Independent Sets in Elimination Graphs with a Submodular Objective

Chandra Chekuri

Department of Computer Science, University of Illinois at Urbana-Champaign, IL, USA

Kent Quanrud

Department of Computer Science, Purdue University, West Lafayette, IN, USA

Abstract

Maximum weight independent set (MWIS) admits a $\frac{1}{k}$-approximation in inductively k-independent graphs [2, 40] and a $\frac{1}{2k}$-approximation in k-perfectly orientable graphs [34]. These are a parameterized class of graphs that generalize k-degenerate graphs, chordal graphs, and intersection graphs of various geometric shapes such as intervals, pseudo-disks, and several others [40, 34]. We consider a generalization of MWIS to a submodular objective. Given a graph $G = (V, E)$ and a non-negative submodular function $f : 2^V \to \mathbb{R}_+$, the goal is to approximately solve $\max_{S \in \mathcal{I}_G} f(S)$ where $\mathcal{I}_G$ is the set of independent sets of G. We obtain an $\Omega(\frac{1}{k})$-approximation for this problem in the two mentioned graph classes. The first approach is via the multilinear relaxation framework and a simple contention resolution scheme, and this results in a randomized algorithm with approximation ratio at least $\frac{1}{e(k+1)}$. This approach also yields parallel (or low-adaptivity) approximations.

Motivated by the goal of designing efficient and deterministic algorithms, we describe two other algorithms for inductively k-independent graphs that are inspired by work on streaming algorithms: a preemptive greedy algorithm and a primal-dual algorithm. In addition to being simpler and faster, these algorithms, in the monotone submodular case, yield the first deterministic constant factor approximations for various special cases that have been previously considered such as intersection graphs of intervals, disks and pseudo-disks.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Theory of computation → Submodular optimization and polymatroids

Keywords and phrases elimination graphs, independent set, submodular maximization, primal-dual

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2023.24

Category APPROX

Related Version *Full Version*: https://arxiv.org/abs/2307.02022

Funding *Chandra Chekuri*: Supported in part by NSF grants CCF-1910149 and CCF-1907937.
Kent Quanrud: Supported in part by NSF grant CCF-2129816.

1 Introduction

Given a graph $G = (V, E)$ a set $S \subseteq V$ of vertices is an independent set (also referred to as a stable set) if there is no edge between any two vertices in S. Let $\alpha(G)$ denote the cardinality of a maximum independent set in G. Finding $\alpha(G)$ is a classical problem with many applications; we refer to the search problem of finding a maximum cardinality independent set as MIS. We also consider the weighted version where the input consists of G and a vertex weight function $w : V \to \mathbb{Z}_+$ and the goal is to find a maximum weight independent set; we refer to the weighted problem as MWIS. MIS is NP-Hard, and moreover it is also NP-Hard to approximate $\alpha(G)$ to within a $\frac{1}{n^{1-\epsilon}}$-factor for any fixed $\epsilon > 0$ [32, 41]. For this reason, MIS and MWIS are studied in various special classes of graphs that capture interesting problems while also being tractable. It is easy to see that graphs with maximum degree k admit a $\frac{1}{k}$-approximation. In fact, the same approximation ratio holds for k-degenerate graphs – a graph $G = (V, E)$ is a k-degenerate if there is an ordering of the vertices $\mathcal{V} = \{v_1, \dots, v_n\}$ such that for each v_i, $|N(v_i) \cap \{v_i, \dots, v_n\}| \le k$. A canonical example is the class of planar graphs which are 5-degenerate.

In this paper we are interested in two parameterized classes of graphs called inductively k-independent graphs [40] and k-perfectly orientable graphs [34]. These graphs are motivated by the well-known class of chordal graphs, and capture several other interesting classes such as intersection graphs of intervals, disks (and hence planar graphs), low-treewidth graphs, t-interval graphs, and many others. A more recent example is the intersection graph of a collection of pseudo-disks which were shown to be inductively 156-independent [38]. Graphs in these classes can be dense and have large cliques. We formally define the classes.

Given a graph $G = (V, E)$ and a vertex v we let $N(v)$ denote the set of neighbors of v (excluding v). A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with n vertices has a *perfect elimination ordering* if there is an ordering of vertices $\mathcal{V} = \{v_1, \dots, v_n\}$ such that for each v_i, $\alpha(G[N(v_i) \cap \{v_i, \dots, v_n\}]) = 1$; in other words $N(v_i) \cap \{v_i, \dots, v_n\}$ is a clique. It is well-known that these graphs are the same as chordal graphs.[1] For example, the intersection graph of a given set of intervals is chordal. One can generalize the perfect elimination property ordering of chordal graphs.

▶ **Definition 1** ([34]). *For a fixed integer $k \ge 1$, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is* k-simplicial *if there is an ordering of vertices $\mathcal{V} = \{v_1, \dots, v_n\}$ such that for each v_i, $G[N(v_i) \cap \{v_i, \dots, v_n\}]$ can be covered by k cliques.*

Note that if $G[N(v_i) \cap \{v_i, \dots, v_n\}]$ is covered by k cliques then $\alpha(G[N(v_i) \cap \{v_i, \dots, v_n\}]) \le k$. Hence one can define a class based on this weaker property.

▶ **Definition 2** ([2, 40]). *For a fixed integer $k \ge 1$, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is* inductively k-independent *if there is an ordering of vertices $\mathcal{V} = \{v_1, \dots, v_n\}$ such that for each v_i, $\alpha(G[N(v_i) \cap \{v_i, \dots, v_n\}]) \le k$. The* inductive independence number *of $\mathcal{G}$ is the minimum k for which $\mathcal{G}$ is inductively k-independent.*

Although inductively k-independent graphs generalize k-simplicial graphs there is no known natural class of graphs that differentiates the two; typically one establishes inductive k-independence via k-simpliciality. The ordering-based definition can be further relaxed based on orientations of G.

▶ **Definition 3** ([34]). *For a fixed integer $k \ge 1$, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is* k-perfectly orientable *if there is an orientation $H = (V, A)$ of $\mathcal{G}$ such that for each vertex $v \in \mathcal{V}$, $G[S_v]$ can be covered by k cliques, where $S_v = N_H^+(v)$ is the out-neighborhood of v in H.*

▶ Remark 4. In this paper we will use the term k-perfectly orientable for the following class of graphs: there is an orientation $H = (V, A)$ of $\mathcal{G}$ such that for each vertex $v \in \mathcal{V}$, $\alpha(G[S_v]) \le k$ where $S_v = N_H^+(v)$ is the out-neighborhood of v in H. This is more general than the preceding definition. We observe that the algorithm in [34] for MWIS works also for this larger class, although there are no known natural examples that differentiate the two.

We observe that if G is inductively k-independent then it is also k-perfectly orientable according to our relaxed definition. Indeed, if $v_1, v_2, \dots, v_n$ is an ordering that certifies inductive k-independence we simply orient the edges of G according to this ordering which yields a DAG. The advantage of the k-perfect orientability is that it allows arbitrary orientations. Note that a cycle is 1-perfectly orientable while it is 2-inductively independent. This factor

[1] A graph is chordal iff there is no induced cycle of length more than 3.

of 2 gap shows up in the known approximation bounds for MWIS in these two classes of graphs. It is known that for arbitrarily large n there are 2-perfectly orientable graphs on n vertices such that the graphs are not inductively $\sqrt{n}$-independent [5]. These come from the intersection graphs of so-called 2-interval graphs. Thus, k-perfect orientability can add substantial modeling power.

Akcoglu et al. [2] described a $\frac{1}{k}$-approximation for the MWIS problem in graphs that are inductively k-independent. They used the local-ratio technique, and subsequently [40] derived it using a stack-based algorithm. Both algorithms require as input an ordering of the vertices that certifies the inductive k-independent property. For k-perfectly orientable graphs [34] described a $\frac{1}{2k}$-approximation for the MWIS problem following the ideas in [5] for a special case. Given a graph $G = (V, E)$ and integer k there is an $n^{O(k)}$-time algorithm to check if G is inductively k-independent [40]. Typically, the proof that a specific class of graphs is inductively k-independent for some fixed value of k, yields an efficient algorithm that also computes a corresponding ordering. This is also true for k-perfect orientability. We refer the reader to [30] for additional discussion on computational aspects of computing orderings. In this paper we will assume that we are given both G and the ordering that certifies inductive k-independence, or an orientation that certifies k-perfect orientability.

1.1 Independent sets with a submodular objective

We consider an extension of MWIS to submodular objectives. A real-valued set function $f : 2^V \to \mathbb{R}$ is modular iff $f(A)+f(B) = f(A\cup B)+f(A\cap B)$ for all $A, B \subseteq V$. It is easy to show that f is modular iff there a weight function $w : V \to \mathbb{R}$ where $f(A) = w(A) = \sum_{v\in A} w(v)$. A real-valued set function $f : 2^V \to \mathbb{R}$ is *submodular* if $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$ for all $A, B \subseteq V$. An equivalent definition is via decreasing marginal value property: for any $A \subset B \subset V$ and $v \in V - B$, $f(A + v) - f(A) \geq f(B + v) - f(B)$. Here $A + v$ is a convenient notation for $A \cup \{v\}$. f is monotone if $f(A) \leq f(B)$ for all $A \subseteq B$. We will confine our attention in this paper to non-negative submodular functions and we will also assume that $f(\emptyset) = 0$. Given a graph $G = (V, E)$ and a non-negative submodular function $f : 2^V \to \mathbb{R}_+$, we consider the problem $\max_{S\subseteq \mathcal{I}_G} f(S)$ where $\mathcal{I}_G$ is the collection of independent sets in G. This problem generalizes MWIS since a modular function is also submodular. We assume throughout that f is available through a value oracle that returns $f(S)$ on query S. Our focus is on developing approximation algorithms for this problem in the preceding graph classes, since even very simple special cases are NP-Hard.

