Advances in Greedy Algorithms

Edited by
Witold Bednorz

I-Tech

Published by In-Teh

In-Teh is Croatian branch of I-Tech Education and Publishing KG, Vienna, Austria.

www.in-teh.org
Additional copies can be obtained from:
publication@ars-journal.com

First published November 2008
Printed in Croatia

A catalogue record for this book is available from the University Library Rijeka under no. 120115050
Advances in Greedy Algorithms, Edited by Witold Bednorz
p. cm.
ISBN 978-953-7619-27-5
1. Advances in Greedy Algorithms, Witold Bednorz

Preface

The greedy algorithm is one of the simplest approaches to solve the optizmization problem in which we want to determine the global optimum of a given function by a sequence of steps where at each stage we can make a choice among a class of possible decisions. In the greedy method the choice of the optimal decision is made on the information at hand without worrying about the effect these decisions may have in the future. Greedy algorithms are easy to invent, easy to implement and most of the time quite efficient. However there are many problems that cannot be solved correctly by the greedy approach. The common example of the greedy concept is the problem of 'Making Change' in which we want to make a change of a given amount using the minimum number of US coins. We can use five different values: dollars (100 cents), quarters (25 cents), dimes (10 cents), nickels (5 cents) and pennies (1 cent). The greedy algorithm is to take the largest possible amount of coins of a given value starting from the highest one (100 cents). It is easy to see that the greedy strategy is optimal in this setting, indeed for proving this it suffices to use the induction principle which works well because in each step either the procedure has ended or there is at least one coin we can use of the actual value. It means that the problem has a certain optimal substructure, which makes the greedy algorithm effective. However a slight modification of 'Making Change', e.g. where one value is missing, may turn the greedy strategy to be the worst choice. Therefore there are obvious limits for using the greedy method: whenever there is no optimal substructure of the problem we cannot hope that the greedy algorithm will work. On the other hand there is a lot of problems where the greedy strategy works unexpectedly well and the purpose of this book is to communicate various results in this area. The key point is the simplicity of the approach which makes the greedy algorithm a natural first choice to analyze the given problem. In this book there are discussed several algorithmic questions in: biology, combinatorics, networking, scheduling or even pure mathematics, where the greedy algorithm can be used to produce the optimal or nearly optimal answer.

The book was written in 2008 by the numerous authors who contributed the publication by presenting their researches in a form of a self-contained chapters. The idea was to coordinate the international project where specialists all over the world can share their knowledge on the greedy algorithms theory. Each chapter comprises a separate study on some optimization problem giving both an introductory look into the theory the problem comes from and some new developments invented by author(s). Usually some elementary knowledge is assumed, yet all the required facts are quoted mostly in examples, remarks or theorems. The publication may be useful for all graduates and undergraduates interested in the algorithmic theory with the focus on the greedy approach and applications of this

method to various concrete examples. Most of scientists involved in the project are young at the full strength of their career, hence the presented content is fresh and acquaints with the new directions where the theory of greedy algorithms evolves to.

On the behalf of authors I would like to acknowledge all who made the publication possible, in particular to Vedran Kordic who coordinated this huge project. Many thanks also for those who helped in the manuscripts preparation making useful suggestions and finding errors.

November 2008

Editor

Witold Bednorz
Warsaw,
Poland,

Contents

Quasi-Concave Functions and Greedy Algorithms

Yulia Kempner[1], Vadim E. Levit[2] and Ilya Muchnik[3]
[1] *Holon Institute of Technology,*
[2] *Ariel University Center of Samaria,*
[3] *Rutgers - the State University of New Jersey,*
[1,2]*Israel*
[3]*USA*

1. Introduction

Many combinatorial optimization problems can be formulated as: for a given *set system* over E (i.e., for a pair $(E, \mathcal{F})$ where $\mathcal{F} \subseteq 2^E$ is a family of feasible subsets of finite set E), and for a given function $F : \mathcal{F} \rightarrow \mathbf{R}$, find an element of $\mathcal{F}$ for which the value of the function F is minimum or maximum. In general, this optimization problem is **NP**-hard, but for some specific functions and set systems the problem may be solved in polynomial time. For instance, greedy algorithms may optimize linear objective functions over matroids [11] and Gaussian greedoids [5], [15], [32], while bottleneck objective functions can be maximized over general greedoids [16]. A generalization of greedoids in the context of dynamic programming is discussed in [1] and [2].

Another example is about set functions defined as minimum values of monotone linkage functions. These functions are known as quasi-concave set functions. Such a set function can be maximized by a greedy type algorithm over the family of all subsets of E [19],[24],[29],[30],[34], over antimatroids and convex geometries [17], [20], [25], join-semilattices [28] and meet-semilattices [21]. A relationship was also established between submodular and quasi-concave functions [28] that allowed to build series of branch and bound procedures for finding maximum of submodular functions.

Originally, quasi-concave set functions were considered [23] on the Boolean 2^E

$$\text{for each } X, Y \subset E,\ F(X \cup Y) \geq \min\{F(X), F(Y)\}. \tag{1}$$

In this work we extend this definition to various set systems. One of the natural extensions is a join-semilattice. Here, $\mathcal{S} \subseteq 2^E$ is a *join-semilattice* if it is closed under union, i.e., $A \cup B \in \mathcal{S}$ for each $A, B \in \mathcal{S}$.

Another direction of our research is to adapt the definition of the quasi-concave set functions to set systems that are not necessarily closed under union. Let E be a finite set, and a pair $(E, \mathcal{F})$ be a set system over E. A minimal feasible subset of E that includes a set X is called a *cover of* X. We will denote by $\mathcal{C}(X)$ the family of covers of X. Then the inequality (1) turns into the following.

Definition 1 *A function F defined on a set system* $(E, \mathcal{F})$ *is quasi-concave if for each* $X, Y \in \mathcal{F}$, *and* $Z \in \mathcal{C}(X \cup Y)$

Source: Advances in Greedy Algorithms, Book edited by: Witold Bednorz,
ISBN 978-953-7619-27-5, pp. 586, November 2008, I-Tech, Vienna, Austria

$$F(Z) \geq \min\{F(X), F(Y)\}. \tag{2}$$

If a set system is closed under union, then the family of covers $\mathcal{C}(X \cup Y)$ contains the unique set $X \cup Y$, and the inequality (2) coincides with the original inequality (1).

This chapter is organized as follows. Section 1 contains an extended introduction. Section 2 gives basic information about monotone linkage functions. We show that for a number of combinatorial structures the class of functions defined as the minimum values of monotone linkage functions coincides with the class of quasi-concave set functions. Section 3 deals with the construction of efficient algorithms for maximizing quasi-concave functions which are associated with monotone linkage functions. It is shown that properties of combinatorial structures affect their corresponding optimization algorithms. Section 4 deals with applications to clustering in bioinformatics. In this section we use a particular class of quasi-concave set functions as natural criteria for cluster analysis. We describe how the Fibonacci heap structure can dramatically reduce the computational complexity. Section 5 contains conclusions and directions of future research.

2. Preliminaries

Here we will give definitions of some set properties that are discussed in the following sections. We will use $X \cup x$ for $X \cup \{x\}$, and $X - x$ for $X - \{x\}$.

A non-empty set system $(E, \mathcal{F})$ is called *accessible* if for each non-empty $X \in \mathcal{F}$, there exists an $x \in X$ such that $X - x \in \mathcal{F}$.

For each non-empty set system $(E, \mathcal{F})$ accessibility implies that $\varnothing \in \mathcal{F}$.

Definition 2 *A closure operator,* $\tau: 2^E \rightarrow 2^E$, *is a map satisfying the closure axioms:*

$C1: X \subseteq \tau(X)$

$C2: X \subseteq Y \Rightarrow \tau(X) \subseteq \tau(Y)$

$C3: \tau(\tau(X)) = \tau(X).$

Definition 3 *The set system* $(E, \mathcal{F})$ *is a closure system if it satisfies the following properties*

$(1)\ \varnothing \in \mathcal{F},\ E \in \mathcal{F}$

$(2)\ X, Y \in \mathcal{F}\ implies\ X \cap Y \in \mathcal{F}.$

Let a set system $(E, \mathcal{F})$ be a closure system, then the operator

$$\tau(A) = \cap\{X : A \subseteq X\ and\ X \in \mathcal{F}\} \tag{3}$$

is a closure operator.

A convex geometries was introduced by Edelman and Jamison [9] as a combinatorial abstraction of "convexity".

Definition 4 *[16] The closure system* $(E, \mathcal{F})$ is *a convex geometry if the family* $\mathcal{F}$ *satisfies the following property*

$$X \in \mathcal{F} - E\ implies\ X \cup x \in \mathcal{F}\ for\ some\ x \in E - X. \tag{4}$$

It is easy to see that property (4) is dual to accessibility. Then, we will call it *up-accessibility*. If in each non-empty accessible set system one can reach the empty set $\varnothing$ from any feasible set $X \in \mathcal{F}$ by moving down, so in each non-empty up-accessible set system $(E, \mathcal{F})$ the set E may be reached by moving up.

It is clear that a complement set system $(E, \overline{\mathcal{F}})$ (system of complements), where $\overline{\mathcal{F}} = \{X \subseteq E : E - X \subseteq \mathcal{F}\}$, is up-accessible if and only if the set system $(E, \mathcal{F})$ is accessible.

