Universität Bielefeld

Technische Fakultät
Abteilung Informationstechnik
Forschungsberichte

2-Stage Fault Tolerant Interval Group Testing

Ferdinando Cicalese José Augusto Amgarten Quitzau

Report 2007-02

Impressum: Herausgeber:
Ellen Baake, Robert Giegerich, Ralf Hofestädt, Franz Kummert,
Peter Ladkin, Ralf Möller, Helge Ritter, Gerhard Sagerer,
Jens Stoye, Holger Theisel, Ipke Wachsmuth

Technische Fakultät der Universität Bielefeld,
Abteilung Informationstechnik, Postfach 10 01 31,
33501 Bielefeld, Germany

ISSN 0946-7831

2-Stage Fault Tolerant Interval Group Testing

Ferdinando Cicalese[1] and José Augusto Amgarten Quitzau[2]

[1] AG Genominformatik, Technical Faculty, Bielefeld University, Germany
Dept. Computer Science and Appl., University of Salerno, Italy
[2] International NRW Graduate School in Bioinformatics and Genome Research, Center of Biotechnology, Bielefeld University, Germany

Abstract. We study the following fault tolerant variant of the interval group testing model: Given three positive integers n, p, e, determine the minimum number of questions needed to identify a (possibly empty) set $P \subseteq \{1, 2, \ldots, n\}$ ($|P| \leq p$), under the following constraints. Questions have the form "Is $I \cap P \neq \emptyset$?", where I can be any interval in $\{1, 2 \ldots, n\}$. Up to e of the answers can be erroneous or lies. Questions are to be organized in batches of non-adaptive questions. Therefore, questions in a given batch can be formulated relying only on the information gathered with the answers to the questions in the previous batches.
The study of interval group testing is motivated by several applications. Among others, it has applications to the problem of identifying splice sites in a genome. To the best of our knowledge, we are the first to consider fault tolerant strategies for interval group testing.
We completely characterize the fully non-adaptive situation and provide tight bounds for the case of two batch strategies. Our bounds only differ by a factor of $\sqrt{11/10}$ for the case $p = 1$ and at most 2 in the general case.

1 Introduction

Problem Statement. In this paper we consider fault tolerant algorithms for the variant of group testing arising when the search space is a linearly ordered set and the queries are constrained to be intervals of this set. This is called *interval group testing*. More precisely, an instance of the problem is given by two non-negative integers p and n and a subset $P \subseteq O = \{1, 2, \ldots, n\}$, such that $|P| \leq p$. The set O is the search space and P is the set of *positive* objects that have to be identified. Queries (tests) are constrained to be intervals $\{i, i+1, \ldots, j-1, j\}$, for some $i, j \in \{1, 2, \ldots, n\}$. The target is to identify P by using the minimum possible number of queries. We assume that tests are arranged in *stages*: in each stage a certain number of tests is performed non-adaptively, while tests of a given stage can be determined depending on the outcomes of the tests in all previous stages. Finally, we assume that up to a finite number e of the answers might be erroneous or lies.

For each value of the parameters n, p, s, e we want to determine $\mathcal{N}(n, p, s, e)$, the worst-case number of tests that are necessary (and sufficient) to successfully

identify all positives in a search space of cardinality n, under the hypothesis that the number of positives is at most p, s-stage algorithms are used and up to e answers are lies.

Our Results. We focus on strategies that use adaptiveness at most once, i.e., strategies with questions organized in one or two batches of non adaptive queries. In fact, according to [**?**] *... the technicians who implement the pooling strategies generally dislike even the* 3*-stage strategies that are often used [...]. The pools are either tested all at once or in a small number of stages (usually at most 2).* We exactly determine $N(n, p, 1, e)$ and provide very tight bounds for the $N(n, p, 2, e)$ that in the case $p = 1$ at most differ by a factor of $\sqrt{11/10}$, and at most by a factor 2 in all the other cases. We remark that these are the first non trivial results on fault tolerant interval group testing procedures.

Motivations and Related Research.

Group testing is a basic paradigm in the theory of combinatorial search which has proved to be useful in a variety of situations such as quality control, multiple access communication, computational molecular biology, data compression, and data streams algorithms among the others (see [5, 6, 8, 13, 16, 3]).

Our main motivation for the study of interval group testing comes from its application to the problem of determining exon-intron boundaries within a gene [14, 17]. In a very simplified model, a gene is a collection of disjoint substrings within a long string representing the DNA molecule. These substrings are called *exons*, and the substrings separating them are called *introns*. Each boundary point linking an exon and an intron is called a *splice site*, because introns are spliced out during transcription. Determining the splice sites is an important task, e.g., when searching for mutations associated with a gene responsible for a disease.

