Minimum Entropy Submodular Optimization (and Fairness in Cooperative Games)

Cosmin Bonchiş[a,c], Gabriel Istrate[b,c]

[a] *Department of Computer Science, West University of Timişoara, Bd. V. Pârvan 4, Timişoara, RO-300223, Romania.* cbonchis@info.uvt.ro
[b] *Center for the Study of Complexity, Babeş-Bolyai University, Fântânele 30, cam. A-14, RO-400294, Cluj Napoca, Romania*
[c] *e-Austria Research Institute, Bd. V. Pârvan 4, cam. 045 B, Timişoara, RO-300223, Romania. email:* gabrielistrate@acm.org

Abstract

We study minimum entropy submodular optimization, a common generalization of the minimum entropy set cover problem, studied earlier by Cardinal et al., and the submodular set cover problem (Wolsey [Wol82], Fujishige [BIKP01], etc).

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) the definition of worst-case approaches to fairness in concave cooperative games, similar to the notion of price of anarchy in noncooperative settings.

Keywords: submodular set cover, minimum entropy, approximation algorithms, cooperative games, fairness.

1. Introduction

Submodularity is a significant structural property of set functions, encoding the notion of *diminishing returns* and plays a crucial role in many scientific areas including combinatorial optimization [Fuj05], cooperative game theory [Sha71, BDT08], information theory [MT10] and in applications involving

clustering [NKI10], *learning* [GK11], *social and sensor networks* [KKT03], *natural language processing* [LB10], *signal processing* [CK11] or *constraint satisfaction* [AGR11], to give just a few examples. Submodular function optimization is a well-established paradigm and reasonably well-understood: minimization has polynomial time algorithms [Sch00, Iwa03, IO09] while maximization is intractable. On the other hand for well-behaved submodular functions (so called integer polymatroids) $f : \mathcal{P}(S) \to \mathbf{Z}_+$ finding the maximum is simple: the ground set set S is a trivial solution. The interesting problem is finding a solution (a subset $A \subseteq S$ satisfying $f(A) = f(S)$) having minimum cost. This is essentially an instance of the *submodular set cover* problem [Fuj00].

Minimizing the cost of A is not the only possible objective function to investigate in this setting: a maximum likelihood approach to an inference problem in computational biology led Halperin and Karp [HK05] to study a *minimum entropy* version of the set cover problem. Finding an optimal solution is in general NP-hard, but Halperin and Karp showed that the GREEDY algorithm produces an approximate solution whose entropy differs only by a constant factor to that of the optimal solution. A tight estimate was obtained by Cardinal et al. [CFJ08a] who subsequently studied other combinatorial problems under minimum entropy objectives [CFJ08b, CFJ12].

It must be stressed that minimizing entropy is an approach that goes beyond the problem studied by Halperin and Karp: for covering-type problems the connection between maximum-likelihood and minimum entropy is quite general. To give just one example, an even earlier problem that exploited the connection between maximum likelihood and minimum entropy is *word segmentation* [Wan01]. Minimizing entropic measures has other applications: for instance, in [JW12] the authors consider a sparse dictionary-based approach to maximum parsimony haplotype inference via minimizing a non-extensive variant of the entropy called *Tsallis entropy*.

In this paper we unify these two directions, submodular optimization and combinatorial optimization under minimum entropy objective function, by investigating a minimum entropy version of the submodular set cover problem. While the problem is clearly NP-complete, our main result show that the approximation performance of the GREEDY algorithm can be quantified using a covering-like 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 [CFJ08b, CFJ12]. We then showcase the power of the method by providing an upper bound on the performance of

the approximation performance of the greedy algorithm for a new problem, the *minimum entropy spanning tree problem*.

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 [BI12]) concerns the development of a worst-case approaches to fairness in concave cooperative games similar in spirit to the *price of anarchy* in noncooperative settings. The measure we propose are based on entropic concepts such as the Shannon divergence. We briefly outline this direction in Section 9. Other potential applications arise in information theory [ST12] and maximum-likelihood approaches to machine learning (in settings inspired by [GB10, GK11]). We plan to further explore and develop these connections in subsequent work.

The plan of the paper is as follows: in Section 2 we briefly review some relevant concepts and notions. In Section 3 we point out the fact that the problems we are interested in are computationally intractable (NP-complete); we also introduce a greedy approach to minimum entropy submodular set cover. In section 4 we discuss an integer programming formulation of this problem. Section 5 contains our main result: we quantify the performance of the GREEDY algorithm with the help of a "covering constant" developed using the IP in Section 4. We then rederive (in Section 6) existing results on the performance of the GREEDY algorithm for the Minimum Entropy Orientations and the Minimum Entropy Set Cover problems [CFJ08b, CFJ12]. 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 a result on the approximability of the Minimum Entropy Spanning Tree problem. We also briefly discuss the intended application to cooperative game theory.

2. Preliminaries

We will assume general familiarity with submodular optimization, see e.g. [Fuj05]. In particular a set function $f : \mathcal{P}(U) \to \mathbf{R}_+$ will be called *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 *polymatroidal* if f is monotone, submodular and satisfies $f(\emptyset) = 0$.

We will use *the Shannon entropy* of a distribution $P = (p_i)_{i \in I}$, defined as $Ent(P) = -\sum_{i \in I} p_i \log_2(p_i)$.

An instance of the classical *(Minimum Cost) Set Cover* (SC) is specified by an universe U and a family $\mathcal{P} = \{\mathcal{P}_1, \ldots, \mathcal{P}_m\}$ of parts of U. Each set $\mathcal{P}_i$ comes with a nonnegative *cost* $c(i)$. The goal is to cover the set U by a family of parts from $\mathcal{P}$ of minimal total cost.

The following classical extension of the set cover problem [Fuj00] shares many properties with problem SC.

Definition 1. [SUBMODULAR SET COVER] (SSC):

1. *[GIVEN:] A set U and a monotone, submodular function $f : \mathcal{P}(U) \to \mathbf{Z}_+$ and a* cost function *$c : U \to \mathbf{R}_+$. The cost of a set S, denoted $c(S)$, is simply the sum of costs of its elements. Without lost of generality we can assume that $U = \{1, 2, \ldots, m\}$ (also denoted $[m]$). Define also $N = f(U)$.*
2. *[SOLUTIONS:] Subsets $S \subseteq U$ with $f(S) = f(U)$ (such a set is called* feasible*).*
3. *[OBJECTIVE:] To find a feasible subset $S \subseteq U$ of minimum cost.*

In particular, the performance of the Greedy algorithm for SSC was studied by Wolsey [Wol82], who showed that results for SC extend to this setup. Other generalizations were given, among other papers, in [BIKP01].

For readers not familiar with SSC it is worth discussing the representation of problem SC as a special case of SSC, since a similar extension will motivate the technical definition of the problem we are interested in.

Given any 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.

Halperin and Karp introduced [HK05] a variation of the SC problem that employs a different objective function:

Definition 2. [MINIMUM ENTROPY SET COVER (MESC)]: *Let $X = \{x_1, x_2, \ldots, x_n\}$ for some $n \geq 1$ and $\mathcal{P} = \{P_1, P_2, \ldots, P_m\}$ be a*

family of subsets of X which covers X. A cover *is a function $g : X \to [m]$ such that for every $1 \leq i \leq n$,*

$$x_i \in P_{g(i)} (\text{“}x_i \text{ is covered by set } P_{g_i}\text{”})$$

The **entropy of cover g** *is defined as*

$$Ent(g) = -\sum_{i \in U} \frac{g(\{i\})}{g(U)} \ln \left(\frac{g(\{i\})}{g(U)} \right). \tag{1}$$

The objective of MESC is to find a cover g of minimum entropy.