In fact, accessibility means that for all sets $X \in \mathcal{F}$ there exists a chain $\varnothing = X_0 \subset X_1 \subset ... \subset X_k = X$ such that $X_i = X_{i-1} \cup x_i$ and $X_i \in \mathcal{F}$ for $0 \le i \le k$, and up-accessibility implies the existence of the corresponding chain $X = X_0 \subset X_1 \subset ... \subset X_k = E$. Consider a set family for which this *chain property* holds for each pair of sets $X \subset Y$.

Definition 5 *A set system* $(E, \mathcal{F})$ *satisfies the chain property if for all* $X, Y \in \mathcal{F}$, *and* $X \subset Y$, *there exists an* $y \in Y - X$ *such that* $Y - y \in \mathcal{F}$. *We call the system a chain system*.

In other words, a set system $(E, \mathcal{F})$ satisfies the chain property if for all $X, Y \in \mathcal{F}$, and $X \subset Y$, there exists a chain $X = X_0 \subset X_1 \subset ... \subset X_k = Y$ such that $X_i = X_{i-1} \cup x_i$ and $X_i \in \mathcal{F}$ for $0 \le i \le k$.

It is easy to see that $(E, \overline{\mathcal{F}})$ is a chain system if and only if $(E, \mathcal{F})$ is a chain system as well.

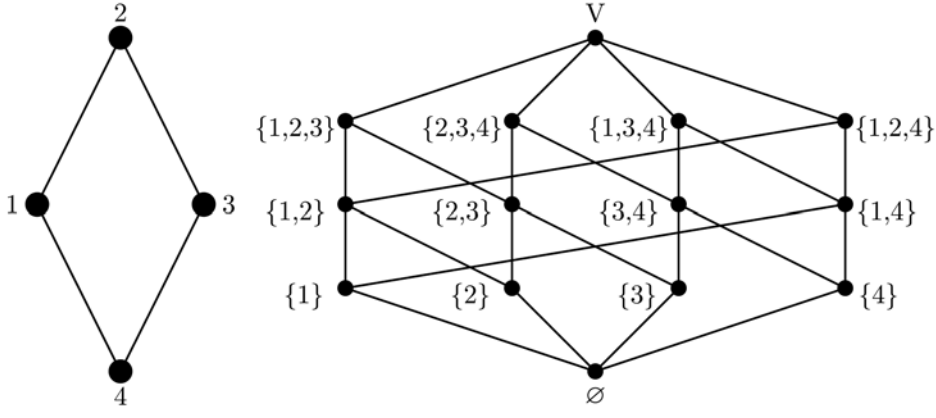
Consider a relation between accessibility and the chain property. If $\varnothing \in \mathcal{F}$, then accessibility follows from the chain property. In general case, there are accessible set systems that do not satisfy the chain property (for example, consider $E = \{1, 2, 3\}$ and $\mathcal{F} = \{\varnothing, \{1\}, \{2\}, \{2, 3\}, \{1, 2, 3\}\}$) and vice versa, it is possible to construct a set system, that satisfies the chain property and it is not accessible (for example, let now $\mathcal{F} = \{\{1\}, \{3\}, \{1, 2\}, \{2, 3\}, \{1, 2, 3\}\}$). In fact, if we have an accessible set system satisfying the chain property, then the same system but without the empty set (or without all subsets of cardinality less then some k) is not accessible, but satisfies the chain property. The analogy statements are correct for up-accessibility.

Examples of chain systems include convex geometries (see proposition 8) and complement systems called antimatroids, matroids and all independence systems (matchings, cliques, independent sets of a graph). Consider a less common example.

Example 6 *For a graph* $G = (V,E)$, *the set system* $(V, \mathcal{S})$ *given by*

$$\mathcal{S} = \{A \subseteq V : (A, E(A)) \text{ is a connected subgraph of } G\},$$

is a chain system. The example is illustrated in Figure 1.



(a) (b)

Fig. 1. $G = (V,E)$ (a) and a family of connected subgraphs (b).

To show that $(V, \mathcal{S})$ is a chain system consider some $A,B \in \mathcal{S}$ such that $A \subset B$. We are to prove that there exists an $b \in B - A$ such that $A \cup b \in \mathcal{S}$. Since B is a connected subgraph, there is an edge $e = (a, b)$, where $a \in A$ and $b \in B - A$. Hence, $A \cup b \in \mathcal{S}$.

For a set $X \in \mathcal{F}$, let $ex(X) = \{x \in X : X - x \in \mathcal{F}\}$ be the set of *extreme points* of X. Originally, this operator was defined for closure systems [9]. An element $e \in A$ was called an *extreme point* if $e \notin \tau(A - e)$. Our definition does not demand the existing of a closure operator, but when the set system $(E, \mathcal{F})$ is a convex geometry $ex(X)$ becomes the set of original extreme points of a convex set X.

Note, that accessibility means that for each non-empty $X \in \mathcal{F}$, $ex(X) \neq \varnothing$.

Definition 7 *The operator* $ex : \mathcal{F} \to 2^E$ *satisfies the heritage property if* $X \subseteq Y$ *implies* $ex(Y) \cap X \subseteq ex(X)$ *for all* $X, Y \in \mathcal{F}$.

We choose the name *heritage property* following B. Monjardet [26]. This condition is well-known in the theory of choice functions where one uses also alternative terms like *Chernoff condition* [7] or *property* α [31]. This property is also known in the form $X - ex(X) \subseteq Y - ex(Y)$.

The heritage property means that $Y - x \in \mathcal{F}$ implies $X - x \in \mathcal{F}$ for all $X, Y \in \mathcal{F}$ with $X \subseteq Y$ and for all $x \in X$.

The extreme point operator of a closure system satisfies the heritage property, but the opposite statement in not correct. Indeed, consider the following example illustrated in Figure 2 (a): let $E = \{1, 2, 3, 4\}$ and

$$\mathcal{F} = \{\varnothing, \{1\}, \{2\}, \{3\}, \{1,2\}, \{2,4\}, \{3,4\}, \{1,2,3\}, E\}.$$

It is easy to check that the extreme point operator ex satisfies the heritage property, but the set system $(E, \mathcal{F})$ is not a closure system ($\{2, 4\} \cap \{3, 4\} \notin \mathcal{F}$). It may be mentioned that this set system does not satisfy the chain property. Another example (Figure 2 (b)) shows that the chain property is also not enough for a set system to be a closure system. Here

$$\mathcal{F} = \{\varnothing, \{1\}, \{4\}, \{1,3\}, \{3,4\}, \{1,2,3\}, \{2,3,4\}, E\},$$

and the constructed set system satisfies the chain property, but is not a closure set ($\{1, 3\} \cap \{3, 4\} \notin F$).

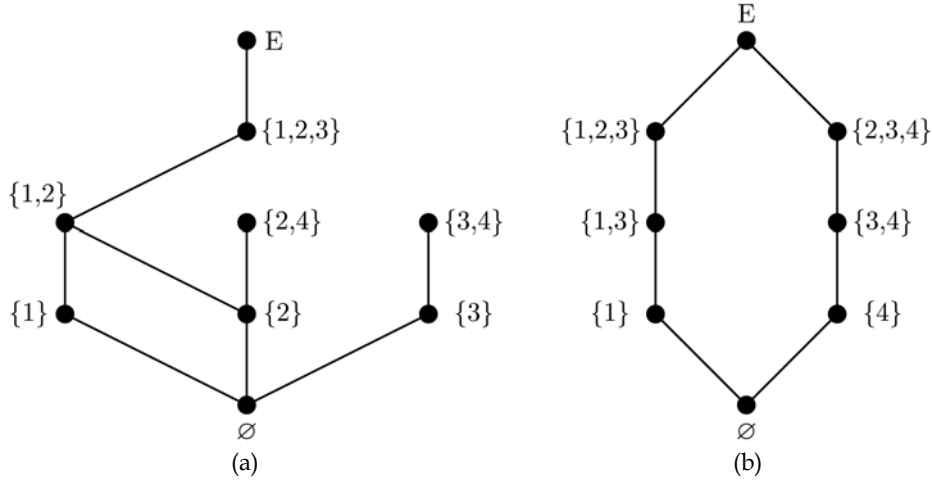


Fig. 2. Heritage property (a) and chain property (b).

Proposition 8 *A set system* $(E, \mathcal{F})$ *is a convex geometry if and only if*

(1) $\varnothing \in \mathcal{F}$, $E \in \mathcal{F}$

(2) *the set system* $(E, \mathcal{F})$ *satisfies the chain property*
(3) *the extreme point operator ex satisfies the heritage property.*

Proof. Let a set system $(E, \mathcal{F})$ be a convex geometry. Then the first condition automatically follows from the convex geometry definition. Prove the second condition. Consider $X, Y \in \mathcal{F}$, and $X \subset Y$. From (4) it follows that there is a chain $X = X_0 \subset X_1 \subset ... \subset X_k = E$ such that $X_i = X_{i-1} \cup x_i$ and $X_i \in$ F for $0 \le i \le k$. Let j be the least integer for which $X_j \supseteq Y$. Then $X_{j-1} \not\supseteq Y$, and $x_j \in Y$. Thus, $Y - x_j = Y \cap X_{j-1} \in \mathcal{F}$. Since $x_j \notin X$, the chain property is proved. To prove that $ex(Y) \cap X \subseteq ex(X)$, consider $p \in ex(Y) \cap X$, then $Y - p \in \mathcal{F}$ and $X \cap (Y - p) = X - p \in \mathcal{F}$, i.e., $p \in ex(X)$.

