A Fast Algorithm for Finding Matching Responses in Survey Data Table

Abstract. The paper addresses an algorithm to perform an analysis on survey data tables with some unreliable entries. The algorithm has almost linear complexity depending on the number of elements in the table. The proposed technique is based on a monotonicity property. An implementation procedure of the algorithm contains a recommendation that might be realistic for clarifying the analysis results.
Keywords: survey; boolean; data table; matrix.

1. INTRODUCTION

Situations in which customer responses being studied are measured by means of survey data arise in the market investigations. They present problems for producing long-term forecasts because the traditional methods based on counting the matching responses in the survey with a large customer population are hampered by unreliable human nature in the answering and recording process. Analysis institutes are making considerable and expensive efforts to overcome this uncertainty by using different questioning techniques, including private interviews, special arrangements, logical tests, “random” data collection, questionnaire scheme preparatory spot tests, etc. However, percentages of responses representing the statistical parameters rely on misleading human nature and not on a normal distribution. It appears thereby impossible to exploit the most simple null hypothesis technique because the distributions of similar answers are unknown. The solution developed in this paper to overcome the hesitation effect of the respondent, and sometimes unwillingness, rests on the idea of searching so-called “agreement lists” of different questions. In the agreement list, a significant number of respondents do not hesitate in choosing the identical answer options, thereby expressing their willingness to answer. These respondents and the agreement lists are classified into some two-dimensional lists – "highly reliable blocks".

For survey analysts with different levels of research experience, or for the people mostly interested in receiving results by their methods, or merely for those who are familiar with only one, "the best survey analysis technique", our approach has some advantages. Indeed, in the survey, data are collected in such a way that can be regarded as respondents answering a series of questions. A specific answer is an option such as displeased, satisfied, well contented, etc. Suppose that all respondents participating in the survey have been interviewed using the same questionnaire scheme. The resulting survey data can then be arranged in a table $X = \langle x_{iq} \rangle$, where x_{iq} is a Boolean vector of options available, while the respondent i is answering the question q. In this respect, the primary table X is a collection of Boolean columns where each column in the collection is filled with Boolean elements from only one particular answer option. Our algorithm will always try to detect some highly reliable blocks in the Table X bringing together similar columns, where only some trustworthy respondents are answering identically. Detecting these blocks, we can separate the survey data. Then, we can reconstruct the data back from those blocks into the primary survey data table $X' = \langle x'_{iq} \rangle$ format, where some "non-matching/ doubtful" answers are removed. Such a "data-switch" is not intended to replace the researchers' own methods, but may be complementary used as a "preliminary data filter" - separator. The analysts' conclusions will be more accurate after the data-switch has been done because each filtered data item is a representative for some "well known sub-tables".

Our algorithm in an ordinary form dates back to Mullat (1971). At first glance, the ordinary form seems similar to the greedy heuristic (Edmonds 1971), but this is not the case. The starting point for the ordinary version of the algorithm is the entire table from which the elements are removed. Instead, the greedy heuristic starts with the empty set, and the elements are added until some criterion for stopping is fulfilled. However, the algorithm developed in the present paper is quite different. The key to our paper is that the properties of the algorithm remain unchanged under the current construction. For matching responses in the Boolean table, it has a lower complexity.

The monotone property of the proposed technique - "monotone systems idea" - is a common basis for all theoretical results. It is exactly the same property (iii) of submodular functions brought up by Nemhauser et al (1978, p.269). Nevertheless, the similarity does not itself diminish the fact that we are studying an independent object, while the property (iii) of submodular set functions is necessary, but not sufficient.

From the very start, the theoretical apparatus called the "monotone system" has been devoted to the problem of finding some parts in a graph that are more "saturated" than any other part with "small" graphs of the same type (see Mullat, 1976). Later, a Markov chain replaced the graph presentation form where the rows-columns may be split implementing the proposed technique into some sequence of submatrices (see Mullat, 1979). There are numerous applications

exploiting the monotone systems ideas; see Ojaveer et al (1975). Many authors have developed a thorough theoretical basis extending the original conception of the algorithm; see Libkin et al (1990) and Genkin and Muchnik (1993).