In [17] a new experimental protocol is proposed that searches for the exons boundaries using group testing. This consists of selecting two positions in the cDNA, a copy of the original genomic DNA from which introns have been spliced out, and determine whether they are at the same distance as they were in the original genomic DNA string. If these distances do not coincide then at least one intron (and hence a splice site) must be present in the genomic DNA between the two selected positions. The formulation of splice sites identification as a group testing problem with interval queries is explicitly stated in [12, 14, 17]. The advantages of splice site detection by distance measurements over sequence-based methods using, e.g., Hidden Markov Models are that this method works without expensive sequencing of genomic DNA and it gives the results directly from experiments, without relying on inference rules. The work [17] and the book [14] report about the experimental evaluation, on real data, of the algorithm ExonPCR, that finds exon-intron boundaries within a gene. The authors of [17] give also a simple asymptotic analysis of their $\Theta(\log n)$-stage algorithm. The question was whether there exist less obvious but more efficient query strategies for Interval Group Testing, and more importantly, algorithms able to cope with the technical limitation of the experiments, remarkably with errors. We remark that non-adaptive strategies are desirable in this context, in order to avoid long

waiting periods, however a totally non-adaptive algorithm (with $s = 1$) needs unreasonably many queries. We can trade more stages for fewer queries. In [1] the first rigorous algorithmic study of the problem was presented, and for the case $s \leq 2$ a precise evaluation of $\mathcal{N}(n, p, s)$ was given. In the present paper we sharpen our earlier results for $s \geq 3$, as stated above. In [2] a sharper asymptotic estimation of $N(n, p, s, 0)$ was given that is optimal up to the constant of the main term in the case of large s.

Interval group testing also arises in other domains like: detecting holes in a gas pipe [5, 4], finding faulty links in an electrical or communication network, data gathering in sensor networks [9–11].

2 Definitions and Notation

In this section we present some concepts necessary to understand the following sections, as well as the notation used in the text. The set of objects where we try to find the positives is usually simply described as a set of size n. In these cases, we always mean the set of integers $[n] = \{1, 2, \ldots, n\}$. By abuse of notation we shall use square bracket to denote intervals in $[n]$. Then, for each $1 \leq i \leq j \leq n$, we shall use $[i, j]$ to denote the set $\{i, i + 1, \ldots, j\}$. By definition each query asks about the intersection of a given interval with the set of positive elements. Therefore, we shall identify a query with the interval it specifies. We say that a query $Q \equiv [i, j]$ covers an element $k \in [n]$ if $k \in [i, j]$.

A query $Q \equiv [i, j]$ has two boundaries: the left, $(i - 1, i)$, and the right, $(j, j + 1)$. For the sake of definiteness, we assume that, for any a, a the query $[1, a]$ has left boundary $(0, 1)$, and the query $[a, n]$ has right boundary $(n, n+1)$. A multiset of queries $\mathcal{Q}$ defines a set of boundaries $\mathcal{B}(\mathcal{Q}) = \{(i_1, i_1 + 1), (i_2, i_2 + 1), \ldots\}$, where $i_k < i_{k+1}$. Every interval of type $\{i_k + 1, i_k + 2, \ldots, i_{k+1}\}$ is called a *piece*. Because every query has two distinct boundaries, but two queries may share some boundaries, we have $|\mathcal{B}(\mathcal{Q})| \leq 2\,|\mathcal{Q}|$. A boundary B of a piece P is said to be *turned to* the piece if there is a query Q such that $P \subset Q$ and B is also a boundary of Q. A piece is called a *2-piece* if both its boundaries are turned to it. A piece that has only one of it boundaries turned to it is called a *1-piece*. If none of the boundaries of a piece are turned to it, the piece is called a *0-piece*. Figure 1 illustrate the definitions given so far.

Fig. 1. Graphic summary of the concepts presented so far. The thicker line represents the set $[n]$ of objects. There are two interval queries, which partition the objects set into 5 pieces.

We also define a *YES set* for $\mathcal{Q}$ as a set of query intervals in $\mathcal{Q}$ that are answered YES consistently with a choice of the positives and of the number of errors in the answers not violating the standing rules of the game. A YES set is called *specific* if the intersection of all its queries correspond to a single piece,

otherwise it is called *unspecific*. More formally, a YES set $\mathcal{Y} \subset \mathcal{Q}$ is specific if and only if there is a piece P of $\mathcal{Q}$ such that

$$\bigcap_{Q \in \mathcal{Y}} Q = P.$$

Notice that a YES set can be unspecific for different reasons, e.g., when more than one positive is in P or when only one or no positives are in P and one or more answers are lies.

3 Non-adaptative Interval Group Testing with One Error

We start our analysis with the case of 1 stage strategies. In fact, the results in this section will be the basis for the analysis of the more complicated two batch case.

The following two theorems completely characterize 1-stage e-fault tolerant interval group testing.