Conversely, let us prove that the set system $(E, \mathcal{F})$ is a convex geometry. We are to prove both up-accessibility and that $X, Y \in \mathcal{F}$ implies $X \cap Y \in \mathcal{F}$. Since $E \in \mathcal{F}$, up-accessibility follows from the chain property. Consider $X, Y \in \mathcal{F}$. Since $E \in \mathcal{F}$, the chain property implies that there is a chain $X = X_0 \subset X_1 \subset ... \subset X_k = E$ such that $X_i = X_{i-1} \cup x_i$ and $X_i \in \mathcal{F}$ for $0 \le i \le k$. If j is the least integer for which $X_j \supseteq Y$, then $X_{j-1} \not\supseteq Y$, and $x_j \in Y$. Since $x_j \in ex(X_j)$, we obtain $x_j \in ex(Y)$. Continuing the process of clearing Y from the elements that are absent in X, eventually we reach the set $X \cap Y \in \mathcal{F}$. ■

3. Monotone linkage functions

Monotone linkage functions were introduced by Joseph Mullat [29].

A function π: $E \times 2^E \to \mathbf{R}$ is called a *monotone linkage function* if

$$\text{for each } X, Y \subseteq E \text{ and } x \in E,\ X \subseteq Y \text{ implies } \pi(x, X) \le \pi(x, Y). \tag{5}$$

For each $X \subseteq E$ define function $F : (2^E - \varnothing) \to \mathbf{R}$ as follows

$$F(X) = \min_{x \in X} \pi(x, X). \tag{6}$$

Example 9 *Consider a graph* $G = (V,E)$, *where* V *is a set of vertices and* E *is a set of edges. Let* $\deg_H(x)$ *denote the degree of vertex* x *in the induced subgraph* $H \subseteq G$. *It is easy to see that function* $\pi(x,H) = \deg_H(x)$ *is monotone linkage function and function* $F(H)$ *returns the minimal degree of subgraph* H.

Example 10 *Consider a proximity graph* $G = (V,E,W)$, *where* w_{ij} *represents the degree of similarity of objects* i *and* j. *A higher value of* w_{ij} *reflects a higher similarity of objects* i *and* j. *Define a monotone linkage function* $\pi(i,H) = \sum_{j \in H} w_{ij}$, *that measures proximity between subset* $H \subseteq V$ *and their element* i. *Then the function* $F(H) = \min_{i \in H} \pi(i, H)$ *can be interpreted as a measure of density of the set* H.

It was shown [23], that for every monotone linkage function π, function F is quasi-concave on the Boolean 2^E. Moreover, each quasi-concave function may be defined by a monotone linkage function. In this section we investigate this relation on different families of sets.

For any function F defined on a set system $(E, \mathcal{F})$, we can construct the corresponding linkage function

$$\pi_F(x, X) = \begin{cases} \max\limits_{A \in [x,X]_{\mathcal{F}}} F(A), & x \in X \text{ and } [x, X]_{\mathcal{F}} \neq \varnothing \\ \min\limits_{A \in \mathcal{F}} F(A), & \text{otherwise} \end{cases} \tag{7}$$

where $[x,X]_{\mathcal{F}} = \{A \in \mathcal{F} : x \in A \text{ and } A \subseteq X\}$.

The function π_F is monotone. Indeed, if $x \in X$ and $[x,X]_{\mathcal{F}} \neq \emptyset$, then $X \subseteq Y$ implies $[x,X]_{\mathcal{F}} \neq \emptyset$ and

$$\pi_F(x,X) = \max_{A \in [x,X]_{\mathcal{F}}} F(A) \leq \max_{A \in [x,Y]_{\mathcal{F}}} F(A) = \pi_F(x,Y).$$

If $x \in X$ and $[x,X]_{\mathcal{F}} = \emptyset$, then $X \subseteq Y$ implies

$$\pi_F(x,X) = \min_{A \in \mathcal{F}} F(A) \leq \pi_F(x,Y).$$

It is easy to verify the remaining cases.

In the sequel we will consider various types of set systems. At first, we investigate the set systems closed under union, i.e., we study quasi-concave functions on join-semilattices.

Theorem 11 *A set function F defined on a join-semilattice $\mathcal{S}$ is a quasi-concave function if and only if there exists a monotone linkage function π such that $F(X) = \min_{x \in X} \pi(x,X)$ for each $X \in \mathcal{S} - \emptyset$.*

Proof. If a monotone linkage function π is given, then $F(X \cup Y) = \pi(x^*, X \cup Y)$, where $x^* \in \arg\min_{x \in X \cup Y} \pi(x, X \cup Y)$ [1]. Without loss of generality, assume that $x^* \in X$. Thus,

$$F(X \cup Y) = \pi(x^*, X \cup Y) \geq \pi(x^*, X) \geq F(X) \geq \min\{F(X), F(Y)\}.$$

Conversely, if we have a quasi-concave set function F, we can define the monotone linkage function $\pi_F(x,X)$ using the definition 7. Let us denote $G(X) = \min_{x \in X} \pi_F(x,X)$, and prove that $F = G$ on $\mathcal{S} - \emptyset$.

Now, for each $X \in \mathcal{S} - \emptyset$

$$G(X) = \min_{x \in X} \pi_F(x,X) = \pi_F(x^*,X) = \max_{A \in [x^*,X]_{\mathcal{S}}} F(A) \geq F(X),$$

where $x^* \in \arg\min_{x \in X} \pi_F(x,X)$.

On the other hand,

$$G(X) = \min_{x \in X} \pi_F(x,X) = \min_{x \in X} F(A^x),$$

where A^x is a set from $[x,X]_{\mathcal{S}}$ on which the value of the function F is maximal i.e.,

$$A^x = \arg\max_{A \in [x,X]_{\mathcal{S}}} F(A).$$

From quasi-concavity of F it follows that

$$\min_{x \in X} F(A^x) \leq F(\bigcup_{x \in X} A^x) = F(X).$$

Therefore, $G(X) \leq F(X)$, and, hence, $F(X) = \min_{x \in X} \pi_F(x,X)$. ■

Now, consider set systems that are not closed under union.

[1] argmin $f(x)$ denote the set of arguments that minimize the function f.

Let $(E, \mathcal{F})$ be an accessible set system. Denote $\mathcal{F}^+ = \mathcal{F} - \varnothing$. Then, having the monotone linkage function $\pi_{\mathcal{F}}$, we can construct for all $X \in \mathcal{F}^+$ the set function

$$G_F(X) = \min_{x \in ex(X)} \pi_F(x, X).$$

It is easy to see that

$$G_F(X) \geq F(X), \text{ for each } X \in \mathcal{F}^+. \tag{8}$$

Indeed, for each $X \in \mathcal{F}^+$

$$G_F(X) = \min_{x \in ex(X)} \pi_F(x, X) = \pi_F(x^*, X) = \max_{A \in [x, X]_{\mathcal{F}}} F(A) \geq F(X),$$

where $x^* \in \arg\min_{x \in ex(X)} \pi_F(x, X)$.

The following theorem finds conditions on the set system $(E, \mathcal{F})$ and on the function F ensuring that the function G_F coincides with F.

Theorem 12 *[18] Let $(E, \mathcal{F})$ be an accessible set system. Then for every quasi-concave set function $F : \mathcal{F}^+ \to \mathbf{R}$*

$$G_F = F \text{ on } \mathcal{F}^+$$

if and only if the set system $(E, \mathcal{F})$ satisfies the chain property.

Thus, for an accessible set system satisfying the chain property each quasi-concave function F determines a monotone linkage function π_F, and a set function defined as a minimum of this monotone linkage function π_F coincides with the original function F.

As examples of such set systems may be considered greedoids [16] that include matroids and antimatroids, and antigreedoids including convex geometries. By an antigreedoid we mean a set system $(E, \mathcal{F})$ such that its complement set system $(E, \overline{\mathcal{F}})$ is a greedoid.

Note, that if F is not quasi-concave, the function G_F does not necessarily equal F. For example, let $\mathcal{F} = \{\varnothing, \{1\}, \{2\}, \{1, 2\}\}$ and let

$$F(X) = \begin{cases} 0, & X = \{1, 2\} \\ 1, & \text{otherwise.} \end{cases}$$

The function F is not quasi-concave, since $F(\{1\} \cup \{2\}) < \min(F(\{1\}), F(\{2\}))$. It is easy to check that here $G_F \neq F$, because $\pi_F(1, \{1, 2\}) = \pi_F(2, \{1, 2\}) = 1$, and so $G_F(\{1, 2\}) = 1$. Moreover, the function G_F is quasi-concave. To understand this phenomenon, consider the opposite process.