In the same way that problem SC was generalized to SSC, we extend problem MESC from Definition 1 to the following:

Definition 3. [MIN-ENTROPY SUBMODULAR SET COVER] (MESSC):

1. *[GIVEN:] A set U and a polymatroidal function $f : \mathcal{P}(U) \to \mathbf{Z}_+$.*
2. *[SOLUTIONS:] A* cover of f, *that is a modular function $g : \mathcal{P}(U) \to \mathbf{Z}_+$ with $g(U) = f(U)$ and $0 \leq g(S) \leq f(S)$ for all $S \subseteq U$.*
   The entropy of cover g is defined as in equation (1).
3. *[OBJECTIVE:] Find a cover g of f of minimum entropy.*

3. Submodular Optimization with restrictions on solution structure: the Minimum Entropy Spanning Tree Problem

Many submodular optimization problems arise from cooperative games on combinatorial structures [Bil00]. In such games solutions are subject to further constraints: A natural example is the setting where solution components form an independent set in a certain matroid.

Games on matroids, or where the possible coalitions form a matroid have been thoroughly investigated in the literature (e.g. [NZKI97, Bil00, BDJLL01, MRB07]). One could naturally define a "min-entropy" version of minimum-base games on matroids. Such games essentially capture all instances of problem MESSC, as any integer polymatroid can be represented using a set of flats in a certain matroid [Oxl06]. At this moment we are

unable to deal with this problem in full generality. Instead we will concentrate on a special case, a combinatorial problem to which our main result will apply in a fairly nontrivial way.

The problem we consider is a variant of *spanning tree games* a classical topic in the area of cooperative games on combinatorial structures (e.g. [CK73, Bir76, GH81, GH84, FKFH97]):

Definition 4. [MIN-ENTROPY SPANNING TREE] (MEST):

1. *[GIVEN:] A connected graph* $G = (V, E)$.
2. *[SOLUTIONS:] Given* $S \subseteq E$, *a* cover *of* S *is a function* $u : S \to V$ *such that for all* $e \in S$, $u(e)$ *is a vertex of* e.
3. *[OBJECTIVES:] A spanning tree* $T = (V, E(T)) \subseteq G$ *and a cover* u *of* $E(T)$ *that minimizes the entropy*

$$Ent(T; u) = -\sum_{i \in V} \frac{|u^{-1}(\{i\}|}{|E(T)|} \log_2 \left[\frac{|u^{-1}(\{i\}|}{|E(T)|} \right].$$

Intuitively in MEST players correspond to nodes of the graph, each of which can only contribute (some of) the edges it is adjacent to, each edge at a unit cost. The goal of the players is to form a spanning tree with the contributed edges. We seek the "most unbalanced" (costwise) spanning tree.

Unlike many of the settings in the papers quoted above we allow a player to control a *set of edges*, rather than a single one. This particular choice ensures the fact that the cost function is submodular. In contrast, in the more classical variants of spanning tree games only a property weaker than submodularity called *permutational convexity* holds [GH82].

Indeed, one can consider MEST as a problem with matroid restrictions on solution structure by considering the *cycle matroid* $M(G)$ of graph G, the matroid whose independent sets consist of sets $\mathcal{I}$ of edges of G that do not contain a cycle. Bases in this matroid correspond to spanning trees of G. For all $S \subseteq V$ define

$$f(S) = \max\{|\mathcal{I}| : \mathcal{I} \in M(G), \forall e \in \mathcal{I}, e \text{ has at least one vertex in } S\} \quad (2)$$

Lemma 1. *Function* f *from equation (2) is submodular.*

Proof. Define $g(S)$ to be the set of edges adjacent to S. Function g satisfies $g(S \cup T) = g(S) \cup g(T)$ and $g(S \cap T) \subseteq g(S) \cap g(T)$. Let r be the rank function of matroid $M(G)$. Clearly, $f(S) = r(g(S))$.

Appling the submodularity of the rank function r to sets $g(S)$ and $g(T)$ we obtain: $f(S \cup T) + f(S \cap T) \leq r(g(S \cup T)) + r(g(S \cap T)) \leq r(g(S)) + r(g(T)) = f(S) + f(T)$ □

3.1. Computational intractability of problems MESSC and MEST.

Problems MESC, MESSC, MEST defined so far are combinatorial *optimization* problems. Turning them into decision problems is easy, though, in the standard manner: we just add an extra cost parameter λ and ask to decide whether the given instance has a solution of cost at most λ. Without risk of confusion we will use the same name for the optimization problems and their corresponding decision variants.

Since problem Minimum Entropy Set Cover Problem is NP-complete [HK05], so is its generalization MESSC. Theorem 1 shows that this is true for problem MEST as well, providing an alternate class of matroid restrictions (beyond those arising from set cover) for which the associated decision problem is hard.

Theorem 1. *Decision problem MEST is NP-complete.*

The proof of Theorem 1 is given in the Appendix.

3.2. The Greedy Algorithm.

Given the previous result, to solve problems MESSC or MEST we have to either resort to heuristic approaches or polynomial time *approximation algorithms* [Vaz04, WS11]. In this paper we pursue the latter alternative.

An approximation algorithm based on the Greedy approach is presented in Figure 1. Note that it is well known that the resulting vector is a solution[1].

We will use, throughout the rest of the paper, the following notations:

[1]This is easiest seen using the dual language of cooperative games. In so-called *concave games* (see Section 9 below) the core is non-empty [Sha71], a polytope whose extremal points are those produced using a greedy approach on a given permutation of the elements of U. Our algorithm simply produces a particular such permutation.

GREEDY MESSC:

INPUT: An instance (U, f) of MESSC

$S := \emptyset$;
Set w to be the zero vector;
While $(f(S) < f(U))$
  choose $i \in U \setminus S$ that maximizes $f(S \cup \{i\}) - f(S)$;
  $w_i = f(S \cup \{i\}) - f(S)$;
  $S := S \cup \{i\}$;
OUTPUT: Vector $\Delta = (w_i)_{i \in U}$.

Figure 1: Greedy algorithm for Minimum Entropy Submodular Set Cover.

- $i_1, i_2, \ldots, i_l$ will be the indices chosen by the GREEDY algorithm, in this order. Furthermore, we define for $1 \leq r \leq l$ the *greedy rank function* by
$$rank(i_r) = r.$$
We extend function *rank* to all elements of U by considering an arbitrary (but fixed) ordering of such elements.

- For $1 \leq r \leq l$, $W_r = \{i_1, \ldots, i_r\}$ is the set of first r elements added by the GREEDY algorithm; also $W_0 = \emptyset$. $\Delta^r_{GREEDY} = w_{i_r}$ is the increase in the objective function caused by the r'th element chosen by $GREEDY$.