The rest of the paper is organized as follows. In Section 2, a reliability criterion will be defined for blocks in the Boolean table B. This criterion guarantees that the shape of the top set of our theoretical construction is a sub-matrix - a block; see the Proposition 1. However, the point of the whole monotone system idea is not limited by our specific criterion as described in Section 2. This idea addresses the question: How to synthesize an analysis model for data matrix using quite simple rules? In order to obtain a new analysis model, the researcher has only to find a family of π-functions suitable for the particular data. The shape of top sets for each particular choice of the family of π-functions might be different; see the note prior to our formal construction. For practical reasons, especially in order to help the process of interpretation of the analysis results, in Section 3 there are some recommendations on how to use the algorithm on the somewhat extended Boolean tables $B^{\pm}$. Section 4 is devoted to an exposition of the algorithm and its formal mathematical properties, which are not yet utilized widely by other authors.

2. RELIABILITY CRITERION

In this Section we deal with the criterion of reliability for blocks in the Boolean tables originating from the survey data. In our case we analyze the Boolean table $B = \langle b_{ij} \rangle$ representing all respondents $\langle 1,..., i,..., n \rangle$, but including only some columns $\langle 1,..., j,...m \rangle$ from the primary survey data table $X = \langle x_{iq} \rangle$; see above. The resulting data of each table B can be arranged in a $n \times m$ matrix. Those Boolean tables are then subjected to our algorithm separately, for which reason there is no difference between any subtable in the primary survey data and a Boolean table. A typical example is respondent satisfaction with services offered, where $b_{ij} = 1$ if respondent i is satisfied with a particular service j level, and $b_{ij} = 0$ if he is unsatisfied. Thus, we analyze any Boolean table of the survey data independently.

Let us find a column j with the *most* significant frequency F of 1-elements among all columns and throughout all rows in table B. Such rows arrange a $g = 1$ *one*-column sub-table pointing out only those respondents who prefer *one* specific *most* significant column j. We will treat, however, a more general criterion. We suggest looking at some significant number of respondents where at least F of them are granting at least g Boolean 1-elements in each single row within the range of a particular number of columns. Those columns arrange what we call an agreement list, $g = 2,3,...$; g is an agreement level.

The problem of how to find such a significant number of respondents, where the F criterion reaches its global maximum, is solved in Section 4. An optimum table S^*, which represents the outcome of the search among all "subsets" H in the Boolean table B, is the solution; see Theorem I. The main result of the Theorem I ensures that there are at least F positive responses in each column in table S^*. No superior sub-table can be found where the number of positive responses in each column is greater F. Beyond that, the agreement level is at least equal to $g = 2,3,...$ in each row belonging to the best sub-table S^*; g is the number of positive responses within the agreement list represented by columns in sub-table S^*. In case of an agreement level $g = 1$, our algorithm in Section 4 will find out only *one* column j with the *most* significant positive frequency F among all columns in table B and throughout all respondents, see above. Needless to say that it is worthless to apply our algorithm in that particular case $g = 1$, but the problem becomes fundamental as soon as $g = 2,3,...$.

Let us look at the problem more closely. The typical attitude of the respondents towards the entire list of options — columns in table B — can be easily "accumulated" by the total number of respondent i positive hits selected:

$$r_i = \sum_{j=1,...,m} b_{ij}.$$

Similarly, each column - option can be measured by means of the entire Boolean table B as

$$c_j = \sum_{i=1,...,n} b_{ij}.$$

It might appear that it should be sufficient to choose the whole table B to solve our problem provided that $r_i \geq g, i = \overline{1,n}$. Nevertheless, let us look throughout the whole table and find the worse case where the number $c_j, j = \overline{1,m}$ reaches its minimum F. Strictly speaking, it does not mean that the whole table B is the best solution just because some "poor" columns (options with rare responses - hits) may be removed in order to raise the worst-case criterion F on the remaining columns. On the other hand, it is obvious that while removing "poor" columns, we are going to decrease some r_i numbers, and now it is not clear whether in each row there are at least $g = 2,3,...$ positive responses. Trying to proceed further and removing those

"poor" rows, we must take into account that some of c_j numbers decrease and, consequently, the F criterion decreases as well. This leads to the problem of how to find the optimum sub-table S^*, where in the worst-case F criterion reaches its *global maximum*? The solution is in Section 4.

