The Minimum Entropy Submodular Set Cover Problem*

Gabriel Istrate[1,2], Cosmin Bonchiş[1,2], and Liviu P. Dinu[3]

[1] Department of Computer Science, West University of Timişoara, Bd. V. Pârvan 4, Timişoara, Romania. Corresponding author's email: gabrielistrate@acm.org
[2] e-Austria Research Institute, Bd. V. Pârvan 4, cam. 045 B, Timişoara, Romania.
[3] Faculty of Mathematics and Computer Science, University of Bucharest, Romania.

Abstract. We study *minimum entropy submodular set cover*, a variant of the submodular set cover problem (Wolsey [22], Fujito [11], etc) that generalizes the minimum entropy set cover problem (Halperin and Karp [12], Cardinal et al. [5])
We give a general bound of the approximation performance of the greedy algorithm using an approach that can be interpreted in terms of a particular type of biased network flows. As an application we rederive known results for the Minimum Entropy Set Cover and Minimum Entropy Orientation problems, and obtain a nontrivial bound for a new problem called the Minimum Entropy Spanning Tree problem.
The problem can be applied to (and is partly motivated by) a worst-case approach to fairness in concave cooperative games.

1 Introduction

Submodularity encodes the notion of *diminishing returns* and plays a crucial role in many problems in combinatorial optimization [10], cooperative game theory [20, 8], information theory [17] and in applications like clustering, learning, natural language and signal processing or constraint satisfaction. Submodular optimization is well-understood: minimization has polynomial time algorithms [19, 13]; maximization is intractable but has efficient approximation algorithms. Minimizing the cost is not the only possible objective for submodular optimization: a problem in computational biology led Halperin and Karp [12] to study a *minimum entropy* version of the set cover problem (MESC). MESC is NP-hard, but the GREEDY algorithm produces [12] an approximate solution with an additive approximation guarantee. The optimal constant is $\log_2(e)$ [5].

It must be stressed that minimizing entropy is a reasonably common scenario: the authors of [5] subsequently studied other combinatorial problems under minimum entropy objectives [6, 7]. Minimal entropy graph coloring is relevant in coding and information theory [2]. Entropy minimization has been applied e.g. to *word segmentation* [21] or (for the non-extensive entropy) to maximum parsimony *haplotype inference* [15].

* Supported by CNCS IDEI Grant PN-II-ID-PCE-2011-3-0981 "Structure and computational difficulty in combinatorial optimization: an interdisciplinary approach."

In this paper we join these two directions, submodularity and combinatorial optimization under a minimum entropy objective, by investigating an *extension of problem MESC* we call *minimum entropy submodular set cover* (MESSC). While the problem is clearly NP-complete (as a generalization of MESC), our main result show that the approximation guarantees of the GREEDY algorithm for MESC extend to MESSC, with the additional appearance of a certain covering parameter that has an interpretation in terms of a type of certain "biased" network flows. This interpretation allows a fairly illuminating rederivation of results in [6, 7] and applications to several new problems, special cases of MESSC.

Besides the conceptual integration, the framework we investigate was developed with several applications in mind. The most important of them (developed in a companion paper [4]) concerns the development of a **worst-case approach to fairness in concave cooperative games** similar in spirit to the *price of anarchy* from noncooperative game theory. We measure inequality of an allocation in the core by the entropy of the associate distribution, and seek allocations in the core minimizing entropy. Here we analyze a concrete example of such a game, the *minimum entropy spanning tree* (MEST) problem.

The plan of the paper is as follows: in Section 2 we briefly review some relevant concepts and notions. In Section 3 define the problems we are interested in and point that they are NP-hard; next we introduce a greedy approach to minimum entropy submodular set cover. Section 4 contains our main result: we quantify the performance of the GREEDY algorithm in terms of an instance-specific "covering constant". We then rederive (in Section 6) existing results on the performance of the GREEDY algorithm for the Minimum Entropy Orientations and Set Cover problems [6, 7]. Section 7 contains an interpretation of the covering constant using network flows that allows us to tighten up our main theorem using a "multi-level" version of our covering constant. As an application we obtain in Section 8 a result on the approximability of the Minimum Entropy Spanning Tree problem that matches the $\log_2(e)$ bound for MESC from [5].

2 Preliminaries

We will use *the Shannon entropy* of a distribution $P = (p_i)_{i\in I}$, defined as $Ent(P) = -\sum_{i\in I} p_i \log_2(p_i)$. We will assume general familiarity with submodular optimization, see e.g. [10]. In particular a set function $f : \mathcal{P}(U) \to \mathbf{R}_+$ will be called **integer** if $range(f) \subseteq \mathbf{Z}$, *monotone* if $f(S) \leq f(T)$ whenever $S \subseteq T \subseteq U$, *submodular* if $f(S) + f(T) \geq f(S \cup T) + f(S \cap T)$ for all $S, T \subseteq U$, *modular* if $f(S) + f(T) = f(S \cup T) + f(S \cap T)$ for all $S, T \subseteq U$, and *polymatroid* if f is monotone, submodular and satisfies $f(\emptyset) = 0$.

An instance of the *(Minimum Cost) Set Cover* (SC) is specified by an universe Z and a family $\mathcal{P} = \{\mathcal{P}_1, \ldots, \mathcal{P}_m\}$ of parts of Z. Each set $\mathcal{P}_i$ comes with a nonnegative *cost* $c(i)$. Given set $X \subseteq Z$, a *cover of* X is a function $g : X \to [m]$ such that for every $x \in X$, $x \in P_{g(x)}$ ("x is covered by $P_{g(x)}$"). The goal is to find a cover g of Z whose cost, defined as $cost[g] = \sum\limits_{j\in range(g)} c(P_j)$, is minimized.

The following classical generalization called *submodular set cover (SSC)* [22, 11] shares many properties with problem SC: we are given an integer polymatroid f and a *cost function* $c : U \to \mathbf{R}_+$. The cost of a set $S \subseteq U$, denoted $cost(S)$, is simply the sum of costs of its elements. Feasible solutions to SSC are subsets $S \subseteq U$ with $f(S) = f(U)$. The goal is to find a feasible subset $S \subseteq U$ of minimum cost $cost(S)$.

For readers not familiar with SSC it is worth discussing the representation of problem SC as a special case of SSC. Given an instance $(X, \mathcal{P})$ of SC of unit costs, define corresponding instance (U, f) of SSC as follows:

1. $U = \{1, 2, \ldots, m\}$.
2. For $S \subseteq U$ define $X_S = \bigcup_{i \in S} P_i$ and $f(S) = |X_S|$.

It is well-known that function f defined above is submodular. A set $S \subseteq U$ with $f(S) = f(U)$ corresponds to a family of parts $\{P_i\}_{i \in S}$ which cover X.

An equivalent restatement of problem SSC relies on the notion of matroid and related concepts (such as basis and flats) we will assume known (see [18]):

Proposition 1. *The following problem is equivalent to SSC: given matroid $M = (U, \mathcal{I})$ and a covering $\mathcal{P} = \{\mathcal{P}_1, \ldots, \mathcal{P}_m\}$ of the universe U find a basis $\mathcal{B}$ of M and a cover $g : \mathcal{B} \to [m]$ of $\mathcal{B}$ such that* c$[g]$ *is minimized.*

Proof. The first part of the statement is a weaker version of a classical result (e.g. Proposition 12.1.9 in [18], where it is further required that sets P_i be flats), stated as follows:

Lemma 1. *Let $g : \mathcal{P}([m]) \to \mathbf{Z}_+$ be an integral polymatroid. Then there exists a matroid $M = (U, \mathcal{I})$ and a family of disjoint parts $P_1, \ldots, P_m$ such that for all $S \subseteq [m]$,*

$$g(S) = rank_M(\cup_{i \in S} P_i).$$

To make the paper self-contained we provide a variant of the proof of Lemma 1 that is simpler to understand than the original version:

Proof. Define

$$U = \{x_{1,1}, x_{1,2}, \ldots x_{1,f(\{1\})}, x_{2,1}, x_{2,2}, \ldots x_{2,f(\{2\})}, \ldots, x_{m,1}, x_{m,2}, \ldots x_{m,f(\{m\})}\}.$$

Also define, for $i \in [m]$, $P_i = \{x_{m,1}, x_{m,2}, \ldots x_{m,f(\{m\})}\}$ Finally, if $A \subseteq U$ define $\lambda_i = |A \cap P_i|$. Then we let $A \in \mathcal{I}$ precisely when there exist nonnegative integers $\mu_1, \mu_2, \ldots, \mu_m$ with

$$\mu_1 + \mu_2 + \ldots + \mu_m = f([m]),$$

for all $S \subseteq [m]$

$$\sum_{i \in S} \mu_i \leq f(S)$$

and for all $i \in [m]$

$$\lambda_i \leq \mu_i$$

It is immediate that $M = (U, \mathcal{I})$ is a matroid (for all bases in $\mathcal{I}$ have the same cardinality $f([m])$.