Motivation and related work. Submodular function maximization subject to various "independence" constraints has been a very active area of research in the last two decades. There have been several important theoretical developments, and a variety of applications ranging from algorithmic game theory, machine learning and artificial intelligence, data analysis, and network analysis; see [10, 6, 22] for some pointers. We are motivated to consider this objective in inductive k-independent graphs and k-perfectly orientable graphs for several reasons. First, it is a natural generalization of MWIS. Second, various special cases of this problem have been previously studied: Feldman [24] considered the case of interval graphs, and Chan and Har-Peled considered the case of intersection graphs of disks and pseudo-disks [16]. Third, previous algorithms have relied on the multilinear relaxation based approach combined with contention resolution schemes for rounding. This is a computationally expensive approach and also requires randomization. The known approximation algorithms for MWIS in inductive k-independent graphs are based on simple combinatorial methods such as local-ratio, and this raises the question of developing similar combinatorial algorithms for submodular objectives. In particular, we are inspired by the connection to *preemptive* greedy algorithms for submodular function maximization that have been developed in the context of streaming algorithms [15, 3, 17]. Although a natural greedy algorithm has been extensively studied for submodular function maximization [36, 28], the utility of the preemptive version for offline approximation has not been explored as far as we are aware of. This is partly due to the fact that the standard greedy algorithm works well for matroid like constraints. More recently [35] developed a primal-dual based algorithm for submodular streaming under b-matching constraints which is inspired by the stack based algorithm of [37] for the modular setting; the latter has close connections to stack based algorithms for inductive k-independent graphs [40]. The algorithm in [35] was generalized to matroid intersection in [29]. Finally, at a meta-level, we are also interested in understanding the relationship in approximability between optimizing with modular objectives and submodular objectives. For many "independence" constraints the approximability of the problem with a submodular objective is often within a constant factor of the approximability with a modular objective, but there are also settings in which the submodular objective is provably harder (see [7]). A substantial amount of research on submodular function optimization is for constraints defined by *exchange* systems such as (intersections of) matroids and their generalizations such as k-exchange systems [27] and k-systems [33, 14]. Independent sets in the graph classes we consider provide a different parameterized family of constraints.

1.2 Results

We obtain an $\Omega(\frac{1}{k})$-approximation for $\max_{S\subseteq\mathcal{I}} f(S)$ in inductively k-independent graphs and in k-perfectly orientable graphs. We explore different techniques to achieve these results since they have different algorithmic benefits.

First, we obtain a randomized algorithm via the multilinear relaxation framework [21] by considering a natural polyhedral relaxation and developing simple contention resolution schemes (CRS). The CRS schemes are useful since one can combine the rounding with other side constraints in various applications.

▶ **Theorem 5.** *There is a randomized algorithm that given a k-perfectly orientable graph G (along with its orientation) and a monotone submodular function f, outputs an independent set S' such that with high probability $f(S') \geq (\frac{1}{k+1} \cdot \frac{1}{(1+1/k)^k}) \max_{A\in\mathcal{I}_G} f(A)$. For non-negative functions there is an algorithm that outputs an independent set S' such that with high probability $f(S') \geq \frac{1}{e(k+1)} \max_{A\in\mathcal{I}_G} f(A)$.*

The multilinear relaxation based approach yields parallel (or low-adaptivity) algorithms with essentially similar approximation ratios, following ideas in [20, 23]. Although the multilinear approach is general and powerful, there are two drawbacks; algorithmic complexity and randomization which are inherent to the approach. An interesting question in the submodular maximization literature is whether one can obtain deterministic algorithms via alternate methods, or by derandomizing the multilinear relaxation approach. There have been several results along these lines [9, 11, 31], and several open problems.

Motivated by these considerations we develop simple and efficient approximation algorithms for inductively k-independent graphs. We show that a preemptive greedy algorithm, inspired by the streaming algorithm in [17], yields a deterministic $\Omega(\frac{1}{k})$-approximation when f is monotone. This can be combined with a simple randomized approach when f is non-monotone. Inspired by [35], we describe a primal-dual algorithm that also yields a $\Omega(\frac{1}{k})$-approximation; the primal-dual approach yields better constants and we state the result below.

▶ **Theorem 6.** *There is a deterministic combinatorial algorithm that given an inductively k-independent graph G (along with its orientation) and a monotone submodular function f, outputs an independent set S' such that $f(S') \geq \frac{1}{k+1+2\sqrt{k}} \max_{A \in \mathcal{I}_G} f(A)$. For non-negative functions there is a randomized algorithm that outputs an independent set S' such that $\mathbf{E}[f(S')] \geq \frac{1}{2k+1+\sqrt{8k}} \max_{A \in \mathcal{I}_G} f(A)$. Both algorithms use $O(|V(G)|)$ value oracle calls to f and in addition take linear time in the size of G.*

▶ Remark 7. We obtain deterministic $1/4$-approximation for monotone submodular function maximization for independent sets in chordal graphs, and hence also for interval graphs. This matches the best ratio known via the multilinear relaxation approach [24], and is the first deterministic algorithm as far as we know. Similarly, this is the first deterministic algorithm for disks and pseudo-disks that were previously handled via the multilinear relaxation approach [16]. Are there deterministic algorithms for k-perfectly orientable graphs? See Section 5.

▶ Remark 8. Matchings in a graph G, when viewed as independent sets in the line graph H of G, form an inductively 2-independent graph. In fact *any* ordering of the edges of G forms a valid 2-inductive ordering of H. Thus our algorithm is also a semi-streaming algorithm. Our approximation bound for monotone functions matches the approximation achieved in [35] for matchings although we use a different LP relaxation and view the problem from a more general viewpoint. However, for non-monotone functions, our ratio is slightly weaker, and highlights some differences.

The primal-dual algorithm is a two-phase algorithm. The preemptive greedy algorithm is a single phase algorithm. It gives slightly weaker approximation bounds when compared to the primal-dual algorithm, but has the advantage that it can be viewed as an *online preemptive* algorithm. Algorithms in such a model for submodular maximization were developed in [12, 25]. Streaming algorithms for submodular function maximization in [15, 17] can be viewed as online preemptive algorithms. Our work shows that there is an online preemptive algorithm for independent sets of inductive k-independent graphs if the vertices arrive in the proper order. There are interesting examples where any ordering of the vertices is a valid k-inductive ordering.

Our main contribution in this paper is conceptual. We study the problem to unify and generalize existing results, understand the limits of existing techniques, and raise some directions for future research (see Section 5). As we mentioned, our techniques are inspired by past and recent work on submodular function maximization [21, 24, 17, 35].

Organization

Section 2 sets up the relevant technical background on submodular functions. Section 3 describes the multilinear relaxation approach and proves Theorem 5. Section 4 describes the primal-dual approach and proves Theorem 6. Section 5 concludes with a discussion of some open problems. The description and analysis of the preemptive greedy algorithm can be found in Appendix A.

2 Preliminaries

Let $f : 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ be a real-valued nonnegative set function defined over a finite ground set $\mathcal{N}$. The function f is *monotone* if $f(S) \leq f(T)$ for any nested sets $S \subseteq T \subseteq \mathcal{N}$, and *submodular* if it has decreasing marginal returns: if $S \subseteq T \subseteq \mathcal{N}$ are two nested sets and $e \in \mathcal{N} \setminus T$ is an element, then $f(S+e) - f(S) \geq f(T+e) - f(T)$. For two sets $A, B \subseteq \mathcal{N}$, we denote the marginal value of adding B to A by $f_A(B) \stackrel{\text{def}}{=} f(A \cup B) - f(A)$.

Incremental values

In this paper, there is always an implicit ordering $<$ over the ground set $\mathcal{N}$. For a set $S \subseteq \mathcal{N}$ and an element $e \in \mathcal{N}$, the *incremental value* of e in S, denoted $\nu(f, S, e)$, is defined as

$$\nu(f, S, e) = f_{S'}(e), \text{ where } S' = \{s \in S : s < e\}.$$

Incremental value has some simple but very useful properties, proved in [17, Lemmas 1–3] and summarized in the following.

▶ **Lemma 9.** *Let $\mathcal{N}$ be an ordered set and $f : 2^{\mathcal{N}} \to \mathbb{R}$ a set function.*
(a) *For any set $S \subseteq \mathcal{N}$, we have $f(S) = \sum_{e \in S} \nu(f, S, e)$.*
(b) *Let $S \subseteq T \subseteq \mathcal{N}$ be two nested subsets of $\mathcal{N}$ and $e \in \mathcal{N}$ an element. If f is submodular, then $\nu(f, T, e) \leq \nu(f, S, e)$.*
(c) *Let $S, Z \subseteq \mathcal{N}$ be two sets, and $e \in S$. If f is submodular, then $\nu(f_Z, S, e) \leq \nu(f, Z \cup S, e)$.*

Multilinear Extension and Relaxation

▶ **Definition 10.** *Given a set function $f : 2^{\mathcal{N}} \to \mathbb{R}$, the* multilinear extension *of f, denoted F, extends f to the product space $[0,1]^{\mathcal{N}}$ by interpreting each point $x \in [0,1]^{\mathcal{N}}$ as an independent sample $S \subseteq \mathcal{N}$ with sampling probabilities given by x, and taking the expectation of $f(S)$. Equivalently,*

$$F(x) = \sum_{S \subseteq \mathcal{N}} \left(\prod_{i \in S} x_i \prod_{i \notin S} (1 - x_i) \right).$$

An *independence family* $\mathcal{I}$ over a ground set $\mathcal{N}$ is a subset of $2^{\mathcal{N}}$ that is downward closed, that is, if $A \in \mathcal{I}$ and $B \subset A$ then $B \in \mathcal{I}$. A polyhedral/convex relaxation P for a given independence family $\mathcal{I}$ over $\mathcal{N}$ is a polyhedra/convex subset of $[0,1]^{\mathcal{N}}$ such that for each $A \in \mathcal{I}$, $\chi_A \in P$ where χ_A is the characteristic vector of A (a vector in $\{0,1\}^{\mathcal{N}}$ with a 1 in coordinate i iff $i \in A$). We say that P is a *solvable* relaxation for $\mathcal{I}$ if there is a polynomial time algorithm to optimize a linear objective over P. Given a ground set $\mathcal{N}$, and a non-negative submodular function f over $\mathcal{N}$, and an independence family $\mathcal{I} \subseteq 2^{\mathcal{N}}$,[2] we are interested in the problem $\max_{S \in \mathcal{I}} f(S)$. For this general problem the multilinear relaxation approach is to approximately solve the multilinear relaxation $\max_{x \in P} F(x)$ followed by rounding – see [14, 21, 10]. For monotone f there is a randomized $(1 - 1/e)$-approximation to the multilinear relaxation when P is solvable [14]. For general non-negative functions there is a 0.385-approximation [11].