Example 1. *Given a graph $G = (V, E)$, an instance of the problem MEST in Definition 4, one can inductively complete the GREEDY algorithm, constructing a solution (S, u). S is a set of edges of G; initially $S_0 = \emptyset$. At stage r, given the current set of edges S_{r-1} constructed so far and the index i_r chosen by the GREEDY algorithm, consider an independent set I_r of cardinality $f(W_r)$ defined in equation (2). Employing the exchange axiom of the cycle matroid we complete the independent set S_{r-1} to an independent set S_r by adding $f(W_r) - f(W_{r-1})$ elements from I_r. Finally, extend u to S_r by defining $u(e) = i_r$ for all $e \in S_r \setminus S_{r-1}$ (all such edges are adjacent to i_r).*

4. Integer Programming Formulation and a Covering Coefficient.

The minimum entropy set cover problem can be formulated as an integer programming problem (Figure 2 (a)). Similarly we can express MESSC (Figure 2 (b), with convention $0\log(0) = 0$) by a rather artificial integer program, whose usefulness will become clearer at a later stage. In a nutshell, the program provides a simple way to define the quantities measuring the performance of the greedy algorithm in our main result.

$$min \left[\sum_{S} -x_S \frac{|S|}{n} \log\left(\frac{|S|}{n}\right) \right] \quad (3)$$

s.t.

$$\sum_{S \subseteq T} x_S \geq 1, \ T \in \mathcal{P}(U)$$

$$x_S \in \{0,1\}$$

(a) MESC

$$\min \left[\sum_{i \in [m]} \sum_{\lambda=0}^{f(\{i\})} -x_{i,\lambda} \frac{\lambda}{n} \log\left(\frac{\lambda}{n}\right) \right] \quad (4)$$

s.t.

$$\sum_{\lambda=0}^{f(\{i\})} x_{i,\lambda} = 1, \ i \in [m]$$

$$\sum_{i \in U-S} \sum_{\lambda=0}^{f(\{i\})} \lambda x_{i,\lambda} \geq f(U) - f(S), \ S \subseteq [m]$$

$$x_{i,\lambda} \in \{0,1\}$$

(b) MESSC

Figure 2: **Minimum Entropy Integer Programming formulations**

Proposition 1. *Given an instance of the MESSC, its solutions are in one-to-one correspondence to solutions of IP problem defined in Figure 2 (b).*

Proof. Given a solution $z = (z_j)_{j \in [m]}$ of MESSC define $\overline{x}_{j,\lambda}$ for $j \in [m]$ and $0 \leq \lambda \leq f(\{j\})$ as follows:

$$\overline{x}_{j,\lambda} = \begin{cases} 1 & \text{if } z_j = \lambda \\ 0 & \text{otherwise.} \end{cases}$$

Claim 1. *If z is a solution to MESSC then $\overline{x}$ is a feasible solution to system (4).*

Proof. Since z is a cover, $0 \leq z_j \leq f(\{j\})$ for every $j \in [m]$ hence obviously for any such j there is exactly one λ, $0 \leq \lambda \leq f(\{j\})$ with $\overline{x}_{j,\lambda} = 1$.

Thus $\sum_{\lambda=0}^{f(\{j\})} \overline{x}_{j,\lambda} = 1$ and $\sum_{\lambda=0}^{f(\{j\})} \lambda \overline{x}_{j,\lambda} = z_j$.

From the definition of cover $\sum_{j \in S} z_j \leq f(S)$ and $\sum_{j \in U} z_j = f(U)$ therefore the second inequality follows. □

Conversely, given a solution w of (4), define for $j \in [m]$

$$X_j = \sum_{\lambda=0}^{f(\{j\})} \lambda w_{j,\lambda}. \tag{5}$$

Claim 2. *$(X_j)_{j \in [m]}$ defined above is a solution to MESSC whose entropy is equal to the value of the objective function of (4) for w.*

Proof. Equation (5) and system (4) ensure the fact that $0 \leq X_j \leq f(\{j\})$ for any $j \in [m]$, and $\sum_{j \in S} X_j \leq f(S)$ for any $S \subseteq [m]$, hence X is a cover.

Since for any $j \in [m]$ we have $\sum_{\lambda=0}^{f(\{j\})} w_{j,\lambda} = 1$, exactly one such term in the above equality is 1, and the result immediately follows. □

□

We now come up to a quantity that will play a fundamental role in our results below: for any $1 \leq r \leq l$ we define

$$a_r^j = f(W_r) - f(W_{r-1}) - \left(f(W_r \cup \{j\}) - f(W_{r-1} \cup \{j\})\right). \tag{6}$$

The best way to make sense of the (admittedly unintuitive) definition of the a_r^j coefficients above is to particularize them in the case of the set cover problem. Observation 1 below shows that 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.

Observation 1. *Let us consider the setting in Example 2. Then*

$$a_r^j = |X_{W_r} \setminus X_{W_{r-1}}| - |X_{W_r} \setminus (X_{W_{r-1}} \cup P_j)| = |(X_{W_r} \setminus X_{W_{r-1}}) \cap P_j|$$

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

$$\min[\sum_{j\in[m]}\sum_{\lambda=1}^{f(\{j\})} -\frac{\lambda}{n}\log\left(\frac{\lambda}{n}\right)x_{j,\lambda}] \qquad (7)$$

s.t.

$$\sum_{\lambda=0}^{f(\{j\})} x_{j,\lambda} = 1,\ j\in[m] \qquad \sum_{i\in U-S} X_i \geq f(U)-f(S),\ S\subseteq[m]$$

$$\sum_{\lambda=0}^{f(\{j\})} \lambda x_{j,\lambda} = X_j,\ j\in[m] \qquad X_j = \sum_{r=1}^{l} Z_r^j, \forall j\in[m]$$

$$x_{i,\lambda}\in\{0,1\} \qquad 0\leq Z_r^j\leq a_r^j, Z_r^j\in\mathbf{Z}$$

Figure 3: Redundant IP formulation of MESSC

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

We will find it useful to introduce a large number of apparently superfluous variables in system (4) as presented in Figure 3. Intuitively Z_r^j is the portion of optimal solution X_j that can be assigned to cover "the r'th set in the greedy solution". This explains the newly introduced constraints: first, one has to allocate all of X_j and no more than that. Second, one cannot allocate to any "set i_r" more than "its intersection with X_j". The quoted statements above make full sense, of course, only for regular set cover (Example 2)

To state the main result we need:

Definition 5. *Given polymatroid G, let $\alpha=\alpha_G$ the smallest positive value such that there exists an optimal solution of system (4) that can be completed to a solution of system (7) by defining Z_r^j so that inequalities*

$$\sum_{j=1}^{m} Z_r^j \leq \alpha\cdot\Delta_r^{GREEDY} \qquad (8)$$

hold for any $1\leq r\leq l$.

Given the discussion above, intuitively α is a covering coefficient. It measures the amount of inherent "redundancy" into coverings of the GREEDY solution by pieces obtained by breaking up the optimal solution. In this sense, measure α has a somewhat similar flavor to the *curvature of a submodular cost function* defined by Wan et al. in [WDPW10]. Of course, there are some differences as well: the latter measure is relevant for *minimum cost* submodular optimization. It also does not directly involve the GREEDY solution (whereas, in the interest of tightness, our concept does).

Proposition 3. *The coefficient α that satisfies system (8) is always greater or equal to* 1.

Proof. Sum all equations (8) for all $r = 1, \ldots, l$.
The left-hand side is

$$\sum_{r=1}^{l}\left(\sum_{j=1}^{m} Z_r^j\right) = \sum_{j=1}^{m}\left(\sum_{r=1}^{l} Z_r^j\right) = \sum_{j=1}^{m} X_j = N. \tag{9}$$

On the other hand the right-hand side is

$$\alpha \cdot \sum_{r=1}^{l} \Delta_r^{GREEDY} \leq \alpha \cdot N, \tag{10}$$

by the GREEDY algorithm. The result follows. □

5. Main result.

In this section we state and prove our main result

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

$$Ent[f_{GREEDY}] \leq \frac{1}{\alpha} \cdot [Ent[f_{OPT}] + \log_2(e)] + [1 - \frac{1}{\alpha}] \log_2(n) \tag{11}$$

Proof.
Let $(X_j)_{j\in[m]}$ an optimal solution of the system from Figure (3) and $(y_i)_{i\in[m]}$ the solution generated by the greedy algorithm.

