C_v| = 3$, we obtain that $x^* = |C_v|\,(1 - \varepsilon_{L-1}) = \sqrt{3}$. Given this value of x^*, it implies that $\beta = 0$. This demonstrates that the **only** tight case in our analysis for $k = 3$ happens when a vertex v has weight 1 and has 3 talons all of which have weight $\sqrt{3}^{-1}$ (as shown in Figure 2.1).

3.9 Conclusion and Open Questions

Together Chapter 2 and 3 almost close the analysis of Berman's algorithm when large swaps are used. In particular, we achieve tight guarantees for all values of $k \ge 3$ except when $k = 4, 5, 6$. For $k = 3$, we obtain an approximation factor equal to $\sqrt{3}$ that improves upon Neuwohner's approximation [Neu21] of $\frac{k+1}{2} - \frac{1}{63'700'992}$. For $k \ge 7$, we obtain a tight $\frac{k+\delta}{2}$-approximation ratio, previously attained in the asymptotic regime [Neu22]. Finally, our algorithm yields state-of-the-art guarantees for all values of k (including $4, 5, 6$) up to $k \le 371$ [Neu23].

We believe our analysis can be improved to obtain a ratio of $k/2$ for all values of $k \geq 4$. When $k = 6$, it seems that selecting the *bridging arc* that minimizes the ratio between the weights of endpoints leads to an approximation ratio of $k/2$. However, it is not enough for $k = 4, 5$.

There are many directions of great interest to obtain even stronger approximation. We list a couple of interesting open questions. All problems have stars $\star$ denoting a combination of their difficulty and interest.

- $(\star)$ Since we reach a local-gap instance in the case $\sqrt{3}$, can we escape it by running another algorithm once Berman's algorithm finishes? Running a local-search algorithm with respect to the original weight function seems a promising direction. $(\star\star)$ Getting a $3/2$-approximation for $k = 3$, would be extremely interesting.
- $(\star)$ For $k \geq 7$, can we use structural results about our instance to improve upon Neuwohner's result [Neu23]. Her analysis goes beyond the factor $1/2$ by employing an unweighted local-search. Can it further be improved? $(\star\star\star)$ up to the factor $\frac{k+1}{3}$?
- $(\star\star\star)$ Can we design smarter local-search algorithms for this problem. In particular, can we smoothly reduce the exponent in the search to converge to the original w while maintaining good guarantees?
- $(\star\star\star)$ What is the integrality gap the hypergraph matching polytope after few rounds of Lasserre hierarchy? Chan and Lau prove an integrality gap of at most $\frac{k+1}{2}$ after 1 round.
- $(\star)$ In the spirit of Chapter 4, can we obtain good approximation guarantees for the (weighted) hypergraph matching problem in the multipass streaming setting.
- $(\star\star\star\star)$ Is it possible to get past the factor $\frac{k+1}{3}$ for the unweighted k-Set Packing Problem [Cyg13] or improve upon the hardness result [HSS06]?

4 Improved Multipass Algorithms for Submodular Maximization with Independence Constraints

A large portion of the material in this chapter was part of a publication in APPROX/RANDOM'20 [HTW20]. Nonetheless, the presentation of the results is specific to this thesis and some additional results are included. Section 4.7 did not appear in [HTW20].

4.1 Introduction

In this chapter, we consider the problem of maximizing both monotone and arbitrary submodular functions over the p-Matchoid class. The resulting family of constraints is quite general and captures both the classes: p-Matroid-Intersection and p-Hypergraph Matching (see Figure 1.2).

In many applications of submodular optimization, such as summarization tasks [Bad+14; LB10; Mir+15; MBK16], we must process datasets so large that they cannot be stored in memory. Thus, there has been recent interest in streaming algorithms for submodular optimization problems. In this context, we suppose the ground set X is initially unknown and elements arrive one-by-one in a stream. We suppose that the algorithm has an efficient oracle for evaluating the submodular function f on any given subset of X, but has only enough memory to store a small number of elements from the stream. Variants of standard greedy and local search algorithms have been developed that obtain a constant-factor approximation in this setting, but their approximation guarantees are considerably worse than that of their simple, offline counterparts.

Given a p-matchoid $\mathcal{M} = (X, \mathcal{I})$, we consider the multipass setting in which the designed algorithm is allowed to perform several passes over a stream. In each pass all of X arrives in some order, and the algorithm is still only allowed to store a small number of elements. In the offline setting, simple variants of greedy [FNW78] or local search [Fel+11; LSV10] algorithms give the best-known approximation guarantees for maximizing submodular functions over p-Matroid-Intersection or p-Matchoid. However, these algorithms potentially require considering all elements in X each time a choice is made. It is natural to ask whether this is truly necessary, or whether we could instead recover an approximation ratio nearly equal to these offline algorithms by using only a constant number of passes through the data stream.

4.1.1 Our Results

The main result of this chapter is to give a positive answer to the previous question. We demonstrate that the standard local-search algorithm for maximizing a submodular function in a stream can be efficiently simulated. In particular, for monotone submodular functions, we show that $O(1/\varepsilon)$-passes suffice to obtain guarantees only $(1+\varepsilon)$ times worse than those guaranteed by the standard offline local search algorithm. More generally, we give an $O(p/\varepsilon)$-pass streaming algorithm that gives a $p+1+\varepsilon$ approximation for maximizing a monotone submodular function subject to an arbitrary p-matchoid constraint. It immediately gives us an $O(1/\varepsilon)$-pass streaming algorithm attaining a $2+\varepsilon$ approximation subject to a single matroid constraint and a $3+\varepsilon$ approximation for a matching constraint in a graph.

Theorem

Over the p-Matchoid class, there is a multipass algorithm for maximizing a monotone submodular function f with approximation factor of $p+1+\varepsilon$ in $O(p/\varepsilon)$-passes.

The formal statement of the above theorem can be found in Theorem 4.4.3. Each pass of our algorithm is equivalent to a single pass of the streaming local search algorithm described by Chakrabarti and Kale [CK15] and Chekuri, Gupta, and Quanrud [CGQ15]. However, obtaining a rapid convergence to a $p+1+\varepsilon$ approximation requires some new insights. We show that if a pass makes either large or small progress in the value of f, then the guarantee obtained at the end of this pass can be improved. Balancing these two effects then leads to a carefully chosen sequence of parameters for each pass. Our general approach is similar to that of Chakrabarti and Kale [CK15], but our algorithm is *oblivious* to the choice of ε. This allows us to give a uniform bound on the convergence of the approximation factor obtained after some number d of passes. This bound is actually available to the algorithm, and so we can certify the quality of the current solution after each pass. In practice, this allows for terminating the algorithm early if a sufficient guarantee has already been obtained. Even in the worst case, however, we improve on the number of passes by a factor of $O(\varepsilon^{-2})$. Our algorithm only stores $O(k)$ elements, where k is the *rank* of the given p-matchoid, defined as the size of the largest independent set of elements.

Building on these ideas, we also give a randomized, multipass algorithm that uses $O(p/\varepsilon)$-passes and attains a $p+1+\bar{\gamma}_{\text{off}}+\varepsilon$ approximation for maximizing an arbitrary submodular function subject to a p-matchoid constraint, where $\bar{\gamma}_{\text{off}}$ is the approximation ratio attained by best-known offline algorithm for the same problem. To the best of our knowledge, ours is the first multipass algorithm when the function is non-monotone with a number of passes independent of n and k, where n is the size of the ground set. In this case, our algorithm requires storing $O(p^3 k \log k/\varepsilon^3)$ elements. We note that if one states approximation factor in the form $1/\alpha$ less than 1, then our results lead to $1/\alpha-\varepsilon$ approximations in which all dependence on p can be eliminated (by setting simply some $\varepsilon' = p\varepsilon$)[1].

[1] Indeed for $\alpha = O(p)$, in $O(1/\varepsilon)$ passes we obtain a solution such that $f(S) \geq \frac{1}{\alpha+O(p\varepsilon)} f(\text{OPT}) = \frac{1}{\alpha} \cdot \frac{1}{1+O(\varepsilon)} f(\text{OPT}) = \frac{1}{\alpha}(1-O(\varepsilon)) f(\text{OPT})$

Constraint	APX	#-passes	
		Previous	Ours
Matroid	$2+\varepsilon$	$O(1/\varepsilon^3)$ [CK15]	$O(1/\varepsilon)$
p-Hyp.Matching	$p+1+\varepsilon$	$O(p^4\log(p)/\varepsilon^3)$[CK15]	$O(p/\varepsilon)$
p-Mat.Inter.	$p+1+\varepsilon$	$O(p^4\log(p)/\varepsilon^3)$[CK15]	$O(p/\varepsilon)$
p-Matchoid	$p+1+\varepsilon$		$O(p/\varepsilon)$

Tab. 4.1: Improvements over the state-of-the-art results (at the time of publication) for monotone submodular functions

Constraint	APX	#-passes
Matroid	$4.589+\varepsilon$	$O(1/\varepsilon)$
p-Hyp.Matching	$p+1+\frac{p^2}{p-1}+\varepsilon$	$O(p/\varepsilon)$
p-Mat.Inter.	$p+1+\frac{p^2}{p-1}+\varepsilon$	$O(p/\varepsilon)$
p-Matchoid	$p+1+\frac{ep}{(1-\varepsilon)(2-o(1))}+\varepsilon$	$O(p/\varepsilon)$

Tab. 4.2: Multipass streaming algorithm results for non-monotone submodular function maximization

Theorem

Over the p-Matchoid class, there is a multipass algorithm maximizing a general submodular function f with approximation factor $p+1+\bar{\gamma}_{\text{off}}+\varepsilon$ in $O(p/\varepsilon)$-passes where $\bar{\gamma}_{\text{off}}$ is the best approximation factor for maximizing f over this class. See exact values in Table 4.2.

The formal statement of the above theorem can be found in Theorem 4.5.2. Table 4.1 and 4.2 offer an overview of our contributions at the time of the submission. From Table 4.1, we observe a great improvement in the number of passes while matching the results of [CK15] and an extension of the results to handle p-matchoid constraints. Our approach is versatile and applies to general submodular functions as shown in Table 4.2.

In this chapter, we also provide a novel multipass algorithm for maximizing regularized monotone submodular functions subject to a cardinality constraint. A regularized monotone submodular function is a set function $f=g-l$ where g is a monotone submodular function and l is a positive linear function. Our approach builds on the work of [Kaz+21] who obtained a $0.382g(\text{OPT})-l(\text{OPT})$ approximation in a single pass. Extending their approach we derive a multipass guarantee equal to $0.4659g(\text{OPT})-l(\text{OPT})$ in $O(\varepsilon^{-1})$-passes. Our approach is inspired by the distorted greedy algorithm of Harshaw et al. [Har+19] which penalizes the linear term l in a decreasing fashion.

4.1.2 Additional Related Work

Due to the large volume of data in modern applications, there has been a line of research focused on developing *fast* algorithms for weighted and submodular function maximization subject to independence systems. Although our results in this chapter focus on submodular function maximization, we cite, for the purpose of this thesis, related work for maximizing

weighted functions in a stream. All the results which we present here assume an adversarial ordering of the ground set.

First, maximizing a weighted function over a matroid can be done exactly. Thus, current research focuses on maximizing submodular functions or considers harder constraints. Given a monotone submodular function, there is a single pass streaming algorithm with approximation factor equal to 2 subject to a uniform matroid [Bad+14; Kaz+19]. The algorithm is fairly simple. A solution is constructed by greedily adding elements if their marginal at the time of arrival is above a certain threshold. The threshold is an approximate guess of the value of OPT. The approximation factor decreases as the number of passes increases. In $O(1/\varepsilon)$-passes, Norouzi-Fard et al. [Nor+18] obtained $\frac{e}{e-1}+\varepsilon$ approximation algorithm. In some sense it is a fast greedy algorithm since it sees the dataset only a few times. For a general matroid constraint, there is a single pass 3.1467-approximation and $\frac{e}{e-1}+\varepsilon$ in $O(\varepsilon^{-3})$-passes [Fel+22][2]. No single pass streaming algorithm for monotone submodular function maximization can beat the factor 2 even subject a cardinality constraint unless its memory is proportional to the size of the ground set [Fel+20]. We point also point out that no constant guarantee for maximizing a weakly submodular function in a single pass is possible unless, again, we use a memory of size $\Omega(n)$ [Ele+17].

Beyond the Matroid class, maximizing a linear function subject to a matching constraint is a popular problem in the streaming literature. However, beating the factor 2 in a single pass is still open. The best hardness result is $1+\ln(2)$ by Kapralov [Kap21]. Recently, Paz and Schwartzmann obtained a $2+\varepsilon$-approximation for weighted functions [PS18]. Their result extends to the weighted p-hypergraph matching problem, where the approximation ratio is $p+\varepsilon$. The weighted b-matching problem can be approximated within a factor $2+\varepsilon$ [HS21]. Currently, the best approximation results for maximizing a weighted function subject to the p-Matroid-Intersection class in a single pass is $2p+\sqrt{p(p-1)}-1$ and p^2 [CK15; CS14]. Numerous multipass streaming algorithms for the unweighted matching problem have been designed. There are various $1+\varepsilon$ approximation algorithms that run in $O(\text{POLY}(1/\varepsilon))$-passes, and research has focused on optimizing the number of passes needed to obtain the desired approximation [AG13; Ass+22; ALT21; FMU22]. To the best of our knowledge, no $1+\varepsilon$ approximation in $O(\text{POLY}(1/\varepsilon))$ is known in the weighted setting.

Subject to b-Matching/2-Matroid-Intersection, there is a single pass streaming algorithm for maximizing a monotone submodular function with approximation factor equal to $3+2\sqrt{2}$ [LW20; HS21; GJS22]. Subject to p-Matchoid, there is a single pass $4p$-approximation algorithm [CGQ15]. The approximation ratio decreases to $p+1+\varepsilon$ using $O(p^4\log(p)/\varepsilon^3)$-passes for the p-Matroid-Intersection [3]. Table 4.3 and 4.4 give an overview of the state-of-the-art results. In the table, results in red highlight our improvements. The dash $-$ symbol means that, to the best of our knowledge, no dedicated results for the given cell are known. Each $-$ can be replaced by an appropriate value using the hierarchy from Figure 1.2.

[2] This result appeared after the publication of our result.

[3] In [CK15] a bound of $O(\log p/\varepsilon^3)$ is stated. We note that there appears to be a small oversight in their analysis, arising from the fact that their convergence parameter κ in this case is $O(\varepsilon^3/p^4)$. Sagar Kale confirmed it in personal communication

Streaming	Single Pass		Multipass		Refs
	APX	Hardness	APX	|passes|	
Matroid	1	1	–	–	
Matching	2	$1+\ln(2)$	$\frac{3}{2}+\varepsilon$	$O\left(\frac{\log(\varepsilon^{-1})}{\varepsilon^2}\right)$	[PS18; Kap21; AG13]
p-Hyp.Matching	p	–	–	–	[PS18]
p-Mat.Inter.	$\min\{p^2; 4p-2\}$	–	$p+\varepsilon$	$O\left(\frac{p^4\log(p)}{\varepsilon^3}\right)$	[CK15; CS14]
p-Matchoid	$4p$	–	–	–	[CGQ15]

Tab. 4.3: State-of-the-art approximation factors for maximizing weighted linear objective functions over various independence systems in the streaming setting.

Streaming	Single Pass		Multipass		Refs
	APX	Hardness	APX	|passes|	
Unif.Mat	2	2	$\frac{e}{e-1}+\varepsilon$	$O(\varepsilon^{-1})$	[Bad+14; Nor+18]
Matroid	3.1467	2	$\frac{e}{e-1}+\varepsilon$	$O(\varepsilon^{-3})$	[Fel+22; Fel+20]
Matching	$3+2\sqrt{2}$	2.69	$p+1+\varepsilon$	$O(\varepsilon^{-1})$	[LW20; Fel+22; HTW20]
p-Hyp.Matching	–	–	–	–	
p-Mat.Inter.	$4p$	–	$p+1+\varepsilon$	$O(p/\varepsilon)$	[CK15; HTW20]
p-Matchoid	$4p$	–	$p+1+\varepsilon$	$O(p/\varepsilon)$	[CGQ15; LSV10; HTW20]

Tab. 4.4: State-of-the-art approximation factors for maximizing monotone submodular objective functions over various independence systems in the streaming setting.

4.2 Single Pass Algorithm

We suppose that we are given a submodular function $f : 2^X \to \mathbb{R}_{\geq 0}$ and a p-matchoid constraint $\mathcal{M} = (X, \mathcal{I})$ on X given as a collection of matroids $\{\mathcal{M}_i = (X_i, \mathcal{I}_i)\}$ (Definition 1.2.5). Our procedure runs for d passes, each of which uses a modification of Chekuri, Gupta and Quanrud's algorithm [CGQ15]. We begin this section by introducing it. StreamingLocalSearch maintains a current solution S, which is initially set to some S_{init}.

Algorithm 2: Streaming Local Search Algorithm by Chekuri et al. [CGQ15]

procedure StreamingLocalSearch($\alpha, \beta, S_{\text{init}}$)
  $S \leftarrow S_{\text{init}}$;
  foreach x in the stream **do**
    if $x \in S_{\text{init}}$ **then** discard x;
    $C_x \leftarrow$ Exchange(x, S);
    if $f(x|S) \geq \alpha + (1+\beta)\sum_{c \in C_x} \nu(c, S)$ **then**
      $S \leftarrow S \setminus C_x + x$;
  return S;

Whenever an element $x \notin S_{\text{init}}$ arrives, the procedure invokes a helper procedure Exchange, given formally in Algorithm 3, to find an appropriate set $C_x \subseteq S$ of up to p elements so that

Algorithm 3: The procedure EXCHANGE(x, S)

```
procedure EXCHANGE(x, S)
    C_x ← ∅;
    foreach M_ℓ = (X_ℓ, I_ℓ) with x ∈ X_ℓ do
        S_ℓ ← S ∩ X_ℓ;
        if S_ℓ + x ∉ I then
            T_ℓ ← {y ∈ S_ℓ : S_ℓ − y + x ∈ I_ℓ};
            C_x ← C_x + arg min_{t∈T_ℓ} ν(t, S);
    return C_x;
```

the solution remains independent, i.e., $S \setminus C_x + x \in \mathcal{I}$. It then exchanges x with C_x if it gives a significantly improved solution.

The improvement is measured with respect to a set of auxiliary weights $\nu(x, S)$ maintained by the algorithm. For $u, v \in X$, let $u \prec v$ denote that "element u arrives before v" in the stream. Then, we define the *incremental value* of an element e with respect to a set T as the marginal contribution of e with respect to the set of elements from T that arrives before e. Formally,

$$\nu(e, T) \triangleq f(e \mid \{t' \in T : t' \prec e\}).$$

Using these incremental values, STREAMINGLOCALSEARCH proceeds as follows. When an element $x \notin S_{\text{init}}$ arrives, it computes a set of elements $C_x \subseteq S$ that can be exchanged for x. STREAMINGLOCALSEARCH replaces C_x with x if and only if the marginal value $f(x \mid S)$ with respect to S is at least $(1 + \beta)$ times larger than the sum of the current incremental values $\nu(c, S)$ of all elements $c \in C_x$ plus some threshold α, where $\alpha, \beta \geq 0$ are given as parameters. In this case, we say that the element x is *accepted*. Otherwise, we say that x is *rejected*. An element $x \in S$ that has been accepted may later be removed from S if $x \in C_y$ for some later element y that arrives in the stream. In this case we say that x is *evicted*.

The approximation ratio obtained by STREAMINGLOCALSEARCH depends on the parameter β in two ways, which can be intuitively understood in terms of the standard analysis of the offline local search algorithm for the problem. Intuitively, if β is chosen to be too large, more valuable elements will be rejected upon arrival and so, in the offline setting, our solution would be only approximately locally optimal, leading to a deterioration of the guarantee by a factor of $(1 + \beta)$. However, in the streaming setting, the algorithm only attempts to exchange an element upon its arrival, and so the final solution will not necessarily be even $(1 + \beta)$-approximately locally optimal. In fact, an element x may be rejected because $f(x \mid S)$ is small when it arrives, but the processing of later elements in the stream can evict some elements of S. After these evictions, we could have $f(x \mid S)$ larger. The key observation in the analyses of [CK15; CGQ15] is that the marginal value of rejected elements can be carefully bounded by $O(\frac{1}{\beta})$ times the final value of $f(S)$ at the end of the algorithm. Intuitively, if β is chosen to be too small, the algorithm will make more exchanges, evicting more elements, which may result in rejected elements being much more valuable with respect to the final

solution. Selecting the optimal value of β thus requires balancing these two effects. The papers [CGQ15; CK15] prove the following result:

Theorem 4.2.1 ([CGQ15; CK15]). *There is a near-linear space streaming algorithm with approximation $4p$ for maximizing a submodular function subject to a p-matchoid constraint.*[4]

4.2.1 Tight Example for Algorithm 2

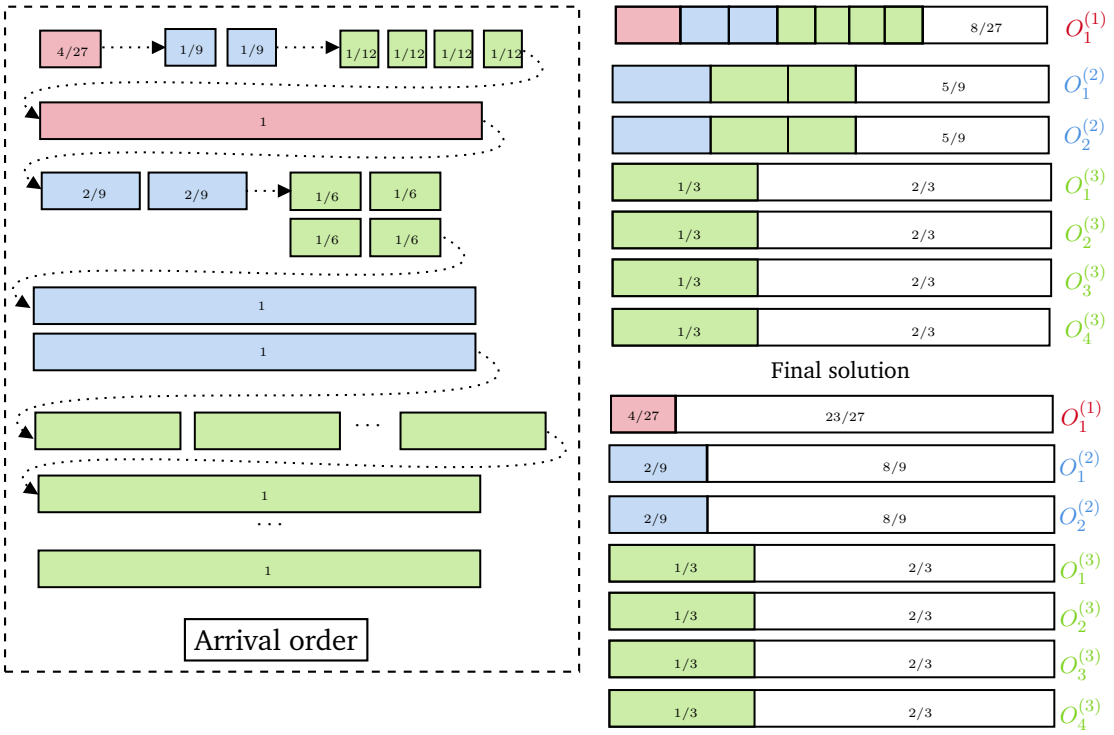


Fig. 4.1: Example when $k = 7$. There is a dotted arrow from e to e' if e arrives before e'. The first 7 sets are covered by $O_1^{(1)}$ as shown in the top right. When $O_1^{(1)}$ arrives, it is discarded. The next 6 sets are added to the solution and are covered by $O_1^{(2)}, O_2^{(2)}$ as shown on the top right. When $O_1^{(2)}$,$O_2^{(2)}$ arrive, they are discarded. The algorithm's output is on the bottom right and has value $f(S) = 4/27 + 4/9 + 4/3 = 52/27$. The optimal solution has value $f(\text{OPT}) = 7$. The approximation factor is $f(\text{OPT})/f(S) = 189/52 = 3.6246$

We prove that Algorithm 2 is tight for a single pass. Our construction uses a coverage function.

Definition 4.2.2 (Coverage Function). Given a universe Ω, and sets $A_1, \ldots, A_m \subseteq \Omega$, the coverage of a collection of sets $T \subseteq \{0, \ldots, m\}$ is the number of elements in the union $\bigcup_{i \in T} A_i$. More generally, each element $i \in \Omega$ has weight $w_i \geq 0$, inducing the function

$$f(T) \triangleq w\left(\bigcup_{i \in T} w_i\right) \quad \text{for all } T \subseteq \{0, \ldots, m\}.$$

[4] [CK15] only prove this for p-Matroid-Intersection

A coverage function is a monotone submodular function. We prove the following result.

Lemma 4.2.3. *There is a converage function subject to a partition matroid for which Algorithm 2 (with $\alpha = 0$ and $\beta = 1$) has approximation factor equal to 4.*

Function Description: Let ℓ be the number of partitions of our partition matroid. The elements of the universe are denoted using 4 indices: i, j, k and m. The range of the indices is the following: For a value k between 1 and ℓ, the index i ranges from k to ℓ and the index m ranges from 1 to 2^{k-1}. Given k and i, the index j takes values between 1 and 2^{i-1}. There are two types of elements: $a_{i,j}^{(k)}$ and $o_m^{(k)}$ and two types of sets: $A_{i,j}^{(k)} \triangleq \{a_{i,j}^{(k)}\}$ and $O_m^{(k)}$. The parameter k denotes the phase of arrival. The sets $A_{i,j}^{(k)}$ have *color* i and the sets $O_m^{(k)}$ have color k. The upper index (k) denotes in which phase the corresponding set arrives. For the set $O_m^{(k)}$, the index k denotes the color of the set as well.

The optimal solution consists of all the sets $O_m^{(k)}$. To describe them, we introduce the *buckets*: $B_{i,m}^{(k)}$. Fixing k, i and m we define $B_{i,m}^{(k)} \triangleq \{A_{i,j}^{(k)} \colon j \in [1 + (m-1)2^{i-k}; m2^{(i-k)}]$ for $j = 1, \ldots, 2^{i-1}\}$. Fixing k and i, the $B_{i,m}^{(k)}$'s partition the sets of color i arriving in the k^{th} phase into groups of 2^{i-k} sets. Then, we define $O_m^{(k)} \triangleq \left(\bigcup_{i=k}^{\ell} \bigcup_{a \in A \colon A \in B_{i,m}^{(k)}} \{a\}\right) \cup \{o_m^{(k)}\}$ that covers all the elements that $\bigcup_{i=k}^{\ell} B_{i,m}^{(k)}$ covers plus the element $o_m^{(k)}$.

Constraint: A independent solution in our partition matroid selects at most 2^{k-1} sets of color k for each $k = 1, \ldots, \ell$.

Arrival Order: The upper index (k) denotes in which phase the corresponding set arrives. In the first phase the sets $A_{i,j}^{(1)}$ arrive first for $i = 1, \ldots, \ell$ and $j = 1, \ldots, 2^{i-1}$. It is followed by the set $O_1^{(1)}$. As we will see later, the sets $A_{i,j}^{(1)}$ will get accepted while $O_1^{(1)}$ will get rejected. In the k^{th}-phase, the set $A_{i,j}^{(k)}$ arrive first for $i = k, \ldots, \ell$ and $j = 1, \ldots, 2^{i-1}$ followed by the sets $O_m^{(k)}$ for $m = 1, \ldots, 2^{k-1}$. As we will see later, the sets $A_{i,j}^{(1)}$ will get accepted whereas $O_m^{(k)}$ will get rejected. Observe that after the k^{th} phase, no more set of color k arrives. It is an invariant that we maintain throughout the stream. We will also prove that our final solution will consists of the sets $A_{k,j}^{(k)}$ for $k = 1, \ldots, \ell$ and $j = 1, \ldots, 2^{k-1}$ and that the optimal solution is the union of the $O_m^{(k)}$. The final ordering is the concatenation each phase, i.e. first phase, then second phase etc... .

Weight/Marginal Value The weight of $a_{i,j}^{(k)}$ is equal to:

$$w\left(a_{i,j}^{(k)}\right) = \frac{2^{\ell-2i+k}}{3^{\ell-i+1}}.$$

Observe that the weight of $a_{i,j}^{(k)}$ is independent of j. Thus, every set in a bucket has the same weight. Note that the $A_{i,j}^{(k)}$'s are disjoint, thus the objective is additive with respect to the $A_{i,j}^{(k)}$'s. The weight of $o_m^{(k)}$ is equal to $w\left(o_m^{(k)}\right) = 1 - \sum_{i=k}^{\ell} \sum_{a \in B_{i,m}^{(k)}} w(a)$. Observe that $w\left(O_m^{(k)}\right) = 1$.

The next Lemma will be useful to bound the value of the solution after each phase.

Lemma 4.2.4. *At the end of the k^{th} phase, the value of the sets in the current solution is equal to:*

$$f(A) = \begin{cases} \frac{2^{\ell-i}}{3^{\ell-i+1}} & A \text{ has color } i \text{ and } i \leq k, \\ \frac{2^{\ell-2i+k}}{3^{\ell-i+1}} & A \text{ has color } i \text{ and } i \geq k. \end{cases}$$

Proof of Lemma 4.2.3. Using Lemma 4.2.4 after the ℓ^{th} phase, we obtain the desired approximation ratio. Let $\tilde{S}$ denote the final solution of Algorithm 2 with $\beta = 1$ and $\alpha = 0$. The final solution contains the set $A_{k,j}^{(k)}$ for $k = 1, \ldots, \ell$ and $j = 1, \ldots, 2^{k-1}$. Their weight is equal to $f\left(A_{k,j}^{(k)}\right) = \frac{2^{\ell-k}}{3^{\ell-k+1}}$. Since all the sets are disjoint, we have

$$f(\tilde{S}) = \sum_{k=1}^{\ell} \sum_{j=1}^{2^{k-1}} f\left(A_{k,j}^{(k)}\right) = \sum_{k=1}^{\ell} 2^{k-1} \cdot \frac{2^{\ell-k}}{3^{\ell-k+1}} = 2^{\ell-1} \sum_{k=1}^{\ell} \left(\frac{1}{3}\right)^k = 2^{\ell-1}\left(\frac{3}{2}(1-(1/3)^{\ell+1}) - 1\right).$$

The optimal solution is equal to $O = \bigcup_{k=1}^{\ell} \bigcup_{m=1}^{2^{k-1}} O_m^{(k)}$ and has weight $f(O) = 2^{\ell} - 1$. Hence, the approximation ratio is equal to:

$$\frac{f(\tilde{S})}{f(O)} = \frac{2^{\ell-1}}{2^{\ell}-1} \cdot \left(\frac{3}{2}(1-(1/3)^{\ell+1}) - 1\right) \underset{\ell \to \infty}{\longrightarrow} 1/4. \qquad \square$$

Recall that in Algorithm 4, a set A of color i enters the solution S if and only if there is a set A' of the same color such that:

$$f(A \mid S) \geq (1+\beta) \cdot \nu\left(A', S\right),$$

where $\nu(A', S)$ is the *incremental value* of A' w.r.t S. In our example, we will not need this notation because all the sets that enter our solution, i.e. the $A_{i,j}^{(k)}$'s, are disjoint from each other. So $\nu(e', S)$ can be replaced by $f(A') = w(a')$.

Proof of Lemma 4.2.4. We prove the lemma by induction of the number of phases. In the first phase, we have $k_i = 2^{i-1}$ disjoint sets of color i arriving of weight $w\left(A_{i,j}^{(1)}\right) = \frac{2^{\ell-2i+1}}{3^{\ell-i+1}}$ where $i = 1, \ldots, \ell$. Since we started with an empty solution, all the sets are accepted. Since the sets in the current solution are disjoint, then, before $O_1^{(1)}$ arrives, the value of the current solution is equal to:

$$\sum_{i=1}^{\ell} k_i \cdot \frac{2^{\ell-2i+1}}{3^{\ell-i+1}} = \sum_{i=1}^{\ell} 2^{i-1} \cdot \frac{2^{\ell-2i+1}}{3^{\ell-i+1}} = \frac{1}{2} \sum_{i=1}^{\ell} \left(\frac{2}{3}\right)^{\ell-i+1} = \frac{1}{2}\left(3(1-(2/3)^{\ell+1}) - 1\right).$$

When $O_1^{(1)}$ arrives (recall that it has color 1), there is only one bucket $B_{1,1}^{(1)}$ that contains all the sets that arrived so far. Thus, its marginal contribution is equal to:

$$f\left(O_1^{(1)} \mid S\right) = w\left(o_1^{(1)}\right) = 1 - \frac{1}{2} \cdot \left(3 \cdot (1 - (2/3)^{\ell+1}) - 1\right) = \left(\frac{2}{3}\right)^{\ell} = (1+\beta) \cdot \frac{2^{\ell-2\cdot 1+1}}{3^{\ell-1+1}},$$

where $\beta = 1$ and the rightmost term is the incremental value of a set of color 1 can be swapped with $O_1^{(1)}$, i.e. $A_{1,1}^{(1)}$. The first equality is because $o_1^{(1)}$ is the only element covered by $O_1^{(1)}$ and not by the current solution. Breaking ties arbitrarily, the algorithm decides to reject $O_1^{(1)}$. Since no set of color 1 will appear in later phase, then a set of color 1 has final weight $\frac{2^{\ell-1}}{3^{\ell}}$.

More generally, we focus on the k^{th} phase, in which we will reject all $O_m^{(k)}$ for $m = 1, \ldots, 2^{k-1}$. All sets $A_{i,j}^{(k)}$ for $i = k, \ldots, \ell$ and $j = 1, \ldots, 2^{i-1}$ arrive before $O_m^{(k)}$. Their marginal contribution/weight is equal to $f(A_{i,j}^{(k)}) = \frac{2^{\ell-2i+k}}{3^{\ell-i+1}}$. By the induction hypothesis and breaking ties accordingly, each such $A_{i,j}^{(k)}$ will be accepted in the current solution because its marginal contribution is twice, i.e., $(1+\beta)$, as much as the set of the same color it is swapped with. Right before the arrival $O_m^{(k)}$, the weight of each bucket $B_{i,m}^{(k)}$ is equal to:

$$\sum_{i=k}^{\ell} 2^{i-k} \cdot \frac{2^{\ell-2i+k}}{3^{\ell-i+1}} = \sum_{i=k}^{\ell} \frac{2^{\ell-i}}{3^{\ell-i+1}} = \frac{1}{3} \sum_{i=k}^{\ell} \left(\frac{2}{3}\right)^{\ell-i} = 1 - (2/3)^{\ell-k+1}.$$

Indeed, by construction the buckets have identical weight, and each contains 2^{i-k} sets of color i. Take the set $O_m^{(k)}$ associated with the bucket $B_{i,m}^{(k)}$. Recall that $O_m^{(k)}$ has color k, and that sets of color k in the current solution have incremental value equal to $\frac{2^{\ell-k}}{3^{\ell-k+1}}$. Each such set of color k in the current solution is a candidate set to be exchanged with $O_m^{(k)}$. Since $o_m^{(k)}$ is the only element that is not covered by the bucket $B_{i,m}^{(k)}$ the marginal contribution $f(O_m^{(k)} \mid S)$ is equal to $w(o_m^{(k)})$. Thus,

$$f\left(O_m^{(k)} \mid S\right) = 1 - \left(1 - (2/3)^{\ell-k+1}\right) = \left(\frac{2}{3}\right)^{\ell-k+1} = (1+\beta)\frac{2^{\ell-k}}{3^{\ell-k+1}} = (1+\beta) w\left(A_{k,j}^{(k)}\right).$$

Thus, breaking ties arbitrarily, the set $O_m^{(k)}$ will be rejected because its marginal contribution at the time of arrival $1+\beta$ times less than any set it is exchangeable with. □

Remark 4.2.5. It is worth noting that the sets in the optimal solution are rejected upon arrival. Hence, even if we remembered all sets being ever added to our solution, we still couldn't produce a better solution.