Concave closure and relaxation

▶ **Definition 11.** *Given a set function $f : 2^{\mathcal{N}} \to \mathbb{R}$, the* concave closure *of f, denoted f^+, extends f to the product space $[0,1]^{\mathcal{N}}$ as follows. For $x \in [0,1]^{\mathcal{N}}$ we let*

$$f^+(x) = \max \left\{ \sum_{S \subseteq \mathcal{N}} \alpha_S f(S) : \sum_{S \ni i} \alpha_S = x_i \textit{ for all } i \in \mathcal{N}, \sum_S \alpha_S = 1, \alpha_S \geq 0 \textit{ for all } S \subseteq N \right\}.$$

[2] We assume that an independence family is specified implicitly via an independence oracle that returns whether a given $A \subseteq \mathcal{N}$ belongs to $\mathcal{I}$.

As the name suggests, f^+ is a concave function over $[0,1]^{\mathcal{N}}$ for any set function f. The definition of $f^+(x)$ involves the solution of an exponential sized linear program. The concave closure of a submodular set function is in general NP-Hard to evaluate. Nevertheless, the concave closure is useful indirectly in several ways. One can relate the concave closure to the multilinear extension via the notion of correlation gap [1, 13, 39, 19]. We can consider a relaxation based on the concave closure for the problem of $\max_{S\in\mathcal{I}} f(S)$, namely, $\max_{x\in P} f^+(x)$ where P is a polyhedral or convex relaxation for the constraint set $\mathcal{I}$. Although we may not be able to solve this relaxation directly, it provides an upper bound on the optimum solution and moreover, unlike the multilinear relaxation, the relaxation can be rewritten as a large linear program when P is polyhedral.

Contention Resolution Schemes

Contention resolution schemes are a way to round fractional solutions for relaxations to packing problems and they are a powerful and useful tool in submodular function maximization [21]. For a polyhedral relaxation P for $\mathcal{I}$ and a real $b \in [0,1]$, bP refers to the polyhedron $\{bx \mid x \in P\}$.

▶ **Definition 12.** *Let $b, c \in [0,1]$. A (b,c)-balanced* CR *scheme π for a polyhedral relaxation P for $\mathcal{I}$ is a procedure that for every $bx \in bP$ and $A \subseteq N$, returns a* random *set $\pi_x(A) \subseteq A \cap support(x)$ and satisfies the following properties:*

(a) *$\pi_x(A) \in \mathcal{I}$ with probability 1 $\quad \forall A \subseteq N, x \in bP$, and*

(b) *for all $i \in support(x)$, $\mathbf{P}[i \in \pi_x(R(x)) \mid i \in R(x)] \geq c \quad \forall x \in bP$.*

The scheme is said to be monotone *if $\mathbf{P}[i \in \pi_x(A_1)] \geq \mathbf{P}[i \in \pi_x(A_2)]$ whenever $i \in A_1 \subseteq A_2$. A $(1,c)$-balanced* CR *scheme is also called a* c-balanced CR scheme. *The scheme is* deterministic *if π is a deterministic algorithm (hence $\pi_x(A)$ is a single set instead of a distribution). It is* oblivious *if π is deterministic and $\pi_x(A) = \pi_y(A)$ for all x, y and A, that is, the output is independent of x and only depends on A. The scheme is* efficiently implementable *if π is a polynomial-time algorithm that given x, A outputs $\pi_x(A)$.*

3 Approximating via Contention Resolution Schemes

Let $G = (V,E)$ be an inductively k-independent graph and let $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ be the corresponding order. Let $\mathcal{I}$ denote the set of independent sets of G. We consider the following simple polyhedral relaxation for $\mathcal{I}$ where there is a variable x_i for each vertex v_i. For notational simplicity we let A_i denote the set $N(v_i) \cap \{v_{i+1}, \ldots, v_n\}$ which is the set of neighbors of v_i that come after v_i in the ordering.

$$x_i + \sum_{v_j \in A_i} x_j \leq k \quad \text{for all } i \in [n]$$

$$x_i \in [0,1] \quad \text{for all } i \in [n]$$

This is a valid polyhedral relaxation for $\mathcal{I}$. Indeed, consider an independent set $S \subseteq V$, and let x be the indicator vector of S. Fix a vertex v_i and consider the first inequality. If $v_i \in S$, then since $A_i \subseteq N(v_i)$, we have $A_i \cap S = \emptyset$, and the left hand side (LHS) is 1. Otherwise $\sum_{v_j \in A_i} x_j = |A_i \cap S| \leq \alpha(A_i) \leq k$, so the LHS is at most k.

In fact, the $\frac{1}{k}$-approximation for MWIS in [2, 40] are implicitly based on this relaxation. Moreover, the relaxation has a polynomial number of constraints and hence is solvable. We refer to this relaxation as Q_G and omit G when clear from the context. The multilinear relaxation is to solve $\max_{x\in Q_G} F(x)$.

Now we consider the case when $G = (V, E)$ is a k-perfectly orientable graph. Let $H = (V, A)$ be an orientation of G. For a given $v \in V$ we let $N_H^+(v) = \{u \in V \mid (v, u) \in A\}$ denote the out-neighbors of v in H. We can write a simple polyhedral relaxation for independent sets in G where we have a variable x_v for each $v \in V$ as follows:

$$x_v + \sum_{u \in N_H^+(v)} x_u \leq k \quad \text{for all } v \in V$$

$$x_v \in [0,1] \quad \text{for all } v \in V$$

To avoid notational overhead we will use Q_G to refer to the preceding relaxation for a k-perfectly orientable graph G. In [34] a stronger relaxation than the preceding relaxation is used to obtain a $\frac{1}{2k}$-approximation for MWIS. It is not hard to see, however, that the proof in [34] can be applied to the simpler relaxation above.

We will only consider k-perfectly orientable graphs in the rest of this section since the CR scheme applies for this more general class and we do not have a better scheme for inductively k-independent graphs. We consider two simple CR schemes for Q. The first is an oblivious deterministic one. Given a set R it outputs S where $S = \{v \in R \mid N_H^+(v) \cap R = \emptyset\}$. In other words it discards from R any vertex v which has an out-neighbor in R. We claim that S is an independent set. To see this suppose $uv \in E(G)$. In H, uv is oriented as (u, v) or (v, u). Thus, both u and v cannot be in S even if they are both are picked in R. It is also easy to see that the scheme is monotone.

We now describe a randomized non-oblivious scheme which yields slightly better constants and is essentially the same as the one from [24] where interval graphs were considered (a special case of $k = 1$). This scheme works as follows. Given R and x it creates a subsample $R' \subseteq R$ by sampling each $v \in R$ independently with probability $(1 - e^{-x_v})/x_v$ (Note that $1 - e^{-y} \leq y$ for all $y \in [0, 1]$.). Equivalently R' is obtained from x by sampling each v with probability $1 - e^{-x_v}$. It then applies the preceding deterministic scheme to R'. Note that this scheme is randomized and non-oblivious since it uses x in the sub-sampling step. It is also easy to see that it is monotone.

We analyze the two schemes.

▶ **Theorem 13.** *For each $b \in [0, 1]$ there is a deterministic, oblivious, monotone $(b/k, 1 - b)$ CR scheme for Q. There is a randomized monotone $(b/k, e^{-b})$ CR scheme for Q.*

Proof. Let $x \in \frac{b}{k} Q$ and Let R be a random set obtained by picking each $v \in V$ independently with probability x_v. We first analyze the deterministic CR scheme. Fix a vertex $v \in \text{support}(x)$ and condition on $v \in R$. The vertex v is included in the final output iff $N_H^+(v) \cap R = \emptyset$. Since $x \in \frac{b}{k} Q$ we have $\sum_{u \in N^+(v)} x_u \leq b - x_v \leq b$.

$$\mathbf{P}[v \in S \mid v \in R] = \mathbf{P}[N^+(v) \cap R = \emptyset] = \prod_{u \in N^+(v)} (1 - x_u) \geq 1 - \sum_{u \in N^+(v)} x_u \geq 1 - b.$$

This shows that the scheme is a $(b/k, 1 - b)$ CR scheme.

Now we analyze the randomized scheme which follows [24]. Consider $v \in R(x)$. We see that $v \in S$ conditioned on $v \in R$, if $v \in R'$ and $R' \cap N^+(v) = \emptyset$. Since the vertices are picked independently,

$$\mathbf{P}[v \in S \mid v \in R] = \mathbf{P}[v \in R' \mid v \in R] \cdot \mathbf{P}[N^+(v) \cap R' = \emptyset] = \frac{(1 - e^{-x_v})}{x_v} \prod_{u \in N^+(v)} e^{-x_u}$$
$$\geq \frac{(1 - e^{-x_v})}{x_v} e^{-(b - x_v)} \geq \frac{(e^{x_v} - 1)}{x_v} e^{-b} \geq e^{-b}.$$

This finishes the proof. ◀

One can apply the preceding CR schemes for Q_G along with the known framework via the multilinear relaxation to approximate $\max_{S\in\mathcal{I}} f(S)$. Let OPT be the value of an optimum solution. For monotone functions the Continuous Greedy algorithm [14] can be used to find a point $x \in \frac{b}{k}Q$ such that $F(x) \geq (1-e^{-b/k})\,\mathrm{OPT}$. When combined with the $(b/k, 1-b)$ CR scheme this yields a $(1-e^{-b/k})(1-b)$-approximation. The randomized CR scheme yields a $(1-e^{-b/k})e^{-b}$-approximation; this bound is maximized when $b = k\ln(1+1/k)$ and the ratio is $\frac{1}{k+1}\cdot\frac{1}{(1+1/k)^k} \geq \frac{1}{e(k+1)}$. For non-negative functions one can use Measured Continuous Greedy [26, 24] to obtain $x \in \frac{b}{k}Q$ such that $F(x) \geq \frac{b}{k}e^{-b/k}\,\mathrm{OPT}$. Combined with the CR scheme this yields a $(\frac{b}{k}e^{-b(1+/k)})$-approximation. Setting $b = k/(k+1)$ yields a $\frac{1}{e(k+1)}$-approximation.

