Identification of Monotone Measures

Bo Guo

Information Science and Technology
University of Nebraska-Omaha, Omaha, NE 68182

Advisor: Dr. Zhenyuan Wang

Department of Mathematics, University of Nebraska-Omaha

Abstract: Identification of monotone measures is a very challenge process. Based on an existing data set of a set function (or, set functions) defined on the power set of a given finite universal set, to determine a monotone measure, simply switching the values of the given set function that violate the monotonicity is not the optimal way and it also causes some side effects. The paper proposes an algorithm to optimally determine a monotone measure using soft computing technique. In the algorithm, the crux is how to guarantee the monotonicity of the obtained set function. In this connection, a sub-algorithm called reordering algorithm is embedded. According to the lattice structure of the power set of the given universal set, a max-min strategy is adopted to reduce the computational complexity. The proposed approach is illustrated by applying it to several random data sets. Our experiments show that this approach is effective.

Key words: Monotone measures, genetic algorithms, optimization.
1. Introduction

With the development and diffusion of information, especially information network and the information highway construction and application, information access, comprehensive analysis and processing have been considered to provide decision support. However, it is very difficult for decision makers to make right decisions in the disorderly and unsystematic data and observations. The extraction of hidden predictive information from large observations is very challenge task. In order to overcome the obstacle, many data mining technologies are recently proposed, such as classification, clustering, decision tree and so on.

Information fusion is used in data mining and consolidation of data from unstructured or semi-structural resources. Information fusion technique can combine data from multiple sources and gather that information in order to achieve inferences, which will be more efficient and potentially more accurate than if they are achieved by means of a single source. In practice, in order or measure the importance and efficiency of the targets, we need to evaluate them by observation. For example, in order to prove the efficiency of the workers under the same conditions, some data are given from different observers. There data are disordered and unsystematic. We need to acquire monotone measure, from these known data, to prove that the more efficient, the more workers.

Moreover, in the modern competitive society, decision making are very important. A tiny wrong decision may cause the unforeseen loss. From the known and unsystematic information, the decision maker may retrieve and acquire the useful information by identifying the measure. For example, medical decision making often involves making a diagnosis and selecting an appropriate treatment from his/her experience and diagnosis database according to the patients’ symptoms.
The rest of this paper is organized as follows. Section 2 describes the preliminary including \( \sigma \)-algebra, measure space, classic measure, monotone measure. Section 3 given the basic concept of optimization. In Section 4, identification of monotone measure is described and argues the feasibility of some simple methods keeping the set monotone, such as simple switch and average value. Section 5 proposed a genetic algorithm. Some basic concepts are introduced in this section, like encoding genes, chromosomes, populations, fitness functions. Also three genetic operators, Two-point crossover, Three-point mutation, and Two-point realignment, are detailed. Main algorithm is described in section 6, in which the reordering algorithm is proposed using the lattice structure of the power set of a universal set. Experiments are implemented and analyzed in section 7 and section 8 is conclusions.

2. Preliminary

2.1 \( \sigma \)-algebra and measure space

In mathematics, a \( \sigma \)-algebra on a set \( X \) is a nonempty collection \( \mathcal{F} \) of subsets of \( X \) (including \( X \) itself) that is closed under complementation and countable unions of its members. For example, if \( X = \{ a, b, c \} \), one possible \( \sigma \)-algebra on \( X \) is \( \mathcal{F} = \{ \emptyset, \{ a, b \}, \{ c \}, \{a, b, c\} \} \). In this paper, we adopt power set \( \mathcal{P}(X) \) as the \( \sigma \)-algebra on set \( X \).

The pair \( ( X, \mathcal{P}(X) ) \) is called a measurable space.

2.2 Classic measure

Let \( ( X, \mathcal{P}(X) ) \) be a measurable space. A classic measure \( \mu \) is a set function on \( \mathcal{P}(X) \) satisfying

i. Non-negativity: \( \mu(\{x_i\}) \geq 0 \) for any \( x_i \in X \)
ii. Countable additivity: For every countable sequence of mutually disjoint measureable sets \( \{A_i \mid i = 1, 2, \ldots \} \), we have

\[
\mu\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} \mu(A_i)
\]

In reality, length, area and volume are instances of measure concept. However, it is not enough to explain all situations by classical measures. There is, for example, a famous aphorism, one boy is a boy; two boys half boy; three boys no boy, which means the efficiency may be reduced when workers’ cooperation is incompatible and unhappy. So we cannot acquire the total efficiency by simply adding their efficiencies.