By the greedy choice we infer

$$y_r = f(W_{r-1} \cup \{i_r\}) - f(W_{r-1})$$

for any $1 \leq r \leq l$, with $y_i = 0$ for other i.

We first prove several auxiliary results:

Claim 3. *For any $j \in [m]$ we have*

$$\sum_{r=1}^{l} a_r^j = f(\{j\})$$

Proof. By the definition of a_r^j:

$$\begin{aligned}
\sum_{r=1}^{l} a_r^j &= \sum_{r=1}^{l} (f(W_r) - f(W_{r-1}) - (f(W_r \cup \{j\}) - f(W_{r-1} \cup \{j\}))) \\
&= f(W_l) - f(W_0) - (f(W_l \cup \{j\}) - f(W_0 \cup \{j\})) \\
&= n - (n - f(\{j\})) = f(\{j\}).
\end{aligned}$$

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

On the other hand, we have:

Claim 4. *For any $1 \leq r \leq l$ and $j \in [m]$ we have:*

$$f(\{j\}) - \sum_{k=1}^{r} a_k^j = f(W_r \cup \{j\}) - f(W_r).$$

Proof.

$$\begin{aligned}
\sum_{k=1}^{r} a_k^j &= \sum_{k=1}^{r} (f(W_k) - f(W_{k-1}) - (f(W_k \cup \{j\}) - f(W_{k-1} \cup \{j\}))) \\
&= \sum_{k=1}^{r} (f(W_k) - f(W_{k-1})) - \sum_{k=1}^{r} (f(W_k \cup \{j\}) - f(W_{k-1} \cup \{j\})) \\
&= f(W_r) - f(W_0) - (f(W_r \cup \{j\}) - f(W_0 \cup \{j\})) \\
&= f(W_r) - f(W_r \cup \{j\}) + f(\{j\}).
\end{aligned}$$

□

Claim 5. *Given coefficient α from the definition (5) we have*

$$\prod_{r=1}^{l} y_r^{\alpha y_r} \geq \prod_{j=1}^{m} X_j!$$

Proof. By the greedy choice, Claim (4) and Claim (3)

$$\begin{aligned} y_r &= f(W_{r-1} \cup \{i_r\}) - f(W_{r-1}) \geq f(W_{r-1} \cup \{j\}) - f(W_{r-1}) \\ &= f(\{j\}) - \sum_{k=1}^{r-1} a_k^j = \sum_{k=r}^{l} a_k^j \end{aligned}$$

By the system in Figure (3) we have $Z_r^j \leq a_r^j$ and we obtain:

$$y_r \geq \sum_{k=r}^{l} Z_k^j = X_j - \sum_{k=1}^{r-1} Z_k^j$$

Therefore,

$$\prod_{j=1}^{m} \left(\prod_{r=1}^{l} (y_r)^{Z_k^j} \right) \geq \prod_{j=1}^{m} \left(\prod_{r=1}^{l} \left(X_j - \sum_{k=1}^{r-1} Z_k^j \right)^{Z_k^j} \right) \geq \prod_{j=1}^{m} (X_j)!$$

On the other hand, by Definition (5)

$$\prod_{j=1}^{m} \left(\prod_{r=1}^{l} (y_r)^{Z_k^j} \right) = \prod_{r=1}^{l} (y_r)^{\sum_{j=1}^{m} Z_k^j} \leq \prod_{r=1}^{l} (y_r)^{\alpha y_r}.$$

□

With Claim (5) in hand, we get

$$\begin{aligned} ENT[f_{GREEDY}] &= -\sum_{r=1}^{l} \frac{y_r}{n} \log_2 \left(\frac{y_r}{n} \right) = -\sum_{r=1}^{l} \frac{y_r}{n} \log_2(y_r) + \log_2(n) \\ &= -\frac{1}{n} \frac{1}{\alpha} \sum_{r=1}^{l} \log_2 y_r^{\alpha y_r} + \log_2 n = -\frac{1}{n} \frac{1}{\alpha} \log_2 \prod_{r=1}^{l} y_r^{\alpha y_r} + \log_2 n \\ &\leq -\frac{1}{n} \frac{1}{\alpha} \log_2 \prod_{j \in OPT} X_j! + \log_2 n \end{aligned}$$

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

$$\begin{aligned}
ENT[f_{GREEDY}] &\leq -\frac{1}{n}\frac{1}{\alpha}\log_2 \prod_{j\in OPT}\left(\frac{X_j}{e}\right)^{X_j} + \log_2 n \\
&= -\frac{1}{n}\frac{1}{\alpha}\sum_{j\in OPT} X_j\left(\log_2 X_j - \log_2 e\right) + \log_2 n \\
&= \frac{1}{\alpha}\left(-\sum_{j\in OPT}\frac{X_j}{n}\log_2 X_j\right) + \frac{1}{\alpha}\log_2 e + \log_2 n \\
&= \frac{1}{\alpha}\left(-\sum_{j\in OPT}\frac{X_j}{n}\log_2 \frac{X_j}{n} - \log_2 n\right) + \frac{1}{\alpha}\log_2 e + \log_2 n \\
&= \frac{1}{\alpha}\left(ENT[f_{OPT}] + \log_2 e\right) + \left(1 - \frac{1}{\alpha}\right)\log_2 n
\end{aligned}$$

□

6. Applications: Minimum Entropy Set Cover, Minimum Entropy Orientation.

A simple problem where one can determine the value of α is the Minimum Entropy Orientation (MEO) problem [CFJ08b, CFJ12]. Indeed, we recover (using a different method) the upper bound on the performance of the GREEDY algorithm for MEO (an algorithm that is, however, not optimal [CFJ08b]). The result will be generalized next to problem MESC (with an even simpler proof). We have chosen to include it here, though, in this form as the proof is going to be useful in the analysis of problem MEST.

Definition 6. [MIN-ENTROPY ORIENTATION (MEO)]:

1. *[GIVEN:] A graph $G = (V, E)$.*
2. *[SOLUTIONS:] 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 edge e.*
3. *[OBJECTIVE:] To find an orientation u of G that minimizes*

$$Ent(S; u) = -\sum_{i\in[V} \frac{|u^{-1}(\{i\}|}{|E|}\log_2\left[\frac{|u^{-1}(\{i\}|}{|E|}\right].$$

The MEO problem is a special case of MESC with sets corresponding to vertices and elements to edges adjacent to the given vertex. Each instance $G = (V, E)$ of MEO can be regarded as an instance of MESC with submodular cost function

$$c(S) = |\{e \in E : E \cap S \neq \emptyset\}|.$$

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

Proof. A simple application of formula (6) yields

$$a_r^j = \begin{cases} 1, \text{ if } i_r \neq j, (i_r, j) \in E, j \notin W_r \\ \Delta_r^{GREEDY}, \text{ if } i_r = j \\ 0, \text{ otherwise.} \end{cases} \tag{12}$$

This choice allows us to turn an orientation of minimum entropy (corresponding to an optimal solution $(X_i)_{i\in[m]}$) into the greedy orientation 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 $a_r^{i_r} = \Delta_r^{GREEDY}$ 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

  $$\sum_j Z_r^j = \Delta_r^{GREEDY}.$$

  (as we add one unit for each edge counted by Δ_r^{GREEDY}).