▶ **Theorem 14.** *There is a randomized algorithm that given a k-perfectly orientable graph G (along with its orientation) and a monotone submodular function f, outputs an independent set S' such that with high probability $f(S') \geq (\frac{1}{k+1}\cdot\frac{1}{(1+1/k)^k})\max_{A\in\mathcal{I}} f(A)$. For non-negative functions there is an algorithm that outputs an independent set S' such that with high probability $f(S') \geq \frac{1}{e(k+1)}\max_{A\in\mathcal{I}} f(A)$.*

Efficiency and Parallelism

Approximately solving the multilinear relaxation is typically a bottleneck. [18] develops faster algorithms via the multiplicative-weight update (MWU) based method. We refer the reader to [18] for concrete running times that one can obtain in terms of the number of oracle calls to f or F. Once the relaxation is solved, rounding via the CR scheme above is simple and efficient. Another aspect is the design of parallel algorithms, or algorithms with low adaptivity – we refer the reader to [4] for the motivation and set up. Via results in [20, 23], and the CR scheme above, we can obtain algorithms with adaptivity $O(\frac{\log^2 n}{\epsilon^2})$ while only losing a $(1-\epsilon)$-factor in the approximation compared to the sequential approximation ratios. We defer details.

4 Primal-Dual Approach for Inductively k-Independent Graphs

We now consider a primal-dual algorithm. This is inspired by previous algorithms for MWIS in inductively k-independent graphs, and the work of Levin and Wajc [35] who considered a primal-dual based semi-streaming algorithm for submodular function maximization under matching constraints.

The stack based algorithm in [40] for MWIS is essentially a primal-dual algorithm. It is instructive to explicitly consider the LP relaxation and the analysis for MWIS before seeing the algorithm and analysis for the submodular setting. An interested reader can find this exposition in the full version.

Following [35] we consider an LP relaxation based on the concave closure of f. For independent sets in an inductively k-independent graph, we consider the relaxation $\max_{x\in Q_G} f^+(x)$. We write this as an explicit LP and describe its dual. See Fig 1. The primal has a variable x_i for each $v_i \in \mathcal{V}$ as we saw in the relaxation for MWIS. In addition to these variables, we have variables $\alpha_L, L \subseteq \mathcal{V}$ to model the objective $f^+(x)$. The dual has three types of variables. μ is for the equality constraint $\sum_L \alpha_L = 1$, y_i is corresponds to the primal packing constraint for x_i coming from the independence constraint, and z_i is for the equality constraint coming from modeling $f^+(x)$.

$$
\begin{aligned}
\max \sum_{S \subseteq V} \alpha_L f(L) & & & \\
\sum_{L \subseteq V} \alpha_L &= 1 & & \\
\sum_{L \ni v_i} \alpha_L &= x_i & & i \in [n] \\
x_i + \sum_{v_j \in A_i} x_j &\leq k & & i \in [n] \\
x_i &\geq 0 & & i \in [n]
\end{aligned}
\qquad\qquad
\begin{aligned}
\min \mu + k \sum_{i=1}^{n} y_i & & & \\
\mu + \sum_{v_i \in L} z_i &\geq f(L) & & L \subseteq V \\
y_i + \sum_{v_j \in B_i} y_j &\geq z_i & & i \in [n] \\
y_i &\geq 0 & & i \in [n]
\end{aligned}
$$

Figure 1 Primal and Dual LPs via the concave closure relaxation for an inductively k-independent graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a given ordering $\{v_1, v_2, \ldots, v_n\}$.

4.1 Algorithm for monotone submodular functions

We describe a deterministic primal-dual algorithm for the monotone case. The algorithm and analysis are inspired by [35] and we note that the algorithm has some similarities to the preemptive greedy algorithm. The primal-dual algorithm takes a two phase approach similar to algorithm for the modular case. In the first phase it processes the vertices in the given order and creates a set $S \subseteq \mathcal{V}$. In the second phase it process the vertices in the reverse order of insertion and creates a maximal independent set. Unlike the modular case, the decision to add a vertex v_i to S in the first phase is based on an inflation factor $(1+\beta)$. The formal algorithm is described in Fig 2. The algorithm creates a feasible dual as it goes along – the variables y, z, μ are from the dual LP. It also maintains and uses auxiliary weight variables w_i, $1 \leq i \leq n$ that will be useful in the analysis.

`primal-dual-monotone-submod(`$f : 2^{\mathcal{V}} \to \mathbb{R}_{\geq 0}$`,`$k \in \mathbb{N}$`,`$\beta \in \mathbb{R}_{>0}$`).`

1. Initialize an empty stack S. Let $\mathcal{V} = \{v_1, \ldots, v_n\}$ be a k-independence ordering of $\mathcal{V}$. Set $w, z, y \leftarrow \mathbb{0}_n$.
2. For $i = 1, \ldots, n$:
   - **A.** Let $C_i = N(v_i) \cap S = \{u \in S : uv_i \in \mathcal{E}\}$
   - **B.** If $(f_S(v_i) > (1+\beta) \sum_{v_j \in C_i} w_j)$ then
     1. Call S`.push(`v_i`)` and set $x_i \leftarrow 1$.
     2. Set $w_i \leftarrow f_S(v_i) - \sum_{v_j \in C_i} w_j$ and $y_i \leftarrow (1+\beta) w_i$.
   - **C.** Otherwise set $z_i \leftarrow f_S(v_i)$
3. Let $\mu \leftarrow f(S)$ and $\hat{S} \leftarrow \emptyset$
4. While S is not empty:
   - **A.** $v \leftarrow S$`.pop()`
   - **B.** If $\hat{S} + v_i$ is independent in $\mathcal{G}$ then set $\hat{S} \leftarrow \hat{S} + v_i$.
5. Return $\hat{S}$

Figure 2 Primal-dual algorithm for monotone submodular maximization. The algorithm creates a feasible dual solution in the first phase along with a set S_{end}. In the second phase it processes S_{end} in reverse order of insertion and creates a maximal independent set.

Let S_{end} be the set of vertices in the stack S at the end of the first phase. S is a monotonically increasing set during the algorithm. Note that $\mu = f(S_{\text{end}})$ at the end of the algorithm. We observe that for each i, the algorithm sets the variables w_i, y_i, z_i exactly once when v_i is processed, and does not alter the values after they are set.

▶ **Lemma 15.** *The algorithm* `primal-dual-monotone-submod` *creates a feasible dual solution* $\mu, \bar{y}, \bar{z}$ *when* f *is monotone.*

Proof. We observe that $z_i = 0$ if $v_i \in S_{\text{end}}$ and $z_i = \nu\big(f, S^-_{v_i}, v_i\big)$ otherwise. By submodularity it follows that if $v_i \notin S_{\text{end}}$, $z_i \geq f_{S_{\text{end}}}(v_i)$ since $S^-_{v_i} \subseteq S_{\text{end}}$.

Consider the first set of constraints in the dual of the form $\mu + \sum_{v_i \in L} z_i \geq f(L)$ for $L \subseteq V$. We have

$$\mu + \sum_{v_i \in L} z_i \geq f(S_{\text{end}}) + \sum_{v_i \in L \setminus S_{\text{end}}} f_{S_{\text{end}}}(v_i) \geq f(S_{\text{end}} \cup L) \geq f(L).$$

We used submodularity in the second inequality and monotonicity of f in the last inequality.

Now consider the second set of constraints in the dual of the form $y_i + \sum_{v_j \in B_i} y_j \geq z_i$ for each i. If $v_i \in S_{\text{end}}$ then $z_i = 0$ and the constraint is trivially satisfied since the y variables are non-negative. Assume $v_i \notin S_{\text{end}}$. The algorithm did not add v_i to S because

$$z_i = \nu\big(f, S^-_{v_i}, v_i\big) \leq (1+\beta) \sum_{v_j \in C_i} w_j = \sum_{v_j \in C_i} y_j$$

which implies that the constraint for v_i is satisfied. ◀

Feasibility of the dual solution implies an upper bound on the optimal value.

▶ **Corollary 16.** $\text{OPT} \leq f(S_{end}) + k(1+\beta) \sum_{i=1}^{n} w_i$.

We now lower bound the value of $f(\hat{S})$.

▶ **Lemma 17.** $f(\hat{S}) \geq \sum_{i=1}^{n} w_i$.

Proof. A vertex v_i is added to S_{end} since $\nu\big(f, S^-_{v_i}, v_i\big) > (1+\beta) \sum_{v_j \in C_i} w_j$. Moreover, we have $w_i + \sum_{v_j \in C_i} w_j = \nu\big(f, S^-_{v_i}, v_i\big)$ via the algorithm. Therefore,

$$f(\hat{S}) = \sum_{v_i \in \hat{S}} \nu\Big(f, \hat{S}, v_i\Big) \geq \sum_{v_i \in \hat{S}} \nu\big(f, S^-_{v_i}, v_i\big) = \sum_{v_i \in \hat{S}} (w_i + \sum_{j \in C_i} w_j).$$

We see that for every i' such that $v_{i'} \in S_{\text{end}}$ the term $w_{i'}$ appears at least once in $\sum_{v_i \in \hat{S}}(w_i + \sum_{j \in C_i} w_j)$; either $v_{i'} \in \hat{S}$ or if it is not then it was removed in the second phase since $v_{i'} \in C_i$ for some $v_i \in \hat{S}$. In the latter case $w_{i'}$ appears in the $\sum_{j \in C_i} w_j$. Thus $f(\hat{S}) \geq \sum_{i=1}^{n} w_i$ (recall that $w_i = 0$ if $v_i \notin S_{\text{end}}$). ◀

We now upper bound $f(S_{\text{end}})$ via the weights.