2.3 Monotone measure

Let \((X, \mathcal{P}(X))\) be a measurable space and \(\mu : \mathcal{P}(X) \to [0, \infty]\) be an extended real-valued set function.

Set function \(\mu : \mathcal{P}(X) \to [0, \infty]\) is called a monotone measure on \((X, \mathcal{P}(X))\) iff it satisfies the following requirements:

I. \(\mu(\emptyset) = 0\) (vanishing at the empty set)

II. \(\mu(E) \leq \mu(F)\) wherever \(E \subseteq F\) (monotonicity)

In this case, \((X, \mathcal{P}(X), \mu)\) is called a monotone measure space.

A monotone measure \(\mu\) is called normalized if \(\mu(X) = 1\).

A monotone measure \(\mu\) is called subadditive if

\[\mu(E \cup F) \leq \mu(E) + \mu(F)\]
A monotone measure \( \mu \) is called **superadditive** if
\[
\mu(E \cup F) \geq \mu(E) + \mu(F) \quad \text{when } E \cap F = \emptyset
\]

It is easy to see that monotone measure \( \mu \) is additive if and only if it is both subadditive and superadditive.

3. **Basic concept of optimization**

Consider \( m \) numerical variables \( t_1, t_2, \ldots, t_m \) and a target \( z = z(t_1, t_2, \ldots, t_m) \) defined on a subset \( D \) of the \( m \)-dimensional Euclidian space \( \mathbb{R}^m \). Given \( F \subseteq D \), we want to find a point \(( t_1^*, t_2^*, \ldots, t_m^* )\) subject to the restriction \(( t_1^*, t_2^*, \ldots, t_m^* ) \in F\) such that \( z(t_1^*, t_2^*, \ldots, t_m^*) \) is the smallest value of \( z \) in \( F \), that is,
\[
z(t_1^*, t_2^*, \ldots, t_m^*) \leq z(t_1, t_2, \ldots, t_m) \quad \text{for any } (t_1, t_2, \ldots, t_m) \in F.
\]

This is a **minimization** problem, where \( z \) is called the **objective function** (or **target**), set \( F \) is called the **feasible region**, any point \(( t_1, t_2, \ldots, t_m ) \in F \) is called a **feasible point**, point \(( t_1^*, t_2^*, \ldots, t_m^* )\) is called the **global minimizer** (simply called **minimizer** if there is no confusion), denoted by \( \text{argmin} z(t_1, t_2, \ldots, t_m) \), of the minimization problem, and value \( z(t_1^*, t_2^*, \ldots, t_m^*) \) is called the **minimum** of \( z \) in \( F \). A point \(( t_1^{'}, t_2^{'}, \ldots, t_m^{'}) \in F \) is called a **local minimizer**, if there exists an open set \( O \) containing \(( t_1^{'}, t_2^{'}, \ldots, t_m^{'}) \) such that
\[
z(t_1^{'}, t_2^{'}, \ldots, t_m^{'}) \leq z(t_1, t_2, \ldots, t_m) \quad \text{for any } (t_1, t_2, \ldots, t_m) \in O \cap F.
\]

Similarly, we have concepts of **maximization**, **global maximizer**, **maximum**, and **local maximizer** correspondingly. Both of minimization and maximization are called **optimization**.
Formally, a minimization problem in $m$-dimensional Euclidian space can be expressed as follows.

$$\min z = z(t_1, t_2, \ldots, t_m)$$

subject to $g_k(t_1, t_2, \ldots, t_m) \geq \alpha_k, \ k = 1, 2, \ldots, r$, \hspace{1cm} (1)

where $\alpha_k, \ k = 1, 2, \ldots, r$, are constants and the inequalities describe the restriction, that is, the feasible region of the minimization problem is

$$F = \{ (t_1, t_2, \ldots, t_m) \mid g_k(t_1, t_2, \ldots, t_m) \geq \alpha_k, \ k = 1, 2, \ldots, r \}$$