□

We generalize the result above as follows:

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

Proof. First remember that, as noted in Observation 1, for MESC

$$a_r^j = |(X_{W_r} \setminus X_{W_{r-1}}) \cap P_j| \tag{13}$$

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 $1 \leq r \leq l$

$$Z_r^j = |U_j \cap (X_{W_r} \setminus X_{W_{r-1}})|. \tag{14}$$

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

$$\sum_{r=1}^{l} Z_r^j = \sum_{r=1}^{l} |U_j \cap (X_{W_r} \setminus X_{W_{r-1}})| = |U_j|$$

$$\sum_{j=1}^{m} Z_r^j = \sum_{j=1}^{m} |U_j \cap (X_{W_r} \setminus X_{W_{r-1}})| = |X_{W_r} \setminus X_{W_{r-1}}|,$$

(as each of the two state systems $(U_j)_{j\in[m]}$ and $((X_{W_r} \setminus X_{W_{r-1}}))_{r=1}^{l}$ consists of disjoint sets) hence $X_j = |U_j|$ and Z_r^j satisfy system in Figure (3) and equation (8) with $\alpha = 1$. □

7. Network flow interpretation and extension of the main result.

Sometimes the application of our main result to specific problems is not quite as straightforward as above. An example is the Minimum Entropy Spanning Tree problem from Definition 4. Intuitively, in this case we would also like to apply Theorem 2 by proving that $\alpha_G = 1$. However, this second result is not easy to obtain. To understand why, we will first reinterpret constant α_G using network flows. This will allow us to eventually define a related constant β_G. The difference between α and β is that roughly α is defined in terms of one-stage network flows, whereas in β we will allow multistage constructions.

We will prove a variant of Theorem 2 for constant β, then we will give a multistage flow construction witnessing that for any instance (G, w) of MEST $\beta = 1$. This will prove the desired result.

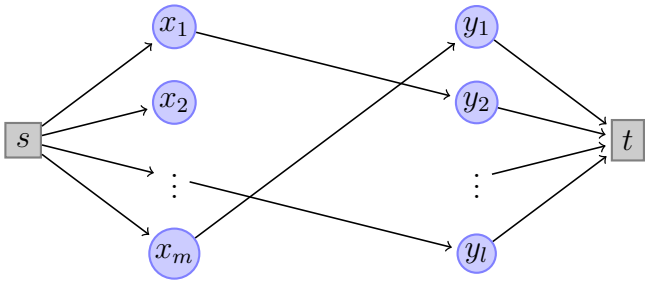

Figure 4: Network flow interpretation of constant α in Definition 5.

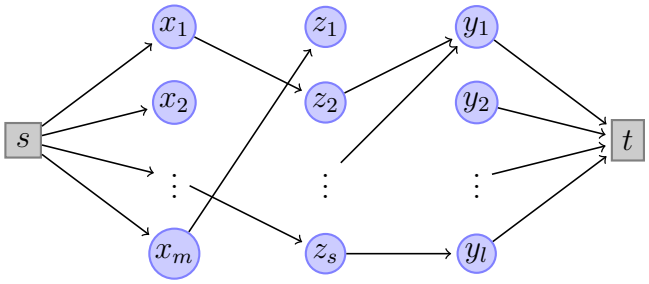

Figure 5: Multistage flow network between solutions

Example 2. *For the MEST problem we have*

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

Similarly

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

Therefore

$$a_r^j = \begin{cases} 1, \ \textit{if } i_r \neq j, i_r \sim j, j \notin W_r \\ |\{k : k \sim i_r, k \sim j, k \not\sim W_{r-1}\}|, \ \textit{if } i_r \neq j, i_r \not\sim j, j \notin W_r \\ \Delta_r^{GREEDY}, \ \textit{if } i_r = j \\ 0, \ \textit{otherwise.} \end{cases} \tag{15}$$

Consider, though, the flow network in Figure 4. 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 Δ_r^{GREEDY}. 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 Δ_r^{GREEDY} to this node)

It follows that $\alpha_G = 1$ amounts to the existence of a flow f of value n in the flow network of Figure 4 (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 the sink* t *in order to accommodate a flow with value* n *from* s *to* t.

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

$$y_r \geq \sum_{k=r}^{l} Z_k^j \tag{16}$$

Condition (16) does not have a direct flow interpretation, since y_r 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 αy_r, 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 (16): $\alpha \cdot y_r \geq \sum_{k=r}^{l} Z_k^j$. We will see (in Proposition 6 below) that the relaxed condition can be applied as well.

We will generalize the setting of Theorem 2 by considering flow networks with $q \geq 1$ levels (see Figure 5 for $q = 2$). The nodes in each level are ordered according to a fixed ordering, the same for all levels, say 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 in this network 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 will use notation $P : [j \dots k]$ to indicate the fact that path P starts at node j on the first level and ends at node k on the last level. We will also use notation $P \sim v$ to indicate the fact that path P is adjacent to node v.

We will consider a total ordering $<$ on paths which will be explicitly constructed later.

Definition 7. *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*

$$\sum_{Q \sim X_j, P \leq Q} f(Q) \leq \sum_{W \sim Y_r} f(W). \tag{17}$$

Consider now a multiple-layer flow network corresponding to the optimal and greedy solutions, respectively (that is, the capacities of edges from s/into t are determined by the values of these solutions). Even with multiple layers it might not be possible to obtain an admissible flow of value n. As before, the solution is to multiply the capacities of edges leading into node t by some fixed amount β.

Definition 8. *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,*

$$\sum_j \left(\sum_{P:[j \dots i_r]} f_P \right) \leq \beta \cdot \Delta_r^{GREEDY}. \tag{18}$$

The reader is requested to compare Definitions 5 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 inequality (16): an extra β factor is needed on the left-hand side (though this is not going to be weaker for the main setting we have in mind, $\beta = 1$).

With this discussion we generalize Theorem 2 as follows:

Proposition 6. *Given an instance $G = (N, f)$, of MESSC the greedy algorithm produces a solution f_{GREEDY} satisfying*

$$\beta \cdot Ent[f_{GREEDY}] \leq Ent[f_{OPT}] + \log_2(e) + \beta \log_2(\beta) + (\beta - 1)\log_2(n). \quad (19)$$

The proof is almost identical to that of Theorem 2: Let $(X_i)_{i\in[m]}$ an optimal solution of the system from Figure (3) and $(y_i)_{i\in[m]}$ the solution generated by the greedy algorithm.

Consider a multi-layer flow network such that equation (18) 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 6. *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 (8)

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

□

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 the minimum entropy spanning tree problem.

Finally we are in position to apply the result in Proposition 6 by proving the following:

Theorem 3. *For any instance G of MEST, $\beta_G = 1$. That is, if f_{GREEDY} is the cover provided by the GREEDY algorithm and f_{OPT} is the optimal cover*

$$Ent[f_{GREEDY}] \leq Ent[f_{OPT}] + \log_2(e).$$

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 1).