▶ **Lemma 18.** $f(S_{end}) \leq \frac{1+\beta}{\beta} \sum_{i=1}^{n} w_i$.

Proof. Let $v_i \in S_{\text{end}}$. Recall that $\nu\big(f, S^-_{v_i}, v_i\big) \geq (1+\beta) \sum_{j \in C_i} w_j$ and $w_i = \nu\big(f, S^-_{v_i}, v_i\big) - \sum_{j \in C_i} w_j$. This implies that $w_i \geq \frac{\beta}{1+\beta} \nu\big(f, S^-_{v_i}, v_i\big)$. Therefore

$$f(S_{\text{end}}) = \sum_{v_i \in S_{\text{end}}} \nu(f, S_{\text{end}}, v_i) = \sum_{v_i \in S_{\text{end}}} \nu\big(f, S^-_{v_i}, v_i\big) \leq \frac{1+\beta}{\beta} \sum_{v_i \in S_{\text{end}}} w_i,$$

as desired. ◀

▶ **Theorem 19.** OPT $\leq (1+\beta)(1/\beta+k)f(\hat{S})$. *In particular, for* $\beta = \frac{1}{\sqrt{k}}$, OPT $\leq (k+1+2\sqrt{k})f(\hat{S})$.

Proof. From Corollary 16 and Lemma 18 and Lemma 17,

$$\begin{aligned}\mathrm{OPT} &\leq f(S_{\mathrm{end}}) + k(1+\beta)\sum_{i=1}^{n} w_i \leq \frac{1+\beta}{\beta}\sum_{i=1}^{n} w_i + k(1+\beta)\sum_{i=1}^{n} w_i \\ &\leq (1+\beta)(\frac{1}{\beta}+k)\sum_{i=1}^{n} w_i \leq (1+\beta)(\frac{1}{\beta}+k)f(\hat{S}),\end{aligned}$$

as desired. ◀

▶ Remark 20. For $k=1$ we obtain a 1/4-approximation which yields a deterministic 1/4-approximation for chordal graphs and interval graphs. For $k=2$ we obtain a bound of $3+2\sqrt{2}$ which is the same as what [35] obtain for matchings. Note that matchings can be interpreted, via the line graph, as inductive 2-independent and in fact any ordering of the edges is an inductive 2-independent order. This explains why the ordering does not matter. [35] use a different LP relaxation for matchings, and hence it is a bit surprising that we obtain the same bound for all 2-independent graphs. For the non-monotone case we obtain a weaker bound for 2-independent graphs than what [35] obtain for matchings.

4.2 Non-monotone submodular maximization

We now consider the case of non-negative submodular function which may not be necessarily monotone. This class of functions requires some additional technical care and a key lemma that is useful in handling non-monotone function is the following.

▶ **Lemma 21** ([8]). *Let $f : 2^V \to \mathbb{R}_+$ be a non-negative submodular function. Fix a set $T \subseteq V$. Let S be a random subset of V such that for any $v \in V$ the probability of $v \in S$ is at most p for some $p < 1$. Then $\mathbf{E}[f(S \cup T)] \geq (1-p)f(T)$.*

We describe a randomized primal-dual algorithm which is adapted from the one form [35]. It differs from the monotone algorithm in one simple but crucial way; even when a vertex v has good value compared to its conflict set it adds it to the stack only with probability p which is a parameter that is chosen later.

As in the monotone case let S_{end} be the set of vertices in the stack at the end of the first phase (note that S_{end} is now a random set). The analysis of the randomized version of the algorithm is technically more involved. The sets $S_{\mathrm{end}}, \hat{S}$ and the dual variables are now random variables. Since very high-value vertices can be discarded probabilistically, the dual values constructed by the algorithm may not satisfy the dual constraints for each run of the algorithm. Levin and Wajc [35] analyze their algorithm for matchings via an "expected" dual solution. We do a more direct analysis via weak duality.

The following two lemmas are essentially the same as in the monotone case and they relate the expected value of $\hat{S}$ and S_{end} to the dual weight values.

▶ **Lemma 22.** *For each run of the algorithm: $f(\hat{S}) \geq \sum_{i=1}^{n} w_i$ and hence $\mathbf{E}\left[f(\hat{S})\right] \geq \sum_{i=1}^{n} \mathbf{E}[w_i]$.*

▶ **Lemma 23.** *For each run of the algorithm, $f(S_{end}) \leq \frac{1+\beta}{\beta}\sum_{i=1}^{n} w_i$ and hence*

$$\mathbf{E}[f(S_{end})] \leq \frac{1+\beta}{\beta}\sum_{i=1}^{n} \mathbf{E}[w_i].$$

```
primal-dual-nonneg-submod(f : 2^V → ℝ≥0, k ∈ ℕ, β ∈ ℝ>0).
```

1. Initialize an empty stack S. Let $\mathcal{V} = \{v_1, \ldots, v_n\}$ be a k-independence ordering of $\mathcal{V}$. Let $w, y, z = \mathbb{0}_n$.
2. For $i = 1, \ldots, n$:
   A. Let $C_i = N(v_i) \cap S = \{u \in S : uv_i \in \mathcal{E}\}$
   B. If $(f_S(v_i) > (1+\beta) \sum_{v_j \in C_i} w_j)$, then with probability p:
      1. Call `S.push(`v_i`)` and $x_i \leftarrow 1$
      2. Set $w_i \leftarrow f_S(v_i) - \sum_{v_j \in C_i} w_j$ and $y_i \leftarrow (1+\beta) w_i$
   C. Otherwise set $z_i \leftarrow f_S(v_i)$.
3. Set $\mu \leftarrow f(S)$ and $\hat{S} \leftarrow \emptyset$
4. While S is not empty:
   A. $v \leftarrow$ `S.pop()`.
   B. If $\hat{S} + v_i$ is independent in $\mathcal{G}$ then set $\hat{S} \leftarrow \hat{S} + v_i$
5. Return $\hat{S}$

Figure 3 Randomized primal-dual algorithm for non-negative submodular maximization.

The next two lemmas provide a way to upper bound the optimum value via the expected dual objective value.

▶ **Lemma 24.** *For each vertex v_i, let $1_{v_i \notin S_{end}}$ indicate if v_i is excluded from S_{end}. Let $B_i' = B_i + v_i$. Then*

$$\mathbf{E}[f(v_i \,|\, S_i) 1_{v_i \notin S_{end}}] \leq \max\left\{\frac{1-p}{p}, 1+\beta\right\} \mathbf{E}[w(B_i' \cap S_{end})].$$

Proof. Let E_i be the event that $f(v_i \,|\, S_i) > (1+\beta) w(B_i \cap S_i)$. Condition on $\bar{E}_i$, that is E_i *not* occurring, in which case v_i is not added to the stack. In this case we have

$$\begin{aligned}\mathbf{E}\big[w(B_i) \,\big|\, \bar{E}_i\big] &= \mathbf{E}\big[w(B_i' \cap S_{\text{end}}) \,\big|\, \bar{E}_i\big] \geq \frac{1}{1+\beta} \mathbf{E}\big[f(v_i \,|\, S_i) \,\big|\, \bar{E}_i\big] \\ &= \frac{1}{1+\beta} \mathbf{E}\big[f(v_i \,|\, S_i) 1_{v_i \notin S_{\text{end}}} \,\big|\, \bar{E}_i\big]\end{aligned}$$

On the other hand, condition on E_i, we have

$$\mathbf{E}[w(B_i' \cap S_{\text{end}}) \,|\, E_i] \overset{\text{(a)}}{\geq} p\, \mathbf{E}[f(v_i \,|\, S_i) \,|\, E_i] \overset{\text{(b)}}{=} \frac{p}{1-p} \mathbf{E}[f(v_i \,|\, S_i) 1_{v_i \notin S_{\text{end}}} \,|\, E_i].$$

(a) is because with probability p, we add v to the stack, in which case $w(B_i' \cap S_{\text{end}}) \geq f(v_i \,|\, S_i)$. (b) is because conditional on E_i and $f(v_i \,|\, S_i)$, $v_i \notin S_{\text{end}}$ with probability $1-p$. We combine the two bounds by taking conditional expectations, as follows:

$$\begin{aligned}\mathbf{E}\big[f(v_i \,|\, S_i) 1_{v_i \notin S_{\text{end}}}\big] &= \mathbf{E}\big[f(v_i \,|\, S_i) 1_{v_i \notin S_{\text{end}}} \,\big|\, E_i\big]\, \mathbf{P}[E_i] + \mathbf{E}\big[f(v_i \,|\, S_i) 1_{v_i \notin S_{\text{end}}} \,\big|\, \bar{E}_i\big]\, \mathbf{P}\big[\bar{E}_i\big] \\ &\leq \frac{1-p}{p} \mathbf{E}\big[w(B_i' \cap S_{\text{end}}) \,\big|\, E_i\big]\, \mathbf{P}[E_i] + (1+\beta)\, \mathbf{E}\big[w(B_i' \cap S_{\text{end}}) \,\big|\, \bar{E}_i\big]\, \mathbf{P}\big[\bar{E}_i\big] \\ &\leq \max\left\{\frac{1-p}{p}, 1+\beta\right\} \Big(\mathbf{E}\big[w(B_i' \cap S_{\text{end}})\big]\, \mathbf{P}[E_i] + \mathbf{E}\big[w(B_i' \cap S_{\text{end}})\big]\, \mathbf{P}\big[\bar{E}_i\big]\Big) \\ &= \max\left\{\frac{1-p}{p}, 1+\beta\right\} \mathbf{E}\big[w(B_i' \cap S_{\text{end}})\big],\end{aligned}$$

as desired. ◀

▶ **Lemma 25.** *For any set T, $\mathbf{E}[f(S_{end} \cup T)] \leq \mathbf{E}[f(S)] + k \max\left\{\frac{1-p}{p}, 1+\beta\right\} \mathbf{E}[w(S_{end})]$.*

Proof. We have

$$\begin{aligned}
\mathbf{E}[f(T \cup S_{\text{end}}) - f(S_{\text{end}})] &\overset{\text{(c)}}{\leq} \mathbf{E}\left[\sum_{v_i \in T \setminus S_{\text{end}}} f(v_i \mid S_{\text{end}})\right] \overset{\text{(d)}}{\leq} \mathbf{E}\left[\sum_{v_i \in T \setminus S_{\text{end}}} f(v_i \mid S_i)\right] \\
&= \sum_{v_i \in T} \mathbf{E}[f(v_i \mid S_i) 1_{v_i \notin S_{\text{end}}}] \\
&\overset{\text{(e)}}{\leq} \max\left\{\frac{1-p}{p}, 1+\beta\right\} \sum_{v_i \in T} \mathbf{E}[w(B'_i \cap S_{\text{end}})] \\
&\overset{\text{(f)}}{\leq} \max\left\{\frac{1-p}{p}, 1+\beta\right\} k\, \mathbf{E}[w(S_{\text{end}})],
\end{aligned}$$

as desired up to rearrangement of terms. Here (c,d) is by submodularity. (e) is by the Lemma 24. (f) is by k-inductive independence. ◀

We now put the lemmas together to relate $\mathbf{E}\left[f(\hat{S})\right]$ to the optimum.