For any maximization problem, $\max z = z(t_1, t_2, \ldots, t_m)$ can be rewritten as $\min z' = -z(t_1, t_2, \ldots, t_m)$, that is, it can be converted into a form of (1). Furthermore, if there is some inequality with “less than or equal to”, say, $g(t_1, t_2, \ldots, t_m) \leq \alpha$ in the restriction, we can rewrite it as $-g(t_1, t_2, \ldots, t_m) \geq -\alpha$. Finally, if some restriction condition is an equality, e.g., $g(t_1, t_2, \ldots, t_m) = \alpha$, then we can separate it into two inequalities $g(t_1, t_2, \ldots, t_m) \geq \alpha$ and $-g(t_1, t_2, \ldots, t_m) \geq -\alpha$. Thus, we call the form shown in (1) the standard form of an optimization problem. We should note that inequalities “greater than” and “less than” are never used in restrictions for optimization problem because they will lead to no solution usually.

In (1), if all functions $z$ and $g$’s are linear, then the optimization problem is called a linear programming problem; otherwise, i.e., if at least one of functions $z$ and $g$’s is nonlinear, it is called a nonlinear programming problem.
4. Identification of monotone measures

Given set function $\mu : \mathcal{P}(X) \rightarrow [0, \infty)$ with $\max\{\mu(E) \mid E \subseteq X\} > 0$, we want to find a monotone measure $\nu : \mathcal{P}(X) \rightarrow [0, \infty)$, such that the total squared error

$$z = \sum_{E \subseteq X} [\nu(E) - \mu(E)]^2 \quad (2)$$

is minimized. Under this criterion, the optimal solution $\nu$ cannot be obtained by only reordering the values of $\mu$ generally, even if $\mu(X) \geq \mu(E)$ for all $E \in \mathcal{P}(X)$. This can be seen from the following counterexample.

Let $X = \{x_1, x_2, x_3\}$ and

$$\mu(E) = \begin{cases} 
0 & \text{if } E = \emptyset \\
7 & \text{if } E = \{x_1\} \\
2 & \text{if } E = \{x_2\} \\
5 & \text{if } E = \{x_1, x_2\} \\
4 & \text{if } E = \{x_3\} \\
8 & \text{if } E = \{x_1, x_3\} \\
9 & \text{if } E = \{x_2, x_3\} \\
10 & \text{if } E = X 
\end{cases}$$

Set function $\mu$ is nonnegative. It is not monotone since $\mu(\{x_1, x_2\}) < \mu(\{x_1\})$.

If we construct set function $\nu$ from $\mu$ by exchanging the values of $\mu(\{x_1, x_2\})$ and $\mu(\{x_1\})$ i.e., let
then \( \nu \) is a monotone revision of \( \mu \). However, it is not the optimal monotone revision of \( \mu \) under the criterion of minimizing \( (2) \). In fact, according \( (2) \), the total squared error of \( \nu \) is \( z(\nu) = 2^2 + 2^2 = 8 \). If we take

\[
\nu'(E) = \begin{cases}
0 & \text{if } E = \emptyset \\
5 & \text{if } E = \{x_1\} \\
2 & \text{if } E = \{x_2\} \\
7 & \text{if } E = \{x_1,x_2\} \\
4 & \text{if } E = \{x_3\} \\
8 & \text{if } E = \{x_1,x_3\} \\
9 & \text{if } E = \{x_2,x_3\} \\
10 & \text{if } E = X
\end{cases}
\]

as a monotone revision of \( \mu \), then its total squared error is \( z(\nu^*) = 1^2 + 1^2 = 2 \), which is much smaller than \( z(\nu) \).

In this example, \( \nu^* \) is obtained only via taking the average of a pair of two \( \mu \)'s values, which violates the required monotonicity to replace the pair. To obtain a monotone revision of a given nonnegative set function vanishing at the empty set, in general case, is not so simple. It is convenient to use a soft computing technique, such as a genetic algorithm, with an embedded reordering algorithm to obtain an
approximate numerical optimal solution for this identification, especially, when the generalized model that is similar to ones discussed in later. The genetic algorithm with an embedded reordering algorithm can be also used for identification of monotone measures based on an input-output nonlinear integral system. One of the advantages of using soft computing techniques is that the algorithm, except the formula of the fitness function, does not depend on the choice of the optimization criterion.