By definition of $\mathcal{I}$, for any $S \subseteq [m]$ any independent set $B \subseteq \cup_{i \in S} P_i$ has cardinal at most $f(S)$. To show that the equality holds it is enough to construct an independent set W with this property.

This is easy to accomplish: for any permutation $\pi \in S_m$ the natural modification of the greedy algorithm that uses π to create an element of the core of the concave cooperative game with cost function f will yield a basis B in M. Any permutation π such that all indices in S appear before any index in $[m] \setminus S$ leads to basis B such that $|B \cap \cup_{i \in S} P_i| = f(S)$.

□

□

On the other hand, Halperin and Karp introduced [12] the following variation on SC called Minimum Entropy Set Cover (MESC): Consider an instance of SC, $Z = \{u_1, u_2, \ldots, u_n\}$, $n \geq 1$, $\mathcal{P} = \{P_1, P_2, \ldots, P_m\}$. Given $X \subseteq Z$ and cover $g : X \to [m]$ of X, the **entropy of g** is $Ent[g] = -\sum_{i \in [m]} \frac{|g^{-1}(\{i\})|}{|X|} \ln \left(\frac{|g^{-1}(\{i\})|}{|X|} \right)$. The goal is to find a cover g of Z of minimum entropy.

3 Minimum Entropy Submodular Set Cover: Definition, Special cases and applications.

The following definition states the problem we are interested in this paper:

Definition 1. **[Min-Entropy Submodular Set Cover] (MESSC):** *Given a matroid $M = (U, \mathcal{I})$ of rank N, find a basis $\mathcal{B}$ and a cover g of $\mathcal{B}$ minimizing $Ent[g]$.*

MESSC indeed generalizes MESC: if M is the transversal matroid of the bipartite graph $G = (U, \mathcal{P})$ naturally associated to instance $X = (U, \mathcal{P})$ of MESC then solving instance M of MESSC is equivalent to solving instance X of MESC.

Problem MESSC has applications to fuzzy set theory: a fuzzy measure (*Choquet capacity*) is an extension of a probability measure. Submodular Choquet capacities coincide with polymatroids. Various definitions of the notion of the entropy of a Choquet capacity have been proposed in the literature. One such definition, due to Dukhovny [9] is of special interest: for submodular capacities, it is equivalent to our definition of minimum entropy cover (details will be given in the journal version of the paper). Thus solving MESSC is equivalently stated as computing the Dukhovny entropy of a submodular Choquet capacity.

Polymatroids are just a different name for *concave games* in theory of cooperative games [8]. If $(M, \mathcal{P})$ is the rank representation of integer polymatroid f, then the convex hull of incidence vectors of bases in M (the matroid polytope) coincides with the core of the cooperative game defined by f. Since entropy is a concave function, its minimum over $core(f)$ is obtained at a extremal point. That is, finding a basis of minimum entropy is equivalent to finding a minimum entropy imputation in the core. MESSC can, therefore, be restated as follows:

Definition 2. *Given integer-valued polymatroid f, find a vector $(x_1, \ldots, x_m)$ of nonnegative reals satisfying $\sum_{i=1}^{m} x_i = f([m])$ and, for all $S \subseteq [m]$, $\sum_{i \in S} x_i \leq f([S])$ minimizing the entropy of distribution $\frac{x_i}{f([m])}$.*

We will freely switch between the two equivalent definitions of MESSC.

A class of matroids that yields a particular interesting class of cooperative games is that of cycle matroids of a connected graph. We will call the corresponding particularization of MESSC the *Minimum Entropy Spanning Tree (MEST)* problem; it can be specified as follows: we define a *cover of a spanning tree* T in a graph G as an orientation of its edges. The *entropy of a cover* is the entropy of the distribution of indegrees. The objective is to find a spanning tree T of G and a cover of T of minimum entropy. Intuitively in MEST players correspond to graph nodes, each of which may contribute the edges it is adjacent to, each at a unit cost. The submodular (cost) function is $f(S) = |\{v \in V : v \in S \text{ or } \exists w \in S, (v,w) \in E\}|$ for $S \subseteq V$. The goal of the players is to form a spanning tree with the contributed edges. We seek the "most unfair" spanning tree.

Theorem 1. *(proved in the full version [14]) Decision problem associated to MEST is NP-hard.*

Proof. We will use an idea related to the strategy employed to prove the NP-completeness of the Minimum Labeling Spanning Tree Problem [**?**]. We will provide a reduction from the NP-complete problem Minimum Entropy Set Cover to MEST.

Indeed, let $M = (U, \mathcal{P})$ be an instance of Minimum Entropy Set Cover problem, with $U = \{1, 2, \ldots, n\}$ and $\mathcal{P} = \{P_1, P_2, \ldots, P_m\}$.

Define a graph G_M as follows:

1. G_M has one super-node R, $m+n-1$ auxiliary nodes $A_1, \ldots A_{m+n-1}$ (connected only to R), m nodes corresponding to sets $P_1, P_2, \ldots, P_m$ and n nodes corresponding to elements $1, 2, \ldots n$, respectively.
2. Nodes corresponding to P_i are connected to R and to nodes corresponding to elements j, $j \in P_i$.
3. These are all edges of G_M.

To relate the minimum entropy spanning tree on G_M and the minimum cover on M we need the following

Claim. Let $1 \leq a \leq b \leq a+b \leq W$. Then

$$-\frac{a}{W}\log_2(\frac{a}{W}) - \frac{b}{W}\log_2(\frac{b}{W}) \geq -\frac{a-1}{W}\log_2(\frac{a-1}{W}) - \frac{b+1}{W}\log_2(\frac{b+1}{W}).$$

Proof. This is equivalent to

$$\frac{a-1}{W}\log_2(\frac{a}{a-1}) + \log_2(\frac{a}{W}) \leq \frac{b}{W}\log_2(\frac{b+1}{b}) + \log_2(\frac{b+1}{W}).$$

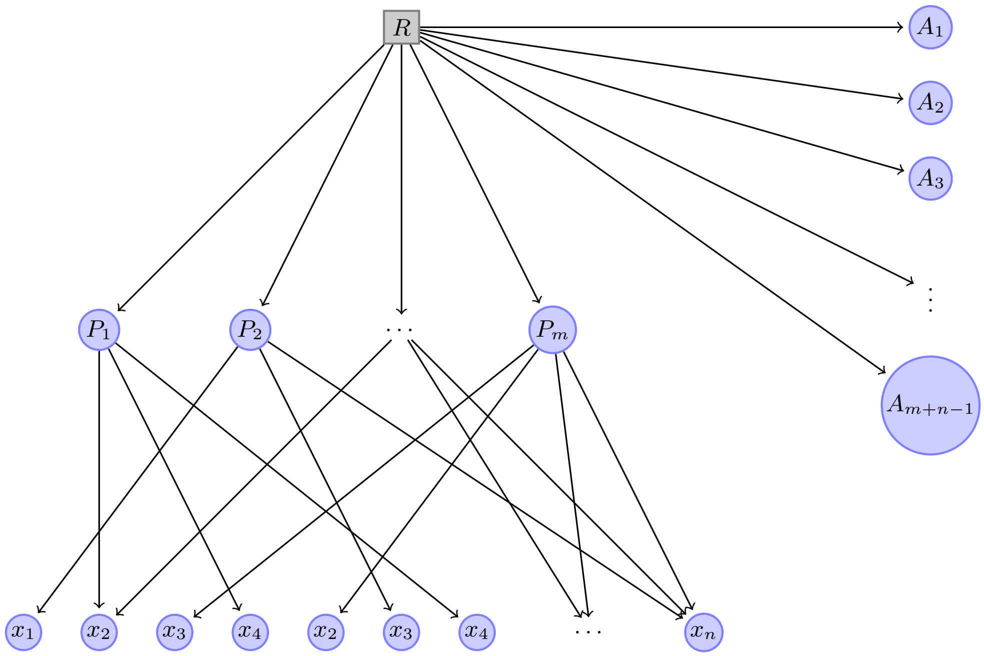

Fig. 1: Graph G_M in the construction from Theorem 1

or

$$\frac{1}{W}\log_2[(1+\frac{1}{a-1})^{a-1}]+\log_2(\frac{a}{W})\leq \frac{1}{W}\log_2[(1+\frac{1}{b})^{b}]+\log_2(\frac{b+1}{W}).$$

This follows easy from the monotone increasing nature of function $g(x) = (1 + \frac{1}{x})^x$. □

So let us consider a spanning tree T_M in G_M of minimum entropy. T_M has to contain edges between R and A_i (as they are the unique edge containing vertex A_i). Moreover, by a simple application of the claim, we may assume without loss of generality that edge (T, A_i) in the minimum entropy solution is contributed by vertex T, who has degree at least $m + n - 1$ from the auxiliary nodes only, thus larger or equal to that of nodes $P_1, \ldots, P_m$, in the spanning tree T_M.