▶ **Lemma 26.** *Let T^* be an optimum independent set with $\mathrm{OPT} = f(T^*)$. Then*

$$\mathrm{OPT} \leq \frac{k \max\left\{\frac{1-p}{p}, 1+\beta\right\} + \left(\frac{1+\beta}{\beta}\right)}{1-p} \mathbf{E}\left[f\left(\hat{S}\right)\right].$$

Proof. Let T be any independent set, in particular T^*. We observe that the algorithm ensures that for any vertex v, $\mathbf{P}[v \in S_{\text{end}}] \leq p$ and hence $\mathbf{P}\left[v \in \hat{S}\right] \leq p$.

$$\begin{aligned}
(1-p) f(T) &\leq \mathbf{E}[f(T \cup S_{\text{end}})] \quad \text{(Lemma 21)} \\
&\leq \mathbf{E}[f(S)] + k \max\left\{\frac{1-p}{p}, 1+\beta\right\} \mathbf{E}[w(S_{\text{end}})] \quad \text{(Lemma 25)} \\
&\leq \left(k \max\left\{\frac{1-p}{p}, 1+\beta\right\} + \left(\frac{1+\beta}{\beta}\right)\right) \mathbf{E}[w(S_{\text{end}})] \quad \text{(Lemma 23)} \\
&\leq \left(k \max\left\{\frac{1-p}{p}, 1+\beta\right\} + \left(\frac{1+\beta}{\beta}\right)\right) \mathbf{E}\left[f\left(\hat{S}\right)\right] \quad \text{(Lemma 22).}
\end{aligned}$$

◀

It remains to choose $p \in [0, 1]$ and $\beta > 0$ to minimize the RHS. Consider the term $\max\{(1-p)/p, 1+\beta\}$. If $(1-p)/p \geq 1+\beta$, then $p \geq 1/2$ (to force $(1-p)/p \geq 1$), and the RHS is minimized by taking β as large as possible – that is, such that $1+\beta = (1-p)/p$. If $(1-p)/p \leq 1+\beta$, then the RHS is minimized by taking p as small as possible – that is, such that $(1-p)/p = 1/p - 1 = 1+\beta$. Thus $(1-p)/p = 1+\beta$ at the optimum. In terms of just p, then, we have

$$\mathrm{OPT} \leq \left(\frac{1}{1-p}\right)\left(\frac{k(1-p)}{p} + \frac{1-p}{1-2p}\right) \mathbf{E}\left[f\left(\hat{S}\right)\right] = \left(\frac{k}{p} + \frac{1}{1-2p}\right) \mathbf{E}\left[f\left(\hat{S}\right)\right].$$

(Here we note that $\beta = (1-2p)/p$, hence $(1+\beta)/\beta = (1-p)/(1-2p)$.)

In the special case of $k = 2$, as in matching, the RHS is

$$\mathrm{OPT} \leq \left(\frac{2}{p} + \frac{1}{1-2p}\right) \mathbf{E}\left[f\left(\hat{S}\right)\right]$$

The RHS is minimized by $p = 1/3$, giving an approximation factor of 9.

For general k, the minimum is $2k + \sqrt{8k} + 1$.

It is easy to see that the primal-dual algorithm makes $O(n)$ evaluation calls to f and the overall running time is linear in the size of the graph. The results for the monotone and non-negative functions, together yield Theorem 6.

5 Concluding Remarks and Open Problems

We described $\Omega(\frac{1}{k})$-approximation algorithms for independent sets in two parameterized families of graphs that capture several problems of interest. Although the multilinear relaxation based framework yields such algorithms, the resulting algorithms are computationally expensive and randomized. We utilized ideas from streaming and primal-dual based algorithms to give simple and fast algorithms for inductively k-independent graphs with the additional property that they are deterministic for monotone functions. Our work raises several interesting questions that we summarize below.

- The CR scheme that we described in Section 3 is unable to distinguish k-perfectly orientable graphs and inductive k-independent graphs. Is a better bound possible for inductively k-independent graphs?
- Our combinatorial algorithms only apply to inductively k-independent graphs. Can we obtain combinatorial algorithms for k-perfectly orientable graphs? Even for MIS the only approach appears to be via primal rounding of the LP solution [34].
- Can we obtain deterministic $\Omega(\frac{1}{k})$-approximation algorithms for these graph classes when f is non-negative? Interval graphs seem to be a natural first step to consider.
- Are better approximation ratios achievable? For instance, can we obtain better than 1/4-approximation for monotone submodular function maximization in interval graphs? Can we prove better lower bounds under complexity theory assumptions or in the oracle model for interval graphs or other concrete special cases of interest?
- For both classes of graphs our algorithms are based on having an ordering that certifies that they belong to the class. For MWIS in k-simplicial and k-perfectly orientable graphs, [30] describes algorithms based on the Lovász number of a graph and the Lovász θ-function of a graph, and these algorithms do not require an ordering. It may be feasible to extend their approach to the submodular setting via the multilinear relaxation. However, the resulting algorithms are computationally quite expensive. It would be interesting to obtain fast algorithms for these classes of graphs (or interesting special cases) when the ordering is not explicitly given.

References

1. Shipra Agrawal, Yichuan Ding, Amin Saberi, and Yinyu Ye. Correlation robust stochastic optimization. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 1087–1096. SIAM, 2010.
2. Karhan Akcoglu, James Aspnes, Bhaskar DasGupta, and Ming-Yang Kao. Opportunity-cost algorithms for combinatorial auctions. In E. J. Kontoghiorghes, B. Rustem, and S. Siokos, editors, *Applied Optimization 74: Computational Methods in Decision-Making, Economics and Finance*, pages 455–479. Kluwer Academic Publishers, 2002.
3. A. Badanidiyuru, B. Mirzasoleiman, A. Karbasi, and A. Krause. Streaming submodular optimization: Massive data summarization on the fly. In *Proc. 20th ACM Conf. Knowl. Disc. and Data Mining* (KDD), pages 671–680, 2014.

4 Eric Balkanski and Yaron Singer. The adaptive complexity of maximizing a submodular function. In *Proceedings of the 50th annual ACM SIGACT symposium on theory of computing*, pages 1138–1151, 2018.

5 Reuven Bar-Yehuda, Magnús M Halldórsson, Joseph Naor, Hadas Shachnai, and Irina Shapira. Scheduling split intervals. *SIAM Journal on Computing*, 36(1):1–15, 2006.

6 Jeff A. Bilmes. Submodularity in machine learning and artificial intelligence. *CoRR*, abs/2202.00132, 2022. arXiv:2202.00132.

7 Simon Bruggmann and Rico Zenklusen. Submodular maximization through the lens of linear programming. *Mathematics of Operations Research*, 44(4):1221–1244, 2019.

8 N. Buchbinder, M. Feldman, J. Naor, and R. Schwartz. Submodular maximization with cardinality constraints. In *Proc. 25th ACM-SIAM Sympos. Discrete Algs.* (SODA), pages 1433–1452, 2014.

9 Niv Buchbinder and Moran Feldman. Deterministic algorithms for submodular maximization problems. *ACM Transactions on Algorithms (TALG)*, 14(3):1–20, 2018.

10 Niv Buchbinder and Moran Feldman. Submodular functions maximization problems. In *Handbook of Approximation Algorithms and Metaheuristics, Second Edition*, pages 753–788. Chapman and Hall/CRC, 2018.

11 Niv Buchbinder, Moran Feldman, and Mohit Garg. Deterministic ($1/2+\varepsilon$)-approximation for submodular maximization over a matroid. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 241–254. SIAM, 2019.

12 Niv Buchbinder, Moran Feldman, and Roy Schwartz. Online submodular maximization with preemption. *ACM Transactions on Algorithms (TALG)*, 15(3):1–31, 2019.

13 G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák. Maximizing a submodular set function subject to a matroid constraint (extended abstract). In *Proc. 12th Int. Conf. Int. Prog. Comb. Opt.* (IPCO), pages 182–196, 2007.

14 Gruia Calinescu, Chandra Chekuri, Martin Pal, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

15 Amit Chakrabarti and Sagar Kale. Submodular maximization meets streaming: matchings, matroids, and more. *Mathematical Programming*, 154:225–247, 2015.

16 Timothy M Chan and Sariel Har-Peled. Approximation algorithms for maximum independent set of pseudo-disks. *Discrete & Computational Geometry*, 48(2):373–392, 2012.

17 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 42*, pages 318–330. Springer, 2015. Longer version: http://arxiv.org/abs/1504.08024.

18 Chandra Chekuri, T. S. Jayram, and Jan Vondrák. On multiplicative weight updates for concave and submodular function maximization. In Tim Roughgarden, editor, *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science, ITCS 2015, Rehovot, Israel, January 11-13, 2015*, pages 201–210. ACM, 2015. doi:10.1145/2688073.2688086.

19 Chandra Chekuri and Vasilis Livanos. On submodular prophet inequalities and correlation gap. *arXiv preprint arXiv:2107.03662*, 2021.