The generalized identification model for monotone measures is expressed as follows.

Given \( l \) rough observations (records) for a monotone measure \( \nu: \mathcal{P}(X) \to [0, \infty) \) with \( \nu(X) > 0 \), denoted by \( \mu_j, \ j = 1, 2, \ldots, l \), where each \( \mu_j \) is a nonnegative set function defined on \( \mathcal{P}(X) \) and \( l \) is a positive integer, we want to find a monotone measure \( \nu \) on \( \mathcal{P}(X) \) such that

\[
z = \sum_{j=1}^{l} \sum_{E \subseteq X} [\nu(E) - \mu_j(E)]^2 \quad (3)
\]

is minimized. Expression (2) can be regarded as a special case of (3) with \( l = 1 \).

We assume that not all observations are trivial, i.e., not all values of every observed set function \( \mu_j \) are zeros. A genetic algorithm now is used to solve this identification problem. It consists of two parts: main algorithm and the embedded reordering algorithm.

5 Genetic Algorithms

Genetic algorithm is one of the soft computing techniques used in optimization. It is a global parallel random search method. Genetic algorithm mimics the natural
evolution process of a species in a given environment to obtain an optimal (or approximate optimal) solution after a large number of generations.

Suppose that a given optimization problem (say, a minimization problem) involves $m$ variables (unknown parameters) with feasible region $F$. A genetic algorithm for solving the optimization problem may contain the following components.

1) **Encoding genes.** By using suitable transformations, the $m$ unknown parameters are respectively represented by $m$ binary bit strings, which are randomly and independently generated obeying the uniform distribution, such that the feasible region $F$ is well covered with a reasonable distribution. Each binary bit string is called a *gene*. The length of each binary bit string is determined according to the required precision of the corresponding parameter in the solution of the optimization problem. For example, if the required precision of a parameter is $10^{-3}$, then 10 bits is needed in the corresponding gene since $2^{-9} > 10^{-3} > 2^{-10}$. Adopting a real coding to replace the binary coding is also workable.

2) **Chromosomes.** According to a fixed order, link the $m$ genes to form a *chromosome*. It is also a string of bits. Its length is the sum of the length of all genes. A chromosome represents a candidate of the set consisting of values of all unknown parameters in the optimization problem. So, each chromosome should be in $F$.

3) **Population.** The *Population* is a set of chromosomes. The number of chromosomes in the population is called the *size*, denoted by $s$, of the population. The size is usually a large positive even integer, such as 100 or 500. The population is initialized by generating chromosomes randomly. Keeping the size, the population will be renewed in the evolution process.
Relatively to the population, each chromosome is called an *individual*.

4) **Fitness function.** According to the goal of the optimization, choose a criterion and, by which, construct a suitable *fitness function* that measures the goodness of each chromosome. Usually, the fitness is a function of the value of the objective function. Decode each individual if it is necessary and then calculate its fitness.

5) **Genetic operators.** Design several *genetic operators* for producing new chromosomes by using existing chromosomes that are selected as the parents. The following common operators are suggested to be used, though the user may design new genetic operators according to the need of the given optimization problem.

   a) **Two-point crossover.** The *crossover* is a binary operator. For selected two chromosomes, randomly choose two points (each point here is a location between two successive bits) to separate each of them into three pieces, and then interchange their middle piece to form two new chromosomes (see Figure 1).

   ![Figure 1: Two-point Crossover](image-url)
b) **Three-point mutation.** The three-point mutation is a unary operator. For each selected chromosome, randomly select three bits and switch their 0 and 1 to obtain a new chromosome (see Figure 2).

![Figure 2: Three-point Mutation](image)

6) **Selecting parents.** According to the individuals’ fitness, randomly select two individuals as a pair of parents. The higher the fitness is, the easier to be selected.

c) **Two-point realignment.** The two-point realignment is also a unary operator. For each selected chromosome, randomly choose two points (the same as in figure 1, each point here is also a location between two successive bits) to separate it into three pieces, and then realign them in a randomly selected order and direction (see Figure 3). There are totally 48 different ways to the realignment. They have the same chance to be selected, i.e., each way has a chance of $1/48$ to be selected.