Assume now, for the sake of contradiction, that some node P_i would be a node unconnected to R in T_M. Thus P_i is connected to some non-leaf node j. Deleting edge P_i, j, adding edge R, P_i (contributed by R) and taking into account that the degree of node j in T_M is at most m we would get a tree of lower entropy.

The conclusion of this argument is that each node P_i is connected to R in T_M, with edge (R, P_i) contributed by R.

From this conclusion it follows easily that every node j is connected in T_M to at most one P_i (or else T_M would have a cycle), thus corresponding to a cover

D in M. Moreover, to be a minimum entropy cover C of T_M we may assume that each such edge is contributed by node P_i.

To compute the entropy of cover C of T_M we first consider the contribution of node R, equal to

$$-\frac{2m+n-1}{2(m+n)} \log_2[\frac{2m+n-1}{2(m+n)}]$$

Assuming node P_i has degree d_i in cover C, the contribution of such nodes to the entropy of cover C is

$$-\sum_{i=1}^{m} \frac{d_i}{2(m+n)} \log_2 \frac{d_i}{2(m+n)} = -\frac{1}{2(m+n)} \left[\sum_{i=1}^{m} d_i \left(\log_2 d_i - \log_2 2(m+n)\right)\right] =$$
$$= -\frac{1}{2(m+n)} \left[\sum_{i=1}^{m} d_i \log_2(d_i) - n \log_2 2(m+n)\right] =$$
$$= \frac{n}{2(m+n)} \left[-\sum_{i=1}^{m} \frac{d_i}{n} \log_2 \left(\frac{d_i}{n}\right)\right] + \frac{n}{2(m+n)} \log_2 \frac{2(m+n)}{n}.$$

Thus

$$Ent(C) = -\frac{2m+n-1}{2(m+n)} \log_2 \frac{2m+n-1}{2(m+n)} + \frac{n}{2(m+n)} \log_2 \frac{2(m+n)}{n} + + \frac{n}{2(m+n)} \cdot Ent(D),$$

in particular instance M has a cover of entropy at most λ if and only if instance G_M of MEST has a cover of entropy at most

$$-\frac{2m+n-1}{2(m+n)} \log_2 \left[\frac{2m+n-1}{2(m+n)}\right] + \frac{n \log_2(2+2m/n)}{2(m+n)} + \frac{n}{2(m+n)} \cdot \lambda.$$

□

Finally, we would like to mention a potential practical application of MESSC to *Web Search Diversification*: Matroids are a natural way to encode diversity in web search results [1]. One could formalize this by following special case of MESSC, a generalization of MESC, called *minimum entropy diverse multicover* (MEDM). We are given a bipartite graph of *queries* and *pages*, and integer $k \geq 1$. Each page $p \in P$ has a *type* $t(p)$ from a set of types T. We assume that each query is adjacent to pages of at least k types. Feasible solutions are *k-covers*, i.e. a set of edges covering the n queries, each query by exactly k pages (a *top-k answer*) and *all the types of pages that cover a given query are distinct*. We seek a partial cover with minimal entropy. A justification for this objective function comes from adapting the maximum likelihood approach developed by Halperin and Karp [12] for MESC to our problem.

3.1 The Greedy Algorithm.

We will denote by X^{OPT} an optimal solution for an instance of MESSC. Also, given a permutation $\sigma \in S_m$ define vector X^{σ} as follows: for $1 \leq i \leq m$, $X^{\sigma}(\sigma(i)) = f(\{\sigma(1), \sigma(2), \ldots, \sigma(i)\}) - f(\{\sigma(1), \sigma(2), \ldots, \sigma(i-1)\})$. It is easy to see that for every $\sigma \in S_m$, X^σ is a feasible (but perhaps not optimal) solution to instance f of MESSC, and that one of X^σ, $\sigma \in S_n$, is an optimal solution. This is easy to see using the language of cooperative games: in concave games the core is non-empty, a polytope whose extremal points are those produced greedily on permutations of U, that is X^σ. Since entropy is a concave function, the optimum is reached at some extremal point X^σ.

A GREEDY approximation algorithm is presented in Figure 2. Note that it is well known that the resulting vector is also one of the vectors X^σ. We will use, throughout the rest of the paper, the following notations: $i_1, i_2, \ldots, i_m$ will be the indices chosen by the GREEDY algorithm, in this order. Furthermore, we define for $1 \leq r \leq m$ the *greedy rank function* by $rank(i_r) = r$. For $1 \leq r \leq m$, $W_r = \{i_1, \ldots, i_r\}$ is the set of first r elements added by the GREEDY algorithm; also $W_0 = \emptyset$. $X_r^{GR} = f(W_r) - f(W_{r-1})$ is the increase in the objective function caused by the choice of the r'th element.

GREEDY MESSC:

INPUT: An instance (U, f) of MESSC
$S := \emptyset$, $r := 1$
While $S \neq U$:
    choose $i \in U \setminus S$ that maximizes $f(S \cup \{i\}) - f(S)$ (may be 0)
    $X_r^{GR} := f(S \cup \{i\}) - f(S)$
    $S := S \cup \{i\}$; r+=1
OUTPUT: Vector $X^{GR} = (X_r^{GR})_{r \in [m]}$.

Fig. 2: Greedy algorithm for Minimum Entropy Submodular Set Cover.

4 Main result and definition of the covering coefficient α

By analogy with MESC, we would expect to upper bound the entropy of the cover produced by the GREEDY algorithm by the entropy of the optimal cover plus $\log_2(e)$. We almost accomplish this: our upper bound further depends on a covering constant α. It can be stated as follows:

Theorem 2. *Given a polymatroid $G = (U, f)$, the greedy algorithm produces a solution X^{GR} to the instance G of MESSC related to the optimal cover X^{OPT} by relation:*

$$Ent[X^{GR}] \leq \frac{1}{\alpha} \cdot [Ent[X^{OPT}] + \log_2(e)] + [1 - \frac{1}{\alpha}] \log_2(n) \tag{1}$$

The rest of the section is devoted to precisely defining α. First we define a quantity that will play a fundamental role in our results: for any $1 \leq r, j \leq m$ let $a_r^j = f(W_r) - f(W_{r-1}) - (f(W_r \cup \{j\}) - f(W_{r-1} \cup \{j\}))$. The best way to make sense of the (admittedly unintuitive) definition of the a_r^j coefficients is to particularize them in the case of the set cover problem. In this case coefficients a_r^j

have a very intuitive description: they represent the size of the intersection of the j'th set P_j to the r'th set in the GREEDY solution. Indeed, $f(W_r) - f(W_{r-1})$ is the number of elements newly added by GREEDY at step r, whereas the subtracted term $f(W_r \cup \{j\}) - f(W_{r-1} \cup \{j\})$ is the number of elements that would still be newly added *if P_j were present too.*

Proposition 2. *For any $1 \leq r, j \leq m$ we have $a_r^j \geq 0$.*

When $j \in W_r$ this follows directly from the monotonicity of f. Assume now that $j \notin W_r$, employ the submodularity of f with $S = W_r$, $T = W_{r-1} \cup \{j\}$. □

To define covering coefficient α we introduce a large number of apparently superfluous variables Z_r^j. Intuitively Z_r^j is the portion of optimal solution X_j^{OPT} that can be assigned to cover X_r^{GR}. This explains the equations below: first, one has to allocate all of X_j^{OPT} and no more than that. Second, one cannot allocate to any "set X_r^{GR}" more than "its intersection with X_j^{OPT}". The quoted statements above make full sense, of course, only for regular set cover.