20 Chandra Chekuri and Kent Quanrud. Submodular function maximization in parallel via the multilinear relaxation. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 303–322. SIAM, 2019.

21 Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Submodular function maximization via the multilinear relaxation and contention resolution schemes. *SIAM Journal on Computing*, 43(6):1831–1879, 2014.

22 Andrew Clark, Basel Alomair, Linda Bushnell, and Radha Poovendran. *Submodularity in dynamics and control of networked systems.* Springer, 2016.

23 Alina Ene, Huy L Nguyen, and Adrian Vladu. Submodular maximization with matroid and packing constraints in parallel. In *Proceedings of the 51st annual ACM SIGACT symposium on theory of computing*, pages 90–101, 2019.
24 Moran Feldman. *Maximization problems with submodular objective functions.* PhD thesis, Computer Science Department, Technion, 2013.
25 Moran Feldman, Amin Karbasi, and Ehsan Kazemi. Do less, get more: Streaming submodular maximization with subsampling. *Advances in Neural Information Processing Systems*, 31, 2018.
26 Moran Feldman, Joseph Naor, and Roy Schwartz. A unified continuous greedy algorithm for submodular maximization. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, pages 570–579. IEEE, 2011.
27 Moran Feldman, Joseph Naor, Roy Schwartz, and Justin Ward. Improved approximations for k-exchange systems. In *Algorithms–ESA 2011: 19th Annual European Symposium, Saarbrücken, Germany, September 5-9, 2011. Proceedings 19*, pages 784–798. Springer, 2011.
28 M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. An analysis of approximations for maximizing submodular set functions – II. *Math. Prog. Studies*, 8:73–87, 1978.
29 Paritosh Garg, Linus Jordan, and Ola Svensson. Semi-streaming algorithms for submodular matroid intersection. *Mathematical Programming*, pages 1–24, 2022.
30 Magnús M. Halldórsson and Tigran Tonoyan. Computing inductive vertex orderings. *Information Processing Letters*, 172:106159, 2021. doi:10.1016/j.ipl.2021.106159.
31 Kai Han, zongmai Cao, Shuang Cui, and Benwei Wu. Deterministic approximation for submodular maximization over a matroid in nearly linear time. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 430–441. Curran Associates, Inc., 2020. URL: https://proceedings.neurips.cc/paper/2020/file/05128e44e27c36bdba71221bfccf735d-Paper.pdf.
32 Johan Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Math*, 182, 1999.
33 Th Jenkyns. The efficacy of the" greedy" algorithm. In *Proc. 7th Southeastern Conf. on Combinatorics, Graph Theory and Computing*, pages 341–350, 1976.
34 Frank Kammer and Torsten Tholey. Approximation algorithms for intersection graphs. *Algorithmica*, 68(2):312–336, 2014.
35 Roie Levin and David Wajc. Streaming submodular matching meets the primal-dual method. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1914–1933. SIAM, 2021.
36 G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions – I. *Math. Prog.*, 14(1):265–294, 1978.
37 Ami Paz and Gregory Schwartzman. A (2+ ε)-approximation for maximum weight matching in the semi-streaming model. *ACM Transactions on Algorithms (TALG)*, 15(2):1–15, 2018.
38 Rom Pinchasi. A finite family of pseudodiscs must include a "small" pseudodisc. *SIAM Journal on Discrete Mathematics*, 28(4):1930–1934, 2014.
39 Jan Vondrák. *Submodularity in combinatorial optimization.* PhD thesis, Univerzita Karlova, Matematicko-fyzikální fakulta, 2007.
40 Yuli Ye and Allan Borodin. Elimination graphs. *ACM Transactions on Algorithms (TALG)*, 8(2):1–23, 2012.
41 David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 681–690, 2006.

A A Preemptive Greedy Algorithm

We now describe a preemptive greedy algorithm for maximizing a monotone submodular function $f : 2^V \to \mathbb{R}_+$ over independent sets of a inductively k-independent graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ assuming that we are also given the ordering. The algorithm is simple and intuitive, and is inspired by algorithms developed in the streaming model.

The pseudocode for the algorithm is given in Figure 4, and is designed as follows. Starting from an empty solution $S = \emptyset$, `preemptive-greedy` processes the vertices in the given ordering one by one. When considering v_i, the algorithm gathers the subset $C_i \subseteq S$ of all vertices in the current set S that are neighbors of v_i (those that conflict with v_i). The algorithm has to decide whether to reject v_i or to accept v_i in which case it has to remove C_i from S. It accepts v_i if the marginal gain $f_S(v_i) \stackrel{\text{def}}{=} f(S + v_i) - f(S)$ of adding v_i directly to S is at least $(1 + \beta)$ times the value $\sum_{u \in C_i} f_{S \setminus C_i}(u)$. Here $\beta > 0$ is a parameter that is fixed based on the analysis. After processing all vertices, we return the final set S.

`preemptive-greedy`$(\mathcal{G} = (\mathcal{V}, \mathcal{E}), f : 2^{\mathcal{V}} \to \mathbb{R}_{\geq 0}, k \in \mathbb{N}, \beta \in \mathbb{R}_{>0})$.

1. Let $S = \emptyset$. Let $\mathcal{V} = \{v_1, \ldots, v_n\}$ by a k-independence ordering of $\mathcal{V}$
2. For $i = 1, \ldots, n$:
  A. Let $C_i = N(v_i) \cap S = \{u \in S : uv_i \in \mathcal{E}\}$
  B. If $f_S(v_i) \geq (1 + \beta) \sum_{u \in C_i} \nu(f, S, u)$
    1. Set $S \leftarrow (S \setminus C_i) + v_i$
3. Return S

Figure 4 The algorithm `preemptive-greedy` for finding an independent set in a inductively k-independent graph to maximize a monotone submodular objective function.

`preemptive-greedy` for inductively k-independent graphs has the following approximation bound. The proof is deferred to the subsection following the theorem statements.

▶ **Theorem 27.** *Given an inductively k-independent graph with a k-inductive ordering, the algorithm* `preemptive-greedy` *returns an independent set $\hat{S}$ such that for any independent set T, $f(T) \leq (k(1 + \beta) + 1)(1 + \beta^{-1}) f(\hat{S})$.*

`preemptive-greedy` can be extended to nonnegative (and non-monotone) submodular functions with a constant factor loss in approximation by random sampling. As a preprocessing step, we let $\mathcal{V}'$ randomly sample each vertex in $\mathcal{V}$ independently with probability $1/2$. We then apply `preemptive-greedy` to the subgraph $\mathcal{G}' = \mathcal{G}[\mathcal{V}']$ induced by $\mathcal{V}'$. It is easy to see that any subgraph of an inductively k-independent graph is also inductively k-independent. The net effect of the random sampling is an approximation factor for nonnegative submodular functions that is a factor 4 worse than for the monotone case. The modified algorithm, called `randomized-preemptive-greedy`, is given in Figure 5.

▶ **Theorem 28.** *Given an inductively k-independent graph with a k-inductive ordering, the algorithm* `randomized-preemptive-greedy` *returns an independent set $\hat{S}$ such that for any independent set T, $f(T) \leq 4(k(1 + \beta) + 1)(1 + \beta^{-1}) f(\hat{S})$.*

`randomized-preemptive-greedy`$(\mathcal{G} = (\mathcal{V}, \mathcal{E}), f : 2^{\mathcal{V}} \to \mathbb{R}_{\geq 0}, k \in \mathbb{N}, \beta \in \mathbb{R}_{>0})$.

1. Let $\mathcal{V}' \subseteq \mathcal{V}$ sample each $v \in \mathcal{V}$ independently with probability $1/2$
2. Let $\mathcal{G}' = \mathcal{G}[\mathcal{V}']$ be the subgraph of $\mathcal{G}$ induced by $\mathcal{V}'$
3. Return `preemptive-greedy`$(\mathcal{G}', f : 2^{\mathcal{V}'} \to \mathbb{R}_{\geq 0}, k \in \mathbb{N}, \beta \in \mathbb{R}_{>0})$.

Figure 5 The algorithm `randomized-preemptive-greedy` for finding an independent set in an inductively k-independent graph to maximize a nonnegative submodular objective function.

▶ Remark 29. The randomized strategy we outline is simple and oblivious. It loses a factor of 4 over the monotone case. One could try to improve the approximation ratio by using randomization within the algorithm which would make the analysis more involved. However, we have not done this since the primal-dual algorithm yields better approximation bounds. This subsampling strategy is not new and has been used previously in [25], and is also implicit in [17].

The rest of the section is devoted to proving the claimed approximation guarantees.

A.1 Analysis of preemptive greedy

We follow the notation of [17]. Let $\hat{S}$ be the final set of vertices returned by `preemptive-greedy`. It is easy to see that the algorithm returns an independent set. For each $u \in \mathcal{V}$ let S_u^- denote the set of vertices in S just before u is processed, and let S_u^+ denote the set after u is processed. Thus a vertex u is added to S iff $S_u^+ \setminus S_u^- = \{u\}$. Let $U = \bigcup_{u \in \mathcal{V}} S_u^+$ be the set of all vertices that were ever (even momentarily) added to S. Alternatively, $\mathcal{V} \setminus U$ is the set of vertices that are discarded by the algorithm when it considers them. For each vertex u, let $\delta_u \stackrel{\text{def}}{=} f(S_u^+) - f(S_u^-)$ be the value added to S from processing u. We have $\delta_u = 0$ for all $u \notin U$, and $f(\hat{S}) = \sum_{u \in \mathcal{V}} \delta_u = \sum_{u \in U} \delta_u$.

Let $T \subseteq \mathcal{V}$ be an independent set in the given graph, in particular an optimum set. We would like to compare $f(\hat{S})$ with $f(T)$. Directly comparing T with $\hat{S}$ is difficult since $\hat{S}$ is obtained by deleting vertices in S along the way; thus a vertex $v \in T \setminus \hat{S}$ may have been discarded due to a vertex $u \in S$ when v was considered but u may not be in $\hat{S}$. Thus, the analysis is broken into two parts that detour through U. First, we relate the value of $f(\hat{S})$ to the value of $f(U)$. This part of the analysis bounds the amount of value lost by kicking out vertices from S during the exchanges. We then relate $f(U)$ and $f(T)$; this is easier because any vertex in T is always compared against *some* subset of vertices in U. Chaining the inequalities from $f(\hat{S})$ to $f(U)$ to $f(T)$ gives the final approximation ratio.