![Figure 3: Two-point Realignment](image)
7) **Produce offspring.** According to a selected probability distribution, randomly choose a genetic operator to produce two individuals from the selected parents.

8) **Renewing the population.** After producing a certain number, e.g., the same as the initial population size $s$, of new individuals, add them into the population. Then, according to the fitness, select the best $s$ individuals to form a new population, called a new *generation*.

9) **Recurrence and stopping.** Based on the new population, recycle procedure 4) - 8) until the stopping condition is satisfied. The *stopping condition* may be chosen anyone from the following or their combinations.
   a) The number of generations (or the number of produced individuals) reaches a given positive integer.
   b) After producing a given number of generations (or individuals), the fitness of the best individual in the population has not been changed (or not been significantly improved).
   c) The population is identical.
   d) The value of the objective function for the best individual in the current population falls in a given small region, such as the error being smaller than a given small positive number $\epsilon$.

10) **Outputting the result.** Once stopping, output the best individual (after decoding) in the current population.

A general flow chat of the genetic algorithm is shown in Figure 4. Since the genetic algorithm is a global search method, we needn’t worry about falling in a valley of local minimizer. This is the advantage of the genetic algorithm. However,
the search process of a genetic algorithm is usually rather slow, even appears a phenomenon so called prematurity, that is, after several generation, the fitness of the best individual has not been significantly improved, though it is still a little far from the global minimizer. To speed the search procedure and reduce the premature, we may adopt two additional approaches as follows. One is to keep the diversity of the population, that is, to select some individuals not very good but with much different feature from the good individuals in the new generation to avoid the population being identical or similar too early. Another is to set up some adaptive mechanisms in the search procedure, i.e., some involved probability distributions are dynamically adjusted. For example, at the beginning of the search procedure, the probability of taking mutation may be smaller than later and the probability of choosing higher bits may be larger than lower bits for mutation.

Figure 4: Flow chart of generic algorithm
6 Main Algorithm

The genetic algorithm shown in previous section can be adopted with a few adjustments as follows. Each chromosome now consists of $2^n - 1$ binary coded genes $g_1, g_2, \ldots, g_{2^n - 1}$ representing the values of a set function at all nonempty subsets of $X$ respectively. The target of the optimization is the total squared error $z$ shown in 3.

The decoding formulas are

\[ \mu(\{x_1\}) = \frac{g_1}{1 - g_1}, \]
\[ \mu(\{x_2\}) = \frac{g_2}{1 - g_2}, \]
\[ \mu(\{x_1, x_2\}) = \frac{g_3}{1 - g_3}, \]
\[ \mu(\{x_3\}) = \frac{g_4}{1 - g_4}, \]
\[ \mu(\{x_1, x_3\}) = \frac{g_5}{1 - g_5}, \]
\[ \ldots \]
\[ \mu(X) = \frac{g_{2^n - 1}}{1 - g_{2^n - 1}} \]

After decoding each chromosome, the following reordering algorithm should be used to convert the nonnegative set function $\mu$ to a monotone measure $\nu$. Then, according (3), calculate the total squared error for $\nu$. 

15
Reordering algorithm: Assume that set function $\mu : \mathcal{P}(X) \to [0, \infty)$ with $\mu(\emptyset) = 0$ is given.

1) According to the lattice structure of the power set $\mathcal{P}(X)$ (see Figure 5 when $n = 4$), divide all $2^n$ subsets of $X$ into $n + 1$ layers: the empty set is at layer 0 (the bottom layer); all singletons are at layer 1; all set consisting of two points are at layer 2; … ; the universal set is at layer $n$ (the top layer). That is, any set consisting of $k$ points is arranged in layer $k$, $k = 0, 1, 2, \ldots, n$. The class of all sets at layer $k$ is denoted as $\mathcal{L}_k$.

2) Initialize $k = 1$

3) For layer $k$, rearrange these $C(n, k)$ sets according their values of $\mu$ in a decreasing order, denoted as $\{E_{kj} | j = 1, 2, \ldots, C(n, k)\}$.