Definition 3. *Given polymatroid G, let $\alpha = \alpha_G$ the smallest positive value such that there exist $Z_r^j \in \mathbf{Z}, Z_r^j \geq 1$ so that the following inequalities hold*

$$\sum_{r=1}^{m} Z_r^j = X_j^{OPT}, 1 \leq j \leq m \tag{2}$$

$$\sum_{j=1}^{m} Z_r^j \leq \alpha \cdot X_r^{GR}, 1 \leq r \leq m \tag{3}$$

Given the discussion above, α is indeed a covering coefficient. It measures the amount of "redundancy" inherent into "assembling" the GREEDY solution from pieces obtained by breaking up the optimal solution.

Proposition 3. *The coefficient α is always greater or equal to 1.*

Proof. Sum all equations (3) for all $r = 1, \ldots, m$. The left-hand side is $\sum\limits_{r=1}^{m} \left(\sum\limits_{j=1}^{m} Z_r^j \right) = \sum\limits_{j=1}^{m} \left(\sum\limits_{r=1}^{m} Z_r^j \right) = \sum_{j=1}^{m} X_j^{OPT} = N.$

On the other hand the right-hand side is $\alpha \cdot \sum\limits_{r=1}^{m} X_r^{GR} \leq \alpha \cdot N$, by the GREEDY algorithm. The result follows. □

5 Proof of main result.

By the greedy choice we infer $X_r^{GR} = f(W_{r-1} \cup \{i_r\}) - f(W_{r-1})$ for any $1 \leq r \leq m$. We first prove several auxiliary results:

Claim. For any $j \in [m]$ we have $\sum\limits_{r=1}^{m} a_r^j = f(\{j\})$.

Proof : $\sum_{r=1}^{m} a_r^j = \sum_{r=1}^{m} (f(W_r) - f(W_{r-1}) - (f(W_r \cup \{j\}) - f(W_{r-1} \cup \{j\}))) =$

$f(W_m) - f(W_0) - (f(W_m \cup \{j\}) - f(W_0 \cup \{j\})) = N - (N - f(\{j\})) = f(\{j\})$

The computations are justified by equalities $f(W_0) = 0, f(W_m \cup \{j\}) = f(W_m) = N$. □

Claim. Given $r, j \in [m]$ we have $f(\{j\}) - \sum_{k=1}^{r} a_k^j = f(W_r \cup \{j\}) - f(W_r)$.

Proof : $\sum_{k=1}^{r} a_k^j = \sum_{k=1}^{r} \big(f(W_k) - f(W_{k-1}) - \big(f(W_k \cup \{j\}) - f(W_{k-1} \cup \{j\})\big)\big) =$

$$= \sum_{k=1}^{r} (f(W_k) - f(W_{k-1})) - \sum_{k=1}^{r} (f(W_k \cup \{j\}) - f(W_{k-1} \cup \{j\})) =$$

$f(W_r) - f(W_0) - (f(W_r \cup \{j\}) - f(W_0 \cup \{j\})) = f(W_r) - f(W_r \cup \{j\}) + f(\{j\})$. □

Claim. We have $\prod_{r=1}^{m} \left(X_r^{GR}\right)^{\alpha X_r^{GR}} \geq \prod_{j=1}^{m} \left(X_j^{OPT}\right)!$.

Proof: By the greedy choice, claims (5), (5) and $Z_r^j \leq a_r^j$:

$$\begin{aligned} X_r^{GR} &= f(W_{r-1} \cup \{i_r\}) - f(W_{r-1}) \geq f(W_{r-1} \cup \{j\}) - f(W_{r-1}) \\ &= f(\{j\}) - \sum_{k=1}^{r-1} a_k^j = \sum_{k=r}^{m} a_k^j \geq \sum_{k=r}^{m} Z_k^j = X_j^{OPT} - \sum_{k=1}^{r-1} Z_k^j \end{aligned}$$

Thus $\prod_{j=1}^{m} \left(\prod_{r=1}^{m} \left(X_r^{GR}\right)^{Z_k^j} \right) \geq \prod_{j=1}^{m} \left(\prod_{r=1}^{m} \left(X_j^{OPT} - \sum_{k=1}^{r-1} Z_k^j \right)^{Z_k^j} \right) \geq \prod_{j=1}^{m} \left(X_j^{OPT}\right)!$

By (3): $\prod_{j=1}^{m} \left(\prod_{r=1}^{m} \left(X_r^{GR}\right)^{Z_k^j} \right) = \prod_{r=1}^{m} \left(X_r^{GR}\right)^{\sum_{j=1}^{m} Z_k^j} \leq \prod_{r=1}^{m} \left(X_r^{GR}\right)^{\alpha X_r^{GR}}$. □

With Claim (5) in hand, we get

$$ENT[X^{GR}] = -\sum_{r=1}^{m} \frac{X_r^{GR}}{n} \log_2 \left(\frac{X_r^{GR}}{n} \right) = -\sum_{r=1}^{m} \frac{X_r^{GR}}{n} \log_2(X_r^{GR}) + \log_2(n) =$$

$$= -\frac{1}{n\alpha} \sum_{r=1}^{m} \log_2 \left(X_r^{GR}\right)^{\alpha X_r^{GR}} + \log_2 n = -\frac{1}{n\alpha} \log_2 \prod_{r=1}^{m} \left(X_r^{GR}\right)^{\alpha X_r^{GR}} + \log_2 n \leq$$

$$= \leq -\frac{1}{n\alpha} \log_2 \prod_{j \in OPT} (X_j^{OPT})! + \log_2 n$$

Using inequality $n! \geq \left(\frac{n}{e}\right)^n$ we infer:

$$\mathbf{ENT}[\mathbf{X}^{\mathbf{GREEDY}}] \leq -\frac{1}{n\alpha}\log_2 \prod_{j\in OPT}\left(\frac{X_j^{OPT}}{e}\right)^{X_j^{OPT}} + \log_2 n =$$

$$-\frac{1}{n\alpha}\sum_{j\in OPT} X_j^{OPT}\left(\log_2 X_j^{OPT} - \log_2 e\right) + \log_2 n = \frac{1}{\alpha}\left(-\sum_{j\in OPT}\frac{X_j^{OPT}}{n}\log_2 X_j^{OPT}\right)$$

$$+\frac{1}{\alpha}\log_2 e + \log_2 n = \frac{1}{\alpha}\big(-\sum_{j\in OPT}\frac{X_j}{n}\log_2\frac{X_j}{n} - \log_2 n\big) + \frac{1}{\alpha}\log_2 e + \log_2 n$$

$$= \frac{\mathbf{1}}{\alpha}\left(\mathbf{ENT}[\mathbf{X}^{\mathbf{OPT}}] + \log_{\mathbf{2}} \mathbf{e}\right) + \left(\mathbf{1} - \frac{\mathbf{1}}{\alpha}\right)\log_{\mathbf{2}} \mathbf{n}$$

6 Applications: Special cases with $\boldsymbol{\alpha = 1}$.

A simple problem where one can determine the value of α is the Minimum Entropy Orientation (MEO) problem [6, 7].The input to MEO is a graph $G = (V, E)$. An *orientation of* G is a function $u : E \to V$ such that for all $e \in E$, $u(e)$ is one of the vertices of e. The entropy of orientation u is defined in an obvious way, as the entropy of the distribution of indegrees. The objective is to find an orientation u of G that minimizes the entropy.

MEO is a special case of MESC: each instance $G = (V, E)$ of MEO can be regarded as an instance of MESC with submodular cost function $f : V \to \mathbf{Z}, f(S) = |\{e \in E : e \cap S \neq \emptyset\}|$. We first recover (using a different method) the upper bound on the performance of the GREEDY algorithm for MEO (an algorithm that is, however, not optimal [6]).

Proposition 4. *For any instance G of MEO $\alpha_G = 1$.*

Proof. A simple application of the definition of f yields $a_r^j = 1$, if $i_r \neq j, (i_r, j) \in E, j \notin W_r; a_r^j = X_r^{GR}$, if $i_r = j; a_r^j = 0$, otherwise. This allows us to turn an orientation of minimum entropy (corresponding to an optimal solution) into the greedy orientation (and define coefficients Z_r^j) as follows:

- At each stage r, after choice of i_r we reorient edges (j, i_r), $j \notin W_r$ that have different orientations in the optimal and greedy solution. Correspondingly define $Z_r^j = 1$ for such edges.
- Also let $Z_r^{i_r}$ be the number of edges (j, i_r) that are oriented towards i_r in both the greedy and the optimal orientation. Note that there are at most $a_r^{i_r} = X_r^{GR}$ such edges.
- Note that an edge that is reoriented at stage r is not reoriented again at a later stage (because of the restriction $j \notin W_r$). Hence the process ends up with the greedy solution. In other words $\sum_{j=1}^{m} Z_r^j = X_r^{GR}$. (as we add one unit for each edge counted by X_r^{GR}).