4.3 The main multipass streaming algorithm

We can now describe our improved multipass algorithm. For monotone functions, our main multipass algorithm is given by the procedure MULTIPASSLOCALSEARCH in Algorithm 4.

Algorithm 4: Multipass algorithm for monotone submodular functions

procedure MULTIPASSLOCALSEARCH($\alpha, \beta_1, \ldots, \beta_d$)
  $S_0 \leftarrow \emptyset$;
  for $i = 1$ **to** d **do**
    Let $\tilde{S}$ be the output of STREAMINGLOCALSEARCH(α, β_i, S_{i-1});
    $S_i \leftarrow \tilde{S}$;
  return S_d;

Our procedure essentially runs d passes of STREAMINGLOCALSEARCH with the following modifications: the initial solution S_{init} used in pass i is the final solution of the previous pass. Secondly, the threshold β is not static. It is set at the beginning of each pass. The crucial property for our improvements is the observation that the marginal contribution of *rejected* elements depends only on the total value of those elements that were accepted *after* they arrive. This effect is bounded by the total value of evicted elements which in turns is bounded by $O(\frac{1}{\beta})$ times the difference between the final and the starting solution.

To use this observation, we measure the ratio $\delta = f(S_{\text{init}})/f(\tilde{S})$ between the value of the initial solution S_{init} of some pass of STREAMINGLOCALSEARCH and the final solution $\tilde{S}$ produced by this pass. If δ is relatively small—and so one pass makes a lot of progress—then this pass gives us an improvement of δ^{-1} over the ratio already guaranteed by the previous pass since $f(\tilde{S}) = \delta^{-1} f(S_{\text{init}})$. On the other hand, if δ is relatively large—and so one pass does not make much progress—then the total increase in the value of our rejected elements can be bounded by $\frac{1-\delta}{\beta} f(\tilde{S})$, and so the potential loss due to only testing these elements at arrival is relatively small. Balancing these two effects allows us to set β smaller in each subsequent passes and obtain an improved guarantee.

Guarantees at the end of a pass

We now turn to the analysis of Algorithm 4. Here we focus on a single pass of STREAMINGLOCALSEARCH. Throughout the rest of the section, we use S to denote the current solution maintained by this pass (initially, $S = S_{\text{init}}$). The following key properties of incremental values will be useful in our analysis. For completeness, they are proved in Appendix 4.A.1 at the end of this chapter.

Lemma 4.3.1. *Given $T, U \subseteq X$ such that $T \subseteq U$, the following properties hold:*

1. $\sum_{e \in T} \nu(e, T) = f(T) - f(\emptyset)$.
2. $\nu(e, U) \leq \nu(e, T)$ *for all* $e \in T$.
3. $f(T \mid U \setminus T) \leq \sum_{t \in T} \nu(t, U)$.
4. *At all times during the execution of* STREAMINGLOCALSEARCH, $\nu(e, S) \geq \alpha$ *for all* $e \in S$, *where S is the solution maintained during this pass.*

Let A denote the set of elements *accepted* during the present pass. These are the elements which were present in the solution S at some previous time during the execution of this pass. Initially we have $A = S = S_{\text{init}}$ and whenever an element is added to S, during this pass we also add this element to A. Let $\tilde{A}$ and $\tilde{S}$ denote the sets of elements A and S at the end of this pass. Note that we regard all elements of S_{init} as having been accepted at the start of the pass.

Each element $e \in \tilde{A} \setminus \tilde{S}$ was accepted but later *evicted* by the algorithm. For any evicted element, we let $\chi(e)$ denote the value of $\nu(e, S)$ at the moment that e was removed from S. More formally, let x be an element that arrives in the stream and let S be the current solution. Suppose that x is *accepted*, then

$$\chi(e) \triangleq \nu(e, S) \text{ for all } e \in \textsc{Exchange}(x, S).$$

To derive guarantees for the i^{th} pass, the proof can be loosely split into 2 parts. Like many standard submodular optimization proofs, we require knowing the marginal contribution of the vertices of OPT at the end of the algorithm, which is derived in the Lemmas 4.3.2 and 4.3.3. Equivalent statement of these lemmas can be found in [CGQ15]. For completeness, their proof is presented in Section 4.A.1.

Lemma 4.3.2. *At the end of* StreamingLocalSearch*, the contribution of elements of OPT with respect to $\tilde{A}$ is at most:*

$$f(OPT \cup \tilde{A}) - f(\tilde{A}) \leq (1+\beta)\left[(p-1)\sum_{e \in \tilde{A} \setminus \tilde{S}} \chi(e) + pf(\tilde{S})\right] + k\alpha.$$

The second part of the proof consists in bounding the value of the set $\tilde{A}$ with respect to the set $\tilde{S}$,

Lemma 4.3.3. *Let $f : 2^X \to \mathbb{R}_{\geq 0}$ be a submodular function. Suppose $\tilde{S}$ is the solution produced at the end of one pass of* StreamingLocalSearch *and $\tilde{A}$ be the set of all elements accepted during this pass. Then,*

$$f(\tilde{A}) \leq f(\tilde{S}) + \sum_{e \in \tilde{A} \setminus \tilde{S}} \chi(e).$$

By combining both Lemma 4.3.2 and Lemma 4.3.3, we get that the value of elements in OPT is bounded by a function of $f(\tilde{S})$ and of the *exit values*: $\chi(\cdot)$. In particular,

Corollary 4.3.4. *At the end of one pass of* StreamingLocalSearch*, we have*

$$f(OPT \cup \tilde{A}) \leq (p + \beta p - \beta)\sum_{e \in \tilde{A} \setminus \tilde{S}} \chi(e) + (p + \beta p + 1)f(\tilde{S}) + k\alpha.$$

We now derive a bound for the summation $\sum_{e \in \tilde{A} \setminus \tilde{S}} \chi(e)$ (representing the value of *evicted* elements) in terms of the total gain $f(\tilde{S}) - f(S_{\text{init}})$ made by the pass, and also bound the total number of accepted elements in terms of $f(\text{OPT})$.

Lemma 4.3.5. *Let $f : 2^X \to \mathbb{R}_{\geq 0}$ be a submodular function. Suppose that $\tilde{S}$ is the solution produced at the end of one pass of* StreamingLocalSearch *and $\tilde{A}$ is the set of all elements accepted during this pass. Then, $|\tilde{A}| \leq f(OPT)/\alpha$ and*

$$\sum_{e \in \tilde{A} \setminus \tilde{S}} \chi(e) \leq \frac{1}{\beta} \left(f(\tilde{S}) - f(S_{\text{init}}) \right) .$$

Proof of Lemma 4.3.5. We consider the quantity $\Phi(A) \triangleq \sum_{e \in A \setminus S} \chi(e)$. Suppose some element a with $C_a \neq \emptyset$ is added to S by the algorithm, evicting the elements of C_a. Then, as each element can be evicted only once, the quantity $\Phi(A)$ increases by precisely $\Delta \triangleq \sum_{e \in C_a} \chi(e)$. Let S_a^-, S_a^+ and A_a^-, A_a^+ be the sets S and A, respectively, immediately before and after a is accepted. Let $\delta_a \triangleq f(S_a^+) - f(S_a^-)$ be the change in the objective function after the exchange between a and C_a. Since a is accepted, we must have $f(a \mid S_a^-) \geq \alpha + (1+\beta) \sum_{e \in C_a} \nu(e, S_a^-)$. Then,

$$\begin{aligned}
\delta_a &= f(S_a^- \setminus C_a + a) - f(S_a^-), \\
&= f(a \mid S_a^- \setminus C_a) - f(C_a \mid S_a^- \setminus C_a), \\
&\geq f(a \mid S_a^-) - f(C_a \mid S_a^- \setminus C_a), && \text{(by submodularity)} \\
&\geq f(a \mid S_a^-) - \sum_{e \in C_a} \nu(e, S_a^-), && \text{(by Lemma 4.3.1 (3))} \\
&\geq \alpha + (1+\beta) \sum_{e \in C_a} \nu(e, S_a^-) - \sum_{e \in C_a} \nu(e, S_a^-), && \text{(since } a \text{ is accepted)} \\
&= \alpha + \beta \sum_{e \in C_a} \chi(e) && \text{(Definition of } \chi(e)) \\
&= \alpha + \beta\Delta.
\end{aligned}$$

It follows that whenever $\Phi(A)$ increases by Δ, $f(S)$ must increase by at least $\beta\Delta$. Initially, $\Phi(A) = 0$ and $f(S) = f(S_{\text{init}})$ and at the end of the algorithm, $\Phi(A) = \sum_{e \in \tilde{A} \setminus \tilde{S}} \chi(e)$ and $f(S) = f(\tilde{S})$. Since $\alpha \geq 0$, we obtain that $\beta \sum_{e \in \tilde{A} \setminus \tilde{S}} \chi(e) \leq [f(\tilde{S}) - f(S_{\text{init}})]$.

It remains to show that $|\tilde{A}| \leq f(\text{OPT})/\alpha$. For this, we note that the above chain of inequalities also implies that every time an element is accepted (and so $|A|$ increases by one), $f(S)$ also increases by at least α. Thus, we have $f(\text{OPT}) \geq f(\tilde{S}) \geq \alpha|\tilde{A}|$. □

Using Lemma 4.3.5 to bound the sum of exit values in Lemma 4.3.4 then immediately gives us the following guarantee for each pass performed in MultipassLocalSearch. In the i^{th} such pass, we will have $S_{\text{init}} = S_{i-1}$, $\tilde{S} = S_i$, and $\beta = \beta_i$. We let A_i denote the set of $\tilde{A}$ of all elements accepted during this particular pass.

Lemma 4.3.6. *Consider the i^{th} pass of* StreamingLocalSearch *performed by* MultipassLocalSearch. *The set A_i of accepted elements in this pass satisfies $|A_i| \leq f(OPT)/\alpha$ and*

$$f(OPT \cup A_i) \leq (p/\beta_i + p - 1)\left[f(S_i) - f(S_{i-1})\right] + (p + p\beta_i + 1)f(S_i) + k\alpha\,.$$

4.4 Analysis for monotone submodular functions

We now show how to use Lemma 4.3.6 together with a careful selection of parameters α and $\beta_1, \ldots, \beta_d$ to derive guarantees for the solution $f(S_i)$ produced after the i^{th} pass made in MultipassLocalSearch. Here, we consider the case that f is a *monotone* submodular function. In this case, we have $f(\text{OPT}) \leq f(\text{OPT} \cup A_i)$ for all i. We set $\alpha = 0$ in each pass. In the first pass, we will set $\beta_1 = 1$. Then, since $f(S_0) = f(\emptyset) \geq 0$, Lemma 4.3.6 immediately gives:

$$f(\text{OPT}) \leq f(\text{OPT} \cup A_1) \leq (2p - 1)\left[f(S_1) - f(\emptyset)\right] + (2p + 1)f(S_1) \leq 4pf(S_1), \tag{4.1}$$

which is tight by Lemma 4.2.3. For subsequent passes, we use the following theorem that relates the approximation guarantee obtained in the current pass to that from the previous pass.

Theorem 4.4.1. *For $i > 1$, suppose that $f(OPT) \leq \gamma_{i-1} \cdot f(S_{i-1})$ and define $\delta_i = \frac{f(S_{i-1})}{f(S_i)}$ as the ratio between the current and the previous pass. Then,*

$$f(OPT) \leq \min\left\{\gamma_{i-1}\delta_i, (\tfrac{p}{\beta_i} + p - 1)(1 - \delta_i) + p + \beta_i p + 1\right\} \cdot f(S_i)\,.$$

Proof of Theorem 4.4.1. From the definition of γ_{i-1} and δ_i, we have:

$$f(\text{OPT}) \leq \gamma_{i-1} f(S_{i-1}) = \gamma_{i-1}\delta_i f(S_i)\,.$$

On the other hand, $f(S_i) - f(S_{i-1}) = (1 - \delta_i)f(S_i)$. Thus, Lemma 4.3.6 with $\alpha = 0$ gives:

$$f(\text{OPT}) \leq \left[(p/\beta_i + p - 1)\,(1 - \delta_i) + p + \beta_i p + 1\right] f(S_i). \qquad \square$$

Equipped with Theorem 4.4.1, we derive exact parameters for δ_i and β_i ensuring a fast convergence with the appropriate approximation guarantees. First, we observe that for any fixed guarantee γ_{i-1} from the previous pass, $\gamma_{i-1}\delta_i$ is an increasing function of δ_i and $(p/\beta_i + p - 1)(1 - \delta_i) + p + \beta_i p + 1$ is a decreasing function of δ_i. Thus, the guarantee we obtain in Theorem 4.4.1 is always at least as good as that obtained when these two values are equal. Setting:

$$\gamma_{i-1}\delta_i = (\tfrac{p}{\beta_i} + p - 1)(1 - \delta_i) + p + \beta_i p + 1,$$

and solving for δ_i gives us:

$$\delta_i = \frac{p(1 + \beta_i)^2}{p + \beta_i(\gamma_{i-1} - 1 + p)}\,. \tag{4.2}$$

In the following analysis, we consider this value of δ_i since the guarantee given by Theorem 4.4.1 will always be no worse than that given by this value. The analysis for a single matroid constraint follows from our results for p-matchoid constraints, but the analysis and parameter values obtained are much simpler, so we present it separately, first.

Theorem 4.4.2. *Suppose we run Algorithm 4 for an arbitrary matroid constraint and monotone submodular function f, with $\beta_i = \frac{1}{i}$. Then $2(1+\frac{1}{i})f(S_i) \geq f(OPT)$ for all $i > 0$. In particular, after $i = \frac{2}{\varepsilon}$ passes, $(2+\varepsilon)f(S_i) \geq f(OPT)$.*

Proof of Theorem 4.4.2. Let γ_i be the guarantee for our algorithm after i passes. Additionally, we introduce the variables $\bar{\gamma}_i \triangleq \frac{2(i+1)}{i}$ that is the worst-case approximation factor at the i^{th} pass. We show by induction on i, that $\gamma_i \leq \frac{2(i+1)}{i} = \bar{\gamma}_i$. For $i = 1$, we have $\beta_1 = 1$ and so from Equation (4.1) we have $\gamma_1 = 4$, as required. For $i > 1$, suppose by induction that $\gamma_{i-1} \leq \frac{2i}{i-1}$, and distinguish two cases depending on whether $\delta_i \leq \frac{p(1+\beta_i)^2}{p+\beta_i(\bar{\gamma}_{i-1}-1+p)}$ or $\delta_i > \frac{p(1+\beta_i)^2}{p+\beta_i(\bar{\gamma}_{i-1}-1+p)}$. Since $p = 1$, $\beta_i = 1/i$, and $\bar{\gamma}_{i-1} = \frac{2i}{i-1}$, the threshold for the case distinction δ_i simplifies to:

$$\frac{p(1+\beta_i)^2}{p+\beta_i(\bar{\gamma}_{i-1}-1+p)} = \frac{(1+\frac{1}{i})^2}{1+\frac{1}{i}(\frac{2i}{i-1})} = \frac{\frac{(i+1)^2}{i^2}}{\frac{(i-1)+2}{i-1}} = \frac{(i-1)(i+1)}{i^2}.$$

Case 1: Suppose $\delta_i \leq \frac{(i-1)(i+1)}{i^2}$. Thus, by Theorem 4.4.1 and the induction hypothesis applied to γ_{i-1}, the i^{th} pass of our algorithm has guarantee γ_i satisfying:

$$\gamma_i \leq \gamma_{i-1}\delta_i \leq \frac{2i}{i-1}\frac{(i-1)(i+1)}{i^2} = \frac{2(i+1)}{i}.$$

Case 2: Suppose now that $\delta_i > \frac{(i-1)(i+1)}{i^2}$. Then, by Theorem 4.4.1 with $p = 1$, the i^{th} pass of our algorithm has guarantee γ_i satisfying:

$$\gamma_i \leq \left(\frac{p}{\beta_i}+p-1\right)(1-\delta_i)+p+p\beta_i+1 \leq i\left(1-\frac{(i-1)(i+1)}{i^2}\right)+2+\frac{1}{i} = 2\left(1+\frac{1}{i}\right). \square$$

The next theorem generalizes the computation and, in fact, proves Theorem 4.4.1 claimed in the introduction of the chapter. The proof is essentially identical but requires a slightly more delicate analysis.

Theorem 4.4.3. *Suppose we run* MULTIPASSLOCALSEARCH *for an arbitrary p-matchoid constraint and monotone submodular function f with β_i set in each pass as:*

$$\beta_i \triangleq \frac{\gamma_{i-1}-1-p}{\gamma_{i-1}-1+p} \qquad \textit{where} \qquad \gamma_i \triangleq 4p\frac{\gamma_{i-1}(\gamma_{i-1}-1)}{(\gamma_{i-1}-1+p)^2}$$

for $i > 1$, and $\beta_1 = 1$ where $\gamma_1 = 4p$. Then $\left(p+1+\frac{4p}{i}\right)f(S_i) \geq f(OPT)$ for all $i > 0$. In particular, after $i = \frac{4p}{\varepsilon}$ passes the approximation is $(p+1+\varepsilon)f(S_i) \geq f(OPT)$.

Proof of Theorem 4.4.3. We first show that approximation guarantee of our algorithm after i passes is given by γ_i. Setting $\beta_1 = 1$, we obtain $\gamma_1 = 4p$ from Equation (4.1), agreeing with our definition.

For subsequent passes with $\beta_i = \frac{\gamma_{i-1}-1-p}{\gamma_{i-1}-1+p}$, Theorem 4.4.1 implies that the guarantee for i^{th} pass will be at most $\delta_i\gamma_{i-1}$, where δ_i is chosen to satisfy Equation (4.2). Specifically, if we set

$$\delta_i = \frac{p\left(1+\frac{\gamma_{i-1}-1-p}{\gamma_{i-1}-1+p}\right)^2}{p+\frac{\gamma_{i-1}-1-p}{\gamma_{i-1}-1+p}(\gamma_{i-1}-1+p)} = \frac{p\left(\frac{2(\gamma_{i-1}-1)}{\gamma_{i-1}-1+p}\right)^2}{\gamma_{i-1}-1} = \frac{4p(\gamma_{i-1}-1)}{(\gamma_{i-1}-1+p)^2},$$

then we have $\delta_i\gamma_{i-1} = \gamma_i$.

We now show by induction on i that $\gamma_i \le p+1+\frac{4p}{i}$. In the case $i = 1$, we have $\gamma_1 = 4p$ and the claim follows immediately from $p \ge 1$. In the general case we may assume without loss of generality that $\gamma_{i-1} \ge 1$. Otherwise, the theorem holds immediately, as each subsequent pass can only increase the value of the solution. Then, we observe that γ_i is an increasing function of γ_{i-1}, for $p \ge 1$ and $\gamma_{i-1} \ge 1$ (shown in Section 4.A.2). By the induction hypothesis, $\gamma_{i-1} \le p+1+\frac{4p}{i-1}$. Therefore:

$$\gamma_i \le \frac{4p\left(p+1+\frac{4p}{i-1}\right)\left(p+\frac{4p}{i-1}\right)}{\left(2p+\frac{4p}{i-1}\right)^2} \le p+1+\frac{4p}{i},$$

as required. The last inequality above follows from straightforward but tedious algebraic manipulations, which can be found in Section 4.A.2. □

4.5 Multipass algorithm for general submodular functions

In this section, we show that the guarantees for monotone submodular maximization can be extended to non-monotone submodular maximization even when dealing with multiple passes. Our main algorithm is given by procedure MULTIPASSRANDOMIZEDLOCALSEARCH in Algorithm 5. In each pass, it calls a procedure RANDOMIZEDLOCALSEARCH, which is an adaptation of STREAMINGLOCALSEARCH, to process the stream. Each such pass produces a pair of feasible solutions S and S', which we now maintain throughout MULTIPASSRANDOMIZEDLOCALSEARCH. The set S is maintained similarly as before and gradually improves by exchanging "good" elements into a solution throughout the pass. The set S' will be maintained by considering the best output of an offline algorithm that we run after each pass as described in more detail below.

To deal with non-monotone submodular functions, we will limit the probability of elements being added to S. Instead of exchanging good elements on arrival, we store them in a buffer B of size m. When the buffer becomes full, an element is chosen uniformly at random from

Algorithm 5: The randomized multipass streaming algorithm

procedure MULTIPASSRANDOMIZEDLOCALSEARCH($\alpha, \beta_1, \ldots, \beta_d, m$)
  $S_0 \leftarrow \emptyset, S'_0 \leftarrow \emptyset$;
  for $i = 1$ **to** d **do**
    Let $(\tilde{S}, S')$ be the output of RANDOMIZEDLOCALSEARCH($S_{i-1}, \alpha, \beta_i, m$);
    $S_i \leftarrow \tilde{S}$, $S'_i \leftarrow \arg\max\{f(S'_{i-1}), f(S')\}$;
  return $\bar{S} = \arg\max\{f(S_d), f(S'_d)\}$;

procedure RANDOMIZEDLOCALSEARCH($S_{\text{init}}, \alpha, \beta, m$)
  $S \leftarrow S_{\text{init}}$; $B \leftarrow \emptyset$;
  foreach x in the stream **do**
    if $f(x \mid S) \geq \alpha + (1+\beta)\sum_{e \in C_x} \nu(e, S)$ **then**
      $B \leftarrow B + x$;
    if $|B| = m$ **then**
      $x \leftarrow$ uniformly random element from B;
      $C_x \leftarrow$ EXCHANGE(x, S);
      $B \leftarrow B - x$; $S \leftarrow S + x - C_x$;
      foreach x' in B **do**
        $C_{x'} \leftarrow$ EXCHANGE(x', S);
        if $f(x' \mid S) < \alpha + (1+\beta)\sum_{e \in C_{x'}} \nu(e, S)$ **then**
          $B \leftarrow B - x'$;
  $S' \leftarrow$ OFFLINE(B);
  return (S, S');

the buffer and added to S. Adding a new element to the current solution may affect the quality of the remaining elements in the buffer. Thus, we need to re-evaluate them and remove the elements that are no longer good.

As before, we let A denote the set of elements that were previously added to S during the current pass of the algorithm. Note that we do not consider an element to be accepted until it has actually been added to S from the buffer. For any fixed set of random choices, the execution of RANDOMIZEDLOCALSEARCH can be considered as the execution of STREAMINGLOCALSEARCH on the following stream: we suppose that an element x arrives whenever it is selected from the buffer and accepted into S. All elements that are discarded from the buffer after accepting x then arrive, and will also be rejected by STREAMINGLOCALSEARCH. Any element remaining in the buffer after the execution of the algorithm does not arrive in the stream. Applying Lemma 4.3.6 with respect to this pretend stream ordering allows us to bound $f(\tilde{S})$ with respect to $f(\text{OPT} \setminus B)$ (that is, the value of the part of OPT that does not remain in the buffer B) after a single pass of RANDOMIZEDLOCALSEARCH. Formally, let $\tilde{B}_i$

be the value of the elements in the buffer after the i^{th} pass of our algorithm. Then, applying Lemma 4.3.6 to the set $\text{OPT} \setminus \tilde{B}_i$, and taking expectation, gives:

$$\mathbb{E}[f(A_i \cup (\text{OPT} \setminus \tilde{B}_i))] \leq (p/\beta + p - 1)\,(\mathbb{E}[f(S_i)] - \mathbb{E}[f(S_{i-1})]) + (p + \beta p + 1)\,\mathbb{E}[f(S_i)] + \alpha k\,. \quad (4.3)$$

In order to bound the value of the elements in $\tilde{B}_i$, we apply any offline $\bar{\gamma}_{\text{off}}$-approximation algorithm OFFLINE to the buffer at the end of the pass to obtain a solution S'. In MULTIPASS-RANDOMIZEDLOCALSEARCH, we then remember the best such offline solution S'_i computed across the first i passes. Then, in the i^{th} pass, we have

$$\mathbb{E}[f(\text{OPT} \cap \tilde{B}_i)] \leq \bar{\gamma}_{\text{off}}\,\mathbb{E}[f(S')] \leq \bar{\gamma}_{\text{off}}\,\mathbb{E}[f(S'_i)]\,. \quad (4.4)$$

From the submodularity of f and $A_i \cap \tilde{B}_i = \emptyset$, we have $f(A_i \cup \text{OPT}) \leq f(A_i \cup (\text{OPT} \setminus \tilde{B}_i)) + f(\text{OPT} \cap \tilde{B}_i)$. Thus, combining Equation (4.3) and Equation (4.4) we have:

$$\mathbb{E}[f(A_i \cup \text{OPT})] \leq (p/\beta + p - 1)\,(\mathbb{E}[f(S_i)] - \mathbb{E}[f(S_{i-1})]) + (p + \beta p + 1)\,\mathbb{E}[f(S_i)] + \bar{\gamma}_{\text{off}}\,\mathbb{E}[f(S'_i)] + \alpha k\,. \quad (4.5)$$

To relate the left-hand side to $f(\text{OPT})$ we use the following result from Buchbinder et al. [Buc+14]:

Lemma 4.5.1 (Lemma 2.2 in [Buc+14])**.** *Let $f\colon 2^X \to \mathbb{R}_{\geq 0}$ be a non-negative submodular function. Suppose that A is a random set where no element $e \in X$ appears in A with probability more than p. Then, $\mathbb{E}[f(A)] \geq (1-p)\,f(\emptyset)$. Moreover, for any set $Y \subseteq X$, it follows that $\mathbb{E}[f(Y \cup A)\,] \geq (1-p) f(Y)$.*

We remark that a similar theorem also appeared earlier in Feige et al. [FMV11] for a random set that contains each element *independently* with probability *exactly* p. Here, the probability that an element occurs in A_i is delicate to handle because such an element may either originate from the starting solution S_{i-1} or be added during the pass. Thus, we use a rougher estimate and bound the probability of an element ever appearing in $A_1 \cup \ldots A_i$. Since the first event is contained in the other, $\Pr[e \in A_i] \leq \Pr[e \in A_i \cup \ldots \cup A_1]$. The number of selections during the j^{th} pass is at most $|A_j|$ and by Lemma 4.3.6 (applied to the set $\text{OPT} \setminus \tilde{B}_j$ due to our pretend stream ordering in each pass j), $|A_j| \leq f(\text{OPT} \setminus \tilde{B}_j)/\alpha \leq f(\text{OPT})/\alpha$ in any pass. Here, the second inequality follows from the optimality of OPT, and the fact that any subset of the feasible solution OPT is also feasible for our p-matchoid constraint. Thus, the total number of selections in the first i passes at most $\sum_{j=1}^{i} |A_j| \leq i \cdot f(\text{OPT})/\alpha$. We select an element only when the buffer is full, and each selection is made independently and uniformly at random from the buffer. Thus, the probability that any given element is selected when the algorithm makes a selection is at most $1/m$ and by a union bound, $\Pr[e \in A_i \cup \ldots \cup A_1] \leq i \cdot f(\text{OPT})/(m\alpha)$. Let d be the number of passes that the algorithm makes and suppose we set $\alpha = \varepsilon f(\text{OPT})/2k$ (in Appendix 4.A.3 we show that this can be accomplished approximately by guessing $f(\text{OPT})$, which can be done at the expense of an

extra factor $O(\log k)$ space). Finally, let $m = 4dk/\varepsilon^2$. Then, applying Lemma 4.5.1, after $i \leq d$ passes we have:

$$\mathbb{E}[f(A_i \cup \text{OPT})] \geq (1 - d \cdot f(\text{OPT})/(m\alpha))\, f(\text{OPT}) \geq (1 - \varepsilon/2)\, f(\text{OPT})\,. \tag{4.6}$$

Our definition of α also implies that $\alpha k \leq \varepsilon/2 f(\text{OPT})$. Using this and Equation (4.6) in (4.5), we obtain:

$$\begin{aligned}(1-\varepsilon) & f(\text{OPT}) \\ & \leq (p/\beta + p - 1)(\mathbb{E}[f(S_i)] - \mathbb{E}[f(S_{i-1})]) + (p + \beta p + 1)\,\mathbb{E}[f(S_i)] + \bar{\gamma}_{\text{off}}\,\mathbb{E}[f(S'_i)]\,. \end{aligned} \tag{4.7}$$

As we show in Section 4.6, the rest of the analysis then follows similarly to that in Section 4.4, using the fact that $f(\bar{S}) = \max\{f(S_d), f(S'_d)\}$. It yields the following theorem

Theorem 4.5.2. *Given a p-matchoid $\mathcal{M} = (X, \mathcal{I})$ of rank k, a submodular function $f : 2^X \to \mathbb{R}_{\geq 0}$, and an algorithm for the offline instance of the problem with approximation factor $\bar{\gamma}_{\text{off}}$. Then, for any $\varepsilon > 0$,* RandomizedLocalSearch *returns a solution $\bar{S} \in \mathcal{I}$ such that*

$$f(OPT) \leq (p + 1 + \bar{\gamma}_{\text{off}} + \varepsilon)\,\mathbb{E}[f(\bar{S})],$$

using a total space of $O\left(\frac{p^3 k \log_2 k}{\varepsilon^3}\right)$ and $O\left(\frac{p}{\varepsilon}\right)$-passes.

4.6 Analysis for non-monotone submodular functions

In this section we give a full proof of Theorem 4.5.2 from Section 4.5. The proof essentially reduces to that of the monotone case of Section 4.3. In particular, we prove that the single pass approximation factor equal to $4p + \bar{\gamma}_{\text{off}}$ improves to $p + 1 + \bar{\gamma}_{\text{off}} + \varepsilon$ using $O(p/\varepsilon)$-passes.

Theorem 4.6.1. *Given a submodular function f, suppose the $(i-1)^{th}$ pass of* MultipassRandomizedLocalSearch *produces a solution $(1-\varepsilon) f(OPT) \leq \gamma_{i-1}\,\mathbb{E}[f(S_{i-1})] + \bar{\gamma}_{\text{off}}\,\mathbb{E}[f(S'_{i-1})]$ with a buffer of size $m = 4dk/\varepsilon^2$, and threshold $\alpha \leq \varepsilon f(OPT)/2k$, then the i^{th} pass satisfies,*

$$(1-\varepsilon) f(OPT) \leq \min\left\{\gamma_{i-1}\delta_i, \left(\frac{p}{\beta_i} + p - 1\right)(1-\delta_i) + p + \beta_i p + 1\right\} \cdot \mathbb{E}[f(S_i)] + \bar{\gamma}_{\text{off}}\,\mathbb{E}[f(S'_i)],$$

where $\delta_i = \frac{\mathbb{E}[f(S_{i-1})]}{\mathbb{E}[f(S_i)]}$.

Proof of Theorem 4.6.1. From the definition of γ_{i-1} and δ_i, it follows that,

$$(1-\varepsilon) f(\text{OPT}) \leq \gamma_{i-1}\,\mathbb{E}[f(S_{i-1})] + \bar{\gamma}_{\text{off}}\,\mathbb{E}[f(S'_{i-1})] \leq \gamma_{i-1}\delta_i\,\mathbb{E}[f(S_i)] + \bar{\gamma}_{\text{off}}\,\mathbb{E}[f(S'_i)] \tag{4.8}$$

where in the last inequality we have used the definition of δ_i and the fact that $f(S'_i) \geq f(S'_{i-1})$, which follows from the way S'_i is defined in Algorithm 5. On the other hand, $\mathbb{E}[f(S_i)] - \mathbb{E}[f(S_{i-1})] = (1-\delta_i)\,\mathbb{E}[f(S_i)]$. Thus, by Equation (4.7) we also have:

$$\begin{aligned}(1-\varepsilon)f(\text{OPT}) &\leq \left(\tfrac{p}{\beta_i} + p - 1\right)(\mathbb{E}[f(S_i)] - \mathbb{E}[f(S_{i-1})]) + (p + \beta p + 1)\,\mathbb{E}[f(S_i)] + \bar{\gamma}_{\text{off}}\,\mathbb{E}[f(S'_i)] \\ &= \left(\left(\tfrac{p}{\beta_i} + p - 1\right)(1-\delta_i) + p + \beta_i p + 1\right)\mathbb{E}[f(S_i)] + \bar{\gamma}_{\text{off}}\,\mathbb{E}[f(S'_i)]\,. \qquad (4.9)\end{aligned}$$

Since the right-hand side of Equation (4.8) is an increasing function of δ_i and the right-hand side of Equation (4.9) is a decreasing function of δ_i, the guarantee we obtain is always at least as good as that obtained when these two values are equal. □

As in the monotone case, the lemma enables us to derive values of β so as to minimize the value of the approximation ratio. The following theorem is by the same calculations as in Section 4.4 and the follow-up computations in Section 4.A.2.

Theorem 4.6.2. *The i^{th} pass of* MULTIPASSRANDOMIZEDLOCALSEARCH *with a buffer of size $m = 4dk/\varepsilon^2$, $\alpha \leq \varepsilon f(OPT)/2k$, and previous passes initialized with $\beta_1 = 1$ and $\beta_i = \frac{\gamma_{i-1}-1-p}{\gamma_{i-1}-1+p}$ where γ_i is given by the recurrence, $\gamma_1 = 4p$ and $\gamma_i = \frac{4p\gamma_{i-1}(\gamma_{i-1}-1)}{(\gamma_{i-1}-1+p)^2}$, has guarantee*

$$(1-\varepsilon)f(OPT) \leq \left(p + 1 + \frac{4p}{i}\right)\mathbb{E}[f(\tilde{S}_i)] + \bar{\gamma}_{\text{off}}\,\mathbb{E}[f(S'_i)].$$

In particular after $d = \frac{4p}{\varepsilon}$ passes, we get $(1-\varepsilon)f(OPT) \leq (p + 1 + \bar{\gamma}_{\text{off}} + \varepsilon)\,\mathbb{E}[f(\bar{S}_d)]$ where $\bar{\gamma}_{\text{off}}$ is the approximation ratio of the best offline algorithm for maximizing f under $\mathcal{I}$.

Corollary 4.6.3. *Using $d = O(\varepsilon^{-1})$ passes with $\beta_i = \frac{1}{i}$, Algorithm 6 has guarantee equal to: $f(OPT) \leq (4.5975 + \varepsilon)\,\mathbb{E}[f(\bar{S}_d)]$, where we use that $\bar{\gamma}_{\text{off}} = 2.5975$ [BF19] for maximizing a submodular function over the Matroid class.*

For others classes such as: p-Hyp.Matching , p-Matroid-Intersection , and p-Matchoid , we recall that $\bar{\gamma}_{\text{off}} = \frac{p^2+\varepsilon}{p-1}, \frac{p^2+(p-1)\varepsilon}{p-1}, \frac{e(p+1)}{2}$, respectively [Fel+11; LSV10; CVZ14]. We conclude the section with the proof of Theorem 4.5.2.

Proof of Theorem 4.5.2. We assume that we know the value of $f(\text{OPT})$ beforehand. We show how to remove this assumption completely in Section 4.A.3. Let $\varepsilon' = \varepsilon/p$ with $1/2 \geq \varepsilon' > 0$ and let $\alpha = \varepsilon' f(\text{OPT})/2k$. We want to obtain an additive error term instead of a multiplicative error term as stated in Theorem 4.6.2. By Theorem 4.6.2,

$$(1-\varepsilon')f(\text{OPT}) \leq \left(p + 1 + \bar{\gamma}_{\text{off}} + \frac{4p}{d}\right)\mathbb{E}[f(\bar{S}_d)]$$

Using the fact that $(1-\varepsilon')^{-1} \leq 1 + 2\varepsilon'$ for $\varepsilon' \in (0, 1/2]$, we get that,

$$f(\text{OPT}) \leq \left(p + 1 + \bar{\gamma}_{\text{off}} + \frac{4p}{d}\right)(1 + 2\varepsilon')\,\mathbb{E}[f(\bar{S}_d)]\,. \qquad (4.10)$$

Since $\varepsilon' = \varepsilon/p$, setting $d = O(p/\varepsilon)$ we finally obtain the desired result:

$$\begin{aligned} f(\text{OPT}) &\leq (p + 1 + \bar{\gamma}_{\text{off}} + O(\varepsilon))(1 + 2\varepsilon/p)\, \mathbb{E}[f(\bar{S}_d)] \\ &\leq (p + 1 + \bar{\gamma}_{\text{off}} + O(\varepsilon/p)(p + 1 + \bar{\gamma}_{\text{off}} + O(\varepsilon)))\, \mathbb{E}[f(\bar{S}_d)] \\ &\leq (p + 1 + \bar{\gamma}_{\text{off}} + O(\varepsilon))\, \mathbb{E}[f(\bar{S}_d)] \end{aligned}$$

where in the final inequality we assume that the approximation factor $\bar{\gamma}_{\text{off}}$ is linear in p subject to a p-matchoid constraint [CVZ14]. The proof terminates with an appropriate rescaling of ε.