Figure 5: Lattice structure for the power set of a universal set with 4 attributes
4) Starting from \( j = 1 \), find the set \( E_{kj}^* \) \arg\min\{\mu(E) \mid E \supset E_{kj}, E \text{ belongs to the layer } k+1\} \), i.e., \( E_{kj}^* \) is the set with the smallest value of \( \mu \) among all sets that include \( E_{kj} \) in \( \mathcal{T}_{k+1} \).

Exchange the values of \( \mu(E_{kj}) \) and \( \mu(E_{kj}^*) \) if \( \mu(E_{kj}) > \mu(E_{kj}^*) \); otherwise, they are unchanged. Then do this for the next \( j \) and continue this procedure until \( j = C(n, k) - 1 \).

5) Add 1 to \( k \) and check whether \( k = n \). If yes, go to 6); if no, go to 3).

6) Check whether for at least one value of \( k \) the exchange in step 4) has been done. If yes, go to 2); if no, go back to the main algorithm.

Once the embedded reordering algorithm finishes and diverts back to the main algorithm, the actual set function \( \mu \) has been reordered to be a monotone measure \( \nu \) on \( \mathcal{P}(X) \). The complexity of the reordering algorithm above can still be reduced a little.

In an iteration beginning from 2), if the first \( r + 1 \) layers (form layer 0 to layer \( r \)) are not involved for any exchange, then \( k \) can be initialized with \( r \) in the next iteration.

If all considered set functions, including the monotone measure \( \nu \) and its Observations \( \mu_j, j = 1, 2, \ldots, l \), are regular, i.e., \( \nu(X) = \mu_1(X) = \mu_2(X) = \ldots = \mu_l(X) = l \), then each chromosome only needs \( 2^n - 2 \) genes. In this case, the total squared error formula 3 is reduced to be

\[
z = \sum_{j=1}^{l} \sum_{E \subset X} (\nu(E) - \mu_j(E))^2
\]
7 Experiments

The proposed approach was applied to several random examples. Details of each of these examples are discussed below. Our initial results are obtained based on implementation in C.

The experimental methodology used is the following. We provided three sets of exports. In each set, three attributes are described where seven observations are given manually. The identification of monotone was computed by bounded for 30 or 35 generations. Many randomizers are employed in the paper such as generation of seven random observations, the choices of three generic operations, the positions’ choice for each generic operation, the parents’ choice and so on. Due to the indeterminacy principle of randomizers, the results are not exactly same and sometimes even have huge difference. In order to retrieve the best results, we implemented 10 times for each example and choose the best results as our final results. In order to study the data further, we draw the figure according to fitness function and error analysis. The fitness function using in this paper is $1000 \frac{1}{1+100z}$; and the mean square error is $\frac{z^*}{l}$ where $z^* = \sum_{j=1}^{l} \sum_{E \subseteq X} [v(E) - \mu^*(E)]^2$.

In order to prove our approach effective, the monotone data is given in the first example. Three exports have the same 8 observations: $\mu (\{x_1\}) = 1, \mu (\{x_2\}) = 2, \mu (\{x_1, x_2\}) = 3, \mu (\{x_3\}) = 4, \mu (\{x_1, x_3\}) = 5, \mu (\{x_2, x_3\}) = 6, \mu (\{x_1, x_2, x_3\}) = 7$. After executing our approach, we acquire the best result: $\mu (\{x_1\}) = 1.036, \mu (\{x_2\}) = 1.653, \mu (\{x_1, x_2\}) = 3.031, \mu (\{x_3\}) = 4.044, \mu (\{x_1, x_3\}) = 4.885, \mu (\{x_2, x_3\}) = 6.014, \mu (\{x_1, x_2, x_3\}) = 7.000$. 
From the figure 6, we observe the fitness values becomes higher and higher from the first generation (fitness = 0.818) to the 19th generation (fitness = 23.418). The tendency becomes flat from the 20th generation (fitness = 23.560) to the 30th generation, which means we cannot produce better result.