Let $(E, \mathcal{F})$ be an accessible set system. If a monotone linkage function $\pi : E \times 2^E \to \mathbf{R}$ is given, we can construct the set function $F_\pi : \mathcal{F}^+ \to \mathbf{R}$:

$$F_\pi(X) = \min_{x \in ex(X)} \pi(x, X). \tag{9}$$

To extend this function to the whole set system $(E, \mathcal{F})$ define

$$F_\pi(\varnothing) = \min_{(x, X)} \pi(x, X).$$

Theorem 13 *[18] Let* $(E, \mathcal{F})$ *be an accessible set system. Then the following statements are equivalent*

(i) the extreme point operator $ex : \mathcal{F} \to 2^E$ *satisfies the heritage property.*

(ii) for every monotone linkage function π *the function* F_π *is quasi-concave.*

Thus, if a set system $(E, \mathcal{F})$ is accessible and the operator *ex* satisfies the heritage property, then for each set function F, defined on $(E, \mathcal{F})$, one can build the quasi-concave set function G_F that is an upper bound of the original function F. If, in addition, the set system has the chain property, the class of set functions defined as the minimum values of monotone linkage functions coincides with the class of quasi-concave set functions.

Corollary 14 *A set function* F *defined on a convex geometry* $(E, \mathcal{F})$ *is quasi-concave if and only if there exists a monotone linkage function* π *such that* $F(X) = \min_{x \in ex(X)} \pi(x, X)$ *for each* $X \in \mathcal{F} - \varnothing$.

Another approach to the result of Theorem 13 is in extending the function F to the Boolean 2^E by building a new linkage function π_{ex}.

Let $(E, \mathcal{F})$ be an accessible set system and π be a monotone linkage function. Define

$$\pi_{ex}(x, X) = \begin{cases} \pi(x, X), & x \in ex(X) \\ \pi(x, X) + c, & \text{otherwise} \end{cases} \tag{10}$$

where $c = \max_{(x,X) \in E \times 2^E} \pi(x, X) + 1$.

Theorem 15 *[25] Let* $(E, \mathcal{F})$ *be an accessible set system and the extreme point operator ex satisfies the heritage property. If function* π *is a monotone linkage function, then*

(i) function π_{ex} *is also monotone and*

(ii) its function $F_{\pi_{ex}}(X) = \min_{x \in X} \pi_{ex}(x, X)$ *coincides with the function* $F_\pi(X) = \min_{x \in ex(X)} \pi(x, X)$ *for each* $X \in \mathcal{F} - \varnothing$.

Now, Theorem 13 immediately follows from the properties of quasi-concave functions on the Boolean [23].

Remark 16 *[25] Any extreme point operator ex satisfying the heritage property may be represented by some monotone linkage function* π *in the following way*

$$ex(X) = \{x \in X : \pi(x, X) \leq u\} \tag{11}$$

and vice versa, if the linkage function π *is monotone, the operator ex defined by (11) satisfies the heritage property.*

4. Maximizers of quasi-concave functions

Consider the following optimization problem: given a monotone linkage function π, and an accessible set system $(E, \mathcal{F})$, find a feasible set $A \in \mathcal{F}^+$, such that $F_\pi(A) = \max\{F_\pi(B) : B \in \mathcal{F}^+\}$, where the function F_π is defined by (9). From quasi-concavity of the function F_π it follows that the set of optimal solutions is a join-semilattice with a unique maximal element. Our goal is to find this maximal element, which we call the $\cup -$ *maximizer.* For instance, for the functions defined in Example 9 $\cup -$ maximizer is the largest subgraph with the maximum minimum degree. In Example 10 we look for the largest subset with the highest density.

A greedy-type algorithm for finding the $\cup -$ maximizer on the Boolean was constructed by Mullat [29] and has been effectively applied in data mining [22], biology [33], and for computer vision [35].

Here we want to investigate the more general set systems.

4.1 Chain algorithm on convex geometries

A convex geometry is a closure system, and so closed under intersection. Hence, each set $X \subseteq E$ has an unique cover which is a closure of X, i.e., $\tau(X)$ and the family of feasible sets F of a convex geometry $(E, \mathcal{F})$ form a join-semilattice $L_{\mathcal{F}}$, with the lattice operation: $X \vee Y = \tau(X \cup Y)$. Hence, for convex geometries the inequality (2) reads as follows $F(X \vee Y) \geq \min\{F(X), F(Y)\}$ for each $X, Y \in L_{\mathcal{F}}$.

Consider the special structure that quasi-concave function F_π determines on a convex geometry. It has been already noted that the family of feasible sets maximizing function F_π is a join-semilattice with a unique maximal element. Denote this family by $\mathcal{T}^0$, and let a^0 be the value of function F_π on the sets from $\mathcal{T}^0$. We denote by $\mathcal{T}^1$ the family of sets, which maximize function F_π over $\mathcal{F}^+ - \mathcal{T}^0$, and by a^1 the value of function F_π on these sets. Continuing this process, we have $\mathcal{F}^+ = \cup_{i=0}^{t} \mathcal{T}^i$, where $t + 1$ is a number of different values of function F_π. It is easy to see that $\mathcal{L}_j = \cup_{i=0}^{j} \mathcal{T}^i$ is a subsemilattice of $L_{\mathcal{F}}$, where $\mathcal{L}_j = \{X \in \mathcal{F}^+ : F_\pi(X) \geq a^j\}$. We call these subsemilattices *upper level semilattices*. Denote by K^j the maximal element - **1** of the upper level semilattice $\mathcal{L}_j$. Since $\mathcal{L}_i \subseteq \mathcal{L}_{i+1}$, we obtain $K^0 \subseteq K^1 \subseteq ... \subseteq K^t$, where K^t is **1** of the join-semilattice $L_{\mathcal{F}}$, i.e., $K^t = E$.

Let $K^0 = H^0 \subset H^1 \subset ... \subset H^r = K^t$ be the subchain of all different **1-s** of the chain $K^0 \subseteq K^1 \subseteq ... \subseteq K^t$. Thus, to find a $\cup -$ maximizer, we have to find just H^0. In fact, we construct an algorithm that finds the complete chain $H^0 \subset H^1 \subset ... \subset H^r = E$ of different **1-s**. This chain of "local maximizers"[2] has a number of interesting applications [24].

For any real number u we define the *u-level set* of a family $\mathcal{F}$ as

$$\mathcal{F}_u = \{X \in \mathcal{F}^+ : F_\pi(X) > u\}.$$

It is clear that if F_π is quasi-concave, then the u-level set of a join-semilattice is a join-semilattice as well. The input of the following algorithm is a threshold u and a set $X \in \mathcal{F}$, while it returns **1** of non-empty $(\mathcal{F}^+ \cap [\varnothing, X])_u$. The algorithm is motivated by procedures from [28] and [29].

The Level-Set Algorithm (u, X)

1. Set $A = X$
3. While $A \neq \varnothing$ do
   3.1 Set $I_u(A) = \{x \in ex(A) : \pi(x, A) \leq u\}$
   3.2 If $I_u(A) = \varnothing$ then stop and return A
   3.3 Set $A = A - I_u(A)$
4. Return A.

Theorem 17 *Let $(E, \mathcal{F})$ be a convex geometry. Then, for every monotone linkage function π and the corresponding function $F_\pi(X) = \min_{x \in ex(X)} \pi(x, X)$ the Level-Set Algorithm (u,X) returns **1** of non-empty semilattice $(\mathcal{F}^+ \setminus [\varnothing, X])_u$ and returns $\varnothing$ when this u-level set is empty.*

Proof. At first, note that $A - I_u(A) = \cap_{x \in I_u(A)} (A - x)$. Since any convex geometry is closed under intersection, then all sets generated by the algorithm belong to the convex geometry.

---

[2] Indeed, for each $A \in \mathcal{F}^+$, and for each null H^l, if $A \nsubseteq H^l$ then $F(A) < F(H^l)$.

Consider the case when the algorithm returns $A \neq \varnothing$. Since $I_u(A) = \varnothing$, then $F_\pi(A) > u$, i.e. $A \in (\mathcal{F}^+ \cap [\varnothing, X])_u$. It remains to be proven that A is the null of the u-level set, i.e., that $B \in (\mathcal{F}^+ \cap [\varnothing, X])_u$ implies $A \supseteq B$. Suppose the opposite was true, and let $X = X_0 \supset X_1 \supset ... \supset X_k = A$ be a sequence of sets generated by the algorithm, where $X_{i+1} = X_i - I_u(X_i)$ for $0 \leq i < k$. Since $B \in (\mathcal{F}^+ \cap [\varnothing, X])_u$, then $X \supseteq B$. On the other hand, since $A \not\supseteq B$, there exists the least integer j for which $X_j \not\supseteq B$. Then $X_{j-1} \supseteq B$, and there is $x_j \in I_u(X_{j-1})$ that belongs to B. So, $X_{j-1} \supseteq B$, $x_j \in B$ and $x_j \in ex(X_{j-1})$, then from heritage property it follows that $x_j \in ex(B)$. Hence, monotonicity of function π implies $F(B) \leq \pi(x_j, B) \leq \pi(x_j, X_{j-}^{-1}) \leq u$, a contradiction.

If the algorithm returns $A = \varnothing$, then $(\mathcal{F}^+ \cap [\varnothing, X])_u = \varnothing$. Assuming the opposite, then there is a non-empty set $B \in (\mathcal{F}^+ \cap [\varnothing, X])_u$. By analogy, with the first part of the proof, we obtain that $F_\pi(B) \leq u$, a contradiction. ■

The following Chain Algorithm finds the chain of all local maximizers for a non-empty join-semilattice $L_{\mathcal{F}}$.