Relating $f(\hat{S})$ to $f(U)$

The analysis is similar to that in [17]. We provide proofs for the sake of completeness. The following claim is easy to see since elements before s can only be deleted from S as the algorithm proceeds.

▷ Claim 30. Over the course of the algorithm, the incremental value $\nu(f, S, s)$ of an element $s \in S$ is nondecreasing.

For a vertex $u \in U \setminus \hat{S}$ we let u' denote the vertex that caused u to be removed from S. And we let $\chi(u)$ denote its incremental value just before it is removed. Therefore, $\chi(u) = \nu\big(f, S_{u'}^-, u\big)$.

▶ **Lemma 31.** *Let $u \in U$ then $\delta_u \geq \beta \sum_{c \in C_u} \nu(f, S_u^-, c)$.*

Proof. Since the vertex u was added to S when it was considered, we have $\delta_u = f(S_u^+) - f(S_u^-)$ where $S_u^+ = S_u^- - C_u + u$. The vertex u was added by the algorithm since $f_S(u) \geq (1+\beta) \sum_{c \in C_u} \nu(f, S, c)$ where $S = S_u^-$. Therefore $\beta \sum_{c \in C_u} \nu(f, S_u^-, c) \leq f_{S_u^-}(u) - \sum_{c \in C_u} \nu(f, S_u^-, c)$. It suffices to prove that $f(S_u^+) - f(S_u^-) \geq f_S(u) - \sum_{c \in C_u} \nu(f, S, c)$ which we do below. For notational convenience let $A = S_u^- - C_u$.

$$
\begin{aligned}
f(S_u^+) - f(S_u^-) &= f(A+u) - f(S_u^-) \\
&= f_A(u) + f(A) - f(S_u^-) \\
&\geq f_{S_u^-}(u) - (f(S_u^-) - f(A)) && \text{by submodularity since } A \subseteq S_u^- \\
&\geq f_{S_u^-}(u) - \sum_{c \in C_u} \nu\big(f, S_u^-, c\big) && \text{by submodularity and defn of } \nu.
\end{aligned}
$$

◀

▶ **Lemma 32.** $\sum_{u \in U \setminus \hat{S}} \chi(u) \leq \beta^{-1} f(\hat{S})$.

Proof. Indeed,

$$
\begin{aligned}
\sum_{u \in U \setminus \hat{S}} \chi(u) &= \sum_{u \in U} \sum_{c \in C_u} \chi(c) && \text{since } \{C_u : u \in U\} \text{ partitions } U \setminus \hat{S} \\
&\leq \sum_{u \in U} \frac{1}{\beta} \sum_{u \in U} \delta_u && \text{from Lemma 31} \\
&= \frac{1}{\beta} f(\hat{S}).
\end{aligned}
$$

◀

The next lemma shows that $f(U)$ is not much larger than $f(\hat{S})$.

▶ **Lemma 33.** $f(U) \leq \big(1 + \beta^{-1}\big) f(\hat{S})$.

Proof. Let $U' = U \setminus \hat{S}$ and let $U' = \{v_{i_1}, \ldots, v_{i_h}\}$ where $i_1 < i_2 \ldots < i_h$. We have $f(U) = f(\hat{S}) + f_{\hat{S}}(U')$. It suffices to upper bound $f_{\hat{S}}(U')$ by $f(\hat{S})/\beta$. For $1 \leq j \leq h$ let $U'_j = \{v_{i_1}, \ldots, v_{i_j}\}$. We have $f_{\hat{S}}(U') = \sum_{j=1}^{h} f_{\hat{S} \cup U'_{j-1}}(v_{i_j})$. We claim that $f_{\hat{S} \cup U'_{j-1}}(v_{i_j}) \leq \chi\big(v_{i_j}\big)$. This follows by submodularity and the fact that $\hat{S} \cup U'_{j-1}$ is a superset of the vertices that are in S when v_{i_j} is deleted. Putting things together,

$$
f_{\hat{S}}(U') = \sum_{j=1}^{h} f_{\hat{S} \cup U'_{j-1}}(v_{i_j}) \leq \sum_{u \in U'} \chi(u) \leq \frac{1}{\beta} f(\hat{S})
$$

where the last inequality follows from Lemma 32. ◀

Relating OPT to $f(U)$

It remains to bound $f(T)$ (for some competing set T) to $f(U)$ and hence to $f(\hat{S})$. The critical question, addressed in the following lemmas, is how to charge the value of elements in T off to elements in U.

▶ **Lemma 34.** *Let $T \subseteq \mathcal{V}$ be an independent set disjoint from U. Each element $u \in U$ appears in the conflict list C_t for at most k vertices $t \in T$.*

Proof. Fix $u \in U$. The set $T \cap N(u) \cap \{v : v > u\}$ consists of precisely the vertices $t \in T$ for which $u \in C_t$. As a subset of T, this set is certainly independent. By definition of k-inductive independence, the cardinality of this set is at most k. ◀

▶ **Lemma 35.** *Let $T \subseteq \mathcal{V}$ be an independent set. Then*

$$
f_U(T) \leq k(1+\beta)(1+\beta^{-1}) f(\hat{S}).
$$

Proof. Since $f_U(T) = f_U(T \setminus U)$, it suffices to assume that T is disjoint from U. For each vertex $t \in T$, since t is not in U, we have $f_{S_t^-}(t) \leq (1+\beta) \sum_{c \in C_t} \nu\big(f, S_t^-, c\big)$. Fix a vertex $u \in C_t$. If $u \in \hat{S}$, then u is in the final output; then we have $\nu\big(f, S_t^-, u\big) \leq \nu\Big(f, \hat{S}, u\Big)$ because the incremental value of an element in S is nondecreasing. If $u \notin \hat{S}$, and u was deleted to make room for some later element u', then we have $\nu\big(f, S_t^-, u\big) \leq \chi(u)$ again because incremental values are nondecreasing.

By the preceding lemma, each element $u \in U$ appears in C_t for at most k choices of t. Therefore, in sum, we have

$$\begin{aligned}
f_U(T) &\leq \sum_{t \in T} f_{S_t^-}(t) && \text{by submodularity,} \\
&\leq (1+\beta) \sum_{t \in T} \sum_{c \in C_t} \nu\big(f, S_t^-, c\big) && \text{since } t \notin U, \\
&\leq k(1+\beta)\left(\sum_{u \in \hat{S}} \nu\Big(f, \hat{S}, u\Big) + \sum_{u \in U \setminus \hat{S}} \chi(u) \right) && \text{Lemma 34 and argument above,} \\
&\leq k(1+\beta)\left(f(\hat{S}) + \sum_{u \in U \setminus \hat{S}} \chi(u) \right) \\
&\leq k(1+\beta)(1+\beta^{-1}) f(\hat{S}) && \text{by Lemma 32}
\end{aligned}$$

as desired. ◀

From here, it is relatively straightforward to get a final approximation bound.

▶ **Theorem 36.** *Given an inductively k-independent graph with a k-inductive ordering, the algorithm* `preemptive-greedy` *returns an independent set $\hat{S}$ such that for any independent set T,*

$$f(T) \leq (k(1+\beta)+1)(1+\beta^{-1}) f(\hat{S}).$$

Proof. Let T be an optimal solution. We have

$$f(T) \leq f_U(T) + f(U) \leq (k(1+\beta)+1)(1+\beta^{-1}) f(\hat{S}) \tag{1}$$

via Lemma 35 and Lemma 33. ◀

The bound is minimized by taking $\beta = \sqrt{1+k^{-1}}$, which at which point

$$f(T) \leq (4k + 2 + o(1)) f(\hat{S}),$$

where the $o(1)$ goes to 0 as k increases. For $k = 1$, the approximation ratio is $3 + 2\sqrt{2}$.

A.2 Randomized preemptive greedy for nonnegative functions

Here we analyze the `randomized-preemptive-greedy` for non-negative submodular functions that may not be monotone. A key observation is that the analysis of `preemptive-greedy` does not invoke the monotonicity of f until the very end, in equation (1). In particular, Lemma 35 and Lemma 33 hold for nonnegative submodular functions.

(1) invokes monotonicity when it takes the inequality $f(U \cup T) \geq f(T)$. Informally speaking, by injecting randomization, we will be able recover a similar inequality, except losing a factor of 4.

Fix a set T. Let $\mathcal{V}'$ sample each element in $\mathcal{V}$ with probability 1/2. Let $T' = T \cap \mathcal{V}'$. Conditional on $\mathcal{V}'$, we have

$$f(U) \leq \left(1 + \beta^{-1}\right) f\left(\hat{S}\right)$$

and

$$f_U(T') \leq k(1+\beta)(1+\beta)^{-1} f\left(\hat{S}\right)$$

via Lemma 33 and Lemma 35 respectively.

Now, conditional on T', $U \setminus T = U \setminus T'$ is a randomized set, where any vertex $v \in \mathcal{V}$ appears in $U \setminus T$ with probability at most 1/2. By Lemma 21,

$$\mathbf{E}[f(U \cup T') \,|\, T'] \geq \frac{1}{2} f(T').$$

We also have, via the concavity of F along any non-negative direction [39],

$$\mathbf{E}[f(T')] = F(\frac{1}{2}\mathbb{1}_T) \geq \frac{1}{2} F(\mathbb{1}_T) = \frac{1}{2} f(T)$$

where $\mathbb{1}_T$ is the indicator vector of T.

Altogether, we have

$$\begin{aligned} f(T) &\leq 2\,\mathbf{E}[f(T')] \leq 4\,\mathbf{E}[f(U \cup T')] \\ &= 4\,\mathbf{E}[f_U(T') + f(U)] \leq 4(k(1+\beta)+1)(1+\beta)^{-1}\,\mathbf{E}\left[f\left(\hat{S}\right)\right], \end{aligned}$$

as desired.