□

We can also rederive the results of Cardinal et al. on MESC using our approach:

Proposition 5. *For any instance G of MESC $\alpha_G = 1$.*

Proof. For MESC the submodular function is $f(S) = |\cup_{i\in S} P_i|$, for $S \subseteq [m]$. By a direct application of their definition $a_r^j = |\big(P_{i_r} \setminus \cup_{k=1}^{r-1} P_{i_k}\big) \cap P_j|$.

Let $u : [N] \to [m]$ be an optimal solution to MESC, i.e. for any $1 \leq i \leq N$, $i \in P_{u(i)}$ and the cover specified by u has minimum entropy. Denote, for $j = 1, \ldots, m$, $U_j = \{x \in [N] : u(x) = j\}$. $U_j \subseteq P_j$ is the set of elements assigned by cover u to set P_j. Define, for $1 \leq r \leq l$

$$Z_r^j = |U_j \cap \big(P_{i_r} \setminus \cup_{k=1}^{r-1} P_{i_k}\big)|. \tag{4}$$

Then $0 \leq Z_r^j \leq a_r^j$. Moreover

$$\sum_{r=1}^{l} Z_r^j = \sum_{r=1}^{l} |U_j \cap \big(P_{i_r} \setminus \cup_{k=1}^{r-1} P_{i_k}\big)| = |U_j|$$

$$\sum_{j=1}^{m} Z_r^j = \sum_{j=1}^{m} |U_j \cap \big(P_{i_r} \setminus \cup_{k=1}^{r-1} P_{i_k}\big)| = |\big(P_{i_r} \setminus \cup_{k=1}^{r-1} P_{i_k}\big)|,$$

(as each of the two set systems $(U_j)_{j\in[m]}$ and $\big(P_{i_r} \setminus \cup_{k=1}^{r-1} P_{i_k}\big)_{r=1}^{m}$ consists of disjoint sets), hence $X_j = |U_j|$ and Z_r^j satisfy conditions for $\alpha = 1$. □

7 Network flow interpretation of α and a multistage approach.

Theorem 2 is, of course, most interesting when $\alpha_G = 1$, matching the $\log_2(e)$ additive guarantee of MESC. However, there exist instances G of MESSC for which the associated constant α_G is *strictly greater than 1*.

To circumvent this problem we will develop a more powerful technique: we first reinterpret constant α_G using one-stage network flows. This will allow us to generalize our method to multistage flows, characterized by a related constant β_G. A variant of Theorem 2 holds for constant β_G as well. The extension allows us to prove that *the* $\log_2(e)$ *additive guarantee is valid for* **all** *instances of MEST*; the result follows from a multistage flow construction witnessing that for any instance G of MEST $\beta_G = 1$.

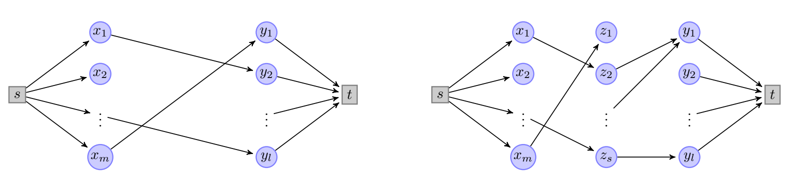

Fig. 3: (a). Network flow interpretation of constant α. (b) Multistage flow network between solutions.

Example 1. For the MEST problem we have

$$f(W_r \cup \{j\}) - f(W_{r-1} \cup \{j\}) = |\{e \in E(G) : e = (i_r, k), k \sim i_r, k \not\sim W_{r-1} \cup \{j\}\}|$$

$$\text{Similarly } f(W_r) - f(W_{r-1}) = |\{e \in E(G) : e = (i_r, k), k \sim i_r, k \not\sim W_{r-1}\}|$$

Therefore $a_r^j = 1$ if $i_r \neq j, i_r \sim j, j \notin W_r; a_r^j = |\{k : k \sim i_r, k \sim j, k \not\sim W_{r-1}\}|$ if $i_r \neq j, i_r \not\sim j, j \notin W_r; a_r^j = X_r^{GR}$ if $i_r = j$; and $a_r^j = 0$ otherwise.

Consider the flow network in Figure 3 (a). In addition to source/sink nodes s, t, F has two layers of nodes; the first layer of nodes corresponding to the optimal solution, the second layer of nodes corresponding to the greedy one. In each layer we have a node for every player in the game. Edges appear between nodes of type k and i_r, with capacity equal to a_k^r. The fact that the first layer of nodes corresponds to the optimal solution is reflected by setting capacity X_j^{OPT} on the edge between node s and node j. Similarly, capacities between node i_r of the second layer and node t are set to value X_r^{GR}. These values are seen as *requests* of node Y_r that may be satisfied by the flow (which in general might send an amount larger than X_r^{GR} to this node)

It follows that $\alpha_G = 1$ amounts to the existence of an integer flow f of value N in the flow network of Figure 3 (a) (that is, f satisfies the request of each node Y_r exactly). More generally, *α is the minimum amount needed to multiply the capacities on the edges entering t in order to accommodate a $s - t$ integer flow with value N.*

Our solution to problem MEO could be easily recast in terms of flows: we construct the flow iteratively, by considering the paths between a node in the first layer and a node in the second layer inductively, in an order determined first by the order of second-layer nodes corresponding to the GREEDY algorithm and then going on nodes in the first layer according to a fixed ordering.

There are lessons to be learned from the construction this flow and our proof of Theorem 2: The key point was that when we had to reorient an edge towards a node in the greedy solution, *we could do so without overflowing this node.* Similarly, the general proof depended on the following the invariant we maintained (*): $X_r^{GR} \geq \sum_{k=r}^{l} Z_k^j$. Condition (*) does not have a direct flow interpretation, since X_r^{GR} is the request, rather than the actual flow value at the given node. However, its relaxation involving α does: the actual flow into node y_r is at most αX_r^{GR}, so requiring that the total flow into node y_r is at least $\sum_{k=r}^{l} Z_k^j$ guarantees the following relaxed version of equation (*): $\alpha \cdot X_r^{GR} \geq \sum_{k=r}^{l} Z_k^j$. We will see (in Proposition 3 below) that the relaxed condition can be applied as well.

We generalize the setting of Theorem 2 by considering flow networks with $q \geq 1$ levels (see Figure 3(b) for $q = 2$). The nodes in each level are ordered according to a fixed ordering, e.g. the ordering induced by the GREEDY algorithm, with nodes not chosen by this algorithm coming after all chosen nodes in a fixed, arbitrary sequence. Capacities correspond either to values a_j^r (if the chosen indices are j and i_r, respectively) or ∞, for edges between nodes with the same index j but on different levels. Note that each path ending in a greedy node with index i_r has finite capacity, at most the capacity of its last edge. We use notation $P : [j \ldots k]$ to indicate the fact that path P starts at node j on the first level and ends at node k on the last. Also write $P \sim v$ to indicate the fact that path P is adjacent to node v. We will also need to consider a total ordering $<$ on paths (explicitly constructed when analyzing particular problems, e.g. MEST):

Definition 4. *A flow f is* admissible with respect to total path ordering $<$ *if for any path P between, say, node X_j and Y_r, the remaining flow into node X_j just before path*

P is considered is at most the final value of the final total flow into node Y_r. Formally $\sum_{Q\sim X_j, P\leq Q} f(Q) \leq \sum_{W\sim Y_r} f(W)$.

Even in a multiple-layer flow network it may not be possible to obtain an admissible flow of value N. As before, the solution is to multiply the capacities of edges into node t by some fixed $\beta \geq 1$.