Regarding space complexity, we observe that MULTIPASSRANDOMIZEDLOCALSEARCH stores the buffer B and maintains two past solutions $S_i, S_i' \in \mathcal{I}$, together with the current solution $S \in \mathcal{I}$. Hence, the total space needed is equal to $O(|B| + |S_i'| + |S_i| + |S|) = O(m + 3k) = O\big(p^3 k \varepsilon^{-3}\big)$, times an additional factor of $O(\log k)$ for guessing $f(\text{OPT})$ (see Section 4.A.3). The number of passes is $d = O(p/\varepsilon)$. □

4.7 Regularized Monotone Submodular Maximization

In this section, we switch gear and focus on multipass streaming algorithms for maximizing regularized monotone submodular functions subject to uniform matroid constraint. The results in this section do not appear in [HTW20].

Definition 4.7.1 (Regularized Submodular Function)**.** We call a *regularized monotone submodular function* a set function f that can be written as $f = g - l$, where $g : 2^X \to \mathbb{R}_{\geq 0}$ is a monotone submodular function and l is a linear function.

Our aim is to maximize f subject to a uniform matroid constraint $\mathcal{I} = \{S \subseteq X : |S| \leq k\}$.

Remark 4.7.2. The performance of a given algorithm is measured via a bicriteria. In particular, we aim at designing an algorithm that returns a solution S such that $f(S) \geq \alpha g(\text{OPT}) - \beta l(\text{OPT})$, where $\alpha, \beta \in [0, 1]$. Note the difference with Definition 1.1.3 where the approximation is greater than 1.

To optimize f, we assume that we have access to the decomposition of f. In particular, we suppose that we have a value oracle for g and for l. Sviridenko et al. [SVW15] give an algorithm that returns a solution S such that $f(S) \geq (1 - e^{-1})g(\text{OPT}) - l(\text{OPT})$ for any monotone submodular function g and any linear (possibly negative) function l. Their result implies an optimal approximation for g and for l, respectively and holds even subject to a matroid constraint. Perhaps surprisingly, greedy approaches yield state-of-the-art results only in a restricted setting where l is non-negative and $\mathcal{I}$ is a uniform matroid [Har+19]. The special case of non-negative l is important in its own right as the function f is no longer strictly positive. This setting was subsequently studied in [Kaz+21; BF22]. In the setting that we consider here, the state-of-the-art streaming algorithm is due to Kazemi et al. [Kaz+21]. They get a solution S such that $f(S) \geq \frac{3-\sqrt{5}}{2} g(\text{OPT}) - l(\text{OPT})$. Our main result is a multipass streaming algorithm for maximizing f that extends Kazemi et al.'s algorithm.

Algorithm

The main idea of Algorithm 6 is to mimic the behavior of the distorted greedy algorithm introduced in [Har+19] which gradually decreases the penalty on the linear term l. Thus, elements with small linear cost are added first. Our algorithm maintains a current solution S throughout each pass. Each pass is initialized with the solution of the previous pass. During the i^{th} pass, a new element is added whenever its distorted weighted marginal contribution is above a fixed threshold τ. More precisely, the algorithm checks whether,

$$g(e \mid S) - \alpha_i l_e \geq \tau,$$

where $\alpha_i \geq 1$ is some *penalty* weight. The penalty of the linear term decreases in each pass so that elements which were rejected may be accepted in a later pass. Kazemi et al. [Kaz+21] is a special instantiation of this algorithm, where they set $\alpha_1 = \frac{3+\sqrt{5}}{2}$ and $\tau k = \frac{3-\sqrt{5}}{2} g(\text{OPT}) - l(\text{OPT})$ to obtain an approximation guarantee of $0.382 g(\text{OPT}) - l(\text{OPT})$ in a single pass.

Algorithm 6: Multipass streaming algorithm for regularized monotone submodular functions

Procedure: REGULARIZEDMULTIPASS:τ, Penalties: $(\alpha_1, \ldots, \alpha_d)$
$S \leftarrow \emptyset$
for $(i \leftarrow 1; i \leq d; i \leftarrow i + 1)$ **do**
  for $e \in X$ **do**
    if $g(e \mid S) - \alpha_i l_e \geq \tau$ **and** $|S| < k$ **then**
      $S \leftarrow S + e$.
return Return S

Compared to MULTIPASSLOCALSEARCH, here, we fix the number of passes in advance to fix the sequence $(\alpha_1, \ldots, \alpha_d)$. Before diving into the analysis, let d be the number of passes, let x^* be the solution of the equation $e^x(x-2)+1=0$, and define $\tau \triangleq \frac{1}{k}\left(\frac{e^{x^*}-1}{2e^{x^*}-1} g(\text{OPT}) - l(\text{OPT})\right)$. Finally, we set $\alpha_i \triangleq \left(1 + \frac{x^*}{d}\right)^{d+1-i}$, and denote by S_i the solution at the end of the i^{th} pass.

4.7.1 Analysis for regularized monotone submodular functions

The main result of this section is the proof of Theorem 4.7.3. To the best of our knowledge, it is the first multipass algorithm for maximizing a regularized monotone submodular function.

Theorem 4.7.3. *For any $\varepsilon > 0$,* MULTIPASS REGULARIZED *(Algorithm 6) produces a solution S of size at most k in $O(\varepsilon^{-1})$-passes such that,*

$$g(S) - l(S) \geq (\gamma - \varepsilon)\, g(OPT) - (1+\varepsilon) l(OPT)\,,$$

where $\gamma \triangleq \frac{e^{x^*}-1}{2e^{x^*}-1} \geq 0.4569$ and x^* is the solution of the equation $e^x(x-2)+1=0$.

The proof follows the same line as the proof of Kazemi et al. [Kaz+21]. It distinguishes two cases: either the solution after a certain number of passes has size k in which case the approximation ratio is at least $k\tau$. Otherwise, we argue that vertices in OPT were rejected and thus must have low marginal contribution. This implies that the value of $g(S_i)$ must be large for the pass we consider. The next two lemmas deal with the first and second case respectively.

Lemma 4.7.4. *At the end of the i^{th} pass, we have*

$$Eq_i^1: \quad g(S_i \mid S_{i-1}) - \alpha_i l(S_i \setminus S_{i-1}) \geq \tau \left|S_i \setminus S_{i-1}\right|.$$

Proof of Lemma 4.7.4. Sort the elements $\{e_1, \ldots, e_n\} = S_i \setminus S_{i-1}$ in the order of their arrival. Recall that the i^{th} pass starts with S_{i-1} as initial solution and that each element in $S_i \setminus S_{i-1}$ has marginal contribution larger than τ when it arrives. Thus,

$$g(S_i \mid S_{i-1}) - \alpha_i l(S_i \setminus S_{i-1}) = \sum_{j=1}^{n} g(e_j \mid S_{i-1} \cup \{e_1, \ldots, e_{j-1}\}) - \alpha_i l_{e_j} \geq \tau \left|S_i \setminus S_{i-1}\right|. \qquad \square$$

Lemma 4.7.5. *Suppose that at the end of the i^{th} pass, we have $|S_i| < k$, then*

$$Eq_i^2: \quad g(S_i) \geq g(OPT) - \alpha_i l(OPT) - \tau k.$$

Proof of Lemma 4.7.5. We look at the elements of OPT which are rejected in the i^{th} pass.

$$\sum_{e \in \text{OPT} \setminus S_i} g(e \mid S_i) - \alpha_i l_e < \tau \left|\text{OPT} \setminus S_i\right| \leq k\tau, \tag{4.11}$$

where the inequality follows from the rejection property. The second inequality is true because $|\text{OPT} \setminus S_i| \leq |\text{OPT}| \leq k$. From submodularity we also get that,

$$\begin{aligned} \sum_{e \in \text{OPT} \setminus S_i} g(e \mid S_i) - \alpha_i l_e &\geq g(\text{OPT} \cup S_i) - g(S_i) - \alpha_i l(\text{OPT} \setminus S_i), \\ &\geq g(\text{OPT}) - g(S_i) - \alpha_i l(\text{OPT}), \end{aligned} \tag{4.12}$$

where the second inequality follows from the monotonicity and positivity of l. Rearranging Equation (4.12) and Equation (4.11) gives the desired result. $\square$

In the following computations we will mention the equations stated in Lemma 4.7.4 and 4.7.5 with respect to their indices. We denote the equations in Lemma 4.7.4 and 4.7.5 at the i^{th} pass by Eq_i^1 and Eq_i^2 respectively. Lemma 4.7.4 holds for any pass but gives a bound which depends on the value of S_{i-1}. Lemma 4.7.5 holds only if $|S_i| < k$ and doesn't take into account the linear part of S_i. The next lemma combines the two lemmas to obtain a bound on $f(S_d)$ with respect to OPT.

Lemma 4.7.6. *Suppose that at the end of the d^{th} pass we have $|S_d| = k$ then,*

$$g(S_d) - l(S_d) \geq \tau k.$$

Else if $|S_d| < k$ then,

$$g(S_d) - l(S_d) \geq \left(1 - \frac{1}{\alpha_1}\right)(g(OPT) - \tau k) - l(OPT)\left(\sum_{i=1}^{d} \frac{\alpha_i}{\alpha_{i+1}} - d\right).$$

Proof of Lemma 4.7.6. We start with the first case where we assume $|S_d| = k$. Let $i \leq d$ be the pass number for which the current solution reaches its maximum size k, i.e., $|S_i| = k$. Later passes will not change the current solution and hence $S_d = S_i$. We apply the lemma 4.7.4 for all passes $j = 1, \ldots, i$. By summing Eq_j^1 over j, we have

$$\tau k = \sum_{j=1}^{i} \tau \, |S_j \setminus S_{j-1}| \leq \sum_{j=1}^{i} g(S_j \mid S_{j-1}) - \alpha_j l(S_j \setminus S_{j-1}) \leq g(S_i) - l(S_i) = g(S_d) - l(S_d),$$

where the last inequality is by the fact that $g(\emptyset) \geq 0$ and that $\alpha_j \geq 1$. Next we assume that $|S_d| < k$. This means that we can apply both lemma 4.7.4 and 4.7.5 for each pass. We combine the equations as follows, $\sum_{i=1}^{d}\left[\frac{\alpha_d}{\alpha_i}\text{Eq}_i^1 + (\frac{\alpha_d}{\alpha_{i+1}} - \frac{\alpha_d}{\alpha_i})\text{Eq}_i^2\right]$, where $\alpha_{d+1} = 1$. The right-hand side simplifies to:

$$\begin{aligned}
& \sum_{i=1}^{d}\left[\frac{\alpha_d}{\alpha_i}(g(S_i \mid S_{i-1}) - \alpha_i l(S_i \setminus S_{i-1})) + \left(\frac{\alpha_d}{\alpha_{i+1}} - \frac{\alpha_d}{\alpha_i}\right) g(S_i)\right] \\
= & \sum_{i=1}^{d}\left[\frac{\alpha_d}{\alpha_{i+1}} g(S_i) - \frac{\alpha_d}{\alpha_i} g(S_{i-1})\right] - \alpha_d l(S_d) \\
= & \ \alpha_d g(S_d) - \frac{\alpha_d}{\alpha_1} g(\emptyset) - \alpha_d l(S_d) \\
\leq & \ \alpha_d (g(S_d) - l(S_d)).
\end{aligned}$$

The inequality follows from the positivity of g, i.e. $g(\emptyset) \geq 0$. Since $\alpha_i \geq \alpha_{i+1} \geq 1$, the sign of Eq_i^2 is preserved when multiplied by $(\frac{\alpha_d}{\alpha_{i+1}} - \frac{\alpha_d}{\alpha_i})$. Combining the previous computation with the left-hand side of $\sum_{i=1}^{d}\left[\frac{\alpha_d}{\alpha_i}\text{Eq}_i^1 + (\frac{\alpha_d}{\alpha_{i+1}} - \frac{\alpha_d}{\alpha_i})\text{Eq}_i^2\right]$ yields

$$\begin{aligned}
\alpha_d(g(S_d) - l(S_d)) &\geq \tau\left(|S_d \setminus S_{d-1}| + \sum_{i=1}^{d-1}\frac{\alpha_d}{\alpha_i}|S_i \setminus S_{i-1}|\right) \\
&\quad + \sum_{i=1}^{d}\left[\left(\frac{\alpha_d}{\alpha_{i+1}} - \frac{\alpha_d}{\alpha_i}\right)(g(\text{OPT}) - \alpha_i l(\text{OPT}) - \tau k)\right] \\
&\geq \sum_{i=1}^{d}\left(\frac{\alpha_d}{\alpha_{i+1}} - \frac{\alpha_d}{\alpha_i}\right)(g(\text{OPT}) - \tau k) - \alpha_d l(\text{OPT})\sum_{i=1}^{d}\left[\alpha_i\left(\frac{1}{\alpha_{i+1}} - \frac{1}{\alpha_i}\right)\right], \\
&= \alpha_d\left(1 - \frac{1}{\alpha_1}\right)(g(\text{OPT}) - \tau k) - \alpha_d l(\text{OPT})\left(\sum_{i=1}^{d}\frac{\alpha_i}{\alpha_{i+1}} - d\right).
\end{aligned}$$

We use the fact that $\tau \geq 0$ for the second inequality. Dividing both sides by α_d yields the desired result. □

The previous lemma tells us that after the d^{th} pass we obtain a solution S_d whose bicriteria guarantee is at least $\min\left\{\tau k; \left(1 - \frac{1}{\alpha_1}\right)(g(\text{OPT}) - \tau k) - l(\text{OPT})\left(\sum_{i=1}^{d}\frac{\alpha_i}{\alpha_{i+1}} - d\right)\right\}$. Observe that the left-hand side of the bracketed expression is an increasing function of τ whereas the right-hand side is a decreasing function w.r.t τ. By setting,

$$\tau k \triangleq \frac{\alpha_1 - 1}{2\alpha_1 - 1}g(\text{OPT}) - \frac{l(\text{OPT})}{2 - 1/\alpha_1}\left(\sum_{i=1}^{d}\frac{\alpha_i}{\alpha_{i+1}} - d\right), \tag{4.13}$$

both sides of the bracketed expression are equal. Hence, if we set τk to be equal to the right-hand side expression of Equation (4.13), we get that the bicriteria guarantee is at least τk. Therefore, it is sufficient to obtain values for $\alpha_i's$ to derive an approximation guarantee for Algorithm 6. More precisely, we want to find the greatest value of α such that $\tau k \geq \alpha g(\text{OPT}) - l(\text{OPT})$ (which is consistent with the guarantees in [SVW15; Har+19; Kaz+21]). The problem of finding the best approximation guarantee is therefore equivalent to the following non-linear program

$$\begin{aligned}
\text{maximize:} \quad & \frac{\alpha_1 - 1}{2\alpha_1 - 1} \\
\text{subject to:} \quad & \sum_{i=1}^{d}\frac{\alpha_i}{\alpha_{i+1}} - d = 2 - \frac{1}{\alpha_1}, \\
& \alpha_i \geq \alpha_{i+1} \geq 1, \text{ for } i = 1, \ldots, d, \quad \text{and } \alpha_{d+1} = 1
\end{aligned}$$

The next computation shows that the set of $\alpha_i \triangleq (1 + \frac{x^*}{d})^{d+1-i}$ is an *approximate* feasible solution to the above program. The set of α_i's is in fact *exactly* feasible as d tends to infinity.

We will later show that $d = O(\varepsilon^{-1})$ is sufficient to obtain Theorem 4.7.3. Substituting them, the constraint can be written as $d\big(1+\frac{x^*}{d}\big)^{d+1} - (d+2)\big(1+\frac{x^*}{d}\big)^{d} + 1 = 0$. Simplifying the left-hand side gives

$$\begin{aligned} d\left(1+\frac{x^*}{d}\right)^{d+1} - (d+2)\left(1+\frac{x^*}{d}\right)^{d} + 1 &= \left(1+\frac{x^*}{d}\right)^{d}\left(d\left(1+\frac{x^*}{d}\right) - (d+2)\right) + 1 \\ &= \left(1+\frac{x^*}{d}\right)^{d}(x^*-2) + 1 \underset{d\to\infty}{\longrightarrow} e^{x^*}(x^*-2) + 1 = 0, \end{aligned}$$

where the last equality is by definition of x^*. Finally, it is easy to see that $\alpha_i \geq \alpha_{i-1}$ and $\alpha_{d+1} = 1$. Finally, substituting α_1 in the objective function yields the guarantee stated in Theorem 4.7.3. It almost concludes this proof as we still need to show that $O(\varepsilon^{-1})$ passes leads to a deterioration of the approximation guarantee by a factor of at most $(1+\varepsilon)$. Additionally, we need to prove that τ can be guessed accurately. The proof of the latter point can be found in Appendix Section 4.A.3.

Convergence in few passes

Let $d = O\big(\varepsilon^{-1}\big)$ be the number of passes and let $\alpha_i = \big(1+\frac{x^*}{d}\big)^{d+1-i}$ for $i = 1, \ldots, d$, where $x^* = 1.84141$ is the root of the equation $e^x(x-2)+1 = 0$. Focusing on the first constraint of the non-linear program, we have

$$\begin{aligned} \frac{d\Big(\big(1+\frac{x^*}{d}\big)^{d+1} - \big(1+\frac{x^*}{d}\big)^{d}\Big)}{2\big(1+\frac{x^*}{d}\big)^{d} - 1} &= d\left(\left(1+\frac{x^*}{d}\right) - 1\right)\frac{\big(1+\frac{x^*}{d}\big)^{d}}{2\big(1+\frac{x^*}{d}\big)^{d} - 1} \\ &= x^*\frac{\big(1+\frac{x^*}{d}\big)^{d}}{2\big(1+\frac{x^*}{d}\big)^{d} - 1} \\ &\leq x^*\frac{e^{x^*}}{2e^{x^*}\left(1-\frac{(x^*)^2}{d}\right) - 1} \\ &\leq x^*\frac{e^{x^*}}{2e^{x^*} - 1 - O(\varepsilon)} \\ &\leq 1 - O(\varepsilon). \end{aligned}$$

The first inequality is by using the following standard inequality: $(1-\frac{x^2}{n})e^x \leq (1+x/n)^n \leq e^x$, where the lower bounds holds for $n > 1$ and $x < n$. The second inequality is by recalling that $d = O(1/\varepsilon)$. Finally, the last inequality is from the observation that x^* is a solution of $e^x(x-2)+1 = 0$. A similar computation bounds the objective value of the NLP. In particular,

$$\frac{\alpha_1 - 1}{2\alpha_1 - 1} = \frac{(1+x^*/d)^d - 1}{2(1+x^*/d)^d - 1} \geq \frac{e^{x^*}(1-\frac{x_\infty^2}{d}) - 1}{2e^{x^*} - 1} \geq \frac{e^{x^*} - 1}{2e^{x^*} - 1} - O(\varepsilon) \simeq 0.4569 - O(\varepsilon)\,.$$

Recall that we have set

$$\tau \triangleq \frac{1}{k}\left(\frac{\alpha_1 - 1}{2\alpha_1 - 1} g(\mathrm{OPT}) - \frac{l(\mathrm{OPT})}{2 - 1/\alpha_1}\left(\sum_{i=1}^{d} \frac{\alpha_i}{\alpha_{i+1}} - d\right)\right)$$

so that the approximation ratio of Algorithm 6 is equal to τk. Substituting the previous two computations into τ with an appropriate rescaling of ε yields the desired result.

4.8 Conclusion and Open Questions

The main result of this chapter is the derivation of a streaming local-search algorithm. For maximizing a monotone submodular function subject to a p-matchoid constraint, its approximation guarantee is at most $(1+\varepsilon)$ times worse than its offline counterpart using only $O(p/\varepsilon)$ passes. Our approach is versatile and capable of handling non-monotone submodular objectives. By doing so, we are the first to derive a multipass streaming algorithm for maximizing general submodular functions with a number of passes independent of the size of the ground set. In the last part of the chapter, we give a multipass streaming algorithm for maximizing a monotone regularized submodular function subject to a cardinality constraint. For this problem as well, we obtain further improvement compared to best single pass streaming algorithm.

There are plenty of directions of great interest. We list a few of open questions. All problems have stars $\star$ denoting a combination of their difficulty and interest.

- $(\star\star\star)$ Is it possible to get a single pass 2-approximation for maximizing a monotone submodular function subject to a matroid constraint? This is the best attainable factor [Fel+20], and the current best factor is equal 3.1467 [Fel+22]. A single pass 2-approximation exists subject to a cardinality constraint [Nor+18].
- $(\star\star)$ We obtain a $p + 1 + \varepsilon$ approximation for maximizing a monotone submodular function subject to a p-matroid intersection in $O(p/\varepsilon)$-passes. Can we improve this approximation further to $p + \varepsilon$ to match the state-of-the-art result by [LSV10]?
- $(\star\star)$ Our algorithm assumes an adversarial arrival of the ground set. Recently, [Nor+18; Liu+21; Fel+22] obtained improved guarantees/number of passes in the random-order model. In the random-order arrival and subject to a matroid constraint, can we improve over the factor 3.1467 in a single pass?
- $(\star\star)$ A slightly open-ended question is to investigate bad instances for multipass algorithm. The situation is clear for single pass algorithm with hardness results [Fel+20]. What about 2-passes?
- $(\star)$ Is it possible to get a solution S such that $f(S) \geq (1-e^{-1})g(\mathrm{OPT}) - l(\mathrm{OPT})$ in $O(\varepsilon^{-1})$ passes? What about streaming algorithms subject to a matroid constraint or even a p-matchoid? To the best of our knowledge, no streaming algorithm for regularized submodular function maximization beyond cardinality constraint is known.

Appendix

4.A Detailed computations for Section 4.3 and 4.6

4.A.1 Analysis of Chekuri et al.'s algorithm

Here, we give a self-contained analysis of the single-pass algorithm of Chekuri, Gupta, and Quanrud [CGQ15] corresponding to Algorithm 4 initialized with $S_{\text{init}} = \emptyset$. First, we prove Lemma 4.3.1 about the properties of the incremental values.

Lemma 4.3.1. *Given $T, U \subseteq X$ such that $T \subseteq U$, the following properties hold:*

1. $\sum_{e \in T} \nu(e, T) = f(T) - f(\emptyset)$.
2. $\nu(e, U) \leq \nu(e, T)$ *for all* $e \in T$.
3. $f(T \mid U \setminus T) \leq \sum_{t \in T} \nu(t, U)$.
4. *At all times during the execution of* StreamingLocalSearch, $\nu(e, S) \geq \alpha$ *for all* $e \in S$, *where S is the solution maintained during this pass.*

Proof of Lemma 4.3.1. Property (1) follows directly from the telescoping summation

$$\sum_{e \in T} \nu(e, T) = \sum_{e \in T} [f(e \cup \{t' \in T : t' \prec e\}) - f(\{t' \in T : t' \prec e\}] = f(T) - f(\emptyset).$$

Property (2) follows from submodularity since $T \subseteq U$ implies that $\{t' \in T : t' \prec e\} \subseteq \{t' \in U : t' \prec e\}$. For property (3), we note that:

$$\begin{aligned} f(T \mid U \setminus T) &= \sum_{t \in T} f(t \mid U \setminus T \cup \{t' \in T : t' \prec t\}) \\ &\leq \sum_{t \in T} f(t \mid \{u' \in U : u' \prec t\}) \\ &= \sum_{t \in T} \nu(t, U)\,, \end{aligned}$$

where the first equation follows from a telescoping summation, and the inequality follows from submodularity, since $\{u' \in U : u' \prec t\} \subseteq U \setminus T \cup \{t' \in T : t' \prec t\}$.

We prove property (4) by induction on the stream of elements arriving. Initially $S = \emptyset$. Thus, the first time that any element x is accepted, we must have $C_x = \emptyset$ and so $f(x \mid S) \geq \alpha \geq 0$. After this element is accepted, we have $\nu(x, S) = \nu(x, \{x\}) = f(x \mid \emptyset) \geq \alpha$. Proceeding inductively, then, let S_x^- and S_x^+ be the set of elements in S before and after some new element x arrives and is processed by Algorithm 2. Suppose that $\nu(s, S_x^-) \geq \alpha$ for all $s \in S_x^-$. Then, if x is rejected, we have $S_x^+ = S_x^-$ and so $\nu(s, S_x^+) = \nu(s, S_x^-) \geq \alpha$ for all $s \in S_x^+$. If x is accepted, then $S_x^+ = S \setminus C_x + x$ and $f(x \mid S_x^-) \geq \alpha + (1+\beta) \sum_{e \in C_x} \nu(e, S_x^-)$. Thus,

$$\nu(x, S_x^+) \geq f(x \mid S_x^+ - x) \geq f(x \mid S_x^-) \geq \alpha + (1+\beta)|C_x|\alpha \geq \alpha\,,$$

where the first inequality follows from property (2) of the lemma, the second from submodularity, and the third from the induction hypothesis and the assumption that x is accepted. For any other $s \in S_x^+$, we have $\{t' \in S \setminus C_x : t' \prec s\} \subseteq \{t' \in S : t' \prec s\}$ and so by property (3) of the lemma, $\nu(s, S_x^+) \geq \nu(s, S_x^-) \geq \alpha$, as required. □

Recall that we let $\tilde{A}$ be the set of all elements that were accepted by this pass of StreamingLocalSearch (and so at some point appeared in S). For each element $x \in X$, we let S_x^- be the current set S at the moment that x arrives and S_x^+ the set after x is processed. For an element e that is accepted but later *evicted* from S, let $\chi(e)$ be the incremental value $\nu(e, S)$ of e at the moment that e was evicted. The following structural lemma from Chekuri et al. [CGQ15] gives a charging argument to bound the value of an *evicted* element e with respect to the value of a subset of $\phi(e) \subseteq \tilde{A}$, where each element $t \in \phi(e)$ appears in ϕ for a small number of evicted elements.

Lemma 4.A.1 (Lemma 9 of [CGQ15]). *Let $T \in \mathcal{I}$ be a feasible solution disjoint from $\tilde{A}$, and $\tilde{S}$ be the output of the streaming algorithm. There exists a mapping $\varphi : T \to 2^{\tilde{A}}$ such that:*

1. *Every $s \in \tilde{S}$ appears in the set $\varphi(t)$ for at most p choices of $t \in T$.*
2. *Every $e \in \tilde{A} \setminus \tilde{S}$ appears in the set $\varphi(t)$ for at most $p-1$ choices of $t \in T$.*
3. *For each $t \in T$:*
$$\sum_{c \in C_t} \nu(c, S_t^-) \leq \sum_{e \in \varphi(t) \setminus \tilde{S}} \chi(e) + \sum_{s \in \varphi(t) \cap \tilde{S}} \nu(s, \tilde{S})\,.$$

Using this charging argument, we can now prove Lemma 4.3.2 and 4.3.3.

Lemma 4.3.2. *At the end of* StreamingLocalSearch*, the contribution of elements of OPT with respect to $\tilde{A}$ is at most:*

$$f(OPT \cup \tilde{A}) - f(\tilde{A}) \leq (1+\beta)\left[(p-1)\sum_{e \in \tilde{A} \setminus \tilde{S}} \chi(e) + p f(\tilde{S})\right] + k\alpha.$$

Proof of Lemma 4.3.2. Let $R = \text{OPT} \setminus \tilde{A}$ and define S_r^- to be the current solution when element r arrives. Since $S_r^- \subseteq \tilde{A}$ for all r, the submodularity of f implies that

$$f(\text{OPT} \cup \tilde{A}) - f(\tilde{A}) = f(R \cup \tilde{A}) - f(\tilde{A}) \leq \sum_{r \in R} \leq f(r \mid S_r^-) \leq \sum_{r \in R} f(r \mid \tilde{A}). \tag{4.14}$$

For any $r \in R$, since r was rejected upon arrival,

$$f(r \mid S_r^-) \leq (1+\beta) \sum_{c \in C_r} \nu(c, S_r^-) + \alpha\,. \tag{4.15}$$

Thus, applying Lemma 4.A.1 we obtain:

$$\begin{aligned}
\sum_{r \in R} f(r \mid S_r^-) &\leq (1+\beta) \sum_{r \in R} \sum_{c \in C_r} \nu(c, S_r^-) + k\alpha, && \text{((4.15) and } |R| \leq k) \\
&\leq \sum_{r \in R} (1+\beta) \Bigg[\sum_{e \in \varphi(r) \setminus \tilde{S}} \chi(e) + \sum_{s \in \varphi(r) \cap \tilde{S}} \nu(s, \tilde{S}) \Bigg] + k\alpha, && \text{(Lemma 4.A.1 (3))} \\
&\leq (1+\beta) \Bigg[(p-1) \sum_{e \in \tilde{A} \setminus \tilde{S}} \chi(e) + p \sum_{s \in \tilde{S}} \nu(s, \tilde{S}) \Bigg] + k\alpha, && \text{(Lemma 4.A.1 (1, 2))}
\end{aligned}$$

where in the last inequality we have also used Lemma 4.3.1 (4), which implies that each $\chi(e)$ and $\nu(s, \tilde{S})$ is non-negative. Combining the above inequality with (4.14), we obtain

$$\begin{aligned}
f(\text{OPT} \cup \tilde{A}) - f(\tilde{A}) &\leq (1+\beta) \left[(p-1) \sum_{e \in \tilde{A} \setminus \tilde{S}} \chi(e) + p \sum_{s \in \tilde{S}} \nu(s, \tilde{S}) \right] + k\alpha \\
&\leq (1+\beta) \left[(p-1) \sum_{e \in \tilde{A} \setminus \tilde{S}} \chi(e) + p f(\tilde{S}) \right] + k\alpha
\end{aligned}$$

where the second inequality is obtained by applying Lemma 4.3.1 (4) and using that $f(\emptyset) \geq 0$. □

Lemma 4.3.3. *Let $f : 2^X \to \mathbb{R}_{\geq 0}$ be a submodular function. Suppose $\tilde{S}$ is the solution produced at the end of one pass of* StreamingLocalSearch *and $\tilde{A}$ be the set of all elements accepted during this pass. Then,*

$$f(\tilde{A}) \leq f(\tilde{S}) + \sum_{e \in \tilde{A} \setminus \tilde{S}} \chi(e).$$

Proof of Lemma 4.3.3. We bound $f(\tilde{A})$ in terms of the values $\nu(s, \tilde{S})$ and $\chi(e)$. Since $S \subseteq \tilde{A}$ at all times during the algorithm, and $\chi(e) = \nu(e, S)$ at the moment e was evicted, we have $\chi(e) \geq \nu(e, \tilde{A})$ by Lemma 4.3.1 (2). Thus,

$$f(\tilde{A}) - f(\emptyset) = \sum_{a \in \tilde{A}} \nu(a, \tilde{A}) = \sum_{s \in \tilde{S}} \nu(s, \tilde{A}) + \sum_{e \in \tilde{A} \setminus \tilde{S}} \nu(e, \tilde{A}) \leq \sum_{s \in \tilde{S}} \nu(s, \tilde{S}) + \sum_{e \in \tilde{A} \setminus \tilde{S}} \chi(e).$$

where the first equation follows from Lemma 4.3.1 (1). The proof then follows by applying Lemma 4.3.1 (1) to the term $\sum_{s\in\tilde{S}}\nu(s,\tilde{S}) = f(\tilde{S}) - f(\emptyset)$. Since $f(\emptyset)\geq 0$, we get the desired result. □

4.A.2 Missing computations in Theorem 4.4.3

There are two facts which are used in the proof of Theorem 4.4.3 which we prove here:

Claim 4.A.2. *The function* $\gamma_i = \frac{4p\gamma_{i-1}(\gamma_{i-1}-1)}{(\gamma_{i-1}-1+p)^2}$ *is an increasing function of* γ_{i-1} *for* $p\geq 1$.

Proof of Claim 4.A.2. A straightforward algebraic manipulation shows that

$$\begin{aligned}\frac{d}{d\gamma_{i-1}}\gamma_i &= \frac{4p(\gamma_{i-1}-1)+4p\gamma_{i-1}}{(\gamma_{i-1}-1+p)^2} - \frac{8p\gamma_i(\gamma_{i-1}-1)}{(\gamma_{i-1}-1+p)^3}\\ &= \frac{4p(\gamma_{i-1}-1)(\gamma_{i-1}-1+p)+4p\gamma_{i-1}(\gamma_{i-1}-1+p)-8p\gamma_{i-1}(\gamma_{i-1}-1)}{(\gamma_{i-1}-1+p)^3}\\ &\geq \frac{4p\gamma_{i-1}(\gamma_{i-1}-1)+4p\gamma_{i-1}^2-8p\gamma_{i-1}(\gamma_{i-1}-1)}{(\gamma_{i-1}-1+p)^3} \geq 0.\end{aligned}$$

The third line follows from $p\geq 1$ and the final inequality is by $\gamma_{i-1}\geq 1$. □

Claim 4.A.3. *For* $p,i\geq 1$, *the following inequality holds:* $\frac{4p\left(p+1+\frac{4p}{i-1}\right)\left(p+\frac{4p}{i-1}\right)}{\left(2p+\frac{4p}{i-1}\right)^2} \leq p+1+\frac{4p}{i}$.

Proof of Claim 4.A.3. Rearranging both sides and placing over a common denominator gives:

$$\begin{aligned}\frac{4p\left(p+1+\frac{4p}{i-1}\right)\left(p+\frac{4p}{i-1}\right)}{\left(2p+\frac{4p}{i-1}\right)^2} &= \frac{4p\left((p+1)(i-1)+4p\right)\left(p(i-1)+4p\right)}{\left(2p(i-1)+4p\right)^2}\\ &= \frac{4p\left((p+1)(i-1)+4p\right)\left(p(i-1)+4p\right)}{\left(2p(i+1)\right)^2}\\ &= \frac{\left((i-1)(p+1)+4p\right)(i+3)}{(i+1)^2}\\ &= \frac{(i-1)(i+3)i(p+1)+i(i+3)4p}{i(i+1)^2}\\ &= \frac{\left(i^2+2i-3\right)i(p+1)+(i^2+3i)4p}{i(i+1)^2},\end{aligned}$$

and

$$\begin{aligned}p+1+\tfrac{4p}{i} &= \frac{(p+1)i+4p}{i}\\ &= \frac{i(i+1)^2(p+1)+(i+1)^2 4p}{i(i+1)^2}\\ &= \frac{\left(i^2+2i+1\right)i(p+1)+\left(i^2+2i+1\right)4p}{i(i+1)^2}.\end{aligned}$$

Then, since $p \geq 1$ and $i \geq 1$,

$$\left(p+1+\tfrac{4p}{i}\right) - \frac{4p\left(p+1+\frac{4p}{i-1}\right)\left(p+\frac{4p}{i-1}\right)}{\left(2p+\frac{4p}{i-1}\right)^2} = \frac{4i(p+1)-4(i-1)p}{i(i+1)^2} \geq 0. \qquad \square$$

4.A.3 Approximately guessing the value of the optimal solution

A common assumption for the analysis of Algorithm 5 and 6 is the access to a threshold α and τ equal to $\frac{\varepsilon f(\text{OPT})}{2k}$ and $0.4569g(\text{OPT}) - l(\text{OPT})$ respectively. We show how to remove those assumptions.