Finally, we argue that the intuitively well-adapted model of 100% matching 1-blocks is ruled out by any approach trying to qualify the real structure of the survey data. It is well known that the survey data matrices arising from questionnaires are fairly empty. Those matrices contain plenty of small 100% matching 1-blocks, whose individual selection makes no sense. We believe that the local worst-case criterion F top set, found by the algorithm, is a reasonable compromise. Instead of 100% matching 1-blocks, we detect somewhat blocks less than 100% filled with 1-elements, but larger in size.

3. RECOMMENDATIONS

We consider the interpretation of the survey analysis results as an essential part of the research. This Section is designed to give guidance on how to make the interpretation process easier. In each survey data it is possible to conditionally select two different types of questions: (1) The answer option is a fact, event, happening, issue, etc.; (2) The answer is an opinion, namely displeased, satisfied, well contented etc.; see above. It does not appear from the answer to options of type 1, which of them is positive or negative, whereas type 2 allows us to separate them. The goal behind this splitting of type 2 opinions is to extract from the primary survey data table two Boolean sub-tables: table B^+, which includes type 1 options mixed with the positive options from type 2 questions, and table B^- where type 1 options are mixed together with the negative type 2 options - opinions. It should be noticed that doing it this way, we are replacing the analysis of primary survey data by two Boolean tables where each option is represented by one column. Tables B^+ and B^- are then subjected to the algorithm separately.

To initiate our procedure, we construct a sub-table K_1^+ implementing the algorithm on table B^+. Then, we replace sub-table K_1^+ in B^+ by zeros, constructing a restriction of table $B^{\pm}$. Next, we implement the algorithm on this restriction and find a sub-table K_2^+, after which the process of restrictions and sub-tables sought by the algorithm may be continued. For practical purposes we suggest stopping the extraction with three sub-tables: K_1^+, K_2^+ and K_3^+. We can use the same procedure on the table B^-, extracting sub-tables K_1^-, K_2^- and K_3^-.

The number of options-columns in the survey Boolean tables $B^{\pm}$ is quite significant. Even a simple questionnaire scheme might have hundreds of options - the total number of options in all questions. It is difficult, perhaps almost impossible, within a short time to observe those options among thousands of respondents. Unlike Boolean tables $B^{\pm}$, the sub-tables $K^{\pm}_{1,2,3}$ have reasonable dimensions. This leads to the following interpretation opportunity: the positive options in $K^{+}_{1,2,3}$ tables indicate some most successful phenomena in the research while the negative options in $K^{-}_{1,2,3}$ point in the opposite direction. Moreover, the positive and negative sub-tables $K^{\pm}_{1,2,3}$ enable the researcher in a short time to "catch" the "sense" in relations between the survey options of type 1 and positive/negative options of the type 2. For instance, to observe all Pearson's *r* correlations a calculator has to perform $O(n \cdot m^2)$ operations depending on the $n \times m$ table dimension, n-rows and m-columns. The reasonable dimensions of the sub-tables $K^{\pm}_{1,2,3}$ can reduce the amount of calculations drastically. Those sub-tables - blocks $K^{\pm}_{1,2,3}$, which we recommend to select in the next Section as index-function $F(H)$ top sets found via the algorithm, are not embedded and may not have intersections; see the Proposition 1. Concerning the interpretation, it is hoped that this simple approach can be of some use to researchers in elaborating their reports with regard to the analysis of results.