Definition 5. *Define β_G as the infimum (over all multi-level flow networks corresponding to the optimal and greedy solution) of all values $\beta > 0$ for which there exists a path ordering $<$ and a flow f admissible w.r.t. $<$ such that for every pair of nodes j and r, $\sum_j \left(\sum_{P:[j\ldots i_r]} f_P\right) \leq \beta \cdot X_r^{GR}$. The reader is requested to compare Definitions 3 and this definition.*

Similarly to the proof of Proposition 3, we obtain $\beta = \beta_G \geq 1$ always. On the other hand, admissibility will guarantee in general only a weaker version of Theorem 2 (though no weaker for the main setting we have in mind, $\beta = 1$):

Theorem 3. *Given an instance $G = (N, f)$, of MESSC the greedy algorithm produces a solution X^{GR} satisfying $\beta \cdot Ent[X^{GR}] \leq Ent[X^{OPT}] + \log_2(e) + \beta \log_2(\beta) + (\beta - 1) \log_2(n)$.*

Proof. The proof is almost identical to that of Theorem 2: Let $(X_i)_{i\in[m]}$ an optimal solution of the system from Figure (**??**) and $(y_i)_{i\in[m]}$ the solution generated by the greedy algorithm.

Consider a multi-layer flow network such that equation (**??**) is satisfied with $\beta = \beta_G + \epsilon$, for some $\epsilon > 0$. Let f be the corresponding admissible flow and define $Z_r^j = \sum_{P:[j\ldots i_r]} f_P$.

Claim. We have

$$\prod_{r=1}^{l} [(\beta_G + \epsilon) \cdot y_r]^{(\beta_G+\epsilon)y_r} \geq \prod_{j\in OPT} X_j!.$$

As flow f is admissible,

$$(\beta_G + \epsilon) \cdot y_r \geq \sum_{k=r}^{l} Z_k^j = X_j - \sum_{k=1}^{r-1} Z_k^j.$$

This follows from considering the situation just before setting the flow on the lexicographically smallest path between node j and i_r: $X_j - \sum_{k=1}^{r-1} Z_k^j$ is the amount of unsent flow at node j, to be sent on a path to one of nodes $i_r, \ldots, i_l$. No such path has been considered yet, as they are lexicographically larger.

Therefore,

$$\prod_{j\in OPT} \prod_{r=1}^{l} ((\beta_G + \epsilon) y_r)^{Z_k^j} \geq \prod_{j\in OPT} \left(\prod_{r=1}^{l} \left(X_j - \sum_{k=1}^{r-1} Z_k^j \right)^{Z_k^j} \right) \geq \prod_{j\in OPT} (X_j)!$$

On the other hand, by Definition (**??**)

$$\prod_{j\in OPT} \prod_{r=1}^{l} ((\beta_G + \epsilon) y_r)^{Z_k^j} = \prod_{r=1}^{l} (\beta_G + \epsilon) y_r^{\sum_j Z_r^j} \leq \prod_{r=1}^{l} ((\beta_G + \epsilon) y_r)^{(\beta_G+\epsilon)y_r}.$$

□

The rest of the computation is similar, except that we also have to take the limit $\epsilon \to 0$ in the end, to obtain the desired result.

8 Application to problem MEST.

Proposition 3 applies to problem MEST, yielding a nontrivial special case of MESSC with the same additive constant as that of minimum entropy set cover:

Theorem 4. *For any instance G of MEST, $\beta_G = 1$. Therefore*

$$Ent[X^{GR}] \leq Ent[X^{OPT}] + \log_2(e).$$

Proof. We will create a flow, admissible with respect to some total path ordering $<$, that will witness the fact that $\beta_G = 1$. To do so we first revisit the GREEDY algorithm for MEST (Example **??**).

As discussed there, the GREEDY algorithm builds an "independent set" (forest, in the particular case of MEST) incrementally: edges are only added, but not removed. After some stage k, $1 \leq k \leq l$ the edges added by the GREEDY algorithm connect nodes in W_k to some other nodes. Denote by $\delta(W_k)$ the set of nodes not in W_k but adjacent to some node in W_k (after stage k).

Consider some stage r, $1 \leq r \leq l$. Denote by $C_1, C_2, \ldots, C_p$ the connected components (trees) created by the GREEDY algorithm after stage $r-1$. Node i_r chosen at stage r will connect to some of its adjacent nodes (not already selected), so that the resulted induced graph contains no cycles.

We infer the following:

- Any edge (i_r, x), with x not in $C_1 \cup C_2 \cup \ldots \cup C_p$ is added by the GREEDY algorithm (charged to i_r) at stage r.
- The GREEDY algorithm also adds some of the edges (i_r, x), where x belongs to a connected component among $C_1, C_2, \ldots, C_p$. It can only add such an edge if i_r is not already connected to some node in that component (necessarily a member of W_{r-1}), thus creating a cycle. More precisely, in this case it will add *exactly one edge for each such component it's adjacent to*, merging in effect these components. Even in this case, note that x cannot belong to W_{r-1}, but to $\delta(W_{r-1})$. Indeed, suppose x were an element in W_{r-1}. Edge (i_r, x) was not added when x was considered because it was creating a cycle. But then adding it would create a cycle now as well.

As a consequence of the previous analysis the following holds:

Lemma 2. *Suppose edge (i_r, x) is added by the GREEDY algorithm at stage r. Then*

$$rank(x) > r,$$

where $rank(x)$ is the GREEDY rank of the element x, the stage when the element x was chosen.

Element x clearly cannot belong to W_{r-1}, if x falls in the first case of the previous discussion. As for the second case, by the argument there $x \in \delta(W_{r-1})$, which implies the fact $rank(x) > r$. □

We will also need a flow property that ensures flow admissibility in the particular case of the MEST problem:

Definition 6. *A flow is* biased (with respect to vertex ordering r) *if, for all nodes j, l*

$$\exists P : [j, l], f_P > 0 \Rightarrow [rank(j) \geq rank(l)]. \quad (5)$$

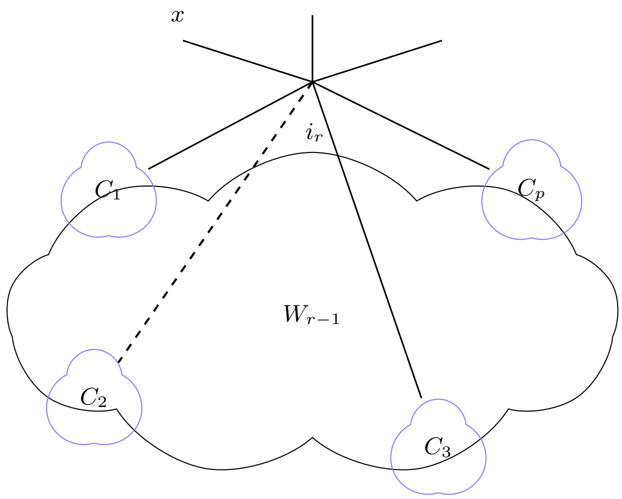

Fig. 4: The GREEDY algorithm for MEST: at stage r, it adds edges from i_r to nodes in components C_1, C_3, C_p (merging them). It does *not* add an edge to component C_2, as there already existed a (shaded) edge from i_r to a node in $C_2 \cap W_{r-1}$. It also adds edges (i_r, x) to nodes x outside of $C_1, C_2, \ldots, C_p$.

The importance of this notion lies in the fact that, while condition (5) is not necessarily satisfied "between the endpoints" of a flow, biased flows can intuitively be "composed", as rank inequality is transitive.

Next we prove the following claim:

Lemma 3. *Given an instance X of the MEST problem let X_{OPT} and Y_{GREEDY} be the vectors corresponding to the optimal and greedy solution, respectively, with elements ordered according to the ordering induced by the greedy algorithm.*

Then there exists a biased flow f with initial values X_{OPT} and final values Y_{GREEDY}.

Let T_{OPT} be the spanning tree (with oriented edges) corresponding to X_{OPT}, and let T_{GREEDY} be the spanning tree with oriented edges corresponding to Y_{GREEDY}. We will construct a multi-level flow network and a greedy flow in stages, corresponding to edge moves that transform T_{OPT} into T_{GREEDY}. Flow values on some node v on an intermediate level correspond to edges oriented towards v at that stage.

Allowed moves are of two basic types:

1. **"Edge reversals".** Consider an edge $e = (w_1, w_2)$ in the current tree, oriented towards w_2. We reorient edge e towards w_1. Biased edge reversals are those with $rank(w_1) < rank(w_2)$.
2. **"Rotations".** Consider an edge $e = (w_1, w_2)$ in the current tree, oriented towards w_1, and let w_3 be another vertex connected to w_1, such that edge (w_1, w_3) is *not* in the tree. Replace (w_1, w_2) by (w_1, w_3).

We will use, in fact, a third type, specified as follows, composed of the previous two moves.