Guessing the value of $f(\text{OPT})$ is a common technique in streaming submodular function maximization. Badanidiyuru et al. [Bad+14] showed how to approximate $f(\text{OPT})$ within a $(1\pm\varepsilon)$- factor using $O(\log(k)/\varepsilon)$ space in a single pass. To avoid further technicalities, we show how to guess α and τ in two passes – which doesn't change much since our algorithms run in $O(1/\varepsilon)$-passes. Since we will only need to guess α within a factor 2, this operation incurs a cost of $O(\log(k))$ without dependence on ε. For τ, we will need guess $0.4569g(\text{OPT}) - l(\text{OPT})$ within a factor $1 \pm \varepsilon$ which uses $O(\log(k)/\varepsilon)$ additional space.

Using one pass find $\Delta_\alpha \triangleq \max_{e\in X} f(e)$ and $\Delta_\tau \triangleq \max_{e\in X} 0.4659g(e) - l_e$. We consider the sets

$$\Lambda_\alpha \triangleq \left\{2^i \colon \frac{\Delta_\alpha}{k} \leq 2^i \leq \Delta_\alpha, i \in \mathbb{N}\right\} \qquad \text{and } \Lambda_\tau \triangleq \left\{(1+\varepsilon)^i \colon \frac{\Delta_\tau}{k} \leq (1+\varepsilon)^i \leq \Delta_\tau, i \in \mathbb{N}\right\}.$$

Case 1 – α: There exists a value $\lambda \in \Lambda_\alpha$ such that $\frac{f(\text{OPT})}{2k} \leq \lambda \leq \frac{f(\text{OPT})}{k}$. Setting $\alpha = \frac{\varepsilon\lambda}{2}$, we get that $\alpha \in [\varepsilon f(\text{OPT})/(4k); \varepsilon f(\text{OPT})/(2k)]$. The defined range of α is sufficient for the analysis[5]. While it is not possible to know which $\lambda \in \Lambda_\alpha$ satisfies the property, it suffices to run parallel instantiations and output the best solution of all the copies. This procedure augments the space by a factor $O(\log_2(k))$, and requires one additional pass through the dataset.

Case 2 – τ: By definition of Λ_τ, there is at least one value in $\tau \in \Lambda$ such that $k\tau \leq 0.4569g(\text{OPT}) - l(\text{OPT}) \leq (1+\varepsilon)k\tau$. Thus, if we run Algorithm 6 for each value of τ in parallel then, for at least one copy, our guess of τ is an approximation of the correct threshold value. It is then sufficient to return the best solution amongst all copies. Observe that the number copies is equal to $|\Lambda_\tau| = \log_{1+\varepsilon}(k) = O(\varepsilon^{-1}\log(k))$. Hence, the space complexity increases by a factor $O(\varepsilon^{-1}\log(k))$ via this operation.

[5] Equation (4.6) and the bound $\alpha k \leq \varepsilon f(\text{OPT})$ are where we need the exact value of α, using upper and lower bounds for α yield the same result up to the hidden constant in the term $O(\varepsilon)$.

5 Sparse Subset Selection Problems under Matroid Constraint

A portion of this chapter is part of a publication which appeared in COLT'22 [TW22]. Nonetheless, the presentation of the results is specific to this thesis. The results in Section 5.7 did not appear in the publication.

5.1 Introduction

Subset selection problems are ubiquitous in statistics and machine-learning since they provide interpretability of high-dimensional models through the selection of a few features of interest. Given a collection of features $\mathcal{X}$, the goal is to find a small collection $\mathcal{S} \in \mathcal{I}$ that best predict a quantity of interest. Despite the low occurrence of the term *independence system* in the machine learning literature, standard applications require $\mathcal{I}$ to be a set of at most k variables. So, the pair $(\mathcal{X}, \mathcal{I})$ usually forms an independence system where $\mathcal{I}$ is a uniform matroid. When exhaustive enumeration is impossible due to the size of $\mathcal{X}$, FORWARD REGRESSION is commonly employed as a heuristic. It constructs $\mathcal{S}$ iteratively, and at each step adds a feature that greedily maximizes the objective.

To explain the success of this approach in practice, Das and Kempe [DK11] connected subset selection problems with submodular optimization. They showed that set functions that are γ-*weakly submodular* (Definition 1.3.1) can be efficiently optimized. The approximation guarantee depends on the *submodularity ratio* γ which measures the deviation of the function from submodularity when considering the aggregate effect of adding elements. By treating FORWARD REGRESSION as a variant of the standard greedy algorithm, they showed that it has an approximation guarantee of $\frac{e^\gamma}{e^\gamma - 1}$.

R^2 **Objective**: As the main problem in their work, Das and Kempe [DK08; DK11] focus on maximizing the *squared multiple correlation objective*, i.e. the R^2 objective. Given a collection $\mathcal{X}$ of predictor variables and a target random variable Z, as well as the covariance between each pair of variables, the goal is to find a small subset $\mathcal{S} \subseteq \mathcal{X}$ that gives the best linear predictor for Z. They show that the R^2 objective is *submodular* in the absence of suppressor variables [DK08]. Intuitively, a random variable $X \in \mathcal{X}$ is a suppressor if there is some other random variable $Y \in \mathcal{X}$ such that observing X *increases* the (conditional) correlation between Y and the target variable Z. In [DK11], they extend this connection and further show that even with suppressor variables the submodularity ratio γ is non-trivially bounded. They proved that it is at least equal to the smallest $2k$-sparse eigenvalue $\lambda_{\min}(C_{\mathcal{X}}, 2k)$ of the covariance matrix for $\mathcal{X}$. The smallest $2k$-sparse eigenvalue of $C_{\mathcal{X}}$ is the minimum eigenvalue of any $2k \times 2k$ submatrix of $C_{\mathcal{X}}$.

In this chapter, we consider a natural generalization of subset selection problems in which we must select a subset $\mathcal{S}$ that is independent in a general matroid $\mathcal{M} = (\mathcal{X}, \mathcal{I})$. Such constraints naturally capture settings in which some observations are mutually exclusive (for example, sensors that may be placed in different configurations) or in which it is desirable or necessary to spread observations amongst multiple different classes (for example by time or location). In contrast to a cardinality constraint, the best known guarantee for maximizing the R^2 objective in a general matroid is a randomized $(1 + \gamma^{-1})^2$-approximation via the ResidualRandomGreedy algorithm due to [CFK18].

However, as γ tends to 1 (i.e. as the function becomes closer to a submodular function) this bound tends to only 4, while ResidualRandomGreedy is known to provide a 2 approximation for submodular objectives under a matroid constraint. Furthermore, we recall that the state-of-the-art algorithm achieves a tight $\frac{e}{e-1}$-approximation [Cal+11; FW14; NW78; Fei98]. A key difficulty is that the definition of weak submodularity considers only the effect of *adding* elements to the current solution. In contrast, the analysis of ResidualRandomGreedy as well other state-of-the art procedures for submodular optimization in a matroid require bounding the losses when elements are *removed* or *swapped* from some solution[1].

In this chapter, we overcome these challenges and show that subset selection problems (including the R^2 objective) satisfy a stronger notion than that of Das and Kempe. We define it as (γ, β)-weak submodularity. This definition allows us to improve over the current analysis of ResidualRandomGreedy and devise an algorithm with asymptotic performance of $\frac{e}{e-1}$.

5.1.1 Main Results

We give a natural extension of the submodularity ratio γ by considering an *upper submodularity ratio* $\beta > 0$, that bounds how far a function deviates from submodularity when considering the effect of *removing* elements.

Definition 5.1.1 (Upper Submodularity Ratio)**.** Given a positive monotone set function $f : 2^X \to \mathbb{R}_{\geq 0}$, the upper submodularity ratio β is the minimum value such that for any $A \subseteq B \subseteq X$, the following holds:

$$\beta \cdot (f(B) - f(A)) \geq \sum_{e \in B \setminus A} f(e \mid B - e). \tag{5.1}$$

Moreover, if f has lower/upper submodularity ratio γ/β respectively (Definition 1.3.1), then we say that f is (γ, β)-*weakly submodular*.

Intuitively, the parameter β compares the loss by removing an entire set to the aggregate individual losses for each element. This parameter is a natural complement of Definition 1.3.1 and smoothly captures the deviation from submodularity. In particular, we have that $\beta \leq 1$ if

[1] Stronger "element-wise" notions of weak submodularity have been proposed that allow the adaptation of such algorithms but in general, these notions may give weaker bounds than those obtained when the submodularity ratio γ can be utilized instead

and only if f is a monotone submodular function. We show that, as with the submodularity ratio γ, our β can be bounded by spectral quantities in the setting of regression. Specifically, we show that

Theorem

The R^2 objective function is $(\gamma, 1/\gamma)$-weakly submodular where $\gamma \geq \lambda_{\min}^{-1}(C)$, and C is the covariance matrix between the variables in $\mathcal{X}$.

The formal statement of the above theorem is Theorem 5.2.1. Since β is defined in terms of removing elements from the solution, the proof requires a different spectral argument than that used to bound γ in [DK18]. While their bound for γ follows directly by considering an appropriate Rayleigh quotient, here we must relate the average value of the quadratic forms obtained from the inverses of all rank $k-1$ principle submatrices of a matrix C to that obtained from C^{-1}.

Spectral bounds on the upper submodularity are attainable for other subset selection problems. In fact, we consider Bayesian A-optimal Design and Column Subset Selection [Alt+16; Far+15]. The first problem has been previously studied via weak submodularity by [Bia+17; Har+19; Has+19]. For both problems, we show in Section 5.6 and 5.7 that β can be bounded by γ^{-1}.

Using the connection between subset selection problems and submodular maximization, we consider the more general problem of maximizing a (γ, β)-*weakly submodular* function under a matroid constraint. We derive improved guarantees for the ResidualRandomGreedy.

Theorem

Given a (γ, β)-weakly submodular function, ResidualRandomGreedy has approximation factor $1 + \frac{\beta}{\gamma}$.

The exact statement of the above theorem is Theorem 5.3.1. For all applications considered in this paper, which are $(\gamma, 1/\gamma)$-weakly submodular, the approximation factor improves over the state-of-the-art guarantee equal to $(1 + \gamma^{-1})^2$ by Chen et al. [CFK18] for all values of γ. Our guarantee approaches 2 as the function f becomes closer to submodular (i.e. as $\gamma, \beta \to 1$). It is natural to ask whether it is possible to obtain an algorithm with guarantee approaching the optimal result of $\frac{e}{e-1}$ for monotone submodular functions. We answer this question affirmatively by giving a local search algorithm guided by a distorted potential. We show that it achieves a guarantee approaching $(\frac{e}{e-1} - \varepsilon)$ for (γ, β)-weakly submodular functions as $\gamma, \beta \to 1$, where $\varepsilon > 0$ is a constant parameter that can be chosen independently of γ and β.

Theorem

There is a local-search algorithm with approximation factor $\frac{\phi e^{\phi}}{\gamma^2(e^{\phi}-1)}$ for maximizing (γ, β)-weakly submodular functions, where $\phi = \gamma^2 + \beta(1-\gamma)$.

The formal statement of the above theorem is given by Theorem 5.4.2. For all applications considered in this paper, which are $(\gamma, 1/\gamma)$-weakly submodular, our approximation factor improves over Theorem 5.3.1 (and thus the state-of-the-art) for all $\gamma > 0.7217$. Figure 5.1 displays the improvements. Our distorted local-search algorithm builds upon similar techniques from the submodular case presented in [FW14]. There, the submodularity of f implies the submodularity of the potential g, which is used to derive the bounds on g necessary for convergence and sampling, as well the crucial bound linking the local optimality of g to the value of f. Here, however, since f is only approximately submodular, these techniques will not work, and so we require a more delicate analysis for each of these components. A further complication in our setting is that the correct potential g depends on the values of γ and β, which may not be known a priori. We give an approach that is based on guessing the value of a joint parameter in γ and β. Each such guess gives a different distorted potential. Inspired (broadly) by simulated annealing, we show that if each such new potential is initialized by the local optimum of the previous potential, then the overall running time can be amortized over all guesses. We present a simplified version of the algorithm and potential function in Section 5.4, and defer the more technical details to Section 5.5.

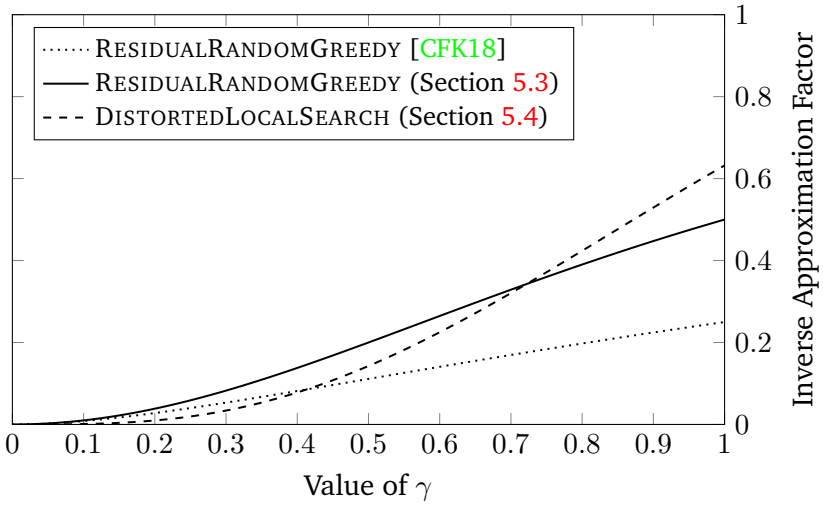


Fig. 5.1: Guarantees for $(\gamma, 1/\gamma)$-weakly submodular function maximization under a matroid constraint.

Finally, given the relationship between γ and β in all problems we consider, it is natural to conjecture that β may be bounded in terms of γ in some generic fashion for *every* set function with lower submodularity ratio γ. However, we show that this is not the case by exhibiting (in Section 5.8) a function on a ground set of size k for which β must be $\Omega\big(k^{1-\gamma}\big)$.

5.1.2 Weak Submodularity and Related Definitions

The (lower) submodularity ratio γ (see Definition 1.3.1) and the corresponding notion of weak submodularity was first introduced to analyze the forward regression and orthogonal matching pursuit algorithms for linear regression [DK18]. It was later related to restricted

strong convexity in [Ele+18] leading to similar guarantees for generalized linear model (GLM) likelihood, graphical model learning objectives, or an arbitrary M-estimator. The submodularity ratio has also been applied to the analysis of greedy algorithms in other modes of computation [Kha+17b; Ele+17]. Together with related algorithmic techniques, it also leads to algorithms for sensor placement problems [Has+20], experimental design [Har+19; Bia+17], low rank optimization [Kha+17a], document summarization [CFK18], and interpretation of neural networks [Ele+17].

Other variants of weak submodularity have been introduced. There have been various approaches based on considering *element-wise* bounds on the deviation of a function from submodularity [BZC18; Non+19; Gon+19] including generalizations to functions over the integer lattice [Qia+18; Kuh+18]. These approaches all involve relaxing the notion of decreasing marginal returns by requiring that a function f satisfy $f(A \cup \{e\}) - f(A) \geq \gamma_e \cdot (f(B \cup \{e\}) - f(B))$ for all $e \notin B$ and $A \subseteq B$, where $\gamma_e \in [0, 1]$ has been variously dubbed the *inverse curvature* [BZC18], *DR ratio* [Kuh+18], or *generic submodularity ratio* [Non+19]. For such functions, it is easy to show that our parameters satisfy $\gamma \geq \gamma_e$ and $\beta \leq 1/\gamma_e$. Unfortunately, as we show in Section 5.2, the resulting inequalities may be very far from tight in our setting. In particular, analyses relying on γ_e may fail to give any non-trivial approximation bounds for regression problems, even in situations when $\lambda_{\min}(C_{\mathcal{X}})$ and the submodularity ratio γ are positive. This observation motivates our consideration of the more general parameter β, which allows spectral bounds to be utilized.

5.2 Sparse Least Square Estimator

We now turn to the sparse least-square estimator problem. Let Z be a target random variable we wish to predict, and let $\mathcal{X} = \{X_1, \ldots, X_n\}$ be a set of n predictor variables (where here and throughout this section we use calligraphic letters to denote sets of random variables to avoid confusion). We suppose that Z and all X_i have been normalized to have mean 0 and variance 1, and let $C_{\mathcal{X}}$ be the $n \times n$ covariance matrix for the variables X_i. Our goal is to find a set $\mathcal{S} \subseteq \mathcal{X}$, that gives the best linear predictor for Z. We additionally require $\mathcal{S}$ to be independent in some given matroid $\mathcal{M} = (\mathcal{X}, \mathcal{I})$. In other words, we want to solve the following optimization problem:

$$\arg\max_{\mathcal{S} \in \mathcal{I}} R^2_{Z,\mathcal{S}} = \arg\max_{\mathcal{S} \in \mathcal{I}} \frac{\mathrm{Var}(Z) - \mathbb{E}[(Z - Z_{\mathcal{S}})^2]}{\mathrm{Var}(Z)}, \tag{5.2}$$

where R^2 is a measure of fitness of the linear predictor using the *squared multiple correlation*, and $Z_{\mathcal{S}} = \sum_{X_i \in \mathcal{S}} \alpha_i X_i$ is the linear predictor over $\mathcal{S}$ which optimally minimizes the mean square prediction error for Z. Thus, the R^2 objective can be regarded as a measure of the fraction of variance of Z that is explained by $\mathcal{S}$. The coefficients of this best linear predictor are given by $\boldsymbol{\alpha} = C_{\mathcal{S}}^{-1} \mathbf{b}_{Z,\mathcal{S}}$, where $C_{\mathcal{S}}$ is the principle submatrix of $C_{\mathcal{X}}$ corresponding to variables in $\mathcal{S}$, and $\mathbf{b}_{Z,\mathcal{S}}$ is a vector of covariances between Z and each $X_i \in \mathcal{S}$, i.e. $(C_{\mathcal{S}})_{i,j} = \mathrm{Cov}(X_i, X_j)$ and $(\mathbf{b}_{Z,\mathcal{S}})_i = \mathrm{Cov}(X_i, Z)$. Therefore, if we let $X_{\mathcal{S}}$ denote the corresponding vector of random variables in $\mathcal{S}$, the best linear predictor can be written as: $Z_{\mathcal{S}} = X_{\mathcal{S}}^T C_{\mathcal{S}}^{-1} \mathbf{b}_{Z,\mathcal{S}}$.

Because Z has unit variance, the objective simplifies to $R^2_{Z,\mathcal{S}} = 1 - \mathbb{E}[(Z - Z_{\mathcal{S}})^2]$. In addition, we can define the *residual* of Z with respect to this predictor as the random variable $\mathrm{Res}(Z, \mathcal{S}) = Z - Z_{\mathcal{S}} = Z - X_{\mathcal{S}}^T C_{\mathcal{S}}^{-1} \mathbf{b}_{Z,\mathcal{S}}$. Therefore,

$$R^2_{Z,\mathcal{S}} = 1 - \mathrm{Var}(\mathrm{Res}(Z, \mathcal{S})) = \mathbf{b}_{Z,\mathcal{S}}^T C_{\mathcal{S}}^{-1} \mathbf{b}_{Z,\mathcal{S}}.$$

Our main result in this section is to prove an analogus result to that of Das and Kempe [DK18]. They show that the R^2 objective (5.2) satisfies Definition 1.2 for all $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{X}$ with $\gamma \geq \lambda_{\min}(C_{\mathcal{X}}, |\mathcal{B}|) \geq \lambda_{\min}(C_{\mathcal{X}})$, where $\lambda_{\min}(C_{\mathcal{X}})$ is the smallest eigenvalue of $C_{\mathcal{X}}$ and $\lambda_{\min}(C_{\mathcal{X}}, |\mathcal{B}|)$ is the smallest $|\mathcal{B}|$-*sparse eigenvalue of* $C_{\mathcal{X}}$. In this section we derive the complementary theorem

Theorem 5.2.1. *The R^2 objective function is $(\gamma, 1/\gamma)$-weakly submodular where $\gamma \geq \lambda_{\min}(C, 2k)$, and $\lambda(C, 2k)$ is the smallest $2k$-sparse eigenvalue of the covariance matrix C between the random variables in $\mathcal{X}$.*

As a warm-up, we consider the following small example that illustrates that the *element-wise* bounds on γ_e (inverse curvature, DR ratio, or generic submodularity ratio) are in general not bounded by $\lambda_{\min}(C_{\mathcal{X}})$. In fact, it shows that we may have $\gamma_e = 0$ (and so approximation bounds based on γ_e fail) even when $\lambda_{\min}(C_{\mathcal{X}})$ is bounded away from 0. It demonstrates that the element-wise bound γ_e is not suited for the problem we consider and that the guarantees that we obtain are in fact stronger than that if we were to replace γ by γ_e.

Example 5.2.2. Let Z,X_1,X_2 be random variables with unit variance and zero mean. Suppose that X_1 is uncorrelated with Z, and $X_2 = (Z + X_1)/\sqrt{2}$. Then, $\mathrm{Cov}(X_1, Z) = 0$ and $\mathrm{Cov}(X_1, X_2) = \mathrm{Cov}(X_2, Z) = 1/\sqrt{2}$. Let $f(\mathcal{S}) = R^2_{Z,\mathcal{S}}$. Then, it can be verified that $f(X_1|\emptyset) = 0$ and $f(X_1|\{X_2\}) = 1/2$. Thus $f(X_1|\emptyset) \geq \gamma_e \cdot f(X_1|\{X_2\})$ is satisfied only for $\gamma_e = 0$. However, $\lambda_{\min}(C_{\mathcal{X}}^{-1}) = 1 - 1/\sqrt{2}$ and, in fact, explicitly computing γ gives $\gamma = 1/2$.

Next, we turn to the proof of Theorem 5.2.1. In order to prove our bounds, we will use the following facts stated in [DK18]:

Lemma 5.2.3. *Given two sets of random variables $\mathcal{S} = \{X_1, \ldots, X_n\}$, and $\mathcal{A}$, and a random variable Z we have:* $\mathrm{Res}(Z, \mathcal{A} \cup \mathcal{S}) = \mathrm{Res}(\mathrm{Res}(Z, \mathcal{A}), \{\mathrm{Res}(X_i, \mathcal{A})\}_{X_i \in \mathcal{S}})$.

Lemma 5.2.4. *Given two sets of random variables $\mathcal{S} = \{X_1, \ldots, X_n\}$, and $\mathcal{A}$, and a random variable Z we have:* $R^2_{Z,\mathcal{A}\cup\mathcal{S}} = R^2_{Z,\mathcal{A}} + R^2_{Z,\{\mathrm{Res}(X_i,\mathcal{A})\}_{X_i \in \mathcal{S}}}$.

We define the following quantities, which we use for the rest of the section. Let $\mathcal{A}, \mathcal{B}$ be some fixed sets of random variables with $\mathcal{A} \subseteq \mathcal{B}$. Let $\mathcal{T} = \mathcal{B} \setminus \mathcal{A}$ and suppose without loss of generality that $\mathcal{T} = \{X_1, \ldots, X_t\}$. For each $X_i \in \mathcal{T}$, let $\hat{X}_i = \mathrm{Res}(X_i, \mathcal{A})$ and suppose further that each $\hat{X}_i$ has been renormalized to have unit variance. Finally, let $\hat{\mathcal{T}} = \{\hat{X}_i, \ldots, \hat{X}_t\}$, $\hat{C}$ be the covariance matrix for $\hat{\mathcal{T}}$, and $\hat{\mathbf{b}}$ to be the vector of covariances between Z and each $\hat{X}_i \in \hat{\mathcal{T}}$. We fix a single random variable X_i. For ease of notation, in the next two lemmas we assume without loss of generality that $\hat{C}$ and $\hat{\mathbf{b}}$ have been permuted so that X_i corresponds to

the last row and column of $\hat{C}$. Then, we define $\mathcal{T}_{-i} = \mathcal{T} \setminus \{X_i\}$, $\hat{\mathcal{T}}_{-i} = \hat{\mathcal{T}} \setminus \{\hat{X}_i\}$, and let $\hat{X}_{-i}$ denote the vector containing the variables of $\hat{\mathcal{T}}_{-i}$ (ordered as in $\hat{C}$ and $\hat{\mathbf{b}}$). Similarly, let $\hat{C}_{-i}$ be the principle submatrix of $\hat{C}$ obtained by excluding the row and column corresponding to $\hat{X}_i$ (i.e., the last row and column), and $\hat{\mathbf{b}}_{-i}$ be the vector obtained from $\hat{\mathbf{b}}$ by excluding the entry for $\hat{X}_i$ (i.e., the last entry). Finally, we let $\mathbf{u}_i$ be the vector of covariances between $\hat{X}_i$ and each $\hat{X}_j \in \hat{\mathcal{T}}_{-i}$. Note that $\mathbf{u}_i$ corresponds to the last column of $\hat{C}$ with its last entry (corresponding to $\mathrm{Var}(\hat{X}_i)$) removed. We begin by computing the loss in $R^2_{Z,\mathcal{B}}$ when removing X_i from $\mathcal{B}$:

Lemma 5.2.5. $R^2_{Z,\mathcal{B}} - R^2_{Z,\mathcal{B}\setminus\{X_i\}} = \mathrm{Cov}(Z, \mathrm{Res}(\hat{X}_i, \hat{\mathcal{T}}_{-i}))^2 / \mathrm{Var}(\mathrm{Res}(\hat{X}_i, \hat{\mathcal{T}}_{-i})) = \hat{\mathbf{b}}^T H_i \hat{\mathbf{b}} / s_i$, *where* $H_i = \begin{pmatrix} \hat{C}_{-i}^{-1}\mathbf{u}_i\mathbf{u}_i^T\hat{C}_{-i}^{-1} & -\hat{C}_{-i}^{-1}\mathbf{u}_i \\ -\mathbf{u}_i^T\hat{C}_{-i}^{-1} & 1 \end{pmatrix}$ *and* $s_i = 1 - \mathbf{u}_i^T\hat{C}_{-i}^{-1}\mathbf{u}_i$.

Proof of Lemma 5.2.5. Note that $\mathcal{B} \setminus \{X_i\} = \mathcal{A} \cup \mathcal{T}_{-i}$. Thus, by Lemma 5.2.4 and Lemma 5.2.3, respectively:

$$R^2_{Z,\mathcal{B}} - R^2_{Z,\mathcal{B}\setminus\{X_i\}} = R^2_{Z,\mathrm{Res}(X_i,\mathcal{A}\cup\mathcal{T}_{-i})} = R^2_{Z,\mathrm{Res}(\mathrm{Res}(X_i,\mathcal{A}),\{\mathrm{Res}(X_j,\mathcal{A})\}_{X_j\in\mathcal{T}_{-i}})}. \tag{5.3}$$

Recall that each $\hat{X}_j$ is obtained from $\mathrm{Res}(X_j, \mathcal{A})$ by renormalization and that $\hat{\mathcal{T}}_{-i} = \hat{\mathcal{T}} \setminus \{\hat{X}_i\} = \{\hat{X}_j\}_{X_j\in\mathcal{T}_{-i}}$. Thus, $\mathrm{Res}(\hat{X}_i, \hat{\mathcal{T}}_{-i})$ is a rescaling of $\mathrm{Res}(\mathrm{Res}(X_i, \mathcal{A}), \{\mathrm{Res}(X_j, \mathcal{A})\}_{X_j\in\mathcal{T}_{-i}})$. Since the R^2 objective is invariant under scaling of the predictor variables, Equation (5.3) then implies that

$$R^2_{Z,\mathcal{B}} - R^2_{Z,\mathcal{B}\setminus\{X_i\}} = R^2_{Z,\mathrm{Res}(\hat{X}_i,\hat{\mathcal{T}}_{-i})} = \mathrm{Cov}(Z, \mathrm{Res}(\hat{X}_i, \hat{\mathcal{T}}_{-i}))^2 / \mathrm{Var}(\mathrm{Res}(\hat{X}_i, \hat{\mathcal{T}}_{-i}))\,, \tag{5.4}$$

where the last line follows directly from the definition of the R^2 objective applied to $\mathcal{S}$ that consists of a single random variable. It remains to express (5.4) in terms of $\hat{C}$, $\hat{\mathbf{b}}$ and $\mathbf{u}$. By definition, $\mathrm{Res}(\hat{X}_i, \hat{\mathcal{T}}_{-i}) = \hat{X}_i - \hat{X}_{-i}^T\hat{C}_{-i}^{-1}\mathbf{u}_i$. Hence,

$$\begin{aligned}\mathrm{Var}(\mathrm{Res}(\hat{X}_i, \hat{\mathcal{T}}_{-i})) &= \mathrm{Var}(\hat{X}_i - \hat{X}_{-i}^T\hat{C}_{-i}^{-1}\mathbf{u}_i) \\ &= \mathbb{E}[\hat{X}_i^2] - 2\,\mathbb{E}[\hat{X}_i\hat{X}_{-i}^T]\hat{C}_{-i}^{-1}\mathbf{u}_i + \mathbf{u}_i^T\hat{C}_{-i}^{-1}\,\mathbb{E}[\hat{X}_{-i}\hat{X}_{-i}^T]\hat{C}_{-i}^{-1}\mathbf{u}_i = 1 - \mathbf{u}_i^T\hat{C}_{-i}^{-1}\mathbf{u}_i,\end{aligned}$$

where the last equality follows from normalization of $\hat{X}_i$, $\mathbb{E}[\hat{X}_i\hat{X}_{-i}^T] = \mathbf{u}_i^T$ and $\mathbb{E}[\hat{X}_{-i}\hat{X}_{-i}^T] = \hat{C}_{-i}$. Furthermore,

$$\begin{aligned}\mathrm{Cov}(Z, \mathrm{Res}(\hat{X}_i, \hat{\mathcal{T}}_{-i}))^2 = \mathrm{Cov}(Z, \hat{X}_i - \hat{X}_{-i}^T\hat{C}_{-i}^{-1}\mathbf{u}_i)^2 &= \left(\mathrm{Cov}(Z, \hat{X}_i) - \mathrm{Cov}(Z, \hat{X}_{-i}^T\hat{C}_{-i}^{-1}\mathbf{u}_i)\right)^2 \\ = \left(\hat{b}_i - \hat{\mathbf{b}}_{-i}^T\hat{C}_{-i}^{-1}\mathbf{u}_i\right)^2 &= \hat{\mathbf{b}}^T\begin{pmatrix} \hat{C}_{-i}^{-1}\mathbf{u}_i\mathbf{u}_i^T\hat{C}_{-i}^{-1} & -\hat{C}_{-i}^{-1}\mathbf{u}_i \\ -\mathbf{u}_i^T\hat{C}_{-i}^{-1} & 1 \end{pmatrix}\hat{\mathbf{b}}.\end{aligned}$$

Substituting the above 2 expressions into Equation (5.4) completes the proof. □

Although the previous lemma has an intricate form, we show that it can be greatly simplified for eigenvectors of $\hat{C}^{-1}$. It will become helpful later to decompose $\hat{\mathbf{b}}$ with respect to eigenvectors of $\hat{C}^{-1}$.

Lemma 5.2.6. *Let $(\lambda, \mathbf{v}), (\mu, \mathbf{w})$ be any 2 eigenpairs of $\hat{C}^{-1}$. Then, $\mathbf{v}^T H_i \mathbf{w} = \lambda\mu s_i^2 v_i w_i$, where H_i and s_i are as defined in the statement of Lemma 5.2.5.*

Proof of Lemma 5.2.6. Applying the formula for block matrix inversion (Lemma 1.B.1) to $\hat{C}^{-1}$, we have

$$\hat{C}^{-1} = \begin{pmatrix} \hat{C}_{-i} & \mathbf{u}_i \\ \mathbf{u}_i^T & 1 \end{pmatrix}^{-1} = \begin{pmatrix} \hat{C}_{-i}^{-1} & 0 \\ 0 & 0 \end{pmatrix} + \frac{1}{1 - \mathbf{u}_i^T \hat{C}_{-i}^{-1} \mathbf{u}_i} \begin{pmatrix} \hat{C}_{-i}^{-1} \mathbf{u}_i \mathbf{u}_i^T \hat{C}_{-i}^{-1} & -\hat{C}_{-i}^{-1} \mathbf{u}_i \\ -\mathbf{u}_i^T \hat{C}_{-i}^{-1} & 1 \end{pmatrix}. \tag{5.5}$$

Now, because $(\mu, \mathbf{w})$ is an eigenpair of $\hat{C}^{-1}$, we must have $(\hat{C}^{-1}\mathbf{w})_i = \mu w_i$. By (5.5), this is equivalent to $(-\mathbf{u}_i^T \hat{C}_{-i}^{-1} \mathbf{w}_{-i} + w_i)/s_i = \mu w_i$ (where, as usual, we let $\mathbf{w}_{-i}$ be the vector obtained from $\mathbf{w}$ by discarding its i^{th} entry). Rearranging this equation gives $\mathbf{u}_i^T \hat{C}_{-i}^{-1} \mathbf{w}_{-i} = w_i(1 - \mu s_i)$. Since $\hat{C}^{-1}$ is symmetric, the same argument implies that $\mathbf{v}_{-i}^T \hat{C}_{-i}^{-1} \mathbf{u}_i = v_i(1 - \lambda s_i)$. Thus,

$$\begin{aligned} \mathbf{v}^T H_i \mathbf{w} &= \mathbf{v}_{-i}^T \hat{C}_{-i}^{-1} \mathbf{u}_i \mathbf{u}_i^T \hat{C}_{-i}^{-1} \mathbf{w}_{-i} - w_i(\mathbf{v}_{-i}^T \hat{C}_{-i}^{-1} \mathbf{u}_i) - v_i(\mathbf{u}_i^T \hat{C}_{-i}^{-1} \mathbf{w}_{-i}) + v_i w_i \\ &= v_i w_i (1 - \lambda s_i)(1 - \mu s_i) - v_i w_i (1 - \lambda s_i) - v_i w_i (1 - \mu s_i) + v_i w_i \\ &= v_i w_i \left((1 - \lambda s_i)(1 - \mu s_i) - (1 - \lambda s_i) - (1 - \mu s_i) + 1 \right) = \lambda\mu s_i^2 v_i w_i \,, \end{aligned}$$

as claimed. □

We can now complete the proof of our main result from this section (Theorem 5.2.1).

Proof of Theorem 5.2.1. Since $\hat{C}$ is a symmetric real matrix of size t, there is an eigenbasis $\{\mathbf{v}_1, \ldots, \mathbf{v}_t\}$ of $\hat{C}^{-1}$ with corresponding real eigenvalues $\lambda_1, \ldots, \lambda_t$. Let V be a matrix with columns given by these $\mathbf{v}_i$. Since $\hat{C}^{-1}$ is symmetric positive semidefinite, the matrix V is orthonormal. Hence, we can write $\hat{\mathbf{b}} = V\mathbf{y}$ for some vector $\mathbf{y}$. By Lemma 5.2.5, $\hat{\mathbf{b}}^T H_i \hat{\mathbf{b}} = \mathrm{Cov}(Z, \mathrm{Res}(\hat{X}_i, \hat{\mathcal{T}}_{-i}))^2 \geq 0$ and $s_i = \mathrm{Var}(\mathrm{Res}(\hat{X}_i, \hat{\mathcal{T}}_{-i})) \leq 1$, for each $i = 1, \ldots, t$ and $R^2_{Z,\mathcal{B}} - R^2_{Z,\mathcal{B}\setminus\{X_i\}} = \hat{\mathbf{b}}^T H_i \hat{\mathbf{b}}/s_i \leq \hat{\mathbf{b}}^T H_i \hat{\mathbf{b}}/s_i^2 = \mathbf{y}^T V^T H_i V \mathbf{y}/s_i^2$. Finally, by Lemma 5.2.6, $(V^T H_i V)_{\ell,m} = \lambda_\ell \lambda_m s_i^2 (\mathbf{v}_\ell)_i (\mathbf{v}_m)_i$. Thus, summing over all i we have:

$$\begin{aligned} \sum_{i=1}^t R^2_{Z,\mathcal{B}} - R^2_{Z,\mathcal{B}\setminus\{X_i\}} &\leq \sum_{i=1}^t \sum_{\ell,m=1}^t (y_\ell y_m \lambda_\ell \lambda_m)(\mathbf{v}_\ell)_i (\mathbf{v}_m)_i \,. \\ &= \sum_{\ell,m=1}^t (y_\ell y_m \lambda_\ell \lambda_m) \sum_{i=1}^t (\mathbf{v}_\ell)_i (\mathbf{v}_m)_i = \sum_{i=1}^t y_i^2 \lambda_i^2 \leq \lambda_{\max}(\hat{C}^{-1}) \sum_{i=1}^t y_i^2 \lambda_i \,, \end{aligned} \tag{5.6}$$

where the last equation follows from the orthonormality of the eigenvectors $\mathbf{v}_i$. Moreover, by Lemma 5.2.4

$$R^2_{Z,\mathcal{B}} - R^2_{Z,\mathcal{A}} = R^2_{Z,\hat{\mathcal{T}}} = \hat{\mathbf{b}}^T \hat{C}^{-1} \hat{\mathbf{b}} = \sum_{i=1}^t y_i^2 \lambda_i \,. \tag{5.7}$$

Combining (5.7) and (5.6), we have $\sum_{i\in\mathcal{S}} R^2_{Z,\mathcal{B}} - R^2_{Z,\mathcal{B}\setminus\{X_i\}} \leq \lambda_{\max}(\hat{C}^{-1})[R^2_{Z,\mathcal{B}} - R^2_{Z,\mathcal{A}}]$ and so inequality (5.1) is satisfied for $\beta = \lambda_{\max}(\hat{C}^{-1}) = 1/\lambda_{\min}(\hat{C})$. It remains to bound $1/\lambda_{\min}(\hat{C})$ in terms of the eigenvalues of $C_{\mathcal{X}}$. Recall that $\hat{C}$ is a normalized covariance matrix for the random variables $\{\mathrm{Res}(X_i, \mathcal{B}\setminus\mathcal{A})\}_{X_i\in\mathcal{A}}$. As shown in [DK18], the normalization and taking the residual of increases the minimum eigenvalue.