4. Definitions and Formal Mathematical Properties

In this Section, our basic approach is formalized to deal with the analysis of the Boolean $n \times m$ table B, n-rows and m-columns. Henceforth, the table B will be the Boolean table $B^{\pm}$ - see above - representing certain options-columns in the survey data table. Let us consider the problem of how to find a sub-table consisting of a subset S_{max} of the rows and columns in the original table B with the properties: (1) that $r_i = \sum_j b_{ij} \geq g$ and (2) the minimum over j of $c_j = \sum_i b_{ij}$ is as large as possible, precisely – the global maximum. The following algorithm solves the problem.

Algorithm.

Step I. Set up the initial values.

1i. Set minimum and maximum bounds a, b on threshold u for c_j values.

Step A. To find that the next step **B** produces a non-empty sub-table.

1a. Using step **B**, test u as $(a+b)/2$.

If it succeeds, replace a by u. If it fails replace b by u.

2a. Go to **1a**.

Step B. To test whether the minimum over j can be at least u.

1b. Delete all rows whose sums $r_i < g$.

This step **B** fails if all must be deleted; return to step **A**.

2b. Delete all columns whose sums $c_j \leq u$.

This step **B** fails if all must be deleted, return to step **A**.

3b. Perform step **T** if none deleted in **1b** and **2b**; otherwise go to **1b**.

Step T. Test that the global maximum is found.

1t. Among numbers c_j find the minimum.

With this new value as u test performing step **B**.
If it succeeds, return to step **A**, otherwise final stop.

Step **B** performed through the step **T** tests correctly whether a sub-matrix of B can have the rows sums at least g and the column sums at least u. Removing row i, we need to perform no more than m operations to recalculate c_j values; removing column j, we need no more than n-operations. We can proceed through **1b** no more than n-times and through **2b**, m-times. Thus, the total number of operations in step **B** is $O(nm)$. The step **A** tests the step **B** no more than $\log_2 n$ times. Thus, the total complexity of the algorithm is $O(\log_2 n \times nm)$ operations.

Note. It is important to keep in mind that the algorithm itself is a particular case of our theoretical construction. As one can see, we are deleting rows and columns including their elements all together, thereby ensuring that the outcome from the algorithm is a sub-matrix. But, in order to expose the properties of the algorithm, we look at the Boolean elements separately. However, in our particular case of π-functions it makes no difference. The difference will be evident if we utilize some other family of π-functions, for instance

$\pi = c_j \max(r_i, c_j)$. We may detect top binary relations, which we call kernels, different from submatrices. It may happen that some kernel includes two blocks - one quite long in the vertical direction and the other - in the horizontal. All elements in the empty area between these blocks in some cases cannot be added to the kernel. In general, we cannot guarantee either the above low complexity of the algorithm for all families of π-functions, but the complexity still remains in reasonable limits.

We now consider the properties of the algorithm in a rigorous mathematical form. Below we use the notation $H \subseteq B$. The notation H contained in B will be understood in an ordinary set-theoretical vocabulary, where the Boolean table B is a set of its Boolean 1-elements. All 0-elements will be dismissed from the consideration. Thus, H, as a binary relation, is also a subset of a binary relation B. However, we shall soon see that the top binary relations - kernels from the theoretical point of view are also sub-matrices for our specific choice of π-functions. Below, we refer to an element we assume that it is a Boolean 1-element.

For an element $\alpha \in B$ in the row i and column j we use the similarity index $\pi = c_j$ if $r_i \geq g$ and $\pi = 0$ if $r_i < g$, counting only on Boolean elements belonging to H. The value of π depends on each subset $H \subseteq B$ and we may thereby write $\pi \equiv \pi(\alpha, H)$: the set H is called the π-function parameter. The π-function values are the real numbers - the similarity indices. In Section 2 we have already introduced these indices on the entire table B. Similarity indices, as one can see, may only concurrently increase with the "expansion" and decrease with the "shrinking" of the parameter H. This leads us to the fundamental definition.