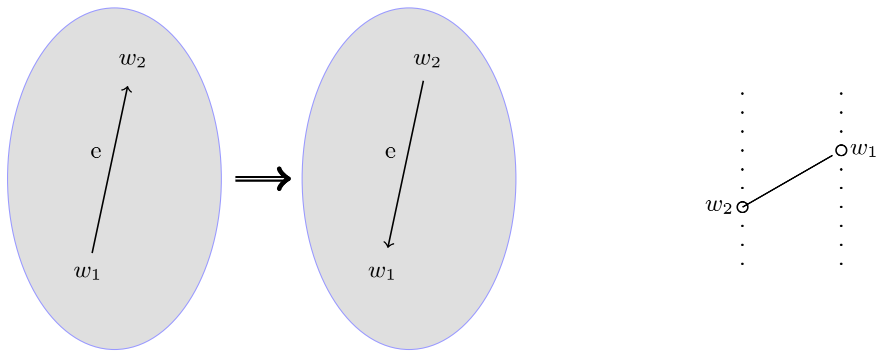

Fig. 5: Edge reversals and associated flow.

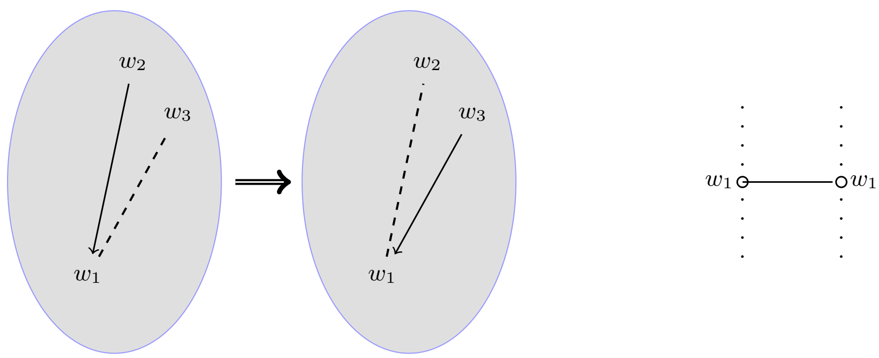

Fig. 6: Edge rotations and associated flow.

3. **"Edge slidings".** Let a, b, c be three nodes Assume that edge (b, c) is in the current tree (oriented towards b) and edge (a, b) is not. Replace edge (b, c) by edge (a, b), oriented towards a by first doing a "rotation" (move of type 2) around node b, and then reorienting edge (a, b) towards a (move of type 1). Biased edge slidings are those corresponding to the case $rank(a) < rank(b) < rank(c)$.

How do these moves correspond to flows ?

1. The edge reversal (w_1, w_2) corresponds (Figure 5) to one unit of flow from node w_2 to node w_1 on the next level.
2. The rotation (w_1, w_2, w_3) as described above corresponds (Figure 6) to one unit of flow from node w_1 to node w_1 on the next level.
3. An edge sliding (Figure 7) involves using two levels of the flow network.

It is easy to see that flows associated to biased edge reversals and rotations, or to preserving an edge (and its orientation) satisfy the biased flow condition. Hence (by composability of biased flows) this holds for biased slidings as well.

It remains to show that we can transform T_{OPT} to T_{GREEDY} using biased moves of type 1,2 and 3. The strategy comprises three parts, described by the following:

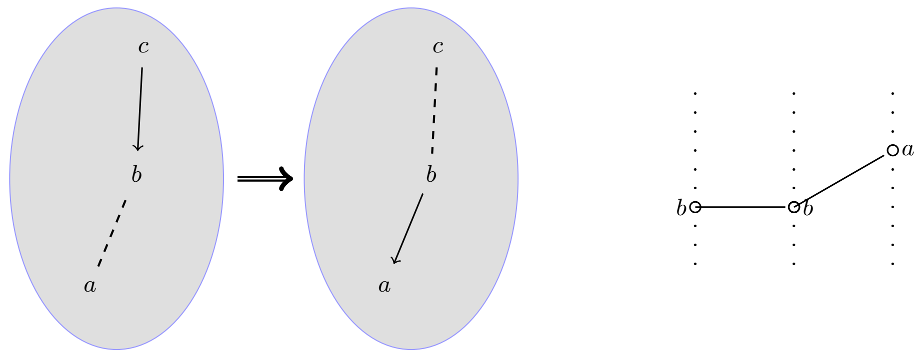

Fig. 7: Edge slidings and associated flow.

(a). Consider all edges $e \in T_{OPT} \cap T_{GREEDY}$, oriented in the same way in both trees. We need to do nothing about them.

(b). Consider all edges $e = (a, b) \in T_{OPT} \cap T_{GREEDY}$, with opposite orientations in the two trees. We will reorient them by an edge reversal.

(c). Consider all edges in $T_{OPT} \setminus T_{GREEDY}$. We will iteratively replace such edges with edges in $T_{GREEDY} \setminus T_{OPT}$, in a way such that the resulting intermediate graphs are in fact trees.

The strategy for iterative replacement employs the current tree, denoted by T_1. Initially $T_1 = T_{OPT}$. Let us consider an edge $e = (a, b) \in T_1 \setminus T_{GREEDY}$. e is in fact in $T_{OPT} \setminus T_{GREEDY}$, as edges added in the iterative process belong to T_{GREEDY}. Assume without loss of generality that $rank(a) > rank(b)$. Since $e \in E$ and $e \notin T_{GREEDY}$, a is connected to a node c with $rank(c) < rank(b)$ in the same component to b (thus creating a cycle that would preclude adding edge e). c is in fact the neighbor of a on the unique path towards the root of T_{GREEDY}.

Eliminating e from T_1 breaks down the set of vertices into two disjoint connected components S and T, with endpoints a, b into disjoint components. Together with the edges of T_{GREEDY}, edge e determines an unique cycle C, consisting of the edges on the path from the root towards a and b, respectively, plus edge e. There exists, therefore an edge $e' \neq e$ on this cycle C, whose endpoints are one in S and one in T. We infer the fact that $e' \in T_{GREEDY} \setminus T_1$.

If e' is on the path from a to the root of T_{GREEDY} then we can use slidings to eliminate edge e from the tree and add edge e' to the tree instead. We may also need to perform the reversal of edge e before we can make the sliding (in case that edge e is oriented towards b in T_{OPT}). But since $rank(c) < rank(b)$ *all resulting flows (including the one corresponding to reorienting edge e and then sliding it) are biased.*

If on the other hand e' is on the path from b to the root of T_{GREEDY} then we first use a greedy edge reversal (possible, as $rank(a) > rank(b)$), then edge slidings to replace e' by e. In both cases, crucially *all resulting flows (including the one corresponding to reorienting edge e and then sliding it) are biased.*

We only have to show that the resulting graph $T_1' = T_1 \setminus e + e'$ is a tree (acyclic), so that the invariant is respected. Indeed, e' has its endpoints in S and T, respectively,

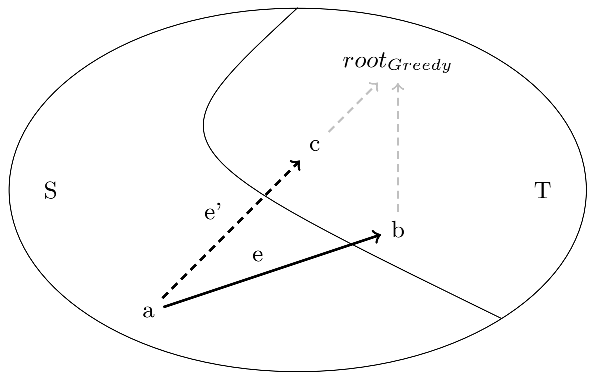

Fig. 8: Iterative transformation of edges from $T_{OPT} \setminus T_{GREEDY}$ into edges in $T_{GREEDY} \setminus T_{OPT}$ in Lemma 3. First the edge e is reoriented towards a. Then we slide it into e'.

and is the unique edge of T_1' with this property. Therefore it is part of no cycle in T_1'. Since T_1 was acyclic, T_1' is acyclic too (hence a tree).

Each edge move of one of the three types above corresponds to a distinct path in the flow network, described as follows:

(a). Edges e shared (with the same orientation, say towards node j) between T_{OPT} and T_{GREEDY} correspond to paths between vertices with the same index j.
(b). Reorienting an edge $e = (j, l)$ from j towards l corresponds to sending one unit of flow from node j on the first level to node l on the next one, and then routing that unit of flow across nodes with label l.
(c). Moves of type [c.] correspond to flows in a similar way, except that they might involve multiple edges (to comply with capacity constraints), and thus multiple nontrivial steps. As argued, though, above, all resulting flows are biased.