The Chain Algorithm $(E, \pi, \mathcal{F})$

1. Set $\Gamma_0 = E$
2. $i = 0$
3. While $\Gamma_i \neq \varnothing$ do
   3.1 $u = F(\Gamma_i)$
   3.2 $\Gamma_{i+1} =$ Level-Set(u, Γ_i)
   3.3 $i = i + 1$
4. Return the chain $\Gamma_0 \supset \Gamma_1 \supset ... \supset \Gamma_{i-1}$.

Theorem 18 *Let $(E, \mathcal{F})$ be a convex geometry. Then, for every monotone linkage function π and the corresponding function $F_\pi(X) = \min_{x \in ex(X)} \pi(x, X)$, the Chain Algorithm returns the chain $\Gamma_0 \supset \Gamma_1 \supset ... \supset \Gamma_p$, which coincides with $H^0 \subset H^1 \subset ... \subset H^r$ - the chain of all different **1-s** of the upper level semilattices.*

Proof. First, prove that for each $l = 0, 1, ..., p$, Γ_l is **1** of some upper level semilattice. It is clear, that if $F_\pi(\Gamma_l) = a_j$, then $\Gamma_l \in L_j$. To prove that Γ_l is **1** of L_j , we have to show that for each $A \in \mathcal{F}^+$, $A \not\subseteq \Gamma_l$ implies $F_\pi(A) < F_\pi(\Gamma_l)$. Suppose that the opposite is true, and let k be the least integer for which there exists $A \in \mathcal{F}^+$, such that $A \not\subseteq \Gamma_k$ and $F_\pi(A) \geq F_\pi(\Gamma_k)$. Note that $k > 0$, because $\Gamma_0 = E$ is **1** of join-semilattice $L_{\mathcal{F}}$, and so $A \not\subseteq \Gamma_0$ never holds. The structure of the Chain Algorithm implies $F_\pi(\Gamma_k) > F_\pi(\Gamma_{k-1})$. Hence $F_\pi(A) > F_\pi(\Gamma_{k-1})$ and, consequently, $A \subseteq \Gamma_{k-1}$. Thus $A \in (\mathcal{F}^+ \cap [\varnothing, \Gamma_{k-1}])_u$, where $u = F_\pi(\Gamma_{k-1})$. On the other hand, from Theorem 17 it follows that Γ_k is **1** of $(\mathcal{F}^+ \cap [\varnothing, \Gamma_{k-1}])_u$, i.e., $A \subseteq \Gamma_k$, a contradiction.

It remains to show that for each H^i there exists $l \in \{0, 1, ...p\}$ such that $\Gamma_l = H^i$. Assume the opposite, and let H^j be a maximal **1** for which the statement is not correct. Since $H^r = \Gamma_0$, then $j < r$, i.e., there exists $l \in \{0, 1, ...p\}$ such that $H^{j+1} = \Gamma_l$. From $F_\pi(H^j) > F_\pi(H^{j+1}) = F_\pi(\Gamma_l)$ and $H^j \subset H^{j+1} = \Gamma_l$, it follows that $H^j \in (\mathcal{F}^+ \cap [\varnothing, \Gamma_l])_u$, where $u = F_\pi(\Gamma_l)$. Thus $H^j \subseteq \Gamma_{l+1}$, where Γ_{l+1} is **1** of $(\mathcal{F}^+ \cap [\varnothing, \Gamma_l])_u$. On the other hand, since Γ_{l+1} is **1** of some upper level semilattice and H^j is the closest **1** to H^{j+1}, then $\Gamma_{l+1} \subseteq H^j \subseteq H^{j+1} = \Gamma_l$. Hence $H^j = \Gamma_{l+1}$, a contradiction. ■

Corollary 19 *Let $(E, \mathcal{F})$ be a convex geometry. Then, for every monotone linkage function π, the Chain Algorithm finds a $\cup -$ maximizer of the quasi-concave function $F_\pi = \min_{x \in ex(X)} \pi(x, X)$.*

Actually, a convex geometry is the unique structure on which the Chain Algorithm produces optimal solutions. To prove it we have to show that for each set system that is not a convex geometry there exists a monotone linkage function for which the Chain Algorithm does not find the $\cup -$ maximizer. It is obvious that if a set system is not up-accessible, then the Chain Algorithm may not reach the optimal solution.

Now, consider an up-accessible set system $(E, \mathcal{F})$ that does not satisfy the heritage property, i.e., there exists $A,B \in \mathcal{F}$ such that $A \subset B$, and there is $a \in A$ such that $B - a \in \mathcal{F}$ and $A - a \notin \mathcal{F}$. Up-accessibility of the set system $(E, \mathcal{F})$ implies that there exists a sequence of feasible sets

$$E = B_0 \supset B_1 \supset \ldots \supset B_p = B \supset B_{p+1} = B - a,$$

where $B_i = B_{i-1} - a_i$ for $1 \le i \le p$, and $a_{p+1} = a$. Define a linkage function π on pairs (x,X) where $X \subseteq E$, $X \neq \varnothing$ and $x \in X$:

$$\pi(x,X) = \begin{cases} 1, & (x = a_{i+1} \text{ and } 0 \le i \le p) \text{ or } (B - a \supseteq X,\ X \neq \varnothing \text{ and } x \in X) \\ 2, & \text{otherwise.} \end{cases}.$$

It is easy to verify that function π is monotone. Then the Chain Algorithm generates only one set E, on which the value of the function F_π is equal to 1, while $F_\pi(A) = 2$. Thus, the Chain Algorithm does not find a feasible set that maximizes the function F_π. So, we have the following theorem.

Theorem 20 *Let* $(E, \mathcal{F})$ *be an accessible and an up-accessible set system. Then the following statements are equivalent*

(1) *the set system* $(E, \mathcal{F})$ *is a convex geometry*

(2) *The Chain Algorithm finds a* $\cup -$ *maximizer of the quasi-concave function* $F_\pi = \min_{x \in ex(X)} \pi(x,X)$ *for every monotone linkage function* π

The Chain Algorithm is of greedy type, since it is based on the best choice principle: it chooses on each step the extreme elements (with respect to the linkage function) and, in such a way, approaches the optimal solution. The run-time of the algorithm depends largely on the efficiency of linkage function computation. For instance, in Example 10 the complexity of computing the initial linkage function values $\pi(x, V)$ for all the vertices in V is $O(|E|)$, where E is a set of edges. For straightforward implementation the time required for finding the minimum value is $O(|V|)$. After deleting the vertex with minimum value of π, the time required for updating the linkage function values for all the neighboring vertices of the deleted vertex is $O(|V|)$, since the update can be carried out in time $O(1)$ by subtracting the corresponding weight w_{ij}. So, the total time required for straightforward implementation of the Chain Algorithm in Example 10 is $O(|E| + |V|^2) = O(|V|^2)$.

In general case, the Chain Algorithm finds the $\cup -$ maximizer of a convex geometry (E,F) in $O(P|E| + U|E|^2)$ time, where P is the maximum complexity of computing the initial linkage function values $\pi(x,E)$ over all $x \in E$, and U is the maximum complexity of updating the linkage function values.

For some special linkage functions the running time can be improved by using more efficient data structure that will be discussed in the next section.

4.2 Chain algorithm on join-semilattices

Now we have a monotone linkage function π, and a join-semilattice $\mathcal{S} \subseteq 2^E$, and we are interested in finding a maximal maximizer of the function F_π defined as $F(X) = \min_{x \in X} \pi(x,X)$ according to (6).

Since a join-semilattice should not to be up-accessible, we have to find another way to reach each feasible set.
Consider the following operator:

$$\omega(X) = \begin{cases} \cup\{A \mid A \subseteq X, A \in \mathcal{S}\}, & [\varnothing, X]_{\mathcal{S}} \neq \varnothing \\ \varnothing, & otherwise \end{cases} . \quad (12)$$

If $\mathcal{S}$ is a join-semilattice, $\omega(X)$ is the largest set in $\mathcal{S}$ contained in X (if such a set exists). In other words, $\omega(X)$ is the **1** of the subsemilattice $[\varnothing, X]_{\mathcal{S}}$ if the subsemilattice is not empty, and $\varnothing$, otherwise.
Note, that a join-semilattice $\mathcal{S}$ should not have the minimum element, and we use the element $\varnothing$ only to complete the definition of the operator ω.
The operator ω is called *interior* (dual to closure) operator:

(i) $\omega(X) \subseteq X$,
(ii) $\omega(X) = \omega(\omega(X))$,
(iii) $X \subseteq Y \Rightarrow \omega(X) \subseteq \omega(Y)$.

$\omega(X)$ is *an interior* of X. The fixed points of ω ($X = \omega(X)$) are called the open sets of ω and forms the dual closure system [27]. A set system $(E, \mathcal{S})$ is a dual closure system if and only if the complement set system $(E, \mathcal{S})$ is a closure system. If $\mathcal{S}$ is a join-semilattice and the operator ω is defined by (12), then the family of open sets coincides with $\mathcal{S}$, excluding, possible, the empty set.