Definition 1. Basic monotone property. *By a monotone system will be understood a family* $\{\pi(\alpha, H) : H \subseteq B\}$ *of* π*-functions, such that the set* H *is to be considered as a parameter with the following monotone property: for any two subsets* $L \subset G$ *representing two particular values of the parameter* H *the inequality* $\pi(\alpha, L) \leq \pi(\alpha, G)$ *holds for all elements* $\alpha \in B$.

We note that this definition indicates exactly that the fulfilment of the inequality is required for all elements $\alpha \in B$. However, in order to prove the Theorems 1,2 and the Proposition 1, it is sufficient to demand the inequality fulfillment only for elements $\alpha \in L$; even the numbers π themselves may not be defined for $\alpha \notin L$. On the other hand, the fulfillment of the inequality is necessary to prove the argument of the Theorem 3 and the Proposition 2. It is obvious that similarity indices $\pi = c_j$ comply with the monotone system requirements.

Definition 2. *Let* $V(H)$ *for a non-empty subset* $H \subseteq B$ *by means of a given arbitrary threshold* u° *be the subset* $V(H) = \{\alpha \in B : \pi(\alpha, H) \geq u^\circ\}$. *The non-empty* H*-set indicated by* S° *is called a stable point with reference to the threshold* u° *if* $S^\circ = V(S^\circ)$ *and there exists an element* $\xi \in S^\circ$, *where* $\pi(\xi, S^\circ) = u^\circ$. See Mullat (1981, p.991) for a similar concept.

Definition 3. *By monotone system kernel will be understood a stable set* S^* *with the maximum possible threshold value* $u^* = u_{max}$.

We will prove later that the very last pass through the step **T** detects the largest kernel $\Gamma_p = S^*$. Below we are using the set function notation $F(X) = \min_{\alpha \in X} \pi(\alpha, X)$.

Definition 4. *An ordered sequence* $\alpha_0, \alpha_1, ..., \alpha_{d-1}$ *of distinct elements in the table B, which exhausts the whole table,* $d = \sum_{i,j} b_{ij}$, *is called a defining sequence if there exists a sequence of sets* $\Gamma_0 \supset \Gamma_1 \supset \ldots \supset \Gamma_p$ *such that:*

A. *Let the set* $H_k = \{\alpha_k, \alpha_{k+1}, ..., \alpha_{d-1}\}$. *The value* $\pi(\alpha_k, H_k)$ *of an arbitrary element* $\alpha_k \in \Gamma_j$, *but* $\alpha_k \notin \Gamma_{j+1}$ *is strictly less than* $F(\Gamma_{j+1})$, $j = 0,1,..., p-1$.

B. *In the set* Γ_p *there does not exist a proper subset* L, *which satisfies the strict inequality* $F(\Gamma_p) < F(L)$.

Definition 5. *A subset* D^* *of the set* B *is called definable if there exists a defining sequence* $\alpha_0, \alpha_1, ..., \alpha_{d-1}$ *such that* $\Gamma_p = D^*$.

Theorem 1. *For the subset* S^* *of* B *to be the largest kernel of the monotone system - to contain all other kernels - it is necessary and sufficient that this set is definable:* $S^* = D^*$. *The definable set* D^* *is unique.*

We note that the Theorem 3 will establish the existence of the largest kernel later.

Proof.

Necessity. If the set S^* is the largest kernel, let's look at the following sequence $B = \Gamma_0 \supset \Gamma_1 = S^*$ of only two sets. Suppose we have found elements $\alpha_0, \alpha_1, ...\alpha_k$ in $B \setminus S^*$ such that for each $i = \overline{1,k}$ the value

$\pi(\alpha_i, B \setminus \{\alpha_0,...,\alpha_{i-1}\})$ is less than $u^o = u_{max}$ and $\alpha_0, \alpha_1,...\alpha_k$ does not exhaust $B \setminus S^*$. Then, in $(B \setminus S^*) \setminus \{\alpha_0,...,\alpha_k\}$ some α_{k+1} exists such that $\pi(\alpha_{k+1}, (B \setminus S^*) \setminus \{\alpha_0,...,\alpha_k\}) < u^*$. Otherwise, the set $(B \setminus S^*) \setminus \{\alpha_0,...,\alpha_k\}$ is a larger kernel than with the same value u^*. Thus, the induction is complete.