Lemma 5.2.7 ([DK18]). *Let $\mathcal{L}$ and $\mathcal{S} = \{X_1, X_2, \ldots, X_n\}$ be two disjoint sets of zero-mean random variables each of which has variance at most 1. Let C be the covariance matrix of the set $\mathcal{L}\cup\mathcal{S}$. Let C_ρ be the covariance matrix of the set $\{\mathrm{Res}(X_1, \mathcal{L}), \ldots, \mathrm{Res}(X_n, \mathcal{L})\}$ after normalization of the random variables to have unit variance. Then $\lambda_{\min}(C_\rho) \geq \lambda_{\min}(C)$.*

This implies that $\lambda_{\min}(\hat{C}) \geq \lambda_{\min}(C_{(\mathcal{B}\setminus\mathcal{A})\cup\mathcal{A}}) \geq \lambda_{\min}(C_{\mathcal{X}}, |\mathcal{B}|) \geq \lambda_{\min}(C_{\mathcal{X}})$ for $\mathcal{A}\subseteq\mathcal{B}$. The claimed bound on β then follows. □

5.3 Improved Analysis of ResidualRandomGreedy

In this section, we derive stronger approximation guarantees for maximizing (γ,β)-weakly submodular functions using ResidualRandomGreedy considered in [Buc+14; CFK18]. Combined with Theorem 5.2.1 from Section 5.2, this gives an improved approximation bound for the sparse least square problem subject to a matroid constraint $\mathcal{M}$. ResidualRandomGreedy (shown in Algorithm 7) proceeds over k iterations. In iteration i, it greedily extends the current solution S_{i-1} to a base $S_{i-1}\cup M_i$ of $\mathcal{M}$ by selecting a set M_i of the $k-|S_{i-1}| = k-i+1$ elements with the largest marginal contribution with respect to the S_{i-1}. Then, it chooses an element s_i uniformly at random from M_i which is added to S_{i-1} to obtain a new solution S_i. After k iterations, the final set S_k is returned.

Algorithm 7: The residual random greedy algorithm

procedure ResidualRandomGreedy$(\mathcal{M}, X, f)$
$S_0 \leftarrow \emptyset$;
for $i = 1, 2, \ldots, k$ **do**
  $M_i \leftarrow \arg\max\{\sum_{e\in T} f(e \mid S) : T\subseteq X, S\cup T \text{ is a base of } \mathcal{M}\}$;
  $s_i \leftarrow$ an element of M_i chosen uniformly at random;
  $S_i \leftarrow S_{i-1}\cup\{s_i\}$;
return S_k;

Theorem 5.3.1. *Given a (γ,β)-weakly submodular function,* ResidualRandomGreedy *has approximation factor $1+\frac{\beta}{\gamma}$.*

Applied to the sparse least square regression problem (Section 5.2), Theorem 5.3.1 attains an approximation factor equal to $1+\gamma^{-2}$, where $\gamma \geq \lambda_{\min}(C_{\mathcal{X}}, 2k)$.

Proof of Theorem 5.3.1. We begin by introducing some auxiliary sets used in the analysis. For each $i = 0, 1, \ldots, k$, we let OPT_i to be a subset of OPT of size $k-i$ such that $S_i\cup\mathrm{OPT}_i$ is

a base of $\mathcal{M}$, as follows. Let $\mathrm{OPT}_0 = \mathrm{OPT}$. For each $i \geq 1$, suppose that $S_{i-1} \cup \mathrm{OPT}_{i-1}$ is a base and consider the bijection $\pi_i : S_{i-1} \cup M_i \to S_{i-1} \cup \mathrm{OPT}_{i-1}$ guaranteed by Proposition 1.B.6. We set $\mathrm{OPT}_i = \mathrm{OPT}_{i-1} - \pi_i(s_i)$. Then, $S_i \cup \mathrm{OPT}_i = S_{i-1} \cup \mathrm{OPT}_{i-1} + s_i - \pi_i(s_i)$ is a base, as required. Moreover, note the choice of π_i is independent of the random choice s_i, which implies that $\pi_i(s_i)$ is an element of OPT_{i-1} chosen uniformly at random. Let $\mathcal{E}$ be the event which fixes the random decisions of the algorithm up to iteration $i-1$. Conditioned on $\mathcal{E}$, we have:

$$\begin{aligned}\mathbb{E}[f(S_i) - f(S_{i-1})] &= \frac{1}{|M_i|}\sum_{e \in M_i} f(e \mid S_{i-1}) = \frac{1}{k-i+1}\sum_{e \in M_i} f(e \mid S_{i-1}) \\ &\geq \frac{1}{k-i+1}\sum_{e \in \mathrm{OPT}_{i-1}} f(e \mid S_{i-1}) \geq \frac{\gamma}{k-i+1}(f(\mathrm{OPT}_{i-1} \cup S_{i-1}) - f(S_{i-1}))\,. \quad (5.8)\end{aligned}$$

Here, the third inequality follows the fact that $S_{i-1} \cup \mathrm{OPT}_{i-1}$ is a base and so OPT_{i-1} is a candidate for M_i. The fourth inequality follows from (1.2) since f is (γ, β)-weakly submodular. Similarly, (5.1) together with the fact that $\pi_i(s_i)$ is a uniformly random element of OPT_{i-1} implies

$$\begin{aligned}\frac{1}{k-i+1}(f(\mathrm{OPT}_{i-1} \cup S_{i-1}) - f(S_{i-1})) &\geq \frac{\beta^{-1}}{k-i+1}\sum_{e \in \mathrm{OPT}_{i-1}} f(e|\mathrm{OPT}_{i-1} \cup S_{i-1} - e), \\ &= \beta^{-1} \cdot \mathbb{E}[f(\pi_i(s_i)|\mathrm{OPT}_{i-1} \cup S_{i-1} - \pi_i(s_i))]. \quad (5.9)\end{aligned}$$

We can bound the expected decrease in $f(\mathrm{OPT}_i \cup S_i)$ in iteration i as:

$$\begin{aligned}&\mathbb{E}[f(\mathrm{OPT}_i \cup S_i) - f(\mathrm{OPT}_{i-1} \cup S_{i-1})] \\ &\quad = \mathbb{E}[f(\mathrm{OPT}_{i-1} \cup S_{i-1} + s_i - \pi_i(s_i)) - f(\mathrm{OPT}_{i-1} \cup S_{i-1})] \\ &\quad = \mathbb{E}[f(s_i|\mathrm{OPT}_{i-1} \cup S_{i-1} - \pi_i(s_i)) - f(\pi_i(s_i)|\mathrm{OPT}_{i-1} \cup S_{i-1} - \pi_i(s_i))] \\ &\quad \geq -\mathbb{E}[f(\pi_i(s_i)|\mathrm{OPT}_{i-1} \cup S_{i-1} - \pi_i(s_i))], \quad (5.10)\end{aligned}$$

where the inequality follows by monotonicity of f. Thus

$$\begin{aligned}\mathbb{E}[f(S_i) - f(S_{i-1})] &\geq \frac{\gamma}{\beta}\mathbb{E}[f(\pi_i(s_i)|\mathrm{OPT}_{i-1} \cup S_{i-1} - \pi_i(s_i))] \\ &\geq \frac{\gamma}{\beta}\mathbb{E}[f(\mathrm{OPT}_{i-1} \cup S_{i-1}) - f(\mathrm{OPT}_i \cup S_i)], \quad (5.11)\end{aligned}$$

where the first inequality follows by combining (5.8) and (5.9) and the second by (5.10).

Removing the conditioning on $\mathcal{E}$ and summing the inequalities (5.11) for $i = 1, \cdots, k$, gives $\mathbb{E}[f(S_k) - f(S_0)] \geq \frac{\gamma}{\beta}\mathbb{E}[f(S_0 \cup \mathrm{OPT}_0) - f(S_k \cup \mathrm{OPT}_k)]$. The claim then follows by observing that $S_0 = \emptyset$, $S_k = S$, $\mathrm{OPT}_0 \cup S_0 = \mathrm{OPT}$ and $\mathrm{OPT}_k \cup S_k = S_k$ and so $(1 + \frac{\beta}{\gamma})\,\mathbb{E}[f(S)] \geq f(\mathrm{OPT})$. $\square$

Remark 5.3.2. We remark that the proof of Theorem 5.3.1 only requires a bound on γ, and β for the following restricted set of elements

$$\gamma \triangleq \min_{i=0,\dots,k} \min_{\substack{S,O:\, O\subseteq \mathrm{OPT} \\ |S|=i,|O|=k-i}} \frac{\sum_{e\in O} f(e \mid S)}{f(O\cup S)-f(S)}, \text{ and } \beta \triangleq \max_{i=0,\dots,k} \max_{\substack{S,O:\, O\subseteq \mathrm{OPT} \\ |S|=i,|O|=k-i}} \frac{\sum_{e\in O} f(e|S\cup O\setminus e)}{f(O\cup S)-f(S)},$$

which gives stronger results for the sparse least square regression problem.

5.4 Distorted Local Search

Here, we present an algorithm for maximizing (γ,β)-weakly submodular functions with a guarantee that smoothly approaches the optimal value of $e/(e-1)$ as $\gamma,\beta\to 1$. Algorithm 8 is a local search routine that attempts to swap a single element into the current solution if and only if it improves the following auxiliary potential function parameterized by $\phi\in\mathbb{R}_{\geq 0}$, which we will set appropriately depending on γ and β:

$$g_\phi(A) \triangleq \int_0^1 \frac{\phi e^{\phi p}}{e^\phi-1} \sum_{B\subseteq A} p^{|B|-1}(1-p)^{|A|-|B|} f(B)\,dp = \sum_{B\subseteq A} m^{(\phi)}_{|A|-1,|B|-1} f(B),$$

where we define

$$m^{(\phi)}_{a,b} \triangleq \int_0^1 \phi e^{\phi p} p^b (1-p)^{a-b}/(e^\phi-1)\,dp = \mathop{\mathbb{E}}_{p\sim\mathcal{D}_\phi}[p^a(1-p)^b],$$

where $\mathcal{D}_\phi$ on $[0,1]$ as a continuous distribution on $[0,1]$ with density function $\mathcal{D}_\phi(x)\triangleq \frac{\phi e^{\phi x}}{e^\phi-1}$. For convenience, we will define $h(x)\triangleq \frac{xe^x}{e^x-1}$ and let $m^{(\phi)}_{a,b}=0$ if either $a<0$ or $b<0$.

Algorithm 8: The distorted local-search algorithm for weakly submodular functions

procedure DISTORTEDLOCALSEARCH$(\mathcal{M},X,f)$.

Suppose that f is (γ,β)-weakly submodular and let $\phi=\gamma^2+\beta(1-\gamma)$;
$A\leftarrow$ an arbitrary base of $\mathcal{M}$;
while $\exists a\in S, b\in X\setminus S$ with $S-a+b\in\mathcal{I}$ and $g_\phi(A-a+b)>g_\phi(A)$ **do**
  $A\leftarrow S-a+b$;
return A;

Algorithm 8 stops when there is no improving exchange with respect to g_ϕ. There are several further issues that must be addressed in order to convert Algorithm 8 to a general, polynomial-time algorithm. First, we cannot compute $g_\phi(A)$ directly, as it depends on the values $f(B)$ for all subsets $B\subseteq A$. In Section 5.5.6 we show that we can efficiently estimate g_ϕ via simple sampling procedure. To bound the number of improvements made, we can instead require that each improvement makes a $(1+\varepsilon)$ increase in g_ϕ. Then at termination, we will instead have $\sum_{i=1}^{|A|}[g_\phi(A)-g_\phi(A-a_i+o_i)]\leq |A|\varepsilon g_\phi(A)$. In order to bound the resulting loss

in our guarantee we must bound the value $g_\phi(A)$ in terms of $f(A)$, which we accomplish in Section 5.5.7. Finally, we address the fact that γ and β may not be known and so we cannot set ϕ a priori. We show that by initializing the algorithm with a solution produced by ResidualRandomGreedy, we can bound the range of values for ϕ that must be considered to obtain our guarantee. It then suffices to enumerate guesses for ϕ from this range. In Section 5.5.4 we show that small changes in ϕ result in small changes to $g_\phi(A)$, and so by initializing the run for each subsequent guess of ϕ with the solution produced for the previous guess, we can amortize the total number of improvements (and work) required across all guesses. The final algorithm, presented in Section 5.5, has the same guarantee as Algorithm 8 minus a small $O(\varepsilon)$ term, and requires $\tilde{O}(nk^4\varepsilon^{-3})$ evaluations of f.

Hence, assuming an oracle access to g_ϕ and to the exact value of ϕ, our main result is the following theorem that analyzes the local-gap at the termination of Algorithm 8.

Theorem 5.4.1. *Suppose that f is (γ,β)-weakly submodular and let $\phi = \phi(\gamma,\beta) \triangleq \gamma^2+\beta(1-\gamma)$. Then, for any base O of a matroid $\mathcal{M}$, Algorithm 8 returns a solution A such that, $f(A) \geq \gamma^2 \frac{e^\phi-1}{\phi e^\phi} f(O)$.*

Applied to the sparse least square regression problem (Section 5.2), Theorem 5.4.1 has approximation factor equal to $\frac{\phi e^\phi}{\gamma^2(e^\phi-1)}$ where $\phi = \gamma^2 + \frac{1}{\gamma} - 1$. Thus, as $\gamma \to 1$, we have that $\phi \to 1$ and implies an asymptotic approximation factor equal to $e/(e-1)$. Algorithm 8 can be transformed into a polynomial-time algorithm at the cost of $O(\varepsilon)$ loss in the approximation factor. Adapting Algorithm 8 to obtain a polynomial time runtime, we get the following theorem.

Theorem 5.4.2. *Given a matroid $\mathcal{M} = (X,\mathcal{I})$, and a (γ,β)-weakly submodular function f. For any $\varepsilon \in (0,1)$, there is a randomized algorithm that with probability $1-o(1)$ returns a set S satisfying $f(S) \geq \left(\frac{\gamma^2(1-e^{-\phi})}{\phi} - O(\varepsilon)\right) f(OPT)$, where $\phi \triangleq \phi(\gamma,\beta) = \gamma^2 + \beta(1-\gamma)$. The algorithm runs in time $\tilde{O}(nk^4\varepsilon^{-3})$.*

Analysis of the distorted local-search algorithm

For the purpose of the analysis, we first assume that f is *normalized*, i.e. $f(\emptyset) = 0$. This reduction doesn't affect the proof as we can run the algorithm with respect to the set function $f_2(S) \triangleq f(S) - f(\emptyset)$. Observe that $f_2(\emptyset) = 0$, and that the upper/lower-submodularity ratio of f_2 are equal to that of f. Moreover, an α approximation for f_2 implies an α approximation for f. Indeed, the optimal solution for f and f_2 is the same. Thus, given an α-approximation algorithm, we have $f(\text{OPT}) - f(\emptyset) \leq \alpha(f(S) - f(\emptyset))$ which for $\alpha \geq 1$ implies that $f(\text{OPT}) \leq f(\text{OPT}) - (1-\alpha)f(\emptyset) \leq \alpha f(S)$.

In the analysis of [FW14], it is shown that if f is submodular, its associated potential g is as well, and this plays a crucial role in the analysis. Here, however, f is only *weakly* submodular, which means we must carry out an alternative analysis to bound the quality of a local optimum for g_ϕ. Our analysis will rely on the following properties of the coefficients $m^{(\phi)}_{a,b}$ (see Section 5.4.1 for a full proof of each):

Lemma 5.4.3. *For any $\phi > 0$, the coefficients $m^{(\phi)}_{a,b}$ satisfy the following:*

1. $g_\phi(e \mid A) = \sum_{B \subseteq A} m^{(\phi)}_{|A|,|B|} f(e \mid B)$, *for any $A \subseteq X$ and $e \notin A$.*
2. $\sum_{B \subseteq A} m^{(\phi)}_{|A|,|B|} = 1$, *for all $A \subseteq X$.*
3. $m^{(\phi)}_{a,b} = m^{(\phi)}_{a+1,b+1} + m^{(\phi)}_{a+1,b}$ *for all $0 \leq b \leq a$.*
4. $\phi m^{(\phi)}_{a,b} = -b m^{(\phi)}_{a-1,b-1} + (a-b) m^{(\phi)}_{a-1,b} + (\phi/(e^{\phi-1}))\mathbf{1}_{b=0} + (\phi e^\phi/(e^\phi - 1))\mathbf{1}_{b=a}$, *for all $a > 0$ and $0 \leq b \leq a$.*

In order to analyze the performance of Algorithm 8, we consider two arbitrary bases A and O of the given matroid $\mathcal{M}$. We index the elements $a_i \in A$ and $o_i \in O$ according to the bijection $\pi : A \to O$ guaranteed by Proposition 1.B.6 so that $A - a_i + o_i$ is a base for all $1 \leq i \leq |A|$. To prove Theorem 5.4.1, we first note that $g_\phi(A - a_i + o_i) - g(A) = g(o_i|A - a_i) - g(a_i|A - a_i)$ and so

$$\sum_{i=1}^{|A|} g(a_i|A - a_i) = \sum_{i=1}^{|A|} [g(A) - g(A - a_i + o_i)] + \sum_{i=1}^{|A|} g(o_i|A - a_i). \tag{5.12}$$

In the next lemma, we bound the final term in Equation (5.12). In particular, we translate the sum of the marginal gain of elements $o \in O$ with respect to g into a guarantee with respect to f.

Lemma 5.4.4. *Suppose that f is (γ, β)-weakly submodular, and let $A, O \subseteq X$ with $A = \{a_1, \ldots, a_{|A|}\}$ and $O = \{o_1, \ldots, o_{|A|}\}$ (so $|A| = |O|$). Then,*

$$\sum_{i=1}^{|A|} g(o_i|A - a_i) \geq \gamma^2 f(O) - \left(\gamma^2 + \beta(1-\gamma)\right) \sum_{B \subseteq A} m^{(\phi)}_{|A|,|B|} f(B).$$

Proof of Lemma 5.4.4. By parts 1 and 3 of Lemma 5.4.3, we have

$$g_\phi(o_i|A - a_i) = \sum_{B \subseteq A - a_i} m^{(\phi)}_{|A|-1,|B|} f(o_i|B) = \sum_{B \subseteq A - a_i} [m^{(\phi)}_{|A|,|B|+1} f(o_i|B) + m^{(\phi)}_{|A|,|B|} f(o_i|B)]. \tag{5.13}$$

Since f is γ-weakly submodular from below

$$f(o_i|B) + f(a_i|B) \geq \gamma f(B \cup \{o_i, a_i\}) - \gamma f(B) = \gamma f(o_i|B + a_i) + \gamma f(a_i|B),$$

and so $f(o_i|B) \geq \gamma f(o_i|B + a_i) - (1-\gamma) f(a_i|B)$. Thus, the right-hand side of (5.13) is at least

$$\sum_{B \subseteq A - a_i} m^{(\phi)}_{|A|,|B|+1} \left[\gamma f(o_i|B + a_i) - (1-\gamma) f(a_i|B)\right] + m^{(\phi)}_{|A|,|B|} f(o_i|B) = P + Q, \tag{5.14}$$

where

$$P \triangleq \gamma \sum_{B \subseteq A - a_i} \left[m^{(\phi)}_{|A|,|B|+1} f(o_i|B + a_i) + m^{(\phi)}_{|A|,|B|} f(o_i|B) \right] = \gamma \sum_{B \subseteq A} m^{(\phi)}_{|A|,|B|} f(o_i|B),$$

$$Q \triangleq (1-\gamma) \sum_{B \subseteq A - a_i} \left[m^{(\phi)}_{|A|,|B|} f(o_i|B) - m^{(\phi)}_{|A|,|B|+1} f(a_i|B) \right] \geq -(1-\gamma) \sum_{B \subseteq A - a_i} m^{(\phi)}_{|A|,|B|+1} f(a_i|B).$$

In the first equation, we have used that for each set $T \subseteq A$, $f(o_i|T)$ appears in the right-hand summation exactly once: if $a_i \in T$ it appears as $T = B + a_i$ with coefficient $m^{(\phi)}_{|A|,|B|+1} = m^{(\phi)}_{|A|,|T|}$ and if $a_i \notin T$ it appears as $T = B$ with coefficient $m^{(\phi)}_{|A|,|B|} = m^{(\phi)}_{|A|,|T|}$. The lower bound on Q simply follows from the monotonicity of f. Summing (5.14) over each $a_i \in A$ we then have

$$\sum_{i=1}^{|A|} g_\phi(o_i|A - a_i) \geq \gamma \sum_{i=1}^{|A|} \sum_{B \subseteq A} m^{(\phi)}_{|A|,|B|} f(o_i|B) - (1-\gamma) \sum_{i=1}^{|A|} \sum_{B \subseteq A - a_i} m^{(\phi)}_{|A|,|B|+1} f(a_i|B). \quad (5.15)$$

Since f is γ-weakly submodular from below and monotone,

$$\begin{aligned} \gamma \sum_{i=1}^{|A|} \sum_{B \subseteq A} m^{(\phi)}_{|A|,|B|} f(o_i|B) &= \gamma \sum_{B \subseteq A} \sum_{i=1}^{|A|} m^{(\phi)}_{|A|,|B|} f(o_i|B) \geq \gamma^2 \sum_{B \subseteq A} m^{(\phi)}_{|A|,|B|} [f(O \cup B) - f(B)] \\ &\geq \gamma^2 \sum_{B \subseteq A} m^{(\phi)}_{|A|,|B|} [f(O) - f(B)] = \gamma^2 f(O) - \gamma^2 \sum_{B \subseteq A} m^{(\phi)}_{|A|,|B|} f(B), \end{aligned}$$

where the last equation follows from part 2 of Lemma 5.4.3. Similarly, since f is β-weakly submodular from above:

$$\begin{aligned} (1-\gamma) \sum_{i=1}^{|A|} \sum_{B \subseteq A - a_i} m^{(\phi)}_{|A|,|B|+1} (f(B + a_i) - f(B)) &= (1-\gamma) \sum_{T \subseteq A} \sum_{i=1}^{|A|} m^{(\phi)}_{|A|,|T|} (f(T) - f(T - a_i)) \\ &\leq \beta(1-\gamma) \sum_{T \subseteq A} m^{(\phi)}_{|A|,|T|} [f(T) - f(\emptyset)] = \beta(1-\gamma) \sum_{B \subseteq A} m^{(\phi)}_{|A|,|B|} f(B), \end{aligned}$$

where the first equation can be verified by substituting $B = T - a_i$ for each $a_i \in T$ and noting that $|T| = |B| + 1$, and the last equation simply follows from $f(\emptyset) = 0$ and renaming T to B. Using the two previous inequalities to bound the right-hand side of (5.15), then gives the claimed result. □

Proof of Theorem 5.4.1. Applying Lemma 5.4.4 to the last term in (5.12) and rearranging gives:

$$\sum_{i=1}^{|A|} g_\phi(a_i|A - a_i) + \left(\gamma^2 + \beta(1-\gamma)\right) \sum_{B \subseteq A} m^{(\phi)}_{|A|,|B|} f(B) \geq \gamma^2 f(O) + \sum_{i=1}^{|A|} [g_\phi(A) - g_\phi(A - a_i + o_i)]. \quad (5.16)$$

From part 1 of Lemma 5.4.3,

$$\sum_{i=1}^{|A|} g_\phi(a_i|A-a_i) = \sum_{i=1}^{|A|} \sum_{B\subseteq A-a_i} m^{(\phi)}_{|A|-1,|B|}(f(B+a_i)-f(B))$$
$$= \sum_{T\subseteq A} |T| m^{(\phi)}_{|A|-1,|T|-1} f(T) - (|A|-|T|) m^{(\phi)}_{|A|-1,|T|} f(T),$$

where the last equation follows from the fact that each $T \subseteq A$ appears once as $T = B + a_i$ for each $a_i \in T$ (in which case it has coefficient $m^{(\phi)}_{|A|-1,|B|} = m^{(\phi)}_{|A|-1,|T|-1}$) and once as $T = B$ for each $a_i \notin T$ (in which case it has coefficient $m^{(\phi)}_{|A|-1,|B|} = m^{(\phi)}_{|A|-1,|T|}$). Thus, we can rewrite (5.16) as:

$$\sum_{B\subseteq A} \left(|B| m^{(\phi)}_{|A|-1,|B|-1} - (|A|-|B|) m^{(\phi)}_{|A|-1,|B|} + \left(\gamma^2 + \beta(1-\gamma)\right) m^{(\phi)}_{|A|,|B|} \right) f(B)$$
$$\geq \gamma^2 f(O) + \sum_{i=1}^{|A|} [g_\phi(A) - g_\phi(A - a_i + o_i)]. \quad (5.17)$$

At the termination of Algorithm 8, each square bracketed term is positive. Thus, it is sufficient to simplify the left-hand term to conclude the proof. Since $\phi = \gamma^2 + \beta(1-\gamma)$, the recurrence in part 4 of Lemma 5.4.3 implies that the left-hand side vanishes for all B except $B = \emptyset$, in which case it is $\frac{\phi}{e^\phi - 1} f(\emptyset) = 0$ or $B = A$, in which case it is $\frac{\phi e^\phi}{e^\phi - 1} f(A)$. The theorem then follows. □

5.4.1 Properties of the coefficients $m^{(\phi)}_{a,b}$

The analysis of Theorem 5.4.2 crucially rely on the properties of the coefficients $m^{(\phi)}_{a,b}$ from Lemma 5.4.3. For convenience, we restate them here and give a proof of each claim in turn.

Lemma 5.4.3. *For any $\phi > 0$, the coefficients $m^{(\phi)}_{a,b}$ satisfy the following:*

1. $g_\phi(e \mid A) = \sum_{B\subseteq A} m^{(\phi)}_{|A|,|B|} f(e \mid B)$, *for any $A \subseteq X$ and $e \notin A$.*
2. $\sum_{B\subseteq A} m^{(\phi)}_{|A|,|B|} = 1$, *for all $A \subseteq X$.*
3. $m^{(\phi)}_{a,b} = m^{(\phi)}_{a+1,b+1} + m^{(\phi)}_{a+1,b}$ *for all $0 \leq b \leq a$.*
4. $\phi m^{(\phi)}_{a,b} = -b m^{(\phi)}_{a-1,b-1} + (a-b) m^{(\phi)}_{a-1,b} + (\phi/(e^{\phi-1})) \mathbf{1}_{b=0} + (\phi e^\phi/(e^\phi - 1)) \mathbf{1}_{b=a}$, *for all $a > 0$ and $0 \leq b \leq a$.*

Proof of Claim 1. Note that by the definition of g_ϕ:

$$\begin{aligned} g_\phi(e \mid A) &= \sum_{B \subseteq A+e} m^{(\phi)}_{|A|,|B|-1} f(B) - \sum_{B \subseteq A} m^{(\phi)}_{|A|-1,|B|-1} f(B) \\ &= \sum_{B \subseteq A} \left[(m^{(\phi)}_{|A|,|B|-1} - m^{(\phi)}_{|A|-1,|B|-1}) f(B) + m^{(\phi)}_{|A|,|B|} f(B+e) \right]. \end{aligned}$$

It thus suffices to show $(m^{(\phi)}_{|A|,|B|-1} - m^{(\phi)}_{|A|-1,|B|-1}) f(B) = -m^{(\phi)}_{|A|,|B|} f(B)$. For $B = \emptyset$, we have $f(\emptyset) = 0$ and so $(m^{(\phi)}_{|A|,-1} - m^{(\phi)}_{|A|-1,-1}) f(\emptyset) = 0 = -m^{(\phi)}_{|A|,0} f(\emptyset)$. When $|B| \geq 1$,

$$\begin{aligned} m^{(\phi)}_{|A|,|B|-1} - m^{(\phi)}_{|A|-1,|B|-1} &= \mathop{\mathbb{E}}_{p \sim \mathcal{D}_\phi} \left[p^{|B|-1}(1-p)^{|A|-|B|+1} - p^{|B|-1}(1-p)^{|A|-|B|} \right] \\ &= \mathop{\mathbb{E}}_{p \sim \mathcal{D}_\phi} \left[-p^{|B|}(1-p)^{|A|-|B|} \right] = -m^{(\phi)}_{|A|,|B|}. \quad \square \end{aligned}$$

Proof of Claim 2. By linearity of expectation:

$$\sum_{B \subseteq A} m^{(\phi)}_{|A|,|B|} = \sum_{b=0}^{|A|} \binom{|A|}{b} \mathop{\mathbb{E}}_{p \sim \mathcal{D}_\phi} \left[p^b (1-p)^{|A|-b} \right] = \mathop{\mathbb{E}}_{p \sim \mathcal{D}_\phi} \left[\sum_{b=0}^{|A|} \binom{|A|}{b} p^b (1-p)^{|A|-b} \right] = 1. \quad \square$$

Proof of Claim 3. When $0 \leq b \leq a$, the definition of $m^{(\phi)}_{a,b}$ immediately gives:

$$m^{(\phi)}_{a,b} = \mathop{\mathbb{E}}_{p \sim \mathcal{D}_\phi} \left[p^b (1-p)^{a-b} \right] = \mathop{\mathbb{E}}_{p \sim \mathcal{D}_\phi} \left[p^b (1-p)^{a-b} p + p^b (1-p)^{a-b} (1-p) \right] = m^{(\phi)}_{a+1,b+1} + m^{(\phi)}_{a+1,b}.$$

$\square$

Proof of Claim 4. For $a > 0$ and $b \leq a$, noting that $\mathcal{D}_\phi(p) = \frac{d}{dp} \frac{\mathcal{D}_\phi(p)}{\phi}$ and applying integration by parts

$$\begin{aligned} m^{(\phi)}_{a,b} &= \int_0^1 \mathcal{D}_\phi(p) \cdot p^b (1-p)^{a-b} \, dp \\ &= \frac{\mathcal{D}_\phi(p)}{\phi} p^b (1-p)^{a-b} \Big|_{p=0}^{p=1} - \int_0^1 \frac{\mathcal{D}_\phi(p)}{\phi} \left(b p^{b-1} (1-p)^{a-b} - (a-b) p^b (1-p)^{a-b-1} \right) dp. \end{aligned}$$

Which is equivalent to:

$$\phi m^{(\phi)}_{a,b} = -b m^{(\phi)}_{a-1,b-1} + (a-b) m^{(\phi)}_{a-1,b} + \mathcal{D}_\phi(p) p^b (1-p)^{a-b} \Big|_{p=0}^{p=1}.$$

This follows immediately from the definition of $m^{(\phi)}_{a,b}$ when $b > 0$, and when $b = 0$ it follows from $-b m^{(\phi)}_{a-1,b-1} = 0 = b p^{b-1} (1-p)^{a-b}$.

To complete the claim, we note that $\lim_{p \to 0^+} \mathcal{D}_\phi(p) p^b (1-p)^{a-b}$ is $\mathcal{D}_\phi(0) = \phi/(e^\phi - 1)$ if $b = 0$ and 0 if $b > 0$, and $\lim_{p \to 1^-} \mathcal{D}_\phi(p) p^b (1-p)^{a-b}$ is $\mathcal{D}_\phi(1) = \phi e^\phi/(e^\phi - 1)$ if $a = b$, and 0 if $0 \leq b < a$. $\square$

5.5 A randomized, polynomial time distorted local-search algorithm

In this section, we will give our final algorithm is shown in Algorithm 9. Before presenting it in detail, we describe the main concerns involved in its formulation.

5.5.1 Initialization

We initialize the algorithm with a solution S_0 by using the guarantee for ResidualRandomGreedy provided by [CFK18] when only γ is bounded. In this case, their analysis shows that the *expected* value of the solution produced by the algorithm is at least $\frac{1}{(1+\gamma^{-1})^2} f(O)$, where O is an optimal solution to the problem. Here, however, we will require a guarantee that holds with high probability. This is easily ensured by independently running ResidualRandomGreedy a sufficient number of times and taking the best solution found.

Formally, suppose we set $\varepsilon' = \min(\varepsilon, \frac{1}{128})$ and run G times ResidualRandomGreedy independently where $G = \frac{2\log(n)}{\varepsilon'^2} = \tilde{O}(\varepsilon^{-2})$. For each $1 \leq l \leq G$, let T_l be the solution produced by the l^{th} instance of the ResidualRandomGreedy. Define the random variables $Z_l = \frac{1}{(1+\gamma^{-1})^2} - \frac{f(T_l)}{f(O)}$, where $O \in \mathcal{I}$ is the optimal solution. Then, $\mathbb{E}[Z_l] = 0$ and $|Z_l| \leq 1$ for all l. Using straightforward concentration arguments we obtain a result that holds with high probability. Let $S_0 = \arg\max_{1 \leq l \leq G} f(T_l)$. Then, by the Chernoff bound (Lemma 1.B.5),

$$\begin{aligned}\Pr\left[f(S_0) < \left((1+\gamma^{-1})^{-2} - \varepsilon'\right) f(O)\right] &\leq \Pr\left[\tfrac{1}{G}\textstyle\sum_{l=1}^{G} f(T_l) < \left((1+\gamma^{-1})^{-2} - \varepsilon'\right) f(O)\right] \\ &= \Pr\left[\textstyle\sum_{l=1}^{G} Z_l > G\varepsilon'\right] < e^{-\frac{\varepsilon'^2 G}{2}} \leq \tfrac{1}{n}.\end{aligned}$$

Thus, with probability at least $1 - \frac{1}{n} = 1 - o(1)$, $f(S_0) \geq \left(\frac{1}{(1+\gamma^{-1})^2} - \varepsilon'\right) f(O)$.

5.5.2 In-depth discussion of the proof strategy

In Theorem 5.4.1, we considered a (γ, β)-weakly submodular function f, and used the potential g_ϕ with $\phi = \phi(\gamma, \beta) = \gamma^2 + \beta(1-\gamma)$ to guide the search. In general, however, the values of γ and β may not be known in advance. One approach to coping with this would be to make an appropriate series of guesses for each of the values, then run our the algorithm for each guess and return the best solution obtained.

Here we describe an alternative and more efficient approach: we guess the value of $\phi(\gamma, \beta)$ directly from an appropriate geometrically decreasing sequence of values for ϕ. Moreover, when running the algorithm for each subsequent guess, we initialize the local search procedure using the solution produced by the algorithm for the previous guess. Lemma 5.5.1 displays the change in the solution value by substituting ϕ by $\phi(1-\varepsilon)$.