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:

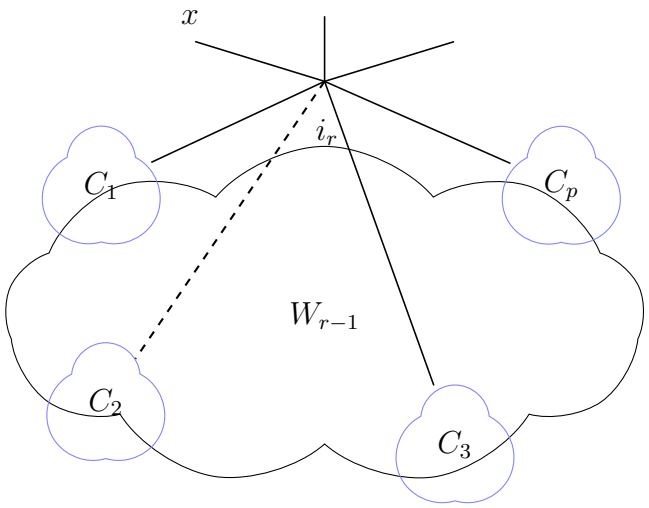

Figure 6: The GREEDY algorithm for MEST: at stage r, it adds edges from i_r to nodes in components C_1, C_3, C_p (merging them). It does *not* add an edge to component C_2, as there already existed a (shaded) edge from i_r to a node in $C_2 \cap W_{r-1}$. It also adds edges (i_r, x) to nodes x outside of $C_1, C_2, \ldots, C_p$.

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

$$\exists P : [j, l], f_P > 0 \Rightarrow [rank(j) \geq rank(l)]. \tag{20}$$

The importance of this notion lies in the fact that, while condition (20) 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 $rank(w_1) < rank(w_2)$.

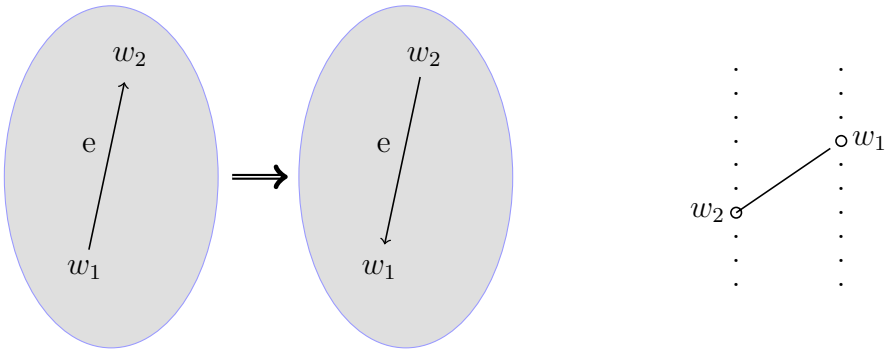

Figure 7: Edge reversals and associated flow.

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.

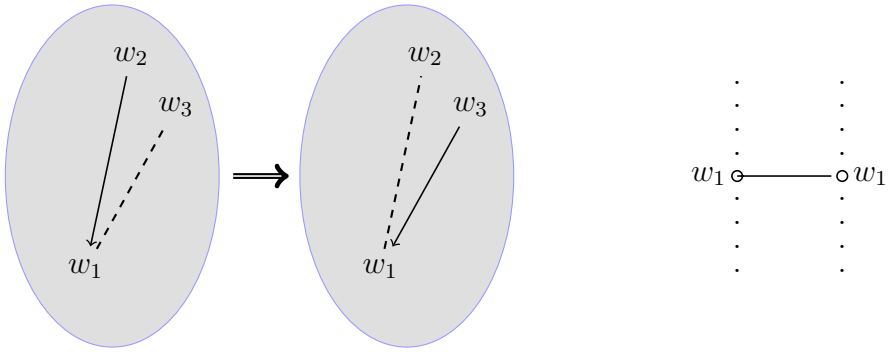

Figure 8: Edge rotations and associated flow.

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 $rank(a) < rank(b) < rank(c)$.

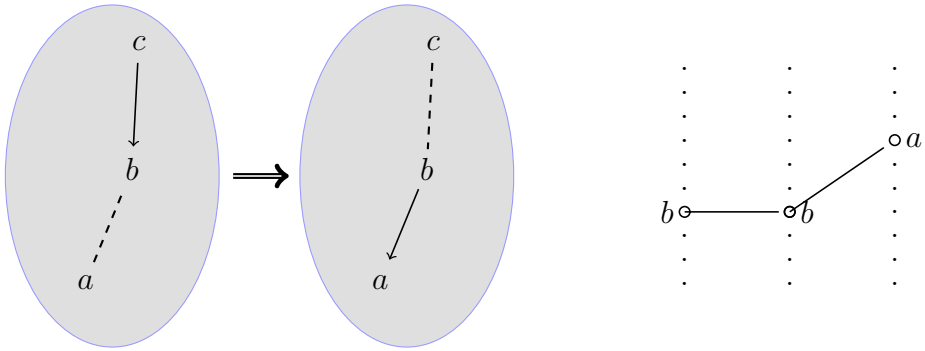

Figure 9: Edge slidings and associated flow.

How do these moves correspond to flows ?

1. The edge reversal (w_1, w_2) corresponds (Figure 7) 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 8) to one unit of flow from node w_1 to node w_1 on the next level.

3. An edge sliding (Figure 9) 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:

(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

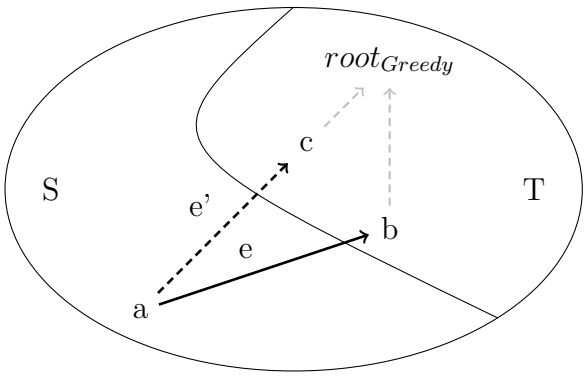

Figure 10: Iterative transformation of edges from $T_{OPT} \setminus T_{GREEDY}$ into edges in $T_{GREEDY} \setminus T_{OPT}$ in Lemma 3. First the edge e is reoriented towards a. Then we slide it into e'.

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, 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.

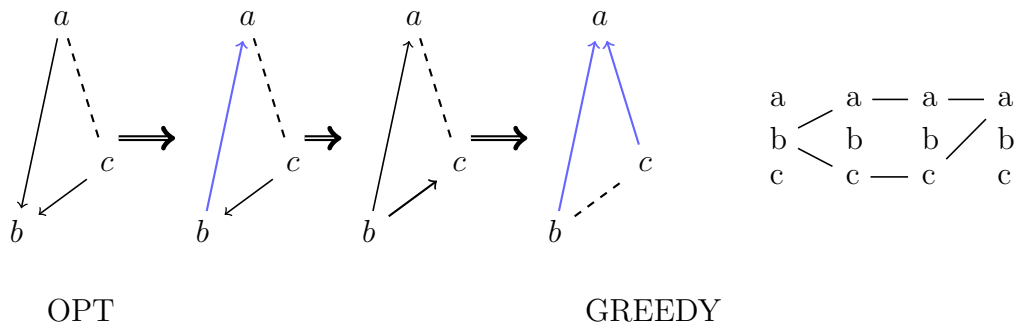

Figure 11: (a). Transfoming the greedy to the optimal solution using the moves from Lemma 3. (b). The associated biased flow.

(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 11. 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 10. *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 $<$.*

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 3, we simply apply Proposition 6. □

9. Application to Cooperative Games

The setting in this paper has an alternative reformulation using the language of cooperative game theory [BDT08]. Indeed, a problem in this area is our main target application and originally motivated our research. It forms the subject of a companion paper [BI12]. Here we only provide a brief outline of our approach.