We assume that for each $X \subseteq E$ a procedure for finding interior $\omega(X)$ is available. Later we will consider some examples of procedures building interior efficiently.
From quasi-concavity of function F_π it follows that the set of maximizers is a join-semilattice with a unique maximal element. It is easy to see that the structure of upper level semilattices investigated for convex geometries holds for join-semilattice as well. To obtain the chain $H^0 \subset H^1 \subset ... \subset H^r = E$ of different ***1-s*** we use the Chain Algorithm with the following transformation: instead of assigning some set we replace it by its interior.

The Level-Set Algorithm-JS (u, X)

1. Set $A = \omega(X)$
3. While $A \neq \varnothing$ do
   3.1 Set $I_u(A) = \{x \in A : \pi(x, A) \leq u\}$
   3.2 If $I_u(A) = \varnothing$ then stop and return A
   3.3 Set $A = \omega(A - I_u(A))$
4. Return A.

The Chain Algorithm-JS (E, π, F)

1. Set $\Gamma_0 = \omega(E)$
2. $i = 0$
3. While $\Gamma_i \neq \varnothing$ do
   3.1 $u = F(\Gamma_i)$
   3.2 Γ_{i+1} = Level-Set(u, Γ_i)
   3.3 $i = i + 1$
4. Return the chain $\Gamma_0 \supset \Gamma_1 \supset ... \supset \Gamma_{i-1}$.

Similarly with the proof of Theorem 18 we obtain the following result.

Theorem 21 *Let* $\mathcal{S} \subseteq 2^E$ *be a non-empty join-semilattice. Then, for every monotone linkage function* π *and the corresponding function* $F(X) = \min_{x \in X} \pi(x, X)$*, the Chain Algorithm-JS returns the chain* $\Gamma_0 \supset \Gamma_1 \supset ... \supset \Gamma_p$*, which coincides with* $H^0 \subset H^1 \subset ... \subset H^r$ *- the chain of all different* ***1-s*** *of the upper level semilattices.*

Consider the complexity of the Chain Algorithm-JS. The run-time of the algorithm depends largely on the efficiency of interior construction. The Chain Algorithm-JS finds the $\cup$ − maximizer of a join-semilattice $(E,\mathcal{S})$ in $O(|E|(P + T + U|E|))$ time, where P is the maximum complexity of computing the initial linkage function values $\pi(x,E)$ over all $x \in E$, U is the maximum complexity of updating the linkage function values, and T is the maximum complexity of interior construction.

4.2.1 Algorithms for interior construction

The efficiency of the interior construction depends on the representation of a join-semilattice. Here we consider a join-semilattice specified by a quasi-concave function. In addition, we consider an antimatroid that is a specific case of a join-semilattice.

1. Quasi-Concave constraints. Assume that the family $\Omega \subseteq 2^E$ of feasible sets is determined by the following constraints: for each $H \in \Omega$, $\widehat{F}(H) > \alpha$, where $\widehat{F}$ is a quasi-concave function defined by a monotone linkage function $\widehat{\pi}$. It is easy to see that the set Ω is an α-level set of 2^E, i.e., $\Omega = \{X \subseteq E : \widehat{F}(X) > \alpha\}$. Since $\widehat{F}$ is a quasi-concave function, the set Ω is a join-semilattice. The problem is to find interior $\omega(X)$ over Ω for every set $X \subseteq E$, i.e., to find **1** of the non-empty join-semilattice $\Omega \cap [\varnothing, X]$. Note that the Level-Set Algorithm(α, X) enables us to find **1** of the non-empty join-semilattice $(2^E \cap [\varnothing, X])_\alpha$, i.e., $\omega(X)$ over Ω. The modified Level-Set Algorithm is as follows:

Quasi-Concave Interior Algortihm (α, X)

1. Set $A = X$
3. While $A \neq \varnothing$ do
   3.1 Set $I_\alpha(A) = \{x \in A : \widehat{\pi}(x,A) \leq \alpha\}$
   3.2 If $I_\alpha(A) = \varnothing$ then stop and return A
   3.3 Set $A = A - I_\alpha(A)$
4. Return A.

The Quasi-Concave Interior Algorithm finds the interior $\omega(X)$ in $O(P|X| + U|X|^2)$ time, where P is the maximum complexity of computing the initial linkage function values $\widehat{\pi}(x,X)$ over all $x \in X$, and U is the maximum complexity of updating the linkage function values.

2. Antimatroids. There are many equivalent axiomatizations of antimatroids, that may be separated into two categories: antimatroids defined as set systems and antimatroids defined as languages. An algorithmic characterization of antimatroids based on the language definition was introduced in [6]. Another algorithmic characterization of antimatroids that depicted them as set systems was developed in [17]. While classical examples of antimatroids connect them with posets, chordal graphs, convex geometries, etc., game theory gives a framework in which antimatroids are interpreted as permission structures for coalitions [4]. There are also rich connections between antimatroids and cluster analysis [20]. In mathematical psychology, antimatroids are used to describe feasible states of knowledge of a human learner [12].

Definition 22 *[16]A non-empty set system* $(E, \mathcal{S})$ *is an antimatroid if*

(A1) $(E, \mathcal{S})$ *is an accessible set system*

(A2) for all $X, Y \in \mathcal{S}$, *and* $X \not\subseteq Y$, *there exist an* $x \in X - Y$ *such that* $Y \cup x \in \mathcal{S}$.

It is easy to see that the chain property follows from (*A*2), but these properties are not equivalent.

Proposition 23 *[5][16]For an accessible set system* $(E, \mathcal{S})$ *the following statements are equivalent:*

(i) $(E, \mathcal{S})$ *is an antimatroid*

(ii) $\mathcal{S}$ *is closed under union* $(X, Y \in \mathcal{S}) \Rightarrow X \cup Y \in \mathcal{S}$.

Therefore an antimatroid is a join-semilattice that includes the empty set. The interior operator ω defined by (12) returns for each set $X \subseteq E$ the maximal feasible subset called the *basis* of X.

Since an antimatroid $(E, \mathcal{S})$ satisfies the chain property, to find $\omega(X)$, one can build the chain $\varnothing \subset X_0 \subset X_1 \subset \ldots \subset X_m = \omega(X)$ belonging to $\mathcal{S}$.

Antimatroid Interior Algortihm$(X, \mathcal{S})$

1. $A = \varnothing$
2. Find $x \in X$ - A, such that $A \cup x \in$ S
   if no such x exists, then stop and return A
3. Set $A = A \cup x$ and go to 2.

The Antimatroid Interior Algorithm returns the basis $\omega(X)$ for each set $X \subseteq E$ that immediately follows from the chain property.

Let an antimatroid $(E, \mathcal{S})$ be given by a membership oracle which for each set $A \subseteq E$ decides whether $A \in \mathcal{S}$ or not. Then the Antimatroid Interior Algorithm finds the interior of a set in at most $k(k + 1)/2$ oracle calls, where $k = |X|$. Thus the complexity of interior construction is $O(|X|^2\theta)$, where θ is the complexity of the membership oracle.

Consider another way to define antimatroids. Let $P = \{x_1 < x_2 < \ldots < x_n\}$ be a linear order on E. Define

$$D_P = \{X_i : \ X_i = \{x_1, x_2, \ldots, x_i\}, \ 1 \le i \le n\} \cup \{\varnothing\}.$$

It is easy to see that (E, D_P) is an antimatroid.

Let $(E, \mathcal{S}_1)$ and $(E, \mathcal{S}_2)$ be two antimatroids. Define

$$\mathcal{S}_1 \vee \mathcal{S}_2 = \{X \cup Y : X \in \mathcal{S}_1, Y \in \mathcal{S}_2\}.$$

Then $(E, \mathcal{S}_1 \vee \mathcal{S}_2)$ is also an antimatroid [16].

Every antimatroid can be represented as a join of a family of its maximal chains. Hence, each antimatroid may be defined by a set T of linear orders as

$$\mathcal{S} = \bigvee_{P \in T} D_P. \tag{13}$$

By analogy with convex realizers of convex geometries [10] the set T is called a realizer. Thus, if $\{P_1, P_2, \ldots, P_k\}$ is a realizer of $(E, \mathcal{S})$, then each element of $\mathcal{S}$ is a join of elements in $D_{P_1}, \ldots, D_{P_k}$. Note, that each $D_{P_i} \subseteq \mathcal{S}$.

Since each (E, D_{P_i}) is an antimatroid , there are k interior operators ω_{P_i}, where $\omega_P(X) = \{y \in E : y \le_P \min \overline{X}\}$, i.e., let $P = \{x_1 < x_2 < \ldots < x_n\}$ and a minimal element of $\overline{X}$ with respect to the order P be $x_i = \min \overline{X}$, then $\omega_P(X) = \{x_1, x_2, \ldots, x_{i-1}\}$.

Proposition 24 $\omega(X) = \bigcup_{P \in T} \omega_P(X)$.

Proof. Let $A = \cup_{P \in T} \omega_P(X)$. Since for each $P \in T$, $\omega_P(X) \subseteq X$ and $\omega_P(X) \in \mathcal{S}$, then $\omega_P(X) \subseteq \omega(X)$, which implies $A \subseteq \omega(X)$. Conversely, from (13) $\omega(X) = \cup_{P \in T} X_P$, where $X_P \in D_P$. Since $\omega(X) \subseteq X$ implies $X_P \subseteq X$ for all $P \in T$, then $X_P \subseteq \omega_P(X)$ and so $\omega(X) \subseteq A$. ■

Let an antimatroid $(E, \mathcal{S})$ be given by a realizer $T = \{P_1, P_2, ..., P_k\}$, then the following algorithm builds the interior set using Proposition 24.