Lemma 5.5.1. *For all ϕ, $\varepsilon \in (0,1)$, and $S \subseteq X$,*

1. $g_{\phi(1-\varepsilon)}(S) \geq e^{-\phi\varepsilon} g_\phi(S)$
2. $h(\phi) \leq e^{\phi\varepsilon} h(\phi(1-\varepsilon))$, *where we recall that* $h(x) \triangleq \frac{xe^x}{e^x-1}$.

Lemma 5.5.1 will allow us to amortize the number of improvements made by the algorithm across all guesses. The appropriate range for the parameter ϕ is determined by the next lemma. It shows that if γ or $\phi(\gamma, \beta)$ is very small, then the guarantee for ResidualRandomGreedy is stronger than that required by our analysis (and so S_0 is already a good solution).

Lemma 5.5.2. *For all* $\gamma \in (0, 1]$ *and* $\beta \geq 1$, $\phi(\gamma, \beta) \geq \frac{3}{4}$. *Moreover, if* $\phi(\gamma, \beta) > 4$ *or* $\gamma < \frac{1}{7}$, *then* $\frac{1}{(1+\gamma^{-1})^2} > \frac{\gamma^2(1-e^{-\phi(\gamma,\beta)})}{\phi(\gamma,\beta)}$.

Lemma 5.5.2 shows that it suffices to consider $\phi(\gamma, \beta) \in [3/4, 4]$ and $\gamma > 1/7$, since otherwise the starting solution already satisfies the claimed guarantee.[2] Thus, our algorithm considers a geometrically decreasing sequence of guesses for the value $\phi \in [3/4, 4]$, given by $\phi_j = 4(1-\varepsilon)^j$, where $0 \leq j \leq \lceil \log_{1-\varepsilon} \frac{3}{16} \rceil$. For the first guess, we initialize our algorithm with the solution S_0 produced using several runs of ResidualRandomGreedy. For each subsequent guess, we initialize S with the approximately locally optimal solution produced for the previous guess.

For each guess, the algorithm proceeds by repeatedly searching for single element swaps that significantly improve the potential $g_\phi(S)$. Specifically, we will exchange an element $a \notin S$ with an element $b \in S$ whenever $\tilde{g}_{\phi_j}(a|S-b) > \tilde{g}_{\phi_j}(b|S-b) + \Delta f(S)$, where $\tilde{g}_{\phi_j}(\cdot|S-b)$ is an estimate of $g_{\phi_j}(\cdot|S-b)$ computed using N samples as described in Section 5.5.6 and Δ is an appropriately chosen parameter.

Lemma 5.5.3. *For any* ϕ, N, *there is a randomized procedure for obtaining an estimate* $\tilde{g}(e|A)$ *of* $g_\phi(e|A)$ *using* N *queries to the value oracle for* f *so that for any* $\delta > 0$,

$$\Pr[\, |g(e|A) - \tilde{g}(e|A)| \geq \delta f(A+e) \,] < 2e^{-\frac{\delta^2 N}{2}},$$

We show that by setting N appropriately, we can ensure that with high probability an approximate local optimum of every g_ϕ is reached after at most some total number M of improvements across all guesses. To successfully prove Lemma 5.5.3, we require the boundedness of g. The final hurdle is to prove the boundedness of g necessary to bound the maximum number of improvements that Algorithm 9 can make. Since, the function g_ϕ might not be weakly submodular the proof of Lemma 5.5.4 is slightly technical and does not directly follow from [FW14].

Lemma 5.5.4. *If* f *is* γ*-weakly submodular, then for all* $A \subseteq X$, $\gamma f(A) \leq g_\phi(A) \leq h(\phi) H_{|A|} f(A)$.

[2] We remark that the use of the ResidualRandomGreedy is not strictly necessary for our results. One can instead initialize the algorithm with a base containing the best singleton as in the standard local search procedure to obtain a guarantee of γ/k for the initial solution. The remaining arguments can then be modified at the cost of a larger running time dependence on the parameter k.

Algorithm 9: Distorted Local Search Implementation

Let $\Delta = \frac{\varepsilon}{k}$, $\delta = \frac{\Delta}{4h(4)\cdot H_k} = \frac{\varepsilon}{4h(4)\cdot H_k k}$, $M = (1+\delta^{-1})(37+\ln(H_k))$, $N = 28\delta^{-2}\ln(Mkn)$, $G = \log(n)/(2\min(\varepsilon, \frac{1}{128})^2)$;
$S_0 \leftarrow$ the best output produced by G independent runs of ResidualRandomGreedy applied to f and $\mathcal{M}$;
$S_{\max} \leftarrow S_0$;
$i \leftarrow 0$;
for $0 \le j \le \lceil \log_{1-\varepsilon} 16/3 \rceil$ **do**
  $\phi \leftarrow 4(1-\varepsilon)^j$;
  $S \leftarrow S_j$;
  repeat
    isLocalOpt $\leftarrow$ **true**;
    foreach $b \in S$ and $a \in X \setminus S$ with $S - b + a \in \mathcal{I}$ **do**
      Compute $\tilde{g}_{\phi_j}(a|S-b)$ and $\tilde{g}_{\phi_j}(b|S-b)$ using N random samples;
      if $\tilde{g}_{\phi_j}(a|S-b) > \tilde{g}_{\phi_j}(b|S-b) + \Delta f(S)$ **then**
        $S \leftarrow S - b + a$;
        $i \leftarrow i + 1$;
        isLocalOpt $\leftarrow$ **false**;
        break
  until isLocalOpt or $i \ge M$;
  $S_{j+1} \leftarrow S$;
  if $f(S_{j+1}) > f(S_{\max})$ **then** $S_{\max} \leftarrow S_{j+1}$;
return $S_{\max}$

5.5.3 The algorithm and its analysis

Our final algorithm is shown in Algorithm 9. Let $\mathcal{M} = (X, \mathcal{I})$ be a matroid, and $f : 2^X \to \mathbb{R}_{\ge 0}$ be a (γ, β)-weakly submodular function. Given some $0 < \varepsilon \le 1$ we set the parameters:

$$\Delta = \frac{\varepsilon}{k} \qquad \text{(threshold for accepting improvements)}$$
$$\delta = \frac{\Delta}{4h(4)\cdot H_k} = \frac{\varepsilon}{4h(4)\cdot H_k k} = \Theta(\varepsilon/(k\log k)) \qquad \text{(bound on sampling accuracy)}$$
$$L = 1 + \lceil \log_{1-\varepsilon} \tfrac{3}{16} \rceil = \mathcal{O}(\varepsilon^{-1}) \qquad \text{(number of guesses for } \phi\text{)}$$
$$M = \log_{1+\delta}(7 \cdot 128 \cdot e^{4L\varepsilon} h(4) \cdot H_k) = \tilde{\mathcal{O}}(\delta^{-1}) = \tilde{\mathcal{O}}(k\varepsilon^{-1}) \qquad \text{(total number of improvements)}$$
$$N = 4 \cdot 7^2 \delta^{-2} \ln(Mkn) = \tilde{\mathcal{O}}(\delta^{-2}) = \tilde{\mathcal{O}}(k^2\varepsilon^{-2}) \qquad \text{(number of samples to estimate } g_\phi\text{)}$$

Equipped with lemmas 5.5.1, 5.5.2, 5.5.3, and 5.5.4 we are ready to derive guarantees for Algorithm 9. In Algorithm 9, we evaluate potential improvements using an estimate $\tilde{g}_{\phi_j}(\cdot|S-b)$ for the marginals of g that is computed using N samples. By Lemma 5.5.3, we then have $|\tilde{g}_{\phi_j}(e|A) - g_{\phi_j}(e|A)| \le \gamma\delta f(A+e)$ for any A, e considered by the algorithm with probability at least $1 - 2e^{-\frac{\delta^2\gamma^2 N}{2}}$. If $\gamma \ge 1/7$, this is at least $1 - 2e^{-\frac{\delta^2}{2\cdot 7^2}N} = 1 - \frac{2}{(Mkn)^2}$. In our algorithm we will limit the total number of improvements made across all guesses for ϕ to be at most M. Note that any improvement can be found by testing at most kn marginal values,

so we must estimate at most Mkn marginal values across the algorithm. By a union bound, we then have $|\tilde{g}_{\phi_j}(e|A) - g_{\phi_j}(e|A)| \leq \gamma\delta f(A+e)$ for *all* A, e considered by Algorithm 9 with probability at least $1 - o(1)$ whenever $\gamma \geq 1/7$. Before proving our main result, let us show that if the algorithm terminates and returns S after making M improvements, we must in fact have an *optimal* solution with high probability.

Lemma 5.5.5. *Suppose that $\gamma \geq 1/7$. Then, if Algorithm 9 makes M improvements, the set S it returns satisfies $f(S) \geq f(O)$ with probability $1 - o(1)$.*

Proof of Lemma 5.5.5. With probability $1 - o(1)$ we have $\left|\tilde{g}_{\phi_j}(e|A) - g_{\phi_j}(e|A)\right| \leq \gamma\delta f(A+e)$ for any e, A considered by Algorithm 9. Whenever the algorithm exchanges some $a \in X \setminus S$ for $b \in S$ for some guess ϕ_j, we have $\tilde{g}_{\phi_j}(a|S-b) - \tilde{g}_{\phi_j}(b|S-b) \geq \Delta f(S)$ and so

$$\begin{aligned} g_{\phi_j}(S-b+a) - g_{\phi_j}(S) &= g_{\phi_j}(a|S-b) - g_{\phi_j}(b|S-b) \\ &\geq \tilde{g}_{\phi_j}(a|S-b) - \delta\gamma f(S-b+a) - \tilde{g}_{\phi_j}(b|S-b) - \delta\gamma f(S) \\ &\geq \tilde{g}_{\phi_j}(a|S-b) - \delta g_{\phi_j}(S-b+a) - \tilde{g}_{\phi_j}(b|S-b) - \delta g_{\phi_j}(S) \\ &\geq \Delta f(S) - \delta g_{\phi_j}(S-b+a) - \delta g_{\phi_j}(S), \end{aligned}$$

where the second inequality follows from the lower bound on g_{ϕ_j} in Lemma 5.5.4. Rearranging and using the upper bound on $g_{\phi_j}(S)$ from Lemma 5.5.4, together with the definition of δ and Δ, we obtain:

$$\begin{aligned} g_{\phi_j}(S-b+a) \geq \frac{\Delta f(S) + (1-\delta)g_{\phi_j}(S)}{1+\delta} &\geq \frac{\frac{\varepsilon}{k}\frac{1}{h(\phi_j)\cdot H_k} + 1 - \delta}{1+\delta} g_{\phi_j}(S) \\ &\geq \frac{\frac{\varepsilon}{k}\frac{1}{h(4)\cdot H_k} + 1 - \delta}{1+\delta} g_{\phi_j}(S) = \frac{1+3\delta}{1+\delta} g_{\phi_j}(S) \geq (1+\delta) g_{\phi_j}(S), \end{aligned} \tag{5.18}$$

where the last inequality follows from $\frac{1+3x}{1+x} \geq \frac{(1+x)^2}{1+x}$ for all $0 \leq x \leq 1$.

Now suppose that $f(S_0) \geq ((1+\gamma^{-1})^{-2} - \varepsilon')f(O)$, which we have shown also occurs with high probability $1 - o(1)$. Then, since $\gamma \geq \frac{1}{7}$ and $\varepsilon' = \min(\frac{1}{128}, \varepsilon)$, we have $f(S_0) \geq \frac{1}{128} f(O)$. Suppose that $i = M$ when the algorithm is considering some guess ϕ_l. We consider how the current value of $g_{\phi_j}(S)$ changes throughout Algorithm 9, both as improvements are made and as j increases. As shown in (5.18), each of our M improvements increases this value by a factor of $(1+\delta)$. Moreover, as shown in Lemma 5.5.1,

$$g_{\phi_j}(S) = g_{(1-\varepsilon)\phi_{j-1}}(S) \geq e^{-\phi_{j-1}\varepsilon} g_{\phi_{j-1}}(S) \geq e^{-4\varepsilon} g_{\phi_{j-1}}(S)\,,$$

for any set S. Thus, each time j is incremented, the value $g_{\phi_j}(S)$ decreases by a factor of at most $e^{4\varepsilon}$. Since we made M improvements, we then have:

$$g_{\phi_l}(S_{\ell+1}) \geq (1+\delta)^M e^{-4l\varepsilon} g_{\phi_0}(S_0) \geq (1+\delta)^M e^{-4l\varepsilon} \gamma f(S_0) \geq (1+\delta)^M e^{-4l\varepsilon} \tfrac{1}{7}\tfrac{1}{128} f(O)\,,$$

where the second inequality follows from the lower bound on g given in Lemma 5.5.4, and the second from $\gamma \geq \frac{1}{7}$. The upper bound on g given by Lemma 5.5.4 implies that: $g_{\phi_l}(S_{l+1}) \leq h(\phi_l) H_k f(S_{l+1}) \leq h(4) H_k f(S_{l+1})$. Thus,

$$f(S_{l+1}) \geq (1+\delta)^M e^{-4l\varepsilon} \frac{1}{7 \cdot 128 \cdot h(4) \cdot H_k} f(O)\,.$$

Since $l \leq L$ and $M = \log_{1+\delta}(e^{4L\varepsilon} 7 \cdot 128 \cdot h(4) \cdot H_k)$, the set $S_{\max}$ returned by the algorithm thus has $f(S_{\max}) \geq f(S_{l+1}) \geq f(O)$, as claimed. □

We are now ready to prove our main claim, from Section 5.4, restated here for convenience:

Theorem 5.4.2. *Given a matroid $\mathcal{M} = (X, \mathcal{I})$, and a (γ, β)-weakly submodular function f. For any $\varepsilon \in (0,1)$, there is a randomized algorithm that with probability $1 - o(1)$ returns a set S satisfying $f(S) \geq \left(\frac{\gamma^2(1-e^{-\phi})}{\phi} - O(\varepsilon)\right) f(OPT)$, where $\phi \triangleq \phi(\gamma,\beta) = \gamma^2 + \beta(1-\gamma)$. The algorithm runs in time $\tilde{O}(nk^4\varepsilon^{-3})$.*

Proof of Theorem 5.4.2. We have shown that $f(S_0) \geq \left((1+\gamma^{-1})^{-2} - \varepsilon'\right) f(O)$ with probability $1 - o(1)$, where $\varepsilon' = \min(\varepsilon, \frac{1}{128})$. If $\gamma < 1/7$ or $\phi(\gamma,\beta) \notin [3/4, 4]$, then Lemma 5.5.2 implies that $(1+\gamma^{-1})^{-2} > \frac{\gamma^2(1-e^{-\phi(\gamma,\beta)})}{\phi(\gamma,\beta)}$, and so the claim follows as $f(S_{\max}) \geq f(S_0)$. Thus, we suppose that $\gamma \geq 1/7$ and $\phi(\gamma,\beta) \in [3/4, 4]$. Then, if Algorithm 9 makes M improvements, Lemma 5.5.5 implies that the set returned by the algorithm is optimal with probability at least $1 - o(1)$.

In the remaining case, we have $\phi(\gamma,\beta) \in [3/4, 4]$, $\gamma \geq 1/7$. Each set S_{j+1} produced by the algorithm must have $\tilde{g}_{\phi_j}(o_l \mid S_{j+1} - s_l) \leq \tilde{g}_{\tilde{\phi}_j}(s_l \mid S_{j+1} - s_l) + \Delta f(S)$ for every $s_l \in S$ and $o_l \in O$ when Algorithm 9 terminates without making M improvements. Since $\gamma \geq 1/7$ and the algorithm makes at most M improvements, with probability $1 - o(1)$, we have $\left|\tilde{g}_{\phi_j}(e|A) - g_{\phi_j}(e|A)\right| \leq \gamma\delta f(A+e)$ for all guesses ϕ_j and e, A considered by the algorithm. Thus,

$$\begin{aligned}
&g_{\phi_j}(S_{j+1} - s_l + o_l) - g_{\phi_j}(S_{j+1}) \\
&\quad = g_{\phi_j}(o_l \mid S_{j+1} - s_l) - g_{\phi_j}(s_l \mid S_{j+1} - s_l) \\
&\quad \leq \tilde{g}_{\phi_j}(o_l \mid S_{j+1} - s_l) + \delta\gamma f(S_{j+1} - s_l + o_l) - \tilde{g}_{\phi_j}(s_l \mid S_{j+1} - s_l) + \delta\gamma f(S_{j+1}) \\
&\quad \leq \Delta f(S_{j+1}) + \delta\gamma f(S_{j+1}) + \delta\gamma f(S_{j+1} - s_l + o_l) \\
&\quad \leq (\Delta + 2\delta) f(\mathrm{OPT}) \\
&\quad = O\left(\tfrac{\varepsilon}{k}\right) \cdot f(\mathrm{OPT}).
\end{aligned}$$

Consider the smallest j such that $\phi_{j+1} \triangleq 4(1-\varepsilon)^{j+1} < \phi(\gamma,\beta)$. Then, $\phi_{j+1} < \phi(\gamma,\beta) \leq \phi_{j+1}/(1-\varepsilon) \triangleq \phi_j$. Let $\tilde{\beta} = \frac{\phi_j - \gamma^2}{1-\gamma}$. Then, $\phi(\gamma,\tilde{\beta}) = \gamma^2 + \frac{\phi_j-\gamma^2}{1-\gamma}(1-\gamma) = \phi_j$ and $\tilde{\beta} \geq \frac{\phi(\gamma,\beta)-\gamma^2}{1-\gamma} = \beta$, so f is also $(\gamma,\tilde{\beta})$-weakly submodular. Theorem 5.4.1 then implies

$$f(S_{j+1}) \geq \frac{\gamma^2}{h(\phi_j)} f(\text{OPT}) + \sum_{i=1}^{k} \left[g_{\phi_j}(S) - g_{\phi_j}(S - s_l + o_l) \right] \geq \left(\frac{\gamma^2}{h(\phi_j)} - \mathcal{O}(\varepsilon) \right) f(\text{OPT})$$

By Lemma 5.5.1 part 2, our choice of j, and $\phi_j \leq 4$,

$$h(\phi_j) \leq e^{\phi_j \varepsilon} h((1-\varepsilon)\phi_j) \leq e^{\phi_j \varepsilon} h(\phi(\gamma,\beta)) \leq e^{4\varepsilon} h(\phi(\gamma,\beta))$$

Thus, $f(S_{j+1}) \geq \left(\frac{\gamma^2}{h(\phi(\gamma,\beta))} - \mathcal{O}(\varepsilon) \right) f(\text{OPT}) = \left(\frac{\gamma^2(1-e^{-\phi(\gamma,\beta)})}{\phi(\gamma,\beta)} - \mathcal{O}(\varepsilon) \right) f(\text{OPT})$.

The running time of the algorithm is dominated by the number of value oracle queries made to f. The initialization requires running ResidualRandomGreedy $\tilde{\mathcal{O}}(\varepsilon^{-2})$ times, each of which requires $\mathcal{O}(nk)$ value queries. The remaining execution makes at most $M = \tilde{\mathcal{O}}(\varepsilon^{-1}k)$ local search improvements, each requiring at most $Nnk = \tilde{\mathcal{O}}(nk^3\varepsilon^{-2})$ value queries to find. Altogether the running time is thus at most $\tilde{\mathcal{O}}(nk^4\varepsilon^{-3})$. □

5.5.4 Warm starting the search using the previous solution

To use the solution of the previous guess of ϕ as the starting solution of the next guess, we must bound the sensitivity of g_ϕ. The following lemma shows that small changes in the parameter ϕ produce relatively small changes in the value $g_\phi(A)$ for any set A.

Lemma 5.5.1. *For all ϕ, $\varepsilon \in (0,1)$, and $S \subseteq X$,*

1. $g_{\phi(1-\varepsilon)}(S) \geq e^{-\phi\varepsilon} g_\phi(S)$
2. $h(\phi) \leq e^{\phi\varepsilon} h(\phi(1-\varepsilon))$, *where we recall that* $h(x) \triangleq \frac{xe^x}{e^x-1}$.

Proof of Lemma 5.5.1. Both claims will follow from the inequality

$$\frac{\phi(1-\varepsilon)e^{\phi(1-\varepsilon)p}}{e^{\phi(1-\varepsilon)}-1} \geq e^{-\phi\varepsilon} \frac{\phi e^{\phi p}}{e^\phi - 1}, \tag{5.19}$$

which we show is valid for all $p \in [0,1]$ and $\varepsilon > 0$. Indeed, under these assumptions,

$$\begin{aligned}
\frac{\phi(1-\varepsilon)e^{\phi(1-\varepsilon)p}}{e^{\phi(1-\varepsilon)}-1} \cdot \frac{e^\phi - 1}{\phi e^{\phi p}} &= (1-\varepsilon)e^{-\phi\varepsilon p} \frac{e^\phi - 1}{e^\phi e^{-\phi\varepsilon} - 1} = (1-\varepsilon)e^{-\phi\varepsilon p} \frac{e^\phi - 1}{e^\phi(1+(e^{-\phi}-1))^\varepsilon - 1} \\
&\geq (1-\varepsilon)e^{-\phi\varepsilon p} \frac{e^\phi - 1}{e^\phi(1+\varepsilon(e^{-\phi}-1)) - 1} = (1-\varepsilon)e^{-\phi\varepsilon p} \frac{e^\phi - 1}{(1-\varepsilon)(e^\phi - 1)} = e^{-\phi\varepsilon p} \geq e^{-\phi\varepsilon}.
\end{aligned}$$

Here the first inequality follows from the generalized Bernoulli inequality $(1+x)^t \leq (1+tx)$, which holds for all $x \geq -1$ and $0 \leq t \leq 1$, and the second inequality follows from $p \in [0,1]$.

For the first claim, applying (5.19) gives

$$g_{\phi(1-\varepsilon)}(A) = \int_0^1 \frac{\phi(1-\varepsilon)e^{\phi(1-\varepsilon)p}}{e^{\phi(1-\varepsilon)p}-1} \sum_{B\subseteq A} p^{|B|-1}(1-p)^{|A|-|B|} f(B)\, dp$$
$$\geq \int_0^1 e^{-\phi\varepsilon} \frac{\phi e^{\phi p}}{e^{\phi}-1} \sum_{B\subseteq A} p^{|B|-1}(1-p)^{|A|-|B|} f(B)\, dp = e^{-\phi\varepsilon} g_{\phi}(A),$$

as required. For the second claim, setting $p = 1$ in (5.19) gives $h(\phi(1-\varepsilon)) \geq e^{-\phi\varepsilon} h(\phi)$ or, equivalently, $h(\phi) \leq e^{\phi\varepsilon} h(\phi(1-\varepsilon))$. □

5.5.5 Restricted Range of guesses

In this section, we prove Lemma 5.5.2, which allows us to restrict the range for ϕ and γ that we consider in Algorithm 9.

Lemma 5.5.2. *For all $\gamma \in (0,1]$ and $\beta \geq 1$, $\phi(\gamma,\beta) \geq \frac{3}{4}$. Moreover, if $\phi(\gamma,\beta) > 4$ or $\gamma < \frac{1}{7}$, then $\frac{1}{(1+\gamma^{-1})^2} > \frac{\gamma^2(1-e^{-\phi(\gamma,\beta)})}{\phi(\gamma,\beta)}$.*

We recall that this theorem shows that whenever γ is too small or ϕ is too large, then ResidualRandomGreedy has an approximation factor greater than desired. Outputting the solution of ResidualRandomGreedy satisfies the guarantees of Theorem 5.4.2.

Proof of Lemma 5.5.2. First, we show that $\phi(\gamma,\beta) \geq 3/4$ for any value of $\gamma \in (0,1]$ and $\beta \geq 1$. Note that $\frac{\partial\phi}{\partial\beta} = 1-\gamma \geq 0$, for all $\gamma \in [0,1]$. Thus, any minimizer of $\phi(\gamma,\beta)$ sets $\beta = 1$. Moreover, $\frac{\partial\phi}{\partial\gamma} = 2\gamma - \beta$ and $\frac{\partial^2\phi}{\partial\gamma^2} = 2$ so a $\phi(\gamma,\beta)$ is minimized by $\gamma = \frac{\beta}{2} = \frac{1}{2}$. It follows that $\phi(\gamma,\beta) \geq \phi\left(\frac{1}{2},1\right) = \frac{3}{4}$ for all $\gamma \in [0,1]$ and $\beta \geq 1$.

Now suppose that $\phi(\gamma,\beta) > 4$. Then, the claim follows, since

$$\frac{\gamma^2(1-e^{-\phi(\gamma,\beta)})}{\phi(\gamma,\beta)} < \frac{\gamma^2}{4} \leq \frac{\gamma^2}{(1+\gamma)^2} = \frac{1}{(1+\gamma^{-1})^2}\,.$$

It remains to consider the case in which $\gamma < \frac{1}{7}$. Recall that $h(x) \triangleq \frac{xe^x}{e^x-1}$ is increasing in x and so $h(\phi(\gamma,\beta)) \geq h(\frac{3}{4}) > \frac{4}{3}$ (where the last inequality follows directly by computation of $h(\frac{3}{4})$). Suppose that $\gamma < \frac{1}{7}$. Then,

$$\frac{\gamma^2(1-e^{-\phi(\gamma,\beta)})}{\phi(\gamma,\beta)} = \frac{\gamma^2}{h(\phi(\gamma,\beta))} < \tfrac{3}{4}\gamma^2\,.$$

Comparing the previous estimation to the approximation ratio of [CFK18] and using that $\gamma < 1/7$, we have

$$\frac{\frac{3}{4}\gamma^2}{(1+\gamma^{-1})^{-2}} = \frac{3}{4}(\gamma+1)^2 < \frac{3}{4}\left(\frac{8}{7}\right)^2 < 1.$$

Thus, $\frac{3}{4}\gamma^2 < \frac{1}{(1+\gamma^{-1})^2}$ and again the claim follows. □

5.5.6 Efficient estimation of the potential via sampling

The definition of g_ϕ requires evaluating $f(B)$ on all $B \subseteq A$, which requires $2^{|A|}$ calls to the value oracle for f. In this section, we show that we can efficiently estimate g_ϕ using only a polynomial number of value queries to f. Our sampling procedure is based on the same general ideas described in [FW14], but here we focus on evaluating only the *marginals* of g_ϕ, which results in a considerably simpler implementation. In particular, our algorithm does not require computation of the coefficients $m^{(\phi)}_{a,b}$.

Lemma 5.5.3. *For any ϕ, N, there is a randomized procedure for obtaining an estimate $\tilde{g}(e|A)$ of $g_\phi(e|A)$ using N queries to the value oracle for f so that for any $\delta > 0$,*

$$\Pr[\, |g(e|A) - \tilde{g}(e|A)| \geq \delta f(A+e)\,] < 2e^{-\frac{\delta^2 N}{2}},$$

Proof of Lemma 5.5.3. We consider the following 2-step procedure given as an interpretation of g in [FW14]: we first sample $p \sim \mathcal{D}_\phi$, then construct a random $B \subseteq A$ by taking each element of A independently with probability p. The probability that any given $B \subseteq A$ is selected by the procedure is then precisely

$$\int_0^1 \frac{\phi e^{\phi p}}{e^\phi - 1} p^{|B|}(1-p)^{|A|}\, dp = m^{(\phi)}_{|A|,|B|}\,.$$

Thus, for a random $\tilde{B} \subseteq A$ sampled in this fashion, $\mathbb{E}[f(e|\tilde{B})] = \sum_{B \subseteq A} m^{(\phi)}_{|A|,|B|} f(e|B) = g(e|A)$, by part 1 of Lemma 5.4.3.

Suppose now that we draw N independent random samples $\{B_i\}_{i=1}^N$ in this fashion and define the random variables $Y_i = \frac{g(e|A) - f(e|B_i)}{f(A+e)}$. Then, $\mathbb{E}[Y_i] = 0$ for all i. Moreover, by monotonicity of f, $0 \leq f(e|B) \leq f(B+e) \leq f(A+e)$ for all $B \subseteq A$ and also $0 \leq \sum_{B\subseteq A} m^{(\phi)}_{|A|,|B|} f(e|B) = g(e|A)$ and $g(e|A) = \sum_{B\subseteq A} m^{(\phi)}_{|A|,|B|} f(e|B) \leq \sum_{B\subseteq A} m^{(\phi)}_{|A|,|B|} f(A+e) = f(A+e)$ by part 2 of Lemma 5.4.3. Thus, $|Y_i| \leq 1$ for all i. Let $\tilde{g}_\phi(e|A) = \frac{1}{N}\sum_{i=1}^N f(e|B_i)$. Applying the Chernoff bound (Lemma 1.B.5), for any $\delta > 0$ we have

$$\Pr[\, |g(e|A) - \tilde{g}(e|A)| \geq \delta f(A+e)\,] \leq \Pr\left[\textstyle\sum_{i=1}^N Y_i > \delta N\right] < 2e^{-\frac{\delta^2 N}{2}}. \qquad \square$$

5.5.7 Proof of Lemma 5.5.4

Here we show that the value of $g_\phi(A)$ can be bounded in terms of $f(A)$ for any set A. In the analysis of [FW14], this follows from submodularity of g, which is inherited from the submodularity of f. Here, we must again adopt a different approach. We begin by proving the following claim. Fix some set $A \subseteq X$ and for all $0 \leq j \leq |A|$ define $F_j \triangleq \sum_{B \in \binom{A}{j}} f(B)$ as the total value of all subsets of A of size j. Note that since we suppose f is normalized, $F_0 = f(\emptyset) = 0$. We start by bounding F_i.

Lemma 5.5.6. *If f is γ-weakly submodular, then $F_i \geq \binom{|A|-1}{i-1}\gamma f(A)$ for all $1 \leq i \leq |A|$.*

Proof of Lemma 5.5.6. Let $k = |A|$. Since f is γ-weakly submodular, for any $B \subseteq A$ we have

$$\sum_{e \in A \setminus B} (f(B+e) - f(B)) \geq \gamma(f(A) - f(B)).$$

Rearranging this, we have

$$\sum_{e \in A \setminus B} f(B+e) \geq \gamma f(A) + (|A| - |B| - \gamma) f(B) \geq \gamma f(A) + (|A| - |B| - 1) f(B)\,, \tag{5.20}$$

for all $B \subseteq A$. Summing (5.20) over all $\binom{k}{j}$ possible subsets B of size j, we obtain

$$(j+1)F_{j+1} \geq \gamma \tbinom{k}{j} f(A) + (k-j-1)F_j, \tag{5.21}$$

since each set T of size $j+1$ appears once as $B+e$ on the left-hand side of (5.20) for each of the $j+1$ distinct choices of $e \in T$ with $B = T - e$.

We now show that $F_i \geq \binom{k-1}{i-1}\gamma f(A)$ for all $1 \leq i \leq k$. The proof is by induction on i. For $i = 1$, the claim follows immediately from (5.21) with $j = 0$, since then $\binom{k}{j} = 1 = \binom{k-1}{i-1}$ and $(k-j-1)F_j = (k-1)F_0 = 0$. For the induction step, (5.21) and the induction hypothesis imply:

$$\begin{aligned}
F_{i+1} &\geq \tfrac{1}{i+1}\left(\gamma\tbinom{k}{i} f(A) + (k-i-1)F_i\right) \geq \tfrac{1}{i+1}\left(\gamma\tbinom{k}{i} f(A) + (k-i-1)\gamma\tbinom{k-1}{i-1} f(A)\right) \\
&= \tfrac{\gamma}{i+1}\left(\tfrac{k}{i}\tbinom{k-1}{i-1} f(A) + (k-i-1)\tbinom{k-1}{i-1} f(A)\right) = \tfrac{\gamma}{i+1}\left(\tfrac{k+ki-i^2-i}{i}\right)\tbinom{k-1}{i-1} f(A) \\
&= \tfrac{\gamma}{i+1}\tfrac{k(i+1)-i(i+1)}{i}\tbinom{k-1}{i-1} f(A) = \gamma\tfrac{k-i}{i}\tbinom{k-1}{i-1} f(A) = \gamma\tbinom{k-1}{i} f(A).
\end{aligned}$$

□

Using the above claim, we now bound the value of $g_\phi(A)$ for any set A.

Lemma 5.5.4. *If f is γ-weakly submodular, then for all $A \subseteq X$, $\gamma f(A) \leq g_\phi(A) \leq h(\phi) H_{|A|} f(A)$.*

Proof of Lemma 5.5.4. Let $k = |A|$. We begin with the lower bound for $g_\phi(A)$. By the definition of the coefficients $m_{a,b}^{(\phi)}$ and Lemma 5.5.6:

$$\begin{aligned}
g_\phi(A) &= \sum_{B \subseteq A} \mathop{\mathbb{E}}_{p \sim \mathcal{D}_\phi}\left[p^{|B|-1}(1-p)^{|A|-|B|}\right] f(B) = \mathop{\mathbb{E}}_{p \sim \mathcal{D}_\phi}\left[\sum_{i=1}^{k} p^{i-1}(1-p)^{k-i} F_i\right] \\
&\geq \mathop{\mathbb{E}}_{p \sim \mathcal{D}_\phi}\left[\sum_{i=1}^{k} \gamma p^{i-1}(1-p)^{k-i}\tbinom{k-1}{i-1} f(A)\right] = \mathop{\mathbb{E}}_{p \sim \mathcal{D}_\phi}\left[\gamma f(A) \sum_{i=0}^{k-1} \tbinom{k-1}{i} p^i (1-p)^{k-i-1}\right] \\
&= \mathop{\mathbb{E}}_{p \sim \mathcal{D}_\phi}[\gamma f(A)] = \gamma f(A).
\end{aligned}$$

For the upper bound, we similarly have:

$$\begin{aligned} g_\phi(A) &= \mathop{\mathbb{E}}_{p\sim\mathcal{D}_\phi}\left[\sum_{i=1}^{k} p^{i-1}(1-p)^{k-i}F_i\right] \le \mathop{\mathbb{E}}_{p\sim\mathcal{D}_\phi}\left[\sum_{i=1}^{k} p^{i-1}(1-p)^{k-i}\binom{k}{i}f(A)\right] \\ &= \int_0^1 \frac{\phi e^{\phi p}}{e^\phi - 1}\frac{\sum_{i=1}^{k}\binom{k}{i}p^i(1-p)^{k-i}f(A)}{p}\,dp = \int_0^1 \frac{\phi e^{\phi p}}{e^\phi - 1}\frac{1-(1-p)^k}{p}f(A)\,dp \\ &\le \frac{\phi e^\phi}{e^\phi - 1}\int_0^1 \frac{1-(1-p)^k}{p}f(A)\,dp = h(\phi)\int_0^1 \sum_{j=0}^{k-1}(1-p)^j f(A)\,dp \\ &= h(\phi)\sum_{j=0}^{k-1}\frac{1}{j+1}f(A) = h(\phi)H_{|A|}f(A), \end{aligned}$$

where the first inequality follows from monotonicity of f and the second inequality from $\frac{1-(1-p)^k}{p} > 0$ for $p \in (0,1]$ and $h(\phi) = \frac{\phi e^\phi}{e^\phi - 1}$ is an increasing function of p. □

5.6 A-optimal design for Bayesian linear regression

In this section, we provide another example of a concrete subset selection problem where β is non-trivially bounded. In Bayesian linear regression, we suppose data is generated by a linear model $\mathbf{y} = X^T\boldsymbol{\theta} + \boldsymbol{\varepsilon}$, where $\mathbf{y} \in \mathbb{R}^n, X \in \mathbb{R}^{p\times n}$ and $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, \sigma^2 I)$, where I is the identity matrix. Here, $X = \begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n\end{bmatrix}$ with $\mathbf{x}_i \in \mathbb{R}^p$ is a vector of data, and $\mathbf{y}$ is a vector corresponding observations for the response variable. The variable $\boldsymbol{\varepsilon}$ represents Gaussian noise with 0 mean and variance σ^2. When the number of columns n (i.e., the number of potential observations) is very large, *experimental design* focuses on selecting a small subset $S \subset \{1, 2, \ldots, n\}$ of columns of X to maximally reduce the variance of the estimator $\boldsymbol{\theta}$.