From the figure 7, we also observe the error results becomes lower and lower from the first generation (mean square error = 0.19381) to the 19th generation (mean square error = 0.006619). Note that the mean square error in 0 generation is 0. The tendency becomes flat from the 20th generation (mean square error = 0.005671), which means the precision of our approach is 0.005671.
Now we go back to our example mentioned in fourth section as our second example. Three exports have the same 8 observations: $\mu (\{x_1\}) = 7$, $\mu (\{x_2\}) = 2$, $\mu (\{x_1, x_2\}) = 5$, $\mu (\{x_3\}) = 4$, $\mu (\{x_1, x_3\}) = 8$, $\mu (\{x_2, x_3\}) = 9$, $\mu (\{x_1, x_2, x_3\}) = 10$. After executing our approach, we acquire the best result: $\mu (\{x_1\}) = 6.314$, $\mu (\{x_2\}) = 1.456$, $\mu (\{x_1, x_2\}) = 6.420$, $\mu (\{x_3\}) = 3.947$, $\mu (\{x_1, x_3\}) = 8.143$, $\mu (\{x_2, x_3\}) = 9.039$, $\mu (\{x_1, x_2, x_3\}) = 9.779$.

![Figure 8: Fitness results for example 2 based on 35 generations](image)

From the figure 8, we observe the fitness values becomes higher and higher from the first generation (fitness = 0.057) to the 30th generation (fitness = 1.155). The tendency becomes flat from the 31st generation to 35th generation (fitness = 1.156), which means we cannot produce better result.
From the figure 9, we also observe the error results becomes lower and lower from the first generation (mean square error = 2.78095) to the 30th generation (mean square error = 0.137286). The tendency becomes flat from the 31st generation to 35th generation (mean square error = 0.136048), which means the precision of our approach is 0.136048.

We randomly choose observations from three exports as our third example. The first export has 8 observations: \( \mu (\{x_1\}) = 4, \mu (\{x_2\}) = 6, \mu (\{x_1, x_2\}) = 1, \mu (\{x_3\}) = 5, \mu (\{x_1, x_3\}) = 8, \mu (\{x_2, x_3\}) = 9, \mu (\{x_1, x_2, x_3\}) = 10 \). The second export: \( \mu (\{x_1\}) = 4, \mu (\{x_2\}) = 6, \mu (\{x_1, x_2\}) = 2, \mu (\{x_3\}) = 5, \mu (\{x_1, x_3\}) = 8, \mu (\{x_2, x_3\}) = 9, \mu (\{x_1, x_2, x_3\}) = 10 \). The third export: \( \mu (\{x_1\}) = 4, \mu (\{x_2\}) = 6, \mu (\{x_1, x_2\}) = 3, \mu (\{x_3\}) = 5, \mu (\{x_1, x_3\}) = 8, \mu (\{x_2, x_3\}) = 9, \mu (\{x_1, x_2, x_3\}) = 10 \). After executing our approach, we acquire the best result: \( \mu (\{x_1\}) = 3.830, \mu (\{x_2\}) = 4.172, \mu (\{x_1, x_2\}) = 4.361, \mu (\{x_3\}) = 4.721, \mu (\{x_1, x_3\}) = 8.394, \mu (\{x_2, x_3\}) = 8.660, \mu (\{x_1, x_2, x_3\}) = 10.011 \).

From the figure 10, we observe the fitness values becomes higher and higher from the first generation (fitness = 0.093) to the 27th generation (fitness = 0.332). The tendency becomes flat from the 28th generation to 35th generation (fitness = 0.334),
which means we cannot produce better result.

![Figure 11: Mean square error results for example 3 based on 35 generations](image)

From the figure 11, we also observe the error results becomes lower and lower from the first generation (mean square error = 1.713952) to the 27th generation (mean square error = 0.478619). The tendency becomes flat from the 28th generation to 35th generation (mean square error = 0.474429), which means the precision of our approach is 0.474429.

8 Conclusions

In order to identifying a monotone measure, we propose an innovative approach to optimally determine a monotone measure using the genetic algorithm. Reordering algorithm is also describe and embedded in the main algorithm. According to the lattice structure of the power set of the given universal set, a max-min strategy is adopted to reduce the computational complexity. The proposed approach is illustrated by applying it to several random data sets. Our experiments show that this approach is effective.
References