Ordering Interior Algorithm(X, $\mathcal{S}$)

1. For $i = 1$ to k do
   1.1 build $\omega_{p_i}(X)$
2. Return $\tau(X) = \cup_{i=1}^{k} \omega_{P_i}(X)$.

A straightforward implementation of the Ordering Interior Algorithm runs in $O(k|E|)$, where k is the cardinality of a realizer.

5. Ortholog clustering

This section deals with applications of quasi-concave functions to clustering in bioinformatics. We concentrate on the one of the problem of comparative genomics. Comparative genomics is a field of biological research in which the genome sequences of different species are compared. Although living creatures look and behave in many different ways, all of their genomes consist of DNA, the chemical chain that includes the genes that code for thousands of different kinds of proteins. Thus, by comparing the sequence of the human genome with genomes of other organisms, researchers can identify regions of similarity and difference. This information can help scientists better understand the structure and function of human genes and thereby develop new strategies to combat human disease. Comparative genomics also provides a powerful tool for studying evolutionary changes among organisms.

A fundamental problem in comparative genomics is the detection of genes from different organisms that are involved in similar biological functions. This requires identification of homologous genes that are similar to each other because they originated from a common ancestor. Such genes are called *orthologs* [13].

We describe an ortholog clustering method where we require that any sequence in an ortholog cluster has to be similar to other sequence from other genomes in that ortholog cluster.

5.1 Ortholog detection using multipartite graph clustering

The input for the ortholog clustering problem is a set of genetic sequences along with information about the organisms they belong to. The goal is to find similar sequences from different organisms. The ortholog detection problem is complicated due to the presence of another type of very similar sequences in the same organism. These sequences, called *paralogs*, are result of duplication events when a gene in an organism is duplicated to occupy two different positions in the same genome. Although both types of genes are similar, only orthologs are likely to be involved in the same biological role. So, for detecting orthologs it is critical to focus on the similarities between genes from different organisms while ignoring the similarities between genes within an organism.

The requirement of selectively ignoring gene similarities for paralogous sequences can be conveniently represented in a multipartite graph. A graph is a multipartite if the set of

vertices in the graph may be divided into non-empty disjoint subsets, called parts, such that no two vertices in the same part have an edge connecting them. We use a multipartite graph, where different genomes correspond to different parts and the genes in a genome correspond to vertices in a part.

Another specific problem in finding ortholog clusters is that orthologous genes from closely related organisms will be much more similar than those from distantly related organisms. Fortunately, we often have estimates of evolutionary relationships between the organisms that define a hierarchical graph over the partite sets. Using this evolutionary graph, called *a phylogenetic tree*, we can correct the observed gene similarities by scaling up the similarities between the orthologs from distantly related organisms.

Consider the ortholog clustering problem with k different genomes, where the genome l, represented by V_l (l = 1, 2, ...k), contains n_l genes. Then, the similarity relationships between genes from different genomes can be represented by an undirected weighted multipartite graph $G = (V,E,W)$, where $V = \cup_{l=1}^{k} V_l$, every set V_l contains n_l vertices corresponding to n_l genes, and $E \subseteq \cup_{i \neq j} V_i \times V_j$ (i, j = 1, 2, ..., k) is a set of weighted edges representing similarities between genes. The example of a multipartite graph is illustrated in Figure 3 (a). The relationship between these genomes is given by the phylogenetic tree relating the species under study (see Figure 3 (b)).

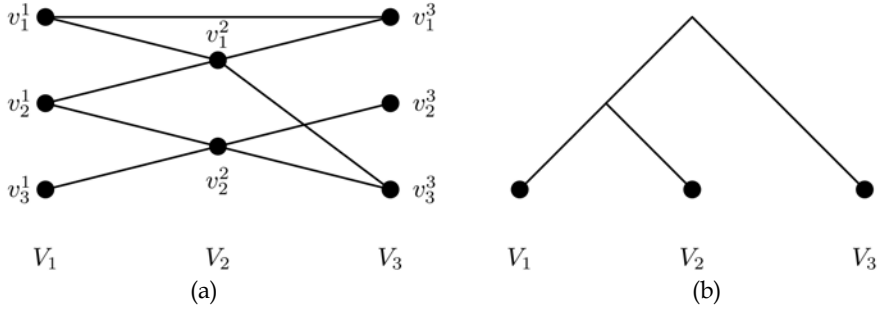


Fig. 3. Multipartite graph (a) and phylogenetic tree (b).

We consider an ortholog cluster as a largest subgraph with the highest density. For finding an ortholog cluster we assign a score $F(H)$ to any subset H of V. A score function F denotes a measure of proximity among genes in H. Then an ortholog cluster H^* is defined as the subset with the largest score value (a maximizer of F). To build a score function $F(H)$ we use Definition 6 that is based on using a linkage function $\pi(i,H)$ which measures the degree of similarity of the gene $i \in H$ to other genes in H.

Our linkage function considers the sequence similarity between genes within the ortholog cluster, their relationship to genes outside the cluster, and the phylogenetic distance between the corresponding genomes.

We require that H contains at least two genomes. So, let $H = \cup_{l=1}^{k} H_l$, where H_l is the subset of genes from V_l present in H. If $m_{ij} \geq 0$ is the similarity value between gene i from genome $g(i)$ and gene j from another genome $g(j)$, and $p(g(i), g(j))$ represents the distance between the two genomes, then the linkage function is defined as

$$\pi(i,H) = \sum_{l \neq g(i)}^{k} p(g(i),l)\Big(\sum_{j \in H_l} m_{ij} - \sum_{j \in V_l - H_l} m_{ij}\Big) \tag{14}$$

For each part $V_l \neq g(i)$ the term $\sum_{j \in H_l} m_{ij}$ aggregates the similarity values between the genes i and all other genes in the subset H_l, while the second term, $\sum_{j \in V_l - H_l} m_{ij}$, estimates the relationship between gene i and genes from genome l that are not included in H_l. A large positive difference between two terms ensures that the gene i is highly similar to genes in H_l and at the same time very dissimilar from the genes not included in H_l. From a clustering point of view, this ensures large values of intra-cluster homogeneity and inter-cluster separability for extracted cluster.

The scaling term $p(g(i),l)$ is used for correcting the observed sequence similarities by magnifying the sequence similarities corresponding to genomes which diverged in ancient times. Given the phylogenetic tree relating the species under study, the distance $p(g(i), g(j))$ between genomes $g(i)$ and $g(j)$ is defined as the height, $h_{g(i),g(j)}$, of the subtree rooted at the last common ancestor of these genomes. When the species are closely related, a function that depends on $h_{g(i),g(j)}$, but grows slower will better model the distance between the species. Choosing an appropriately growing function is critical because a faster growing function will have the undesirable effect of clustering together sequence from distance species but leaving out the sequence from closely related species. So, in this case the distance $p(g(i), g(j))$ may be defined as $(1 + \log_2 h_{g(i),g(j)})$.

It is easy to verify that function π defined in (14) is monotone. Firstly note that the distance $p(g(i), g(j)) \geq 0$ has no effect on the monotonicity. Consider the case when H is extended by some gene p. If $i \in g(p)$, then $\pi(i,H \cup p) = \pi(i,H)$, otherwise $\pi(i,H \cup p) - \pi(i,H) = 2p(g(i), g(p))m_{ip} \geq 0$

So, the function $F_\pi(X) = \min_{x \in X} \pi(x, X)$ is quasi-concave and we can use the Chain Algorithm to find the orthogonal cluster.

5.2 Analysis and implementation

The performance of the Chain Algorithm depends on the type of data structure one chooses to maintain the set of linkage function values. In Example 10 the total time required for straightforward implementation of the Chain Algorithm is $O(|V|^2)$. Here we build the efficient data structure that enables us to reduce the run-time of the algorithm. There are three operation that are performed at each iteration of the algorithm.

i. find-min - this operation performs in Step 3.1 of the Chain Algorithm where the value $F(\Gamma_i)$ is determined.
ii. delete-min - this operation performs in Step 3.2 of the Chain Algorithm when the Level-Set Algorithm finds set $I_u(A)$ of elements with the minimum value of function π and removes this set from the set A.
iii. decrease-key - this operation performs inside the Level-Set Algorithm. Deleting set $I_u(A)$ entails updating the linkage function values for all neighbors of elements from this set.

If $|V|$ elements are organized into a Fibonacci heap [14], we can perform a delete-min operation in $O(\log V)$ amortized time and a decrease-key operation in $O(1)$ amortized time, while a find-min operation can be done in constant time [8].

Proposition 25 *[33] With a Fibonacci heap, the Chain Algorithm finds an ortholog cluster in time* $O(|E| + |V| \log |V|)$.

Proof. The initialization of the algorithm includes computing $\pi(i, V)$ for each $i \in V$. The value $\pi(i, V)$ depends on the weights on edges incident to i and on the relationship of the

genome $g(i)$ with other genomes. We assume that the number of genomes is very small compared to the number of genes, i.e., $k << n$. Thus computing the initial linkage function values for all the vertices takes $O(|E|)$.