Let $X_S, \mathbf{y}_S$ be the matrix X (the vector $\mathbf{y}$ respectively) restricted to columns (rows respectively) indexed by S. From classical statistical theory, the optimal choice of parameters for any such S is given by $\hat{\boldsymbol{\theta}}_S = (X_S^T X_S)^{-1} X_S \mathbf{y}_S$ and satisfies $\mathrm{Var}(\hat{\boldsymbol{\theta}}_S) = \sigma^2 (X_S^T X_S)^{-1}$. Because the variance of $\hat{\boldsymbol{\theta}}_S$ is a matrix, there is not a universal function which one tries to minimize to find the appropriate set S. Instead, there are multiple objective functions depending on the context leading to different optimality criteria.

As in [KSG08; Bia+17; Har+19], we consider the *A-optimal* design objective. We suppose our prior probability distribution has $\boldsymbol{\theta} \sim \mathcal{N}(0, \Lambda)$. We start by stating a standard result from Bayesian linear regression.

Lemma 5.6.1 ([CV95]). *Given the previous assumption, and the prior on $\boldsymbol{\theta} \sim \mathcal{N}(0, \Lambda)$, The posterior distribution of $\boldsymbol{\theta}$ follows a normal distribution $p(\boldsymbol{\theta}|\mathbf{y}_S) \sim \mathcal{N}(M_S^{-1} X_S \mathbf{y}_S,\, M_S^{-1})$, where $M_S^{-1} \triangleq \left(\sigma^{-2} X_S X_S^T + \Lambda^{-1}\right)^{-1}$.*

In A-optimal design, our objective function seeks to reduce the variance of the posterior distribution of $\boldsymbol{\theta}$ by reducing the trace of M_S^{-1}, i.e., the sum of the variance of the regression coefficients. Mathematically, we seek to maximize the following objective function

$$F(S) = \operatorname{tr}(\Lambda) - \operatorname{tr}(M_S^{-1}) = \operatorname{tr}(\Lambda) - \operatorname{tr}((\Lambda^{-1} + \sigma^{-2} X_S X_S^T)^{-1}). \tag{5.22}$$

The function F is not submodular as shown in [KSG08]. The current tightest estimation of the submodularity ratio γ of F is due to Harshaw et al. [Har+19]. They show that $\gamma \geq (1 + \frac{s^2}{\sigma^2}\lambda_{\max}(\Lambda))^{-1}$, where $s = \max_{i\in[n]} \|\mathbf{x}_i\|$. Here we give a bound on the upper weak-submodularity ratio β.

Theorem 5.6.2. *Assume a prior distribution $\boldsymbol{\theta} \sim \mathcal{N}(0, \Lambda)$, and let $s = \max_{i\in[n]} \|\mathbf{x}_i\|$. The function F is $(1/c, c)$-weakly submodular with $c = 1 + \frac{s^2}{\sigma^2}\lambda_{\max}(\Lambda)$.*

Observe that like for the R^2 objective, our upper bound for β is the inverse of the lower bound for γ.

Proof of Theorem 5.6.2. The lower bound on γ is shown is [Har+19]. It remains to prove the upper bound on β. Let B be some set of observations and $A \subseteq B$ with $k = |A|$ and for convenience, define $T = B \setminus A$. By the Sherman-Morrisson-Woodbury formula (see Lemma 1.B.2), we have

$$\begin{aligned} F(B) - F(A) &= \operatorname{tr}(M_A^{-1}) - \operatorname{tr}(M_B^{-1}) \\ &= \operatorname{tr}\big((M_B - \sigma^{-2} X_T X_T^T)^{-1}\big) - \operatorname{tr}(M_B^{-1}) \\ &= \operatorname{tr}\big(M_B^{-1} + M_B^{-1} X_T (\sigma^2 I - X_T^T M_B^{-1} X_T)^{-1} X_T^T M_B^{-1}\big) - \operatorname{tr}(M_B^{-1}) \\ &= \operatorname{tr}\big(M_B^{-1} X_T (\sigma^2 I - X_T^T M_B^{-1} X_T)^{-1} X_T^T M_B^{-1}\big) \\ &= \operatorname{tr}\big((\sigma^2 I - X_T^T M_B^{-1} X_T)^{-1} X_T^T M_B^{-2} X_T\big). \end{aligned} \tag{5.23}$$

The fourth equality uses the linearity of the trace while the last equality uses the cyclic property of the trace. We use the previous equation to derive an upper and lower bound for the numerator and denominator of the submodularity ratio respectively. Applying (5.23) with $A = B \setminus \{i\}$ (and so $T = \{i\}$) we obtain

$$F(B) - F(B - i) = \frac{\operatorname{tr}(\mathbf{x}_i^T M_B^{-2} \mathbf{x}_i)}{\sigma^2 - \mathbf{x}_i^T M_B^{-1} \mathbf{x}_i}.$$

Let $\preceq$ be the Loewner ordering of positive semidefinite matrices, where $A \preceq B$ if and only if $B - A \succeq 0$. First, observe that $\Lambda^{-1} \preceq M_R$ for any set R, which implies that $\Lambda \succeq M_R^{-1}$. Using

a second time the Sherman-Morrison-Woodbury formula (Lemma 1.B.2) together with the previous observation, we get

$$\begin{aligned}\left(\sigma^2 - \mathbf{x}_i^T M_B^{-1} \mathbf{x}_i\right)^{-1} &= \sigma^{-2} + \sigma^{-4} \mathbf{x}_i^T \left(M_B - \sigma^{-2} \mathbf{x}_i \mathbf{x}_i^T\right)^{-1} \mathbf{x}_i, \\ &= \sigma^{-2} + \sigma^{-4} \mathbf{x}_i^T M_{B\setminus\{i\}}^{-1} \mathbf{x}_i, \\ &\leq \sigma^{-2} + \sigma^{-4} \mathbf{x}_i^T \Lambda \mathbf{x}_i, \\ &\leq \sigma^{-2} + \sigma^{-4} \lambda_{\max}(\Lambda) s^2,\end{aligned}$$

where $s = \max_i \|\mathbf{x}_i\|_2$ and the last inequality follows by the Courant-Fischer min-max theorem. Summing over all $i \in T = B \setminus A$ and using the linearity of the trace, we have

$$\begin{aligned}\sum_{i\in T} F(i|B-i) = \sum_{i\in T} \frac{\operatorname{tr}(\mathbf{x}_i^T M_B^{-2} \mathbf{x}_i)}{\sigma^2 - x_i^T M_B^{-1} x_i} &\leq \left(\sigma^{-2} + s^2\sigma^{-4}\lambda_{\max}(\Lambda)\right) \sum_{i\in T} \operatorname{tr}(\mathbf{x}_i^T M_B^{-2} \mathbf{x}_i) \\ &= \left(\sigma^{-2} + s^2\sigma^{-4}\lambda_{\max}(\Lambda)\right) \operatorname{tr}(X_T^T M_B^{-2} X_T). \qquad (5.24)\end{aligned}$$

Returning to the expression of $F(B) - F(A)$, we note that M_B is positive definite, which implies that M_B^{-1} is positive definite. This in turn implies that $-X_T^T M_B^{-1} X_T \preceq 0$ and so $\sigma^2 I - X_T^T M_B^{-1} X_T \preceq \sigma^2 I$. Thus, $(\sigma^2 I - X_T^T M_B^{-1} X_T)^{-1} \succeq \sigma^{-2} I \succ 0$. Therefore,

$$\operatorname{tr}((\sigma^2 I - X_T^T M_B^{-1} X_T)^{-1} X_T^T M_B^{-2} X_T) \geq \operatorname{tr}\left(\sigma^{-2} X_T^T M_B^{-2} X_T\right) = \sigma^{-2} \operatorname{tr}\left(X_T^T M_B^{-2} X_T\right).$$

Combining this with the bound (5.24), we have:

$$\frac{\sum_{i\in T} F(i|B-i)}{F(B) - F(A)} \leq \frac{\left(\sigma^{-2} + \sigma^{-4}\lambda_{\max}(\Lambda) \cdot s^2\right) \operatorname{tr}\left(X_T^T M_B^{-2} X_T\right)}{\sigma^{-2} \operatorname{tr}\left(X_T^T M_B^{-2} X_T\right)} \leq 1 + \frac{s^2}{\sigma^2} \lambda_{\max}(\Lambda).$$

Recalling that $T = B \setminus A$, this completes the proof. □

5.7 The Column Subset Selection Problem

In this section, we demonstrate that the column subset selection problem is (γ, β)-weakly submodular. The problem appears in [Far+15; Alt+16]. Like the R^2 objective, it is an important machine learning problem which provides interpretability of high-dimensional statistics through the selection of the best columns of a given matrix. Given an $m \times n$ matrix A, the goal is to find a subset of columns $\mathcal{S}$ such that $\mathcal{S} \in \mathcal{I}$ for some matroid constraint $\mathcal{I}$. It is described by the following objective function:

$$f(\mathcal{S}) \triangleq \|P_{\mathcal{S}} A\|_F^2, \qquad (5.25)$$

where $P_{\mathcal{S}}$ is the *best* $m \times m$ projection matrix which projects the columns of A on the span of the candidates columns $A_{\mathcal{S}}$. For a matrix A, $\|A\|_F \triangleq \sqrt{\sum_{i=1}^m \sum_{j=1}^n |a_{i,j}|^2} = \operatorname{tr}(A^T A)$. Given a set $\mathcal{S}$ of columns the best projection matrix $P_{\mathcal{S}}$ has a closed form equal to $P_{\mathcal{S}} = A_{\mathcal{S}}(A_{\mathcal{S}}^T A_{\mathcal{S}})^{-1} A_{\mathcal{S}}^T$, or equivalently, $P_{\mathcal{S}} = A_{\mathcal{S}} A_{\mathcal{S}}^+$ where $A_{\mathcal{S}}^+ \triangleq (A_{\mathcal{S}}^T A_{\mathcal{S}})^{-1} A_{\mathcal{S}}^T$ is the Moore-Penrose inverse. Some

key properties of the projection matrix $P_{\mathcal{S}}$ include: $P_{\mathcal{S}}P_{\mathcal{S}} = P_{\mathcal{S}}$ and $P_{\mathcal{S}}^T = P_{\mathcal{S}}$. We prove that the objective function (5.25) is weakly submodular and satisfies similar properties to that of the R^2 objective.

Theorem 5.7.1. *Objective* (5.25) *is* $(\gamma, 1/\gamma)$*-weakly submodular where* $\gamma \geq \sigma^2_{\min}(A)$*. Restricted to two sets* $\mathcal{A} \subseteq \mathcal{B}$*, the submodularity property in Equation* (1.2) *and* (5.1) *is satisfied with* $\gamma = 1/\beta \geq \sigma^2_{\min}(A, |\mathcal{B}|)$*, where* $\sigma^2_{\min}(A, k)$ *is the squared minimum singular value of a subset of* k *columns of* A*.*

The proof of Theorem 5.7.1 follows the lines of the proof of Theorem 5.2.1. In fact, Theorem 5.2.1 is a special case of Theorem 5.7.1. To simplify the proof, we define $\mathrm{Res}(\mathcal{R}, \mathcal{S})$ as the set of residual vectors/columns in $\mathcal{R}$ after being projected onto $\mathcal{S}$. The matrix associated with $\mathrm{Res}(\mathcal{R}, \mathcal{S})$ is $\Pi_{\mathcal{R},\mathcal{S}} \triangleq A_{\mathcal{R}} - P_{\mathcal{S}}A_{\mathcal{R}}$ where columns of $\Pi_{\mathcal{R},\mathcal{S}}$ are the vectors in $\mathrm{Res}(\mathcal{R}, \mathcal{S})$. We define $P_{\mathrm{Res}(\mathcal{R},\mathcal{S})} \triangleq \Pi_{\mathcal{R},\mathcal{S}}(\Pi_{\mathcal{R},\mathcal{S}}^T\Pi_{\mathcal{R},\mathcal{S}})^{-1}\Pi_{\mathcal{R},\mathcal{S}}^T$ as the projection matrix onto $\mathrm{Res}(\mathcal{R}, \mathcal{S})$. Observe that $P_{\mathcal{S}}$ is equivalent to $P_{\mathrm{Res}(\mathcal{S},\emptyset)}$.

5.7.1 Decomposition Properties

Similar to the R^2 objective, the objective function satisfies the following lemmas that are analog to Lemma 5.2.3 and 5.2.4.

Lemma 5.7.2 ([Far+15])**.** *Let* $\mathcal{T} = \mathcal{S} \cup \mathcal{R}$ *be a subset of columns of a given matrix* A*. Suppose that* $\mathcal{S} \cap \mathcal{R} = \emptyset$*. Then, we have*

$$P_{\mathcal{T}} = P_{\mathcal{S}} + P_{\mathrm{Res}(\mathcal{R},\mathcal{S})}.$$

This lemma tells that the projection onto the span of the columns indexed by $\mathcal{T}$ is equivalent to the projection onto the span of the columns indexed by $\mathcal{S}$ plus the projection onto the *residual* columns indexed by $\mathcal{R}$, where *residual* vectors are the component of the original vectors perpendicular to the space spanned by $\mathcal{S}$. Thus, we can intuitively think of the projection as a 2-stage procedure, first projecting the columns on a subset of columns indexed by $\mathcal{S}$ and secondly on a subset of *residual* columns indexed by $\mathcal{R}$. The next lemma uses this idea to compute the projection to understand the 2-stage projection as acting on orthogonal subspaces with respect to the Frobenius norm.

Lemma 5.7.3 ([Far+15])**.**

$$\|P_{\mathcal{R}\cup\mathcal{S}}A\|_F^2 = \|P_{\mathcal{S}}A\|_F^2 + \|P_{\mathrm{Res}(\mathcal{R},\mathcal{S})}A\|_F^2,$$

From straightforward application of Lemma 5.7.3, we obtain the following lemma

Lemma 5.7.4. *Let* $\mathcal{S} \cup \mathcal{R} \cup \mathcal{T}$ *be pairwise disjoint subsets of columns of a given matrix* A*. Then, we have*

$$\|P_{\mathrm{Res}(\mathcal{T},\mathcal{S}\cup\mathcal{R})}A\|_F^2 = \|P_{\mathrm{Res}(\mathrm{Res}(\mathcal{T},\mathcal{S}),\mathrm{Res}(\mathcal{R},\mathcal{S}))}A\|_F^2.$$

Proof of Lemma 5.7.4. By Lemma 5.7.3, we have that $f(\mathcal{T} \mid \mathcal{R} \cup \mathcal{S}) = \|P_{\mathrm{Res}(\mathcal{T},\mathcal{R}\cup\mathcal{S})}A\|_F^2$.

$$\begin{aligned}
f(\mathcal{T} \mid \mathcal{R} \cup \mathcal{S}) &= f(\mathcal{T} \cup \mathcal{R} \mid \mathcal{S}) - f(\mathcal{R} \mid \mathcal{S}),\\
&= \|P_{\mathrm{Res}(\mathcal{T}\cup\mathcal{R},\mathcal{S})}A\|_F^2 - \|P_{\mathrm{Res}(\mathcal{R},\mathcal{S})}A\|_F^2,\\
&= \|P_{\mathrm{Res}(\mathcal{T},\mathcal{S})\cup\mathrm{Res}(\mathcal{R},\mathcal{S})}A\|_F^2 - \|P_{\mathrm{Res}(\mathcal{R},\mathcal{S})}A\|_F^2,\\
&= \|P_{\mathrm{Res}(\mathcal{R},\mathcal{S})}A\|_F^2 + \|P_{\mathrm{Res}(\mathrm{Res}(\mathcal{T},\mathcal{S}),\mathrm{Res}(\mathcal{R},\mathcal{S}))}A\|_F^2 - \|P_{\mathrm{Res}(\mathcal{R},\mathcal{S})}A\|_F^2,\\
&= \|P_{\mathrm{Res}(\mathrm{Res}(\mathcal{T},\mathcal{S}),\mathrm{Res}(\mathcal{R},\mathcal{S}))}A\|_F^2,
\end{aligned}$$

where the before last equality is by Lemma 5.7.3. □

5.7.2 Proof of Theorem 5.7.1

We prove Theorem 5.7.1. We assume that we are given two disjoint subsets of columns of A which we denote by $\mathcal{S}$ and $\mathcal{R}$. Our goal is to derive a bound on the submodularity ratio of the objective function. By a renumbering of the columns of A, we let $\mathcal{R} \triangleq \{\mathbf{a}_1, \ldots, \mathbf{a}_\ell\}$ where $\ell = |\mathcal{R}|$. As our proof requires projecting $\mathcal{R}$ onto the span of $\mathcal{S}$, we denote by $\boldsymbol{\pi}_i \triangleq \mathbf{a}_i - P_{\mathcal{S}}\mathbf{a}_i$ their residual projection. The vectors $\boldsymbol{\pi}_i$ are the columns of the matrix $\Pi_{\mathcal{R},\mathcal{S}}$. We will further assume that $\boldsymbol{\pi}_i$'s are normalized so that $\boldsymbol{\pi}_i^T\boldsymbol{\pi}_i = 1$. The marginal gain of adding $\mathcal{R}$ to $\mathcal{S}$ is equal to:

$$\begin{aligned}
f(\mathcal{R} \mid \mathcal{S}) &= \|P_{\mathcal{R}\cup\mathcal{S}}A\|_F^2 - \|P_{\mathcal{S}}A\|_F^2\\
&= \|P_{\mathrm{Res}(\mathcal{R},\mathcal{S})}A\|_F^2 && \text{(Lemma 5.7.3)}\\
&= \mathrm{tr}\big(A^T P_{\mathrm{Res}(\mathcal{R},\mathcal{S})} P_{\mathrm{Res}(\mathcal{R},\mathcal{S})} A\big)\\
&= \mathrm{tr}\big(A^T P_{\mathrm{Res}(\mathcal{R},\mathcal{S})} A\big) && (P^2 = P)\\
&= \mathrm{tr}\big(A^T \Pi_{\mathcal{R},\mathcal{S}} (\Pi_{\mathcal{R},\mathcal{S}}^T \Pi_{\mathcal{R},\mathcal{S}})^{-1} \Pi_{\mathcal{R},\mathcal{S}}^T A\big)\,. && (5.26)
\end{aligned}$$

We observe that $\Pi_{\mathcal{R},\mathcal{S}}^T\Pi_{\mathcal{R},\mathcal{S}}$ is a positive semidefinite matrix. In particular, $\Pi_{\mathcal{R},\mathcal{S}}^T\Pi_{\mathcal{R},\mathcal{S}} \succeq \lambda_{\min}(\Pi_{\mathcal{R},\mathcal{S}}^T\Pi_{\mathcal{R},\mathcal{S}})I$, where I is the identity matrix. Thus, $(\Pi_{\mathcal{R},\mathcal{S}}^T\Pi_{\mathcal{R},\mathcal{S}})^{-1} \preceq \lambda_{\min}^{-1}(\Pi_{\mathcal{R},\mathcal{S}}^T\Pi_{\mathcal{R},\mathcal{S}})I$, which implies that:

$$f(\mathcal{R} \mid \mathcal{S}) = \mathrm{tr}\big(A^T \Pi_{\mathcal{R},\mathcal{S}} (\Pi_{\mathcal{R},\mathcal{S}}^T \Pi_{\mathcal{R},\mathcal{S}})^{-1} \Pi_{\mathcal{R},\mathcal{S}}^T A\big) \leq \lambda_{\min}^{-1}(\Pi_{\mathcal{R},\mathcal{S}}^T\Pi_{\mathcal{R},\mathcal{S}})\mathrm{tr}\big(A^T \Pi_{\mathcal{R},\mathcal{S}} \Pi_{\mathcal{R},\mathcal{S}}^T A\big)\,,$$

where the inequality follows from: $\mathrm{tr}(AB) \leq \mathrm{tr}(AC)$ if $A, B, C \succeq 0$ and $B \preceq C$. On the other hand,

$$\begin{aligned}\sum_{i=1}^{\ell} f(\mathbf{a}_i \mid \mathcal{S}) &= \sum_{i=1}^{\ell} \mathrm{tr}\left(A^T P_{\mathrm{Res}(\mathbf{a}_i,\mathcal{S})} A\right) \\ &= \sum_{i=1}^{\ell} \mathrm{tr}\left(A^T \boldsymbol{\pi}_i (\boldsymbol{\pi}_i^T \boldsymbol{\pi}_i)^{-1} \boldsymbol{\pi}_i^T A\right) \\ &= \sum_{i=1}^{\ell} \mathrm{tr}\left(A^T \boldsymbol{\pi}_i \boldsymbol{\pi}_i^T A\right) \\ &= \mathrm{tr}\left(A^T \Pi_{\mathcal{R},\mathcal{S}} \Pi_{\mathcal{R},\mathcal{S}}^T A\right),\end{aligned}$$

where the third equality follows by normalization of $\boldsymbol{\pi}_x$, and the last inequality is by linearity of the trace. Using Equation (5.26) we derive our lower bound on γ.

$$f(\mathcal{R} \mid \mathcal{S}) \leq \lambda_{\min}^{-1}(\Pi_{\mathcal{R},\mathcal{S}}^T \Pi_{\mathcal{R},\mathcal{S}}) \cdot \mathrm{tr}\left(A^T \Pi_{\mathcal{R},\mathcal{S}} \Pi_{\mathcal{R},\mathcal{S}}^T A\right) = \lambda_{\min}^{-1}(\Pi_{\mathcal{R},\mathcal{S}}^T \Pi_{\mathcal{R},\mathcal{S}}) \cdot \sum_{i=1}^{\ell} f(\mathbf{a}_i \mid \mathcal{S}). \quad (5.27)$$

Next we turn our attention to the upper submodularity ratio β. To simplify the notations, we let $\mathrm{Res}(\mathcal{R},\mathcal{S}) = \hat{\mathcal{R}} = \{\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots, \boldsymbol{\pi}_\ell\}$, and define $\hat{\mathcal{R}}_{-i} \triangleq \hat{\mathcal{R}} \setminus \{\boldsymbol{\pi}_i\}$ and $\mathcal{R}_{-i} \triangleq \mathcal{R} \setminus \{\mathbf{a}_i\}$. Fix an index $i = 1, \ldots, \ell$, and let $\boldsymbol{\pi}_i \in \hat{\mathcal{R}}$. By permutation of the columns of A, we may assume that the row and the column corresponding to the index i are the last row and last column of $(\Pi_{\mathcal{R},\mathcal{S}}^T \Pi_{\mathcal{R},\mathcal{S}})$. We expand the term $f(\mathbf{a}_i \mid \mathcal{S} \cup \mathcal{R}_{-i})$.

$$f(\mathbf{a}_i \mid \mathcal{S} \cup \mathcal{R} - \mathbf{a}_i) = \|P_{\mathrm{Res}(\mathrm{Res}(\mathbf{a}_i,\mathcal{S}),\mathrm{Res}(\mathcal{R}_{-i},\mathcal{S}))} A\|_F^2 = \|P_{\mathrm{Res}(\boldsymbol{\pi}_i,\hat{\mathcal{R}}_{-i})} A\|_F^2 = \mathrm{tr}\left(A^T P_{\mathrm{Res}(\boldsymbol{\pi}_i,\hat{\mathcal{R}}_{-i})} A\right).$$

The key idea of the proof is to prove that $\sum_{i=1}^{\ell} P_{\mathrm{Res}(\boldsymbol{\pi}_i,\hat{\mathcal{R}}_{-i})} \preceq \lambda_{\min}^{-1}(\Pi_{\mathcal{R},\mathcal{S}}^T \Pi_{\mathcal{R},\mathcal{S}}) P_{\hat{\mathcal{R}}}$. To prove it, it is sufficient to prove that: $\mathbf{v}^T \left(\sum_{i=1}^{\ell} P_{\mathrm{Res}(\boldsymbol{\pi}_i,\hat{\mathcal{R}}_{-i})}\right) \mathbf{v} \leq \lambda_{\min}^{-1}(\Pi_{\mathcal{R},\mathcal{S}}^T \Pi_{\mathcal{R},\mathcal{S}}) \mathbf{v}^T P_{\hat{\mathcal{R}}} \mathbf{v}$ for any vector $\mathbf{v}$. Applying the block inverse formula 1.B.1 to the matrix $\Pi_{\mathcal{R},\mathcal{S}}^T \Pi_{\mathcal{R},\mathcal{S}}$, we get:

$$(\Pi_{\mathcal{R},\mathcal{S}}^T \Pi_{\mathcal{R},\mathcal{S}})^{-1} = \begin{bmatrix} (\Pi_{\mathcal{R}_{-i},\mathcal{S}}^T \Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1} & 0 \\ 0 & 0 \end{bmatrix} + \frac{1}{s_i} H_i,$$

where $s_i \triangleq 1 - \boldsymbol{\pi}_i^T \Pi_{\mathcal{R}_{-i},\mathcal{S}} (\Pi_{\mathcal{R}_{-i},\mathcal{S}}^T \Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1} \Pi_{\mathcal{R}_{-i},\mathcal{S}}^T \boldsymbol{\pi}_i$, and

$$H_i \triangleq \begin{bmatrix} (\Pi_{\mathcal{R}_{-i},\mathcal{S}}^T \Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1} \Pi_{\mathcal{R}_{-i},\mathcal{S}}^T \boldsymbol{\pi}_i \boldsymbol{\pi}_i^T \Pi_{\mathcal{R}_{-i},\mathcal{S}} (\Pi_{\mathcal{R}_{-i},\mathcal{S}}^T \Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1} & -(\Pi_{\mathcal{R}_{-i},\mathcal{S}}^T \Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1} \Pi_{\mathcal{R}_{-i},\mathcal{S}}^T \boldsymbol{\pi}_i \\ -\boldsymbol{\pi}_i^T \Pi_{\mathcal{R}_{-i},\mathcal{S}} (\Pi_{\mathcal{R}_{-i},\mathcal{S}}^T \Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1} & 1 \end{bmatrix}.$$

We start by showing the following proposition which rewrites $P_{\mathrm{Res}(\boldsymbol{\pi}_i,\hat{\mathcal{R}}_{-i})}$ in a simpler form:

Proposition 5.7.5. $P_{\mathrm{Res}(\boldsymbol{\pi}_i,\hat{\mathcal{R}}_{-i})} = \frac{1}{s_i} \Pi_{\mathcal{R},\mathcal{S}} H_i \Pi_{\mathcal{R},\mathcal{S}}^T$

Proof of Proposition 5.7.5. By definition of the projection matrix, we have $P_{\mathrm{Res}(\boldsymbol{\pi}_i,\hat{\mathcal{R}}_{-i})} = \Pi_{\boldsymbol{\pi}_i,\hat{\mathcal{R}}_{-i}}(\Pi^T_{\boldsymbol{\pi}_i,\hat{\mathcal{R}}_{-i}}\Pi_{\boldsymbol{\pi}_i,\hat{\mathcal{R}}_{-i}})^{-1}\Pi^T_{\boldsymbol{\pi}_i,\hat{\mathcal{R}}_{-i}}$. We start by observing that,

$$\Pi_{\boldsymbol{\pi}_i,\hat{\mathcal{R}}_{-i}} = \boldsymbol{\pi}_i - P_{\hat{\mathcal{R}}_{-i}}\boldsymbol{\pi}_i = \boldsymbol{\pi}_i - P_{\mathrm{Res}(\mathcal{R}_{-i},\mathcal{S})}\boldsymbol{\pi}_i = \boldsymbol{\pi}_i - \Pi_{\mathcal{R}_{-i},\mathcal{S}}(\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1}\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\boldsymbol{\pi}_i. \tag{5.28}$$

On the one hand, we have

$$\begin{aligned}\Pi^T_{\boldsymbol{\pi}_i,\hat{\mathcal{R}}_{-i}}\Pi_{\boldsymbol{\pi}_i,\hat{\mathcal{R}}_{-i}} &= \boldsymbol{\pi}_i^T\boldsymbol{\pi}_i - \boldsymbol{\pi}_i^T\Pi_{\mathcal{R}_{-i},\mathcal{S}}(\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1}\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\boldsymbol{\pi}_i,\\ &= 1 - \boldsymbol{\pi}_i^T\Pi_{\mathcal{R}_{-i},\mathcal{S}}(\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1}\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\boldsymbol{\pi}_i.\end{aligned}$$

The first equality follows by expanding the term $\Pi^T_{\boldsymbol{\pi}_i,\hat{\mathcal{R}}_{-i}}$ using Equation (5.28), computing the product and then using that $\Pi_{\mathcal{R}_{-i},\mathcal{S}}(\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1}\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}$ is a projection matrix. The last equality holds because the $\boldsymbol{\pi}_i$ have norm equal to 1. On the one hand,

$$\begin{aligned}&\Pi_{\boldsymbol{\pi}_i,\hat{\mathcal{R}}_{-i}}\Pi^T_{\boldsymbol{\pi}_i,\hat{\mathcal{R}}_{-i}}\\ =\ &(\boldsymbol{\pi}_i - \Pi_{\mathcal{R}_{-i},\mathcal{S}}(\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1}\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\boldsymbol{\pi}_i)(\boldsymbol{\pi}_i - \Pi_{\mathcal{R}_{-i},\mathcal{S}}(\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1}\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\boldsymbol{\pi}_i)^T,\\ =\ &\boldsymbol{\pi}_i\boldsymbol{\pi}_i^T - \boldsymbol{\pi}_i\boldsymbol{\pi}_i^T\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}(\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1}\Pi_{\mathcal{R}_{-i},\mathcal{S}} - \Pi_{\mathcal{R}_{-i},\mathcal{S}}(\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1}\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\boldsymbol{\pi}_i\boldsymbol{\pi}_i^T\\ &\quad + \Pi_{\mathcal{R}_{-i},\mathcal{S}}(\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1}\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\boldsymbol{\pi}_i\boldsymbol{\pi}_i^T\Pi_{\mathcal{R}_{-i},\mathcal{S}}(\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1}\Pi^T_{\mathcal{R}_{-i},\mathcal{S}},\\ =\ &\Pi_{\mathcal{R},\mathcal{S}}H_i\Pi^T_{\mathcal{R},\mathcal{S}}.\end{aligned}$$

Combining both computations yield the desired result. □

Let $\mathbf{v}$ be any vector and let $\mathbf{u} \triangleq \Pi^T_{\mathcal{R},\mathcal{S}}\mathbf{v}$. Applying the matrix block inverse theorem 1.B.1 to the matrix $(\Pi^T_{\mathcal{R},\mathcal{S}}\Pi_{\mathcal{R},\mathcal{S}})^{-1}$, we get

$$\mathbf{v}^T\Pi_{\mathcal{R},\mathcal{S}}(\Pi^T_{\mathcal{R},\mathcal{S}}\Pi_{\mathcal{R},\mathcal{S}})^{-1}\Pi^T_{\mathcal{R},\mathcal{S}}\mathbf{v} = \mathbf{u}^T\begin{bmatrix}(\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1} & 0\\ 0 & 0\end{bmatrix}\mathbf{u} + \frac{1}{s_i}\mathbf{u}^T H_i\mathbf{u}.$$

The previous expression drastically simplifies when $\mathbf{u}$ is an eigenvector of $(\Pi^T_{\mathcal{R},\mathcal{S}}\Pi_{\mathcal{R},\mathcal{S}})^{-1}$. In fact, let $(\lambda, \mathbf{w})$ be an eigenpair of the matrix $(\Pi^T_{\mathcal{R},\mathcal{S}}\Pi_{\mathcal{R},\mathcal{S}})^{-1}$, where $\mathbf{w} = \Pi^T_{\mathcal{R},\mathcal{S}}\mathbf{v}$. Consider the i^{th} index of the vector $(\Pi^T_{\mathcal{R},\mathcal{S}}\Pi_{\mathcal{R},\mathcal{S}})^{-1}\mathbf{w}$ that without loss of generality corresponds to last row and last column of the matrix, then

$$\begin{aligned}\lambda w_i = \left((\Pi^T_{\mathcal{R},\mathcal{S}}\Pi_{\mathcal{R},\mathcal{S}})^{-1}\mathbf{w}\right)_i &= \left(\begin{bmatrix}(\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1} & 0\\ 0 & 0\end{bmatrix}\mathbf{w}\right)_i + \left(\frac{1}{s_i}H_i\mathbf{w}\right)_i,\\ &= \left(\frac{1}{s_i}H_i\mathbf{w}\right)_i,\\ &= \frac{1}{s_i}\left(-\boldsymbol{\pi}_i^T\Pi_{\mathcal{R}_{-i},\mathcal{S}}(\Pi^T_{\mathcal{R}_{-i},\mathcal{S}}\Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1}\mathbf{w}_{-i} + w_i\right),\end{aligned}$$

where $\mathbf{w}_{-i}$ corresponds to the vector $\mathbf{w}$ minus its ith (and hence last) index. Therefore, we obtain the following identity:

$$(1-\lambda s_i)w_i = \boldsymbol{\pi}_i^T \Pi_{\mathcal{R}_{-i},\mathcal{S}} (\Pi_{\mathcal{R}_{-i},\mathcal{S}}^T \Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1} \mathbf{w}_{-i}. \tag{5.29}$$

Take two such eigenpairs $(\lambda, \mathbf{w})$ and $(\mu, \mathbf{y})$ of the matrix $(\Pi_{\mathcal{R},\mathcal{S}}^T \Pi_{\mathcal{R},\mathcal{S}})^{-1}$. By applying Equation (5.29), we get that

$$\begin{aligned}
\mathbf{y}^T H_i \mathbf{w} &= \mathbf{y}_{-i}^T (\Pi_{\mathcal{R}_{-i},\mathcal{S}}^T \Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1} \Pi_{\mathcal{R}_{-i},\mathcal{S}}^T \boldsymbol{\pi}_i \boldsymbol{\pi}_i^T \Pi_{\mathcal{R}_{-i},\mathcal{S}} (\Pi_{\mathcal{R}_{-i},\mathcal{S}}^T \Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1} \mathbf{w}_{-i} \\
&\quad - \mathbf{y}_{-i}^T (\Pi_{\mathcal{R}_{-i},\mathcal{S}}^T \Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1} \Pi_{\mathcal{R}_{-i},\mathcal{S}}^T \boldsymbol{\pi}_i w_i - u_i \boldsymbol{\pi}_i^T \Pi_{\mathcal{R}_{-i},\mathcal{S}} (\Pi_{\mathcal{R}_{-i},\mathcal{S}}^T \Pi_{\mathcal{R}_{-i},\mathcal{S}})^{-1} \mathbf{w}_{-i} \\
&\quad + y_i w_i, \\
&= (1-\lambda s_i)(1-\mu s_i) u_i w_i - (1-\lambda s_i) u_i w_i - (1-\mu s_i) u_i w_i + u_i w_i \\
&= u_i w_i \left[1 - \mu s_i - \lambda s_i + \lambda \mu s_i^2 - 1 + \lambda s_i - 1 + \mu s_i + 1\right], \\
&= s_i^2 \lambda \mu u_i w_i
\end{aligned}$$

Let $\{\mathbf{w}_1, \ldots, \mathbf{w}_i\}$ be an eigenbasis of $(\Pi_{\mathcal{R},\mathcal{S}}^T \Pi_{\mathcal{R},\mathcal{S}})^{-1}$ with associated eigenvalues $\lambda_1 \leq \ldots \leq \lambda_i$. Recall, that $\mathbf{u} = \Pi_{\mathcal{R},\mathcal{S}}^T \mathbf{v}$. Let W be a matrix with columns given by these $\{\mathbf{w}_j\}_{j=1}^i$. Since $(\Pi_{\mathcal{R},\mathcal{S}}^T \Pi_{\mathcal{R},\mathcal{S}})^{-1}$ is a symmetric positive semidefinite matrix, the matrix W is orthonormal. Hence, we can write $\mathbf{u} = W\mathbf{y}$ for some vector $\mathbf{y}$. Thus,

$$\mathbf{u}^T H_i \mathbf{u} = \mathbf{y}^T W^T H_i W \mathbf{y} \tag{5.30}$$

By the previous computation, the index (ℓ, m) of the matrix $W^T H_i W$ is equal to

$$(W^T H_i W)_{\ell,m} = \lambda_\ell \lambda_m s_i^2 \mathbf{w}_\ell^T \mathbf{w}_m = \begin{cases} 0 & \text{if } \ell \neq m, \\ \lambda_\ell^2 s_i^2 & \text{otherwise.} \end{cases}$$