A *transferable utility (TU) coalitional (cost) game* consists of a finite set of players U and a monotone function (called *characteristic cost function*) $c : \mathcal{P}(U) \to \mathbf{R}$ that satisfies $c(\emptyset) = 0$. A set $T \subseteq U$ is called a *coalition*, with $T = U$ called *the grand coalition*. A TU game is called *concave* if its cost function is submodular. Denote N the cardinal of U. A *cost allocation (imputation)* is simply a vector $x = (x_1, \ldots, x_N) \in \mathbf{R}_+^N$.

The notion of rationality of a cost allocation is embodied by the *core of a cooperative game* $G = (U, c)$ defined as the set of cost allocations that satisfy the following conditions:

1. **Efficiency:** $\sum_{i \in U} x_i = c(U)$.
2. **Individual rationality:** $0 \leq x_i \leq c(\{i\})$, for all $i \in U$.
3. **Coalitional rationality:**

$$\sum_{i \in S} x_i \leq c(S), \forall\, \emptyset \neq S \subseteq U;$$

Clearly core elements in a concave game can be regarded as solutions to the corresponding instance of submodular set cover.

Allocations in the core may be seen as "rational", but they need not necessarily be "fair". The classical approach to fairness in cost allocations is axiomatic, and identifies the celebrated *Shapley Value* [Rot88] as the unique cost allocation satisfying several natural conditions. However, despite its intrinsic attractiveness and conceptual power, the Shapley value may be inappropriate as a "fair" solution concept for many reasons, including the setting of coalitions with a dynamic structure , in games (necessarily not concave) for which the Shapley value is not in the core, or in the presence of social preferences in favor of other social arrangements (e.g. egalitarianism [DR89]).

This is not to say that any particular alternative to the Shapley value would fare better: for one, any such measure would violate at least one of the axioms that uniquely identify the Shapley value, and may have other drawbacks.

The situation is somewhat similar to the issues arising in non-cooperative settings with respect to Nash equilibria, the canonical concept of rationality in that setting. Just as the core of a cooperative game may be very large, a non-cooperative game may have multiple Nash Equilibria. Some of these equilibria may be "suboptimal" the selfish nature of goals pursued by individual agents may lead to suboptimal system performance. As long as we cannot exclude some of these equilibria there is no way to rule out suboptimal behavior.

The seminal work of Roughgarden and Tardos [RT02],[Rou05] has opened an interesting avenue in dealing with the multiplicity of equilibria in noncooperative settings. Instead of attempting to propose a normative solution to equilibrium selection, their price of anarchy measure takes a worst-case perspective, quantifying the degradation in system performance due to uncoordinated behavior, measured on the *worst* strategy profile still compatible with individual rationality.

The objective of paper [BI12] is to propose approach with a similar philosophy for cooperative games. Rather than attempting to postulate any particular "fair solution" of a cooperative game, we will investigate the fairness of an arbitrary allocation in the core.

Fairness will be measured with respect to a "baseline" cost distribution q, deemed "reasonable". Given an arbitrary cost distribution p we will use the *Shannon divergence* $D(p||q) = \sum_{i \in I} p_i \log_2(p_i/q_i)$ to measure the "distance" of distribution p with respect to the baseline distribution q. Note that D is not a metric (as it does not satisfy the triangle inequality) but is a *pseudo-metric* and has been employed before as a "distance" between two distributions.

Given a cooperative game G, its *worst-case fairness* with respect to cost allocation q is defined as the supremum of $D(p||q)$ over all distributions p arising from an imputation in $core(G)$.

Depending on the way to select q we may have several versions of the worst-case fairness measures, including the following ones:

- *strictly egalitarian:* q is the uniform distribution $q_i = 1/N$.
- *egalitarian:* q is the egalitarian solution of Dutta and Ray [DR89].
- *marginalist:* q is the cost distribution corresponding to the Shapley value.

The strictly egalitarian worst-case fairness can be directly related to the setting of this paper, as in that case Shannon divergence directly relates to entropy. Certainly this seting is the the most controversial in terms of applicability, though strict egalitarianism as an approach has featured in a substantive manner in economics and other social sciences.One can further justify its study on grounds related to *mechanism design*: suppose the cooperative game is not "given" but can be imposed on the set of players. Viewed this way strictly egalitarian worst-case fairness is a measure of the *design*, rather than the resulting cost allocation.

On the other hand the connection between divergence and entropy extends (perhaps in a more limited setting) to marginalist approaches too: for certain games (including the *induced subgraph games* from [DP94]) one can quantify the performance of the GREEDY and other approximation algorithms to the marginalist worst-case fairness. Once again we refer to [BI12], where full details will be provided.

10. Conclusions

Obtaining a tight result for entropy minimization problem on matroids remains a topic for further research. We believe that $\beta = 1$ at least for a large class of particular versions of this problem. Formulating and proving such a result remains still open, though. Even in the case of MEST we don't know whether our result is optimal.

The game theoretic investigations opened by our results have multiple variations: notions of "worst-case fairness" can be investigated for a variety of combinatorial games [BDT08], for various other solution concepts such as the ϵ-core, the least core, the kernel or the nucleolus.

Finally, as mentioned, the problem we studied has some promising potential applications. It would be interesting to develop these applications.

Acknowledgments

Both authors contributed equally to this work.

The first author has been supported by a project on Postdoctoral national programs, contract CNCSIS PD_575/2010.

The corresponding author has been supported by a project on Postdoctoral programs for sustainable development in a knowledge-based society, contract POSDRU/89/1.5/S/60189, cofinanced by the European Social Fund

via the Sectorial Operational Program for Human Resource Development 2007-2013.

Appendix: Proof of Theorem 1

We will use an idea related to the strategy employed to prove the NP-completeness of the Minimum Labeling Spanning Tree Problem [CL97]. 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 7. *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

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

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

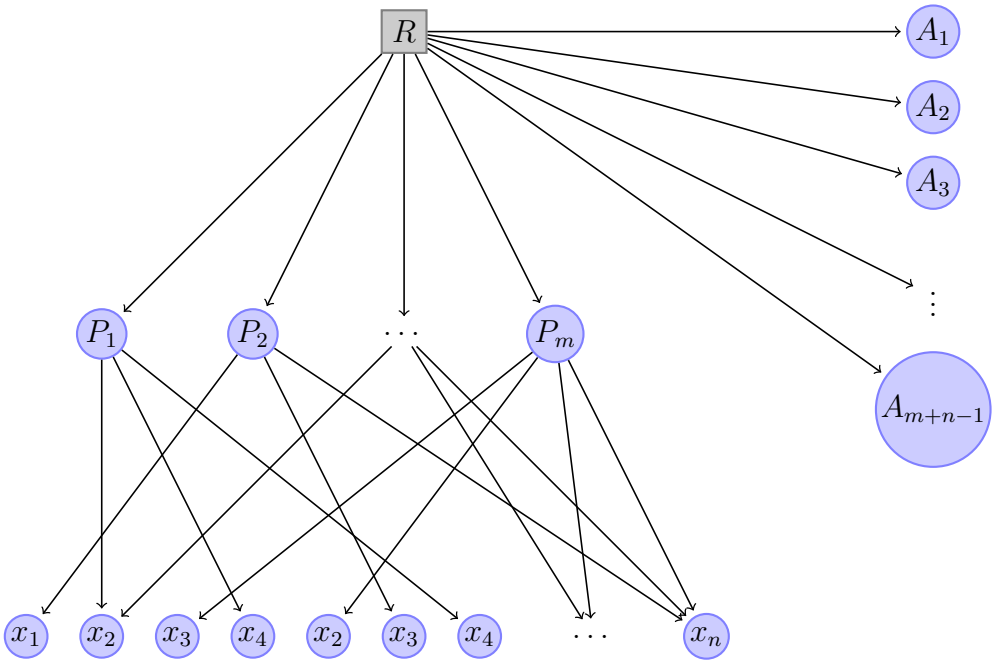

Figure 12: Graph G_M in the construction from Theorem 1

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

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

□

References

[AGR11] Y. Azar, I. Gamzu, and R. Roth. Submodular MAX-SAT. *Proceedings of ESA 2011*, pages 323–334, 2011.