We use Fibonacci heap to store vertices according to their linkage function values. So, the value $F(\Gamma_i)$ can be found in $O(1)$ time, and since each deletemin operation takes $O(\log V)$ amortized time, the total time for all calls to delete-min is $O(V \log V)$.

Each deleting of an element with minimum value of linkage function π leads to updating the linkage function values for all neighbours of the element. Due to the additive property of the linkage function (14), the update can be carried out in time $O(1)$ by subtracting the corresponding value $2p(g(i), g(p))m_{ip}$ due to the deleted edge (i, p).

Decreasing the value of function π involves an implicit decrease-key operation, which can be implemented in $O(1)$ amortized time. As each edge is deleted once, all linkage function updates together require $O(|E|)$ time. Thus, the algorithm runs in $O(|E| + |V| \log |V|)$ time. ■

The proposed ortholog clustering method was applied to the protein sequences from complete genomes of seven eukaryotes present in the eukaryotic orthologous groups [33]. The analysis of these results shows that clusters obtained using proposed method show a high degree of correlation with the manually curated ortholog clusters.

6. Conclusions

In this article, we have investigated monotone linkage functions defined on convex geometries, antimatroids, and semilattices in general. It has been shown that the class of functions defined as minimum values of monotone linkage functions has close relationship with the class of quasi-concave set functions. Quasi-concave functions defined on semilattices, antimatroids and convex geometries determine special substructures of these set families. This structures allow building efficient algorithms that find minimal sets on which values of quasi-concave functions are maximum.

The mutual critical step of these algorithms is how to describe the set closure operator. If an efficient algorithm of the closure construction exists it causes the optimization algorithm to be efficient as well. On the other hand, we think that the closure construction problem is interesting enough to be investigated separately. Thus, we suppose that for an arbitrary semilattice the problem of closure construction has exponential complexity.

An interesting direction for future work is to develop our methods for relational databases, where a polynomial algorithm for closure construction is known [3].

We have considered some applications of quasi-concave functions to clustering a structured data set, where together with pair-wise similarities between objects, we are also given additional information about objects organization.

We focused on a simple structure - a partition model of data where the objects are a priori partitioned into groups. While clustering such data, we also considered an additional requirement of being able to differentiate between pairwise similarities across different partite sets. Existing clustering methods do not solve this problem, since they are limited to finding clusters in a collection of isolated objects.

The requirement of differentially treating pair-wise relationships across different groups was modeled by a multipartite graph along with a hierarchical relationship between these groups. The problem was reduced to finding the cluster (subgraph) of the highest density in the multipartite graph.

This problem is usually formulated as finding the maximum weight multipartite clique. However, no efficient procedure exists for solving this problem. Due to this, clusters are often modeled as quasi-cliques or dense graphs.
Traditionally, quasi-cliques are defined, using a threshold, as a relaxation of a complete subgraph - the relaxation can be on the degree of a vertex or on the total number of edges in the quasi-clique. In contrast to traditional quasi-clique definition, our definition does not use any threshold parameters.
The proposed multipartite graph clustering method was successfully applied to the ortholog clustering problem. It may be also adapted to other clustering problem both in comparative genomics and in computer vision [35].

7. References

[1] Bagotskaya, N.V.; Levit, V.E. & Losev, I.S. (1988). On one generalization of matroids insuring applicability of dynamic programming method, In: *Information Transmission and Processing Systems*, Vol. 2, IPIT USSR Academy of Sciences, Moscow, (33–36). (in Russian)
[2] Bagotskaya, N.V.; Levit, V.E. & Losev, I.S. (1990). A combinatorial structure insuring applicability of dynamic programming method, *Automation and Remote Control* 50, (1414-1420).
[3] Beeri,C. & Bernstein, P.A. (1979). Computational problems related to the design of normal form relational schemes, *ACM Transactions on Database Systems* 4, No.1, (30-59).
[4] Bilbao, J.M. (2003). Cooperative games under augmenting systems, *SIAM Journal of Discrete Mathematics* 17, (122-133).
[5] Björner, A. & Ziegler, G.M. (1992). Introduction to greedoids, In: *Matroid applications*, ed. N. White, Cambridge Univ.Press, Cambridge, UK.
[6] Boyd, E.A. & Faigle, U. (1990). An algorithmic characterization of antimatroids, *Discrete Applied Mathematics* 28, (197-205).
[7] Chernoff, H. (1954). Rational selection of decision functions, *Economica* 22, (422-443).
[8] Cormen, T.H.; Leiserson, C. E.; Rivest, R. L. & Stein, C. (2001). *Introduction to Algorithms*, second ed. MIT Press and McGraw-Hill, (476-497).
[9] Edelman, P.H. & Jamison, R.E. (1985). The theory of convex geometries, *Geom. Dedicata* 19, (247-270).
[10] Edelman, P.H. & Saks, M.E. (1988). Combinatorial representation and convex dimension of convex geometries, *Order* 5, No.1, (23-32).
[11] Edmonds, J. (1971). Matroid and the greedy algorithm, *Mathematical Programming* 1, (127-136).
[12] Eppstein, D. (2008). Upright-Quad Drawing of st-planar learning spaces, *Journal of Graph Algorithms and Applications* 12, No. 1, (51-72).
[13] Fitch, W.M. (1970). Distinguishing homologous from analogous proteins, *Systematic Zoology* 19, (99-113).
[14] Fredman, M.L. & Tarjan, R.E. (1987). Fibonacci heaps and their uses in improved network optimization algorithms, *Journal of the ACM* 34, No.3, (596-615).
[15] Goecke, O. (1988). A greedy algorithm for hereditary set systems and a generalization of the Rado-Edmonds characterization of matroids, *Discrete Applied mathematics* 20, (39-49).

[16] Korte, B.; Lovász, L. & Schrader, R. (1991). *Greedoids*, Springer-Verlag, New York/Berlin.

[17] Kempner, Y. & Levit, V.E. (2003). Correspondence between two antimatroid algorithmic characterizations, *The Electronic Journal of Combinatorics* 10, R44.

[18] Kempner, Y. & Levit, V.E. (2008). Duality between quasi-concave functions and monotone linkage functions, arXiv:0808.3244 [math.CO].

[19] Kempner, Y.; Mirkin, B. & Muchnik, I. (1997). Monotone linkage clustering and quasi-concave functions, *Appl.Math.Lett.* 10, No.4 (19-24).

[20] Kempner, Y. & Muchnik, I. (2003). Clustering on antimatroids and convex geometries, *WSEAS Transactions on Mathematics* 2, Issue 1, (54-59).

[21] Kempner, Y. & Muchnik, I. (2008). Quasi-concave functions on meet-semilattices, *Discrete Applied Mathematics* 156, No. 4, (492-499).

[22] Kuusik, R. & Lind, G. (2004). Generator of Hypotheses - An approach of data mining based on monotone system theory, *International Journal of Computational Intellegence* 1, No. 1, (43-47).

[23] Malishevski, A. (1998). Properties of ordinal set functions, In: *A.Malishevski, Qualitative Models in the Theory of Complex Systems*, Nauka, Moscow, (169-173) (in Russian).

[24] Mirkin, B. & Muchnik, I. (2002). Layered clusters of tightness set functions, *Appl. Math. Lett.* 15, (147-151).

[25] Mirkin, B. & Muchnik, I. (2002). Induced layered clusters, Hereditary Mappings, and Convex Geometry, *Appl. Math. Lett.* 15, (293-298).

[26] Monjardet, B. & Raderanirina, V. (2001). The duality between the antiexchange closure operators and the path independent choice operators on a finite set, *Mathematical Social Sciences* 41, (131-150).

[27] Monjardet, B. (2003). The presence of lattice theory in discrete problems of mathematical social sciences. Why. *Mathematical Social Sciences* 46, (103-144).

[28] Muchnik, I. & Shvartser, L.V. (1989). Kernels of Monotonic Systems on a Semi-lattice of Sets, *Automation and Remote Control* 50, No. 8, part 2, (1095-1102).

[29] Mullat, J. (1976). Extremal subsystems of monotone systems: I, II, *Automation and Remote Control* 37, (758-766); (1286-1294).

[30] Mullat, J. (1995). A fast algorithm for finding matching responses in survey data table, *Mathematical Social Sciences* 30, (195-205).

[31] Sen, A.K. (1971). Choice functions and revealed preference, *Review of Economic Studies* 38, (307-317).

[32] Serganova, V.V.; Bagotskaya, N.V.; Levit, V.E. & Losev, I.S. (1988). Greedoids and the greedy algorithm, In: *Information Transmission and Processing Systems*, Vol. 2, IPIT USSR Academy of Sciences, Moscow, (49-52). (in Russian)

[33] Vashist, A.; Kulikowski, C.A. & Muchnik, I. (2007). Ortholog clustering on a multipartite graph, *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 4, No. 1 (17-27).

[34] Zaks (Kempner), Y. & Muchnik, I. (1989). Incomplete classifications of a finite set of objects using monotone systems, *Automation and Remote Control* 50, (553-560).

[35] Zhang, R.; Vashist, A.; Muchnik, I.; Kulikowski, C. A. & Metaxas, D. N. (2005). A new combinatorial approach to supervised learning : Application to gait recognition, *LNCS* 3723 (55-69).