Summing over all indices $i = 1, \ldots, |\mathcal{R}|$, and applying Proposition 5.7.5 we get:

$$\mathbf{v}^T \left(\sum_{i=1}^{|\mathcal{R}|} P_{\mathrm{Res}(\boldsymbol{\pi}_i, \hat{\mathcal{R}}_{-i})} \right) \mathbf{v} = \sum_{i=1}^{|\mathcal{R}|} \frac{\mathbf{u}^T H_i \mathbf{u}}{s_i} \leq \sum_{i=1}^{|\mathcal{R}|} \frac{\mathbf{u}^T H_i \mathbf{u}}{s_i^2} = \sum_{i=1}^{|\mathcal{R}|} \frac{\mathbf{y}^T W^T H_i W \mathbf{y}}{s_i^2} = \sum_{i=1}^{|\mathcal{R}|} y_i^2 \lambda_i^2,$$

where the inequality is because $s_i \leq 1$, and the positivity of the marginal contribution $f(\mathbf{a}_i \mid \mathcal{S} \cup \mathcal{R}_{-i})$. The second inequality is by applying Equation (5.30). The final equality is by the orthonormality of W. Using the previous computation

$$\begin{aligned}
\mathbf{v}^T \left(\sum_{i=1}^{|\mathcal{R}|} P_{\mathrm{Res}(\boldsymbol{\pi}_i, \hat{\mathcal{R}} - \boldsymbol{\pi}_i)} \right) \mathbf{v} &\leq \lambda_{\max}((\Pi_{\mathcal{R},\mathcal{S}}^T \Pi_{\mathcal{R},\mathcal{S}})^{-1}) \sum_{i=1}^{|\mathcal{R}|} \lambda_i y_i^2 \\
&= \lambda_{\max}((\Pi_{\mathcal{R},\mathcal{S}}^T \Pi_{\mathcal{R},\mathcal{S}})^{-1}) \mathbf{v}^T \Pi_{\mathcal{R},\mathcal{S}} (\Pi_{\mathcal{R},\mathcal{S}}^T \Pi_{\mathcal{R},\mathcal{S}})^{-1} \Pi_{\mathcal{R},\mathcal{S}}^T \mathbf{v}.
\end{aligned}$$

Using that $\lambda_{\max}((\Pi_{\mathcal{R},\mathcal{S}}^T\Pi_{\mathcal{R},\mathcal{S}})^{-1}) = \lambda_{\min}^{-1}(\Pi_{\mathcal{R},\mathcal{S}}^T\Pi_{\mathcal{R},\mathcal{S}})$, and because the previous computation holds for any vector $\mathbf{v}$ we have:

$$\sum_{i=1}^{|\mathcal{R}|} P_{\mathrm{Res}(\boldsymbol{\pi}_i,\hat{\mathcal{R}}-\boldsymbol{\pi}_i)} \preceq \lambda_{\min}^{-1}(\Pi_{\mathcal{R},\mathcal{S}}^T\Pi_{\mathcal{R},\mathcal{S}})P_{\hat{\mathcal{R}}}.$$

Using monotonicity of the trace operator for semidefinite matrices under the Lowner ordering, we have that:

$$\begin{aligned}\sum_{i=1}^{|\mathcal{R}|} f(\mathbf{a}_i \mid \mathcal{S}\cup\mathcal{R}_{-i}) = \mathrm{tr}\left(A^T\left[\sum_{i=1}^{|\mathcal{R}|} P_{\mathrm{Res}(\boldsymbol{\pi}_i,\hat{\mathcal{R}}-\boldsymbol{\pi}_i)}\right]A\right) &\leq \lambda_{\min}^{-1}(\Pi_{\mathcal{R},\mathcal{S}}^T\Pi_{\mathcal{R},\mathcal{S}})\mathrm{tr}\big(A^TP_{\hat{\mathcal{R}}}A\big)\\ &\leq \lambda_{\min}^{-1}(\Pi_{\mathcal{R},\mathcal{S}}^T\Pi_{\mathcal{R},\mathcal{S}})f(\mathcal{R}\mid\mathcal{S}).\end{aligned}$$

To conclude the proof, we note that we have show that $\gamma = 1/\beta \geq \lambda_{\min}^{-1}(\Pi_{\mathcal{R},\mathcal{S}}^T\Pi_{\mathcal{R},\mathcal{S}})$. We point out that $\Pi_{\mathcal{R},\mathcal{S}}^T\Pi_{\mathcal{R},\mathcal{S}}$ plays the same role has the covariance matrix C with respect to set $\{\mathrm{Res}(X_1,\mathcal{S}),\ldots,\mathrm{Res}(X_n,\mathcal{S})\}$. Applying Lemma 5.2.7 to the matrix $\Pi_{\mathcal{R},\mathcal{S}}^T\Pi_{\mathcal{R},\mathcal{S}} = A_{\mathcal{R}}^TA_{\mathcal{R}} - A_{\mathcal{R}}^TP_{\mathcal{S}}A_{\mathcal{R}}$ shows that $\lambda_{\min}(\Pi_{\mathcal{R},\mathcal{S}}^T\Pi_{\mathcal{R},\mathcal{S}}) \geq \lambda_{\min}(A_{\mathcal{R}\cup\mathcal{S}}^TA_{\mathcal{R}\cup\mathcal{S}}) \geq \lambda_{\min}(A^TA,|\mathcal{R}\cup\mathcal{S}|) = \sigma_{\min}^2(A,|\mathcal{R}\cup\mathcal{S}|)$.

5.8 How large is the upper submodularity ratio

We have shown that the R^2 objective (Section 5.2), the A-optimal design objective (Section 5.6), and the column subset objective (Section 5.7) are $(c,1/c)$-weakly submodular for some parameter c. A natural question to ask is whether, given $\gamma > 0$, there is a small non-trivial bound for β independent of the size of the ground set. Here we show that this is not true in general and prove that

Theorem 5.8.1. *For any $\gamma > 0$ and $k > 0$ there exists a function on a ground set of size k that is not (γ,β)-weakly submodular for any $\beta < \binom{k-\gamma}{k-1} = \Theta\big(k^{1-\gamma}\big)$.*

The intuition behind the construction is simple. We build a set function recursively with lower submodularity ratio exactly equal to γ. The recurrence relation holds until the $(k-1)^{\text{th}}$ marginal, which allows us to have a large value for the final marginal and thus increase β.

Proof of Theorem 5.8.1. We construct a monotone set function f on a ground set of k elements. The elements are indistinguishable, meaning that for any given set S, two elements $e, e' \in X\setminus S$ have the same marginal contribution. Therefore, because elements are indistinguishable, the value of a set is a function of its size. Let x_i be the value of any set of size $i = 0,1,\ldots,k$. Additionally, let $x_0 = f(\emptyset) = 0$ and $x_k = 1$. We define x_i inductively with the following recurrence for $i = 0,1,\ldots,k-2$:

$$x_{i+1} = \frac{k-i-\gamma}{k-i}\cdot x_i + \frac{\gamma}{k-i} \quad \text{or equivalently} \quad x_{i+1} - x_i = \frac{\gamma}{k-i}(1-x_i). \tag{5.31}$$

It can easily be shown (by induction) that the described sequence is valid, i.e. it is monotone and each $x_i \in [0,1]$. Additionally, we note that the sequence satisfies:

$$1 - x_{i+1} = 1 - \left(\frac{k-i-\gamma}{k-i} \cdot x_i + \frac{\gamma}{k-i}\right) = \left(1 - \frac{\gamma}{k-i}\right)(1-x_i)\,, \tag{5.32}$$

for all $i = 0, 1, \ldots, k-2$. First, we show that f has a lower submodularity ratio at most γ. We prove that for any B and $A \subset B$ such that $|B| = j$ and $|A| = i$:

$$\frac{\sum_{e \in B \setminus A} f(e|A)}{f(B) - f(A)} = \frac{(j-i)(x_{i+1} - x_i)}{x_j - x_i} \geq \gamma. \tag{5.33}$$

First, we consider the case in which $j = k$. If $i = k-1$, then the left-hand side of (5.33) is 1. If $i \leq k-2$, then applying the identity (5.31), and recalling that $x_k = 1$ gives:

$$\frac{(k-i)(x_{i+1} - x_i)}{x_k - x_i} = \frac{(k-i) \cdot \frac{\gamma}{k-i}(1-x_i)}{1 - x_i} = \gamma,$$

for any $i = 1, \ldots, k-1$. Next, we consider the case in which $j \leq k-1$ and so $i \leq k-2$. Then, by employing recursively the identity (5.32) we obtain

$$x_j - x_i = (1-x_i) - (1-x_j) = (1-x_i)\left(1 - \prod_{\ell=i}^{j-1}\left(1 - \frac{\gamma}{k-\ell}\right)\right). \tag{5.34}$$

Since $\gamma \in [0,1]$, we can use the generalized Bernoulli inequality $(1-1/n)^x \leq 1 - x/n$ for $x \in [0,1]$, and $n \geq 1$ to bound each term in the product above. This gives:

$$\prod_{\ell=i}^{j-1}\left(1 - \frac{\gamma}{k-\ell}\right) \geq \prod_{\ell=i}^{j-1}\left(1 - \frac{1}{k-\ell}\right)^{\gamma} = \left(\frac{k-j}{k-i}\right)^{\gamma} = \left(1 - \frac{j-i}{k-i}\right)^{\gamma} \geq 1 - \left(\frac{j-i}{k-i}\right),$$

where in the last inequality we used the fact that $(1-x)^\gamma \geq 1 - x$ for $x \in [0,1]$ and $\gamma \in [0,1]$. Thus we have:

$$\frac{(j-i)(x_{i+1} - x_i)}{x_j - x_i} = \frac{(j-i)\frac{\gamma}{k-i}(1-x_i)}{(1-x_i)\left(1 - \prod_{\ell=i}^{j-1}\left(1 - \frac{\gamma}{k-\ell}\right)\right)} \geq \frac{(j-i)\frac{\gamma}{k-i}(1-x_i)}{(1-x_i)\frac{j-i}{k-i}} = \gamma,$$

where we have used (5.31) and (5.34) in the first equation. Combining these cases, we find f is γ-weakly submodular from below. To complete the proof we now show that f is not β-weakly submodular from above for any $\beta < \binom{k-\gamma}{k-1}$. Here we consider the case in which $A = \emptyset$ and $B = X$ and show that:

$$\frac{\sum_{e \in B} f(e|B-e)}{f(B) - f(\emptyset)} = \frac{k(x_k - x_{k-1})}{x_k - x_0} = \binom{k-\gamma}{k-1}.$$

Recall that $x_k = 1$, and $x_0 = 0$, which implies that the denominator is equal to 1. Recursively applying the identity (5.32) gives

$$k(1 - x_{k-1}) = k \cdot \prod_{\ell=0}^{k-2}\left(1 - \frac{\gamma}{k-\ell}\right) = k\prod_{\ell=2}^{k}\left(1 - \frac{\gamma}{\ell}\right)$$
$$= \frac{\prod_{\ell=2}^{k}(\ell-\gamma)}{\prod_{\ell=1}^{k-1}\ell} = \frac{\prod_{\ell=1}^{k-1}(\ell+1-\gamma)}{\prod_{\ell=1}^{k-1}\ell} = \binom{k-\gamma}{k-1},$$

as required. □

5.9 Conclusion and Open Questions

In this chapter we gave improved algorithms for sparse subset selection problems subject to a matroid constraint. The applications that we consider include: Sparse Least Square Regression, Bayesian A-Optimal Design, and Matrix Column Selection. Increased performances are obtained by showing that the objective functions are (γ, β)-weakly submodular and by designing improved approximation algorithms for maximizing such functions. The submodularity ratios γ and β measure the deviation of the set function from submodularity. In particular, as γ, β tends to 1 the set function becomes submodular. For sparse subset selections applications considered here, we show that $\beta \leq 1/\gamma$.

We obtain two new algorithmic results for maximizing (γ, β)-weakly submodular functions: a novel analysis of ResidualRandomGreedy, and a distorted local-search algorithm. The asymptotic approximation factors are equal to 2 and $\frac{e}{e-1}$, respectively. The latter being optimal [FNW78; Fei98].

There are many directions of great interest. We list a few of open questions. All problems have stars $\star$ denoting a combination of their difficulty and interest.

- $(\star\star)$ Elenberg et al. [Ele+18] show that *Restricted Strong Concavity implies γ-weak submodularity*. Is there an analogous definition that implies both the upper and lower weak submodularity definition? It could have a high impact for a wide variety of statistical problems and perhaps lead to the development of local-search-type of algorithms for solving them.
- $(\star\star)$ Calinescu et al. [Cal+11] obtained an $e/(e-1)$ approximation algorithm for maximizing a monotone submodular function through a relax-and-round approach. The *multilinear* relaxation that they use is also defined for weak submodular functions. However, rounding the fractional solution is the main bottleneck in extending Calinescu et al.'s result to weakly submodular functions. Is it possible to overcome this obstacle?
- $(\star)$ A somewhat surprising fact is that it is still not clear whether Greedy is a constant factor approximation algorithm for maximizing a weakly submodular function subject to a matroid constraint. Gatmiry et al. [GG18] show that Greedy has guarantee at least $\frac{\sqrt{\gamma}k+1}{0.4\gamma^2}$, where k is the rank of the matroid.

- Due to the wide number of applications that the weakly submodular definition encompasses, there are obvious directions consisting in designing approximation algorithms for (γ, β)-weakly submodular functions subject to general independence systems in various computational environments.

Bibliography

[AG13] Kook Jin Ahn and Sudipto Guha. „Linear programming in the semi-streaming model with application to the maximum matching problem“. In: *Information and Computation* (2013) (cit. on pp. 85, 86).

[AH98] Esther M Arkin and Refael Hassin. „On local search for weighted k-set packing“. In: *Mathematics of Operations Research* 23.3 (1998), pp. 640–648 (cit. on pp. 17, 21, 46).

[Alt+16] Jason M. Altschuler, Aditya Bhaskara, Gang Fu, et al. „Greedy Column Subset Selection: New Bounds and Distributed Algorithms“. In: *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*. 2016 (cit. on pp. 13, 116, 141).

[ALT21] Sepehr Assadi, S Cliff Liu, and Robert E Tarjan. „An auction algorithm for bipartite matching in streaming and massively parallel computation models“. In: *Symposium on Simplicity in Algorithms (SOSA)*. 2021 (cit. on p. 85).

[AS16] Noga Alon and Joel H. Spencer. *The Probabilistic Method*. 4th. Wiley Publishing, 2016 (cit. on pp. 14, 15).

[Ass+22] Sepehr Assadi, Arun Jambulapati, Yujia Jin, Aaron Sidford, and Kevin Tian. „Semi-Streaming Bipartite Matching in Fewer Passes and Optimal Space“. In: *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 2022 (cit. on p. 85).

[Bad+14] Ashwinkumar Badanidiyuru, Baharan Mirzasoleiman, Amin Karbasi, and Andreas Krause. „Streaming submodular maximization: massive data summarization on the fly“. In: *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*. 2014 (cit. on pp. 82, 85, 86, 113).

[Bai+15] Ramakrishna Bairi, Rishabh Iyer, Ganesh Ramakrishnan, and Jeff Bilmes. „Summarization of multi-document topic hierarchies using submodular mixtures“. In: *Proc. 53rd Annual Meeting of the Association for Computational Linguistics and 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 2015, pp. 553–563 (cit. on p. 12).

[Ber00] Piotr Berman. „A d/2 approximation for maximum weight independent set in d-claw free graphs“. In: *Scandinavian Workshop on Algorithm Theory*. Springer. 2000, pp. 214–219 (cit. on pp. 4, 17, 19–25, 45).

[BF19] Niv Buchbinder and Moran Feldman. „Constrained submodular maximization via a non-symmetric technique“. In: *Mathematics of Operations Research* 44.3 (2019), pp. 988–1005 (cit. on pp. 14, 101).

[BF22] Kobi Bodek and Moran Feldman. „Maximizing Sums of Non-Monotone Submodular and Linear Functions: Understanding the Unconstrained Case“. In: *30th Annual European Symposium on Algorithms, ESA 2022, September 5-9, 2022, Berlin/Potsdam, Germany*. 2022 (cit. on p. 102).

[Bia+17] Andrew An Bian, Joachim M Buhmann, Andreas Krause, and Sebastian Tschiatschek. „Guarantees for greedy maximization of non-submodular functions with applications“. In: *Proc. 34th ICML*. 2017, pp. 498–507 (cit. on pp. 12, 116, 118, 139).

[BK03a] Piotr Berman and Marek Karpinski. „Improved Approximation Lower Bounds on Small Occurrence Optimization“. In: *ECCC* (2003) (cit. on pp. 3, 17).

[BK03b] Piotr Berman and Piotr Krysta. „Optimizing misdirection“. In: *SODA*. 2003, pp. 192–201 (cit. on pp. 19, 21).

[Buc+14] Niv Buchbinder, Moran Feldman, Joseph Naor, and Roy Schwartz. „Submodular Maximization with Cardinality Constraints“. In: *Proc. ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 2014, pp. 1433–1452 (cit. on pp. 99, 122).

[BZC18] Ilija Bogunovic, Junyao Zhao, and Volkan Cevher. „Robust Maximization of Non-Submodular Objectives“. In: *Proc. 21st AISTATS*. 2018, pp. 890–899 (cit. on pp. 12, 118).

[Cal+11] Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. „Maximizing a Monotone Submodular Function Subject to a Matroid Constraint“. In: *SIAM Journal on Computing* 40.6 (2011), pp. 1740–1766 (cit. on pp. 10, 15, 115, 149).

[CFK18] Lin Chen, Moran Feldman, and Amin Karbasi. „Weakly submodular maximization beyond cardinality constraints: Does randomization help greedy?“ In: *Proc. 35th ICML*. 2018, pp. 804–813 (cit. on pp. 12, 13, 115–118, 122, 130, 136).

[CGM13] Marek Cygan, Fabrizio Grandoni, and Monaldo Mastrolilli. „How to sell hyperedges: The hypermatching assignment problem“. In: *SODA*. 2013, pp. 342–351 (cit. on pp. 3, 21).

[CGQ15] Chandra Chekuri, Shalmoli Gupta, and Kent Quanrud. „Streaming Algorithms for Submodular Function Maximization“. In: *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*. 2015 (cit. on pp. 11, 83, 85–88, 93, 109, 110).

[CH01] Barun Chandra and Magnús M Halldórsson. „Greedy local improvement and weighted set packing approximation“. In: *Journal of Algorithms* 39.2 (2001), pp. 223–240 (cit. on p. 21).

[CK15] Amit Chakrabarti and Sagar Kale. „Submodular maximization meets streaming: matchings, matroids, and more“. In: *Mathematical Programming* 154.1-2 (2015), pp. 225–247 (cit. on pp. 11, 83–88).

[CL12] Yuk Hei Chan and Lap Chi Lau. „On linear and semidefinite programming relaxations for hypergraph matching“. In: *Mathematical programming* 135.1 (2012), pp. 123–148 (cit. on p. 22).

[CS14] Michael Crouch and Daniel M Stubbs. „Improved streaming algorithms for weighted matching, via unweighted matching“. In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2014)*. 2014 (cit. on pp. 85, 86).

[CV95] Kathryn Chaloner and Isabella Verdinelli. „Bayesian experimental design: A review“. In: *Statistical science* (1995), pp. 273–304 (cit. on p. 139).

[CVZ14] Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. „Submodular Function Maximization via the Multilinear Relaxation and Contention Resolution Schemes“. In: *SIAM Journal on Computing* 43.6 (2014), pp. 1831–1879 (cit. on pp. 15, 101, 102).

[Cyg13] Marek Cygan. „Improved approximation for 3-dimensional matching via bounded pathwidth local search“. In: *SODA*. 2013, pp. 509–518 (cit. on pp. 3, 14, 17, 19, 21, 81).

[DK08] Abhimanyu Das and David Kempe. „Algorithms for subset selection in linear regression“. In: *Proc. 40th STOC*. 2008, pp. 45–54 (cit. on pp. 12, 13, 114).

[DK11] Abhimanyu Das and David Kempe. „Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection“. In: *Proc. 28th ICML*. 2011 (cit. on pp. 12, 13, 114).

[DK18] Abhimanyu Das and David Kempe. „Approximate Submodularity and its Applications: Subset Selection, Sparse Approximation and Dictionary Selection“. In: *J. Mach. Learn. Res.* 19 (2018), 3:1–3:34 (cit. on pp. 116, 117, 119, 122).

[Edm65] Jack Edmonds. „Paths, trees, and flowers“. In: *Canadian Journal of mathematics* 17 (1965), pp. 449–467 (cit. on pp. 3, 14, 17).

[Edm71] Jack Edmonds. „Matroids and the greedy algorithm“. In: *Mathematical programming* 1 (1971), pp. 127–136 (cit. on p. 6).

[Ele+17] Ethan R Elenberg, Alexandros G Dimakis, Moran Feldman, and Amin Karbasi. „Streaming Weak Submodularity: Interpreting Neural Networks on the Fly“. In: *Proc. 31st NeurIPS*. 2017, pp. 4044–4054 (cit. on pp. 12, 85, 118).

[Ele+18] Ethan R Elenberg, Rajiv Khanna, Alexandros G Dimakis, Sahand Negahban, et al. „Restricted strong convexity implies weak submodularity“. In: *The Annals of Statistics* 46.6B (2018), pp. 3539–3568 (cit. on pp. 118, 149).

[Far+15] Ahmed K. Farahat, Ahmed Elgohary, Ali Ghodsi, and Mohamed S. Kamel. „Greedy column subset selection for large-scale data sets“. In: *Knowl. Inf. Syst.* 45.1 (2015), pp. 1–34 (cit. on pp. 13, 116, 141, 142).

[Fei+05] Joan Feigenbaum, Sampath Kannan, Andrew McGregor, Siddharth Suri, and Jian Zhang. „On graph problems in a semi-streaming model“. In: *Theoretical Computer Science* (2005) (cit. on p. 9).

[Fei98] Uriel Feige. „A threshold of $\ln n$ for approximating set cover“. In: *Journal of the ACM* 45.4 (1998), pp. 634–652 (cit. on pp. 15, 115, 149).

[Fel+11] Moran Feldman, Joseph (Seffi) Naor, Roy Schwartz, and Justin Ward. „Improved approximations for k-exchange systems“. In: *Proc. European Symposium on Algorithms (ESA)*. 2011, pp. 784–798 (cit. on pp. 82, 101).

[Fel+20] Moran Feldman, Ashkan Norouzi-Fard, Ola Svensson, and Rico Zenklusen. „The One-way Communication Complexity of Submodular Maximization with Applications to Streaming and Robustness“. In: *Proc. ACM Symposium on Theory of Computing (STOC)*. 2020, pp. 1363–1374 (cit. on pp. 85, 86, 108).

[Fel+22] Moran Feldman, Paul Liu, Ashkan Norouzi-Fard, Ola Svensson, and Rico Zenklusen. „Streaming Submodular Maximization Under Matroid Constraints“. In: *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*. 2022 (cit. on pp. 85, 86, 108).

[FMU22] Manuela Fischer, Slobodan Mitrović, and Jara Uitto. „Deterministic (1+ epsilon)-approximate maximum matching with poly (1/epsilon) passes in the semi-streaming model and beyond“. In: *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*. 2022 (cit. on p. 85).

[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 (cit. on p. 99).

[FNW78] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. „An Analysis of Approximations for Maximizing Submodular Set Functions II“. In: *Mathematical Programming Study* (1978) (cit. on pp. 8, 82, 149).

[FW14] Yuval Filmus and Justin Ward. „A Tight Combinatorial Algorithm for Submodular Maximization Subject to a Matroid Constraint“. In: *SIAM J. Computing* 43.2 (2014), pp. 514–542 (cit. on pp. 10, 13, 15, 115, 117, 125, 131, 137).

[FY14] Martin Fürer and Huiwen Yu. „Approximating the k-set packing problem by local improvements“. In: *ISCO*. 2014, pp. 408–420 (cit. on pp. 17, 21).

[GG18] Khashayar Gatmiry and Manuel Gomez-Rodriguez. „Non-submodular function maximization subject to a matroid constraint, with applications“. In: *arXiv preprint arXiv:1811.07863* (2018) (cit. on p. 149).

[GJS22] Paritosh Garg, Linus Jordan, and Ola Svensson. „Semi-streaming algorithms for submodular matroid intersection“. In: *Mathematical Programming* (2022) (cit. on p. 85).

[Gon+19] Suning Gong, Qingqin Nong, Wenjing Liu, and Qizhi Fang. „Parametric monotone function maximization with matroid constraints“. In: *J. Global Optimization* 75.3 (2019), pp. 833–849 (cit. on p. 118).

[GV11] Shayan Oveis Gharan and Jan Vondrák. „Submodular maximization by simulated annealing“. In: *Proc. ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 2011, pp. 1098–1116 (cit. on p. 14).

[Hal95] Magnús M. Halldórsson. „Approximating Discrete Collections via Local Improvements“. In: *SODA*. 1995, pp. 160–169 (cit. on pp. 3, 21).

[Har+19] Chris Harshaw, Moran Feldman, Justin Ward, and Amin Karbasi. „Submodular Maximization beyond Non-negativity: Guarantees, Fast Algorithms, and Applications“. In: *Proc. 36th ICML*. Vol. 97. 2019, pp. 2634–2643 (cit. on pp. 12, 13, 84, 102, 103, 106, 116, 118, 139, 140).

[Has+19] Abolfazl Hashemi, Mahsa Ghasemi, Haris Vikalo, and Ufuk Topcu. „Submodular observation selection and information gathering for quadratic models“. In: *Proc. 36th ICML*. 2019, pp. 2653–2662 (cit. on p. 116).

[Has+20] Abolfazl Hashemi, Mahsa Ghasemi, Haris Vikalo, and Ufuk Topcu. „Randomized greedy sensor selection: Leveraging weak submodularity“. In: *IEEE Trans. on Automatic Control* 66.1 (2020), pp. 199–212 (cit. on p. 118).

[HS21] Chien-Chung Huang and François Sellier. „Semi-Streaming Algorithms for Submodular Function Maximization Under b-Matching, Matroid, and Matchoid Constraints“. In: *arXiv preprint arXiv:2107.13071* (2021) (cit. on p. 85).

[HS89] Cor A. J. Hurkens and Alexander Schrijver. „On the size of systems of sets every t of which have an SDR, with an application to the worst-case ratio of heuristics for packing problems“. In: *SIAM Journal on Discrete Mathematics* 2.1 (1989), pp. 68–72 (cit. on pp. 3, 17, 21).

[HSS06] Elad Hazan, Shmuel Safra, and Oded Schwartz. „On the complexity of approximating k-set packing“. In: *Computational Complexity* 15.1 (2006), pp. 20–39 (cit. on pp. 3, 14, 15, 17, 81).

[HTW20] Chien-Chung Huang, Theophile Thiery, and Justin Ward. „Improved Multi-Pass Streaming Algorithms for Submodular Maximization with Matroid Constraints“. In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX-/RANDOM 2020, August 17-19, 2020, Virtual Conference*. 2020 (cit. on pp. iv, 11, 82, 86, 102).

[Jen75] TA Jenkyns. „Matchoids: a generalization of matchings and matroids.“ In: (1975) (cit. on pp. 5, 6).

[Kap21] Michael Kapralov. „Space lower bounds for approximating maximum matching in the edge arrival model“. In: *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM. 2021, pp. 1874–1893 (cit. on pp. 85, 86).

[Kar72] Richard M. Karp. „Reducibility Among Combinatorial Problems“. In: *Complexity of Computer Computations*. The IBM Research Symposia Series. Plenum Press, New York, 1972, pp. 85–103 (cit. on pp. 2, 17).

[Kaz+19] Ehsan Kazemi, Marko Mitrovic, Morteza Zadimoghaddam, Silvio Lattanzi, and Amin Karbasi. „Submodular Streaming in All Its Glory: Tight Approximation, Minimum Memory and Low Adaptive Complexity“. In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*. 2019 (cit. on p. 85).

[Kaz+21] Ehsan Kazemi, Shervin Minaee, Moran Feldman, and Amin Karbasi. „Regularized Submodular Maximization at Scale“. In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. 2021 (cit. on pp. 84, 102–104, 106).

[KH78] Bernhard Korte and Dirk Hausmann. „An analysis of the greedy heuristic for independence systems“. In: *Annals of Discrete Mathematics*. 1978 (cit. on p. 14).

[Kha+17a] Rajiv Khanna, Ethan Elenberg, Alexandros G. Dimakis, and Sahand Negahban. „On Approximation Guarantees for Greedy Low Rank Optimization“. In: *Proc. 34th ICML*. 2017, pp. 1837–1846 (cit. on p. 118).

[Kha+17b] Rajiv Khanna, Ethan Elenberg, Alexandros G Dimakis, Sahand Negahban, and Joydeep Ghosh. „Scalable greedy feature selection via weak submodularity“. In: *Proc. 20th AISTATS*. 2017, pp. 1560–1568 (cit. on p. 118).

[KKT03] David Kempe, Jon Kleinberg, and Éva Tardos. „Maximizing the spread of influence through a social network“. In: *Proc. 9th KDD*. 2003, pp. 137–146 (cit. on p. 7).

[KSG08] Andreas Krause, Ajit Paul Singh, and Carlos Guestrin. „Near-Optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies“. In: *J. Machine Learning Research* 9 (2008), pp. 235–284 (cit. on pp. 13, 139, 140).

[Kuh+18] Alan Kuhnle, J. David Smith, Victoria G. Crawford, and My T. Thai. „Fast Maximization of Non-Submodular, Monotonic Functions on the Integer Lattice“. In: *Proc. 35th ICML*. 2018, pp. 2791–2800 (cit. on pp. 12, 118).

[Kuh55] Harold W Kuhn. „The Hungarian method for the assignment problem“. In: *Naval research logistics quarterly* 2.1-2 (1955), pp. 83–97 (cit. on p. 3).

[LB10] Hui Lin and Jeff A. Bilmes. „Multi-document Summarization via Budgeted Maximization of Submodular Functions“. In: *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 2-4, 2010, Los Angeles, California, USA*. The Association for Computational Linguistics, 2010, pp. 912–920 (cit. on pp. 7, 12, 82).

[Liu+21] Paul Liu, Aviad Rubinstein, Jan Vondrák, and Junyao Zhao. „Cardinality constrained submodular maximization for random streams“. In: *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*. 2021 (cit. on p. 108).

[LSV10] Jon Lee, Maxim Sviridenko, and Jan Vondrák. „Submodular Maximization over Multiple Matroids via Generalized Exchange Properties.“ In: *Mathematics of Operations Research* 35.4 (2010), pp. 795–806 (cit. on pp. 10, 14, 15, 82, 86, 101, 108).

[LSV13] Jon Lee, Maxim Sviridenko, and Jan Vondrák. „Matroid Matching: The Power of Local Search“. In: *SIAM J. Comput.* (2013) (cit. on p. 14).

[LW20] Roie Levin and David Wajc. „Streaming Submodular Matching Meets the Primal-Dual Method“. In: *arXiv preprint arXiv:2008.10062* (2020) (cit. on pp. 85, 86).

[MBK16] Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, and Amin Karbasi. „Fast Constrained Submodular Maximization: Personalized Data Summarization.“ In: *ICML*. 2016, pp. 1358–1367 (cit. on p. 82).

[Mir+15] Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, Amin Karbasi, Jan Vondrák, and Andreas Krause. „Lazier Than Lazy Greedy“. In: *Proc. AAAI Conference on Artificial Intelligence (AAAI)*. 2015, pp. 1812–1818 (cit. on p. 82).

[Neu21] Meike Neuwohner. „An Improved Approximation Algorithm for the Maximum Weight Independent Set Problem in d-Claw Free Graphs“. In: *STACS*. Vol. 187. 2021, 53:1–53:20 (cit. on pp. 4, 18, 19, 50, 80).

[Neu22] Meike Neuwohner. „The Limits of Local Search for Weighted k-Set Packing“. In: *IPCO*. Springer. 2022, pp. 415–428 (cit. on pp. 4, 5, 18–20, 50, 74, 75, 80).

[Neu23] Meike Neuwohner. „Passing the Limits of Pure Local Search for Weighted k-Set Packing“. In: *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*. SIAM, 2023 (cit. on pp. 4, 14, 18, 19, 75, 80, 81).

[Non+19] Qingqin Nong, Tao Sun, Suning Gong, et al. „Maximize a monotone function with a generic submodularity ratio“. In: *Proc. International Conference on Algorithmic Applications in Management*. 2019, pp. 249–260 (cit. on p. 118).

[Nor+18] Ashkan Norouzi-Fard, Jakub Tarnawski, Slobodan Mitrović, et al. „Beyond 1/2-Approximation for Submodular Maximization on Massive Data Streams“. In: *Proc. International Conference on Machine Learning (ICML)*. 2018, pp. 3826–3835 (cit. on pp. 85, 86, 108).

[NW78] G L Nemhauser and L A Wolsey. „Best Algorithms for Approximating the Maximum of a Submodular Set Function“. In: *Mathematics of Operations Research* 3.3 (1978), pp. 177–188 (cit. on p. 115).

[NWF78] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. „An analysis of approximations for maximizing submodular set functions—I“. In: *Mathematical programming* (1978) (cit. on p. 8).

[PS18] Ami Paz and Gregory Schwartzman. „A (2+ ε)-approximation for maximum weight matching in the semi-streaming model“. In: *ACM Transactions on Algorithms (TALG)* (2018) (cit. on pp. 85, 86).

[Qia+18] Chao Qian, Yibo Zhang, Ke Tang, and Xin Yao. „On Multiset Selection With Size Constraints“. In: *Proc. 32nd AAAI*. 2018, pp. 1395–1402 (cit. on pp. 12, 118).

[RW05] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2005 (cit. on p. 14).

[Sch+03] Alexander Schrijver et al. *Combinatorial optimization: polyhedra and efficiency*. Vol. 24. Springer, 2003 (cit. on pp. 6, 14, 16).

[ST10] Mohit Singh and Kunal Talwar. „Improving integrality gaps via Chvátal-Gomory rounding“. In: *APPROX*. 2010, pp. 366–379 (cit. on p. 22).

[Ste04] J Michael Steele. *The Cauchy-Schwarz master class: an introduction to the art of mathematical inequalities*. Cambridge University Press, 2004 (cit. on pp. 14, 15).

[SVW15] Maxim Sviridenko, Jan Vondrák, and Justin Ward. „Optimal approximation for submodular and supermodular optimization with bounded curvature“. In: *Proc. 26th SODA*. 2015, pp. 1134–1148 (cit. on pp. 102, 106).

[SW13] Maxim Sviridenko and Justin Ward. „Large neighborhood local search for the maximum set packing problem“. In: *ICALP*. Springer. 2013, pp. 792–803 (cit. on pp. 3, 21).

[SY20] Richard Santiago and Yuichi Yoshida. „Weakly Submodular Function Maximization Using Local Submodularity Ratio“. In: *arXiv preprint arXiv:2004.14650* (2020) (cit. on p. 12).

[TW22] Theophile Thiery and Justin Ward. „Two-Sided Weak Submodularity for Matroid Constrained Optimization and Regression“. In: *Conference on Learning Theory, 2-5 July 2022, London, UK*. Proceedings of Machine Learning Research. 2022 (cit. on pp. iv, 13, 114).

[TW23] Theophile Thiery and Justin Ward. „An Improved Approximation for Maximum Weighted k-Set Packing“. In: *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*. 2023 (cit. on pp. iv, 5, 14, 17, 19).

[War12a] Justin Ward. „A (k+ 3)/2-approximation algorithm for monotone submodular k-set packing and general k-exchange systems“. In: *STACS'12 (29th Symposium on Theoretical Aspects of Computer Science)*. Vol. 14. LIPIcs. 2012, pp. 42–53 (cit. on p. 15).

[War12b] Justin Ward. *Oblivious and non-oblivious local search for combinatorial optimization*. University of Toronto (Canada), 2012 (cit. on p. 6).

List of Figures

List of Tables