□

We exemplify the transformation from the previous lemma in the example from Figure 9. The graph G consists of three nodes, considered in the order $rank(a) < rank(b) < rank(c)$ by the GREEDY algorithm. To go from the optimal solution to the greedy one we first reverse orientation on the edge (a,b). This corresponds to one unit of flow from node b to node a (and subsequently being routed to nodes labeled a). The second transformation consists of first performing an edge reversal on edge (b, c) and then sliding edge (b, c) towards a. The associated flow goes from b to a, going through nodes labeled c, exemplifying the fact that the biased condition is only valid at the extremities of the flow.

Definition 7. *Let $<$ be any total path ordering such that:*

1. *All paths (j, s), $j \neq s$ come before all paths of type (p, p).*
2. *Among paths of the first type $P_i = (j_i, l_i)$, $i = 1, 2$, $l_1 < l_2 \Rightarrow P_1 < P_2$.*

Lemma 4. *The flow f constructed in the proof of Lemma 3 is admissible with respect to $<$.*

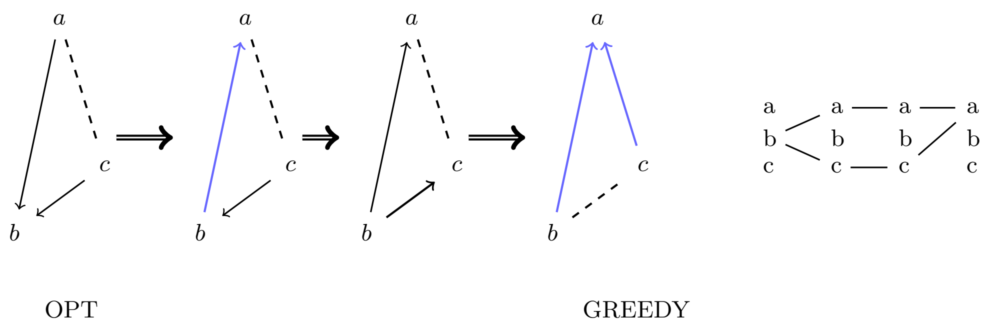

Fig. 9: (a). Transfoming the greedy to the optimal solution using the moves from Lemma 3. (b). The associated biased flow.

Consider a path P between nodes j and i_m, such that $f(P) > 0$. There are two cases:
Case 1: $j \neq i_m$.
Then, by the biased nature of the flow, $j \notin W_{m-1}$, that is node j is a candidate for the greedy algorithm at stage m. Since i_m was chosen instead, the number of edges that would be oriented towards j, should it be chosen at stage m, is less or equal to the number of edges oriented towards i_m at that stage. The second quantity is (by the definition of the GREEDY algorithm) nothing but Y_{i_m}.

To interpret the first quantity, we will associate edges in $T_{OPT} \setminus T_{GREEDY}$ to paths of unit flow starting from j, in such a way that all edges mapped to some path Q, $P \leq Q$, could be oriented towards j should this node be chosen at stage m.

First, note that every possible edge (j, l) with $l \in W_{m-1}$, oriented towards j in the spanning tree T_{OPT} is either present in T_{GREEDY} (but necessarily oriented towards l) or has been replaced (using the process of step [c]) by an edge rooted at some vertex v of even lower rank than l. Thus the edge corresponds to a one unit of flow on some path Q between j and l (or v), a path that is lexicographically smaller than P.

Similarly, if l is not itself in W_{m-1} but is in $\delta(W_{m-1})$, and j is connected to a node in the same connected component as l (after step $m-1$ of the GREEDY algorithm) , then edge (j, l) cannot be in T_{GREEDY}, under any orientation (or else it would create a cycle C). When considered by the GREEDY algorithm it is swapped under step [c.]. Note that cycle C (except edge (j, l)) is contained in $W_{m-1} \cup \delta(W_{m-1})$, with every edge in this cycle being assigned to a vertex in W_{m-1}. Hence the edge also corresponds to a one unit of flow on some path Q between j and l (or v), a path that is lexicographically smaller than P.

Thus any unit of flow from node j sent on a path that is scheduled after path P corresponds to some edge (j, l) not covered by one of the previous two cases. All remaining such edges are among those available for j at stage m, were it to be chosen by the GREEDY algorithm. Their number is, as we saw, at most Y_{i_m}, the flow into i_m in the GREEDY solution.
Case 2: $j = i_m$.
This is trivial, as P is the only path leaving node j at this stage (and is among those that arrive at j).

Hence the flow is admissible. □

To complete the proof of Theorem 4, we simply apply Proposition 3.

□

We conjecture that a similar result holds for problem MEDM, and that it can be proved using multilevel flows (single level ones do not seem powerful enough).

9 Conclusions and open problems

The most important open question raised by our work is whether $\log_2(e)$ is an additive approximation guarantee for **all** instances of MESSC. A deeper matroid-theoretic study of MESSC would be useful in this respect. Finding the optimal additive approximation guarantee for MEST is an open problem as well.

Second, a problem in information theory called *minimum entropy coupling* [16] problem led us to consider an extension of the framework from this paper to *string submodular* functions. Various game-theoretic variations on "worst-case fairness" are an interesting topic for further study, given the large variety of interesting combinatorial games [3].

Finally, we believe that problem MESSC has many potential practical applications, including the outlined applications to diversifying web search results. It would be quite interesting to study more realistic version of problem MEDM and, more generally, to develop such applications.

Bibliography

[1] Z. Abbassi, V. Mirrokni and M. Thakur. Diversity Maximization under Matroid Constraints. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'2013)*, pp. 32–40, ACM.

[2] N. Alon and A. Orlitsky. Source coding and graph entropies. *IEEE Trans. Inform. Theory*, vol.42(5), pp. 1329 – 1339, 1996.

[3] J.M. Bilbao. Cooperative Games on Combinatorial Structures. Kluwer, 2000.

[4] C. Bonchiş and G. Istrate. A parametric worst-case approach to fairness in TU-Cooperative Games. arXiv.org:1208.0283. Revised version forthcoming.

[5] J. Cardinal, S. Fiorini, and G. Joret. Tight results on minimum entropy set cover. *Algorithmica*, 51(1):49–60, 2008.

[6] J. Cardinal, S. Fiorini, and G. Joret. Minimum entropy orientations. *Operations Research Letters*, 36(6):680–683, 2008.

[7] J. Cardinal, S. Fiorini, and G. Joret. Minimum entropy combinatorial optimization problems. *Theory of Computing Systems*, 51(1):4–21, 2012.

[8] T. Driessen. Cooperative Games, Solutions and Applications. Kluwer, 1988.'

[9] Alexander Dukhovny. General entropy of general measures. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(03):213–225, 2002.

[10] S. Fujishige. *Submodular functions and optimization.* Elsevier, 2005.

[11] T. Fujito. Approximation algorithms for submodular set cover with applications. *IEICE Transactions on Information and Systems*, E83-D(3):480–487, 2000.

[12] E. Halperin and R. Karp. The minimum entropy set cover problem. *Theoretical Computer Science*, 348(2–3):340–350, 2005.

[13] S. Iwata and J.B. Orlin. A simple combinatorial algorithm for submodular function minimization. In *Proceedings of the 20th SODA*, pp. 1230–1237.

[14] G. Istrate, C. Bonchiş and L.P. Dinu. The Minimum Entropy Submodular Set Cover Problem. Manuscript, available at http://tcs.ieat.ro/wp-content/uploads/2015/10/lata.pdf

[15] G.Jajamovich, X. Wang. Maximum-parsimony haplotype inference based on sparse representations of genotypes. *IEEE Trans. Sign. Proc.*, 60:2013–2023, 2012.

[16] M. Kovačević, I. Stanojević, and V. Šenk. On the entropy of couplings. *Information and Computation*, 242:369–382, 2015.

[17] M. Madiman and P. Tetali. Information inequalities for joint distributions, with interpretations and applications. *IEEE Trans. Inf. Theory*, 56(6):2699–2713, 2010.

[18] J.G. Oxley. *Matroid theory*, volume 3. Oxford University Press, 2006.

[19] A. Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *J. Combinatorial Theory, Series B*, 80(2):346–355, 2000.

[20] L. Shapley. Cores of convex games. *Int. J. Game Theory*, 1(1):11–26, 1971.

[21] B. Wang. Minimum entropy approach to word segmentation problems. *Physica A: Statistical Mechanics and its Applications*, 293(3):583–591, 2001.

[22] L. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2:385–393, 1982.