[BDJLL01] J.M. Bilbao, T. Driessen, A. Jiménez Losada, and E. Lebrón. The Shapley value for games on matroids: The static model. *Mathematical Methods of Operations Research*, 53(2):333–348, 2001.

[BDT08] R. Branzei, D. Dimitrov, and S. Tijs. *Models in cooperative game theory*. Springer Verlag, 2008.

[BI12] C. Bonchiş and G. Istrate. A parametric worst-case approach to fairness in TU-Cooperative Games. Submitted manuscript. arXiv.org:1208.0283, 2012, 2012.

[BIKP01] J Bar-Ilan, G. Kortsarz, and D. Peleg. Generalized submodular cover problems and applications. *Theoretical Computer Science*, 250(1–2):179–200, 2001.

[Bil00] J.M. Bilbao. *Cooperative games on combinatorial structures*. Springer, 2000.

[Bir76] C.G. Bird. On cost allocation for a spanning tree: a game theoretic approach. *Networks*, 6(4):335–350, 1976.

[CFJ08a] J. Cardinal, S. Fiorini, and G. Joraet. Tight results on minimum entropy set cover. *Algorithmica*, 51(1):49–60, 2008.

[CFJ08b] J. Cardinal, S. Fiorini, and G. Joret. Minimum entropy orientations. *Operations Research Letters*, 36(6):680–683, 2008.

[CFJ12] J. Cardinal, S. Fiorini, and G. Joret. Minimum entropy combinatorial optimization problems. *Theory of Computing Systems*, 51(1):4–21, 2012.

[CK73] A. Claus and D.J. Kleitman. Cost allocation for a spanning tree. *Networks*, 3(4):289–304, 1973.

[CK11] V. Cevher and A. Krause. Greedy dictionary selection for sparse representation. *IEEE Journal of Selected Topics in Signal Processing*, 5(5):979, 2011.

[CL97] R.S. Chang and S.J. Leu. The minimum labeling spanning trees. *Information Processing Letters*, 63(5):277–282, 1997.

[DP94] X. Deng and C.H. Papadimitriou. On the complexity of cooperative solution concepts. *Mathematics of Operations Research*, pages 257–266, 1994.

[DR89] B. Dutta and D. Ray. A concept of egalitarianism under participation constraints. *Econometrica: Journal of the Econometric Society*, pages 615–635, 1989.

[FKFH97] U. Faigle, W. Kern, S.P. Fekete, and W. Hochstättler. On the complexity of testing membership in the core of min-cost spanning tree games. *International Journal of Game Theory*, 26(3):361–366, 1997.

[Fuj00] T. Fujita. Approximation algorithms for submodular set cover with applications. *IEICE Transactions on Information and Systems*, E83-D(3):480–487, 2000.

[Fuj05] S. Fujishige. *Submodular functions and optimization*. Elsevier, 2005.

[GB10] A. Guillory and J. Bilmes. Interactive submodular set cover. *The Twenty-Seventh International Conference on Machine Learning (ICML '10)*, 2010.

[GH81] D. Granot and G. Huberman. Minimum cost spanning tree games. *Mathematical Programming*, 21(1):1–18, 1981.

[GH82] D. Granot and G. Huberman. The relationship between convex games and minimum cost spanning tree games: A case for permutationally convex games. *SIAM Journal on Algebraic and Discrete Methods*, 3:288, 1982.

[GH84] D. Granot and G. Huberman. On the core and nucleolus of minimum cost spanning tree games. *Mathematical Programming*, 29(3):323–347, 1984.

[GK11] D. Golovin and A. Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, pages 427–286, 2011.

[HK05] E. Halperin and R. Karp. The minimum entropy set cover problem. *Theoretical Computer Science*, 348(2–3):340–350, 2005.

[IO09] S. Iwata and J.B. Orlin. A simple combinatorial algorithm for submodular function minimization. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1230–1237. Society for Industrial and Applied Mathematics, 2009.

[Iwa03] S. Iwata. A faster scaling algorithm for minimizing submodular functions. *SIAM Journal on Computing*, 32(4):833–840, 2003.

[JW12] G.H. Jajamovich and X. Wang. Maximum-parsimony haplotype inference based on sparse representations of genotypes. *Signal Processing, IEEE Transactions on*, 60(4):2013–2023, 2012.

[KKT03] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

[LB10] H. Lin and J. Bilmes. Multi-document summarization via budgeted maximization of submodular functions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 912–920. Association for Computational Linguistics, 2010.

[MRB07] F. Maffioli, R. Rizzi, and S. Benati. Least and most colored bases. *Discrete Applied Mathematics*, 155(15):1958–1970, 2007.

[MT10] M. Madiman and P. Tetali. Information inequalities for joint distributions, with interpretations and applications. *Information Theory, IEEE Transactions on*, 56(6):2699–2713, 2010.

[NKI10] K. Nagano, Y. Kawahara, and S. Iwata. Minimum average cost clustering. *Advances in Neural Information Processing Systems*, 23:1759–1767, 2010.

[NZKI97] H. Nagamochi, D.Z. Zeng, N. Kabutoya, and T. Ibaraki. Complexity of the minimum base game on matroids. *Mathematics of Operations Research*, pages 146–164, 1997.

[Oxl06] J.G. Oxley. *Matroid theory*, volume 3. Oxford University Press, 2006.

[Rot88] A. Roth, editor. *The Shapley Value: essays in honor of Lloyd S. Shapley*. Cambridge University Press, 1988.

[Rou05] T. Roughgarden. *Selfish Routing and the Price of Anarchy*. M.I.T. Press, 2005.

[RT02] T. Roughgarden and É. Tardos. How bad is selfish routing? *Journal of the ACM*, 49(2):236–259, 2002.

[Sch00] A. Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *Journal of Combinatorial Theory, Series B*, 80(2):346–355, 2000.

[Sha71] L.S. Shapey. Cores of convex games. *International Journal of Game Theory*, 1(1):11–26, 1971.

[ST12] Marek Smieja and Jacek Tabor. Partition reduction for lossy data compression problem. *CoRR*, abs/1204.0078, 2012.

[Vaz04] V. Vazirani. *Approximation Algorithms*. Springer Verlag, 2004.

[Wan01] B. Wang. Minimum entropy approach to word segmentation problems. *Physica A: Statistical Mechanics and its Applications*, 293(3):583–591, 2001.

[WDPW10] P.J. Wan, D.Z. Du, P. Pardalos, and W. Wu. Greedy approximations for minimum submodular cover with submodular cost. *Computational Optimization and Applications*, 45(2):463–474, 2010.

[Wol82] L. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2:385–393, 1982.

[WS11] D.P. Williamson and D.B. Shmoys. *The design of approximation algorithms*. Cambridge University Press, 2011.