This gives the ordering with the property (a). If the property (b) failed, then u^* would not be a maximum, contradicting the definition of the kernel. This proves the necessity.

Sufficiency. Note that every time the algorithm — see above — goes through step **T**, some stable point, a set S° is put in the form of a set $\Gamma_j = S^\circ$, $j = 0,1,..., p-1$, where $u_j = \min_{\alpha \in S^\circ} \pi(\alpha, S^\circ)$. Obviously, these stable "layering" points (stable sets) form an embedded chain of sets $B = \Gamma_0 \supset \Gamma_1 \supset ... \supset \Gamma_p = D^*$. Let the set $L \subseteq B$ be the largest core. Suppose that this L is a proper subset of D^*, then by property (b) $F(D^*) \geq F(L)$ and hence D^* is also a kernel. The set L as the largest kernel cannot be a proper subset of D^* and therefore must be equal to D^*.

Suppose now that L is not the subset of D^*. Let H_s be the smallest set $H_k = \{\alpha_k, \alpha_{k+1},..., \alpha_{d-1}\}$, which includes L. The value $\pi(\alpha_s, H_s)$ by our basic monotone property must be grater than, or at least equal to u^*, since α_s is an element of H_s and it is also an element of the kernel L and $L \subseteq H_s$. By property (a) this value is strictly less than $F(\Gamma_{j+1})$ for some $j = 0,1,..., p-1$. But that contradicts the maximality of u^*. This proves the sufficiency. Moreover, it proves that any largest kernel equals D^* so that it is the unique largest kernel. This concludes the proof. ■

Proposition 1. *The largest kernel is a sub-matrix of the table* B.

Proof. Let S^* be the largest kernel. If we add to S^* any element lying in a row and a column where S^* has existing elements, then the threshold value u^* cannot decrease. So by maximality of the set S^* this element must already be in S^*. ■

Now, we need to focus on the individual properties of the sets $\Gamma_0 \supset \Gamma_1 \supset ... \supset \Gamma_p$, which have a close relation to the case $u < u_{max}$ - a subject for a separate inquiry. Let us look at the step **T** of the algorithm originating the series of mapping initiating from the whole table B in form of $V(B), V(V(B)),...$ with some particular threshold u. We denote $V(V(B))$ by $V^2(B)$, etc.

Definition 6. *The chain of sets* $B, V(B), V^2(B),...$ *with some particular threshold u is called the central series of monotone system; see Mullat (1981) for exactly the same notion.*

Theorem 2. *Each set* $\Gamma_0 \supset \Gamma_1 \supset ... \supset \Gamma_p$ *in the defining sequence* $\alpha_0, \alpha_1,..., \alpha_{d-1}$ *is the central series convergence point* $\lim_{k=2,3,...} V^k(B)$ *as well as the stable point for some particular thresholds values* $F(W) = u_0 < u_1 < ... < u_n = F(S^*)$. *Each* Γ_j *is the largest stable point - including all others for threshold values* $u \geq u_j = F(\Gamma_j)$.

It is not our intention to prove the statement of Theorem 2 since this proof is similar to that of Theorem 1. Theorem 1 is a particular case for Theorem 2 statement regarding threshold value $u = u_p$.

Next, let us look at the formal properties of all kernels and not only the largest one found by the algorithm. It can easily be proved that with respect to the threshold $u_{max} = u_p$ the subsystem of all kernels classifies a structure, which is known as an upper semilattice in lattice theory.

Theorem 3. *The set of all kernels - stable points - for* u_{max} *is a full semilattice.*

Proof. Let Ω be a set of kernels and let $K_1 \in \Omega$ and $K_2 \in \Omega$. Since the inequalities $\pi(\alpha, K_1) \geq u$, $\pi(\alpha, K_2) \geq u$ are true for all K_1 and K_2 elements on each K_1, K_2 separately, they are also true for the union set $K_1 \cup K_2$ due to the basic monotone property. Moreover, since $u = u_{max}$, we can always find an element $\xi \in K_1 \cup K_2$ where $\pi(\xi, K_1 \cup K_2) = u$. Otherwise, the set $K_1 \cup K_2$ is some H -set for some u' greater than u_{max}. Now, let us look at the sequence of sets $V^k(K_1 \cup K_2)$, $k = 2,3,...$, which certainly converges to some non empty set - stable point K. If there exists any other kernel $K' \supset K_1 \cup K_2$, it is obvious, that applying the basic monotone property we get that $K' \supseteq K$. ■

With reference to the highest-ranking possible threshold value $u_p = u_{max}$, the statement of Theorem 3 guarantees the existence of the largest stable point and the largest kernel S^* (compare this with equivalent statement of Theorem 1).

Proposition 2. *Monotone system Kernels are sub-tables of the table* B.

Proof. The proof is similar to proposition 1. However, we intend to repeat it. In the monotone system all elements outside a particular kernel lying in a row and a column where the kernel has existing elements belong to the kernel. Otherwise, the kernel is not a stable point because these elements may be added to it without decreasing the threshold value u_{max}.

Note that Propositions 1,2 are valid for our specific choice of similarity indices $\pi = c_j$. The point of interest might be to verify what π-function properties guarantee that the shape of the kernels still is a sub-matrix. The defining sequence of table B elements constructed by the algorithm represents only some part $u_0 < u_1 < u_2 < ... < u_p$ of the threshold values existing for central series in the monotone system. On the other hand, the original algorithm, Mullat (1971), similar to the inverse Greedy Heuristic, produces the entire set of all possible threshold values u for all possible central series, what is sometimes unnecessary from a practical point of view. Therefore, the original algorithm always has the higher complexity.

ACKNOWLEDGMENTS

The author is grateful to an anonymous referee for useful comments, style corrections and especially for the suggestion regarding the induction mechanism in the proof of the necessity of the main theorem argument.

REFERENCES

Edmonds J. (1971) Matroids and the Greedy Algorithm, Math. Progr., No. 1 pp. 127-136.

Genkin, A.V. and I.B. Muchnik. (1993) Fixed Points Approach to Clustering, Journal of Classification 10, pp. 219-240.

Libkin, L.O., Muchnik, I.B. and L.V. Shvartser. (1990) Quasilinear monotone systems, Automation and Remote Control 50 1249-1259,

Mullat, J.E. a) (1971) On the Maximum Principle for Some Set Functions, Tallinn Techn. Univ. Proceedings., Ser. A, No. 313, pp. 37-44; b) (1976, 1977) Extremal Subsystems of Monotonic Systems, I,II,III, Automation and Remote Control 37, pp. 758-766, 37, pp. 1286-1294, 38, pp. 89-96; c) (1979) Application of Monotonic system to study of the structure of Markov chains, Tallinn Technical University Proceedings, No. 464, 71;
(d) (1981) Counter monotonic Systems in the Analysis of the Structure of Multivariate Distributions, Automation and Remote Control 42, pp. 986-993.

Nemhauser, G.L., Walsey, L.A. and M.L. Fisher. (1978) An Analysis of Approximations for Maximizing Submodular Set Functions, Mathematical Programming 14, pp. 265-294.

Ojaveer, E., Mullat, J. E. and L.K. Vôhandu. (1975) A Study of Infraspecific Groups of the Baltic East Coast Autumn Herring by two new Methods Based on Cluster Analysis, Estonian Contributions to the International Biological Program 6, Tartu, pp. 28-50.