Theorem 1. *For all $n \geq 1$ and $e \geq 0$, it holds that $\mathcal{N}(n,1,1,e) = \left\lceil \frac{(2e+1)(n+1)}{2} \right\rceil$.*

Proof. The lower bound directly follows from the following claim.
Claim. In any strategy identifying the (only) positive or correctly reporting $P = \emptyset$, there are at least $2e+1$ questions' boundaries $(i, i+1)$ for each $i = 0, 1, \ldots, n$.

Fix a strategy where for a specific $i \in [n]$ there are $b \leq 2e$ questions with a boundary $(i, i+1)$. Let $\mathcal{Q}$ be the set of such questions and $\mathcal{Q}_1$ the set of all questions in $\mathcal{Q}$ that contain i. Assume, without loss of generality, that $|\mathcal{Q}_1| \geq |\mathcal{Q} \setminus \mathcal{Q}_1|$.

Suppose that the adversary answers i) NO to all the questions having empty intersection with $\{i, i+1\}$, ii) YES to all questions including both i and $i+1$, iii) YES to exactly $\lceil |\frac{\mathcal{Q}}{2}| \rceil$ questions in $\mathcal{Q}_1$ and NO to the remaining ones in $\mathcal{Q}_1$, iv) answers YES to all the questions in $\mathcal{Q} \setminus \mathcal{Q}_1$.

A moment reflection shows that, due to the possibility of having up to e erroneous answers, the above set of answers does not allow to determine whether i or $i+1$ is an element of P[3].

Therefore, any strategy that is able to correctly identify P must use in total at least $(2e+1)(n+1)$ boundaries. Then, the desired results follows by observing that each question can cover at most 2 boundaries.

We now turn to the upper bound. Direct inspection shows that for $n \leq 3$ there exists a strategy attaining the desired bound.

For each $k \geq 2$, let $\mathcal{A}_{2k+1} = \{[1,2],[2,4],[4,6],\ldots,[2k-2,2k],[2k,2k+1]\}$ and $\mathcal{A}^1_{2k} = \{[2,2k-1],[3,2k-2],\ldots,[k,k+1]\}$, $\mathcal{A}^2_{2k} = \{[1,k],[k+1,2k]\}$, and $\mathcal{A}^3_{2k} = \{[1,k]\}$.

For $n \geq 4$, the following strategy attains the desired bound.

[3] In particular, for the cases, $i = 0$ (respectively $i = n$) the ambiguity is whether P contains no elements or the element is 1 (resp. n).

If n is odd, the strategy consists of asking $2e+1$ times the questions in $\mathcal{A}_n$. These amount to $(2e+1)\lceil (n+1)/2 \rceil = \lceil (2e+1)(n+1)/2 \rceil \rceil$ questions and they clearly cover $2e+1$ times each boundary $(i, i+1)$, for each $i = 0, 1, \ldots, n$.

If n is even, let $k = n/2$. Now, the strategy consists of asking $2e+1$ times the questions in $\mathcal{A}_n^1$, plus $e+1$ times the questions in $\mathcal{A}_n^2$, plus e times the questions in $\mathcal{A}_n^3$. In total, in this case, the strategy uses $(2e+1)(k-1) + 2(e+1) + e = (2e+1)k + e + 1 = \lceil (2e+1)(2k+1)/2 \rceil = \lceil (2e+1)(n+1)/2 \rceil$, as desired.

For the case of more positives we can generalize the above statement as follows.

Theorem 2. *For all integers $n \geq 1, p \geq 2, e \geq 0$, it holds that $\mathcal{N}(n, p, 1, e) = (2e+1)n$*

Proof. The upper bound is trivially obtained by a strategy made of $(2e+1)$ copies of the singleton questions $\{1\}, \{2\}, \ldots, \{n\}$.

The lower bound is obtained proceeding as in the previous theorem. Here, we argue that every strategy that correctly identifies P must ask, for each $i = 1, 2, \ldots, n-1$, at least $2e+1$ questions with boundary $(i, i+1)$ and including i, and at least $2e+1$ questions with boundary $(i, i+1)$ and including $i+1$. Moreover, it must ask at least $2e+1$ questions with boundary $(0, 1)$ and $2e+1$ questions with boundary $(n, n+1)$. For otherwise, assume that there exists $i \in \{1, 2, \ldots, n-1\}$, such that one of the above $4e+2$ boundaries $(i, i+1)$ is missing. Proceeding as in the proof of the previous theorem, it is possible to define an answering strategy for the adversary such that it is not possible to discriminate between the case $P = \{i\}$ and the case $P = \{i, i+1\}$. Alternatively, if some of the above boundaries $(0, 1)$ (resp. $(n, n+1)$) are missing, the adversary can answer in such a way that it is not possible to discriminate between the case $P = \emptyset$ or $P = \{1\}$ (resp. $P = \{n\}$).

4 Bounds for Two-Stage Strategies with One Positive

We adapt the bounds for the 2-stage strategy for 2 positives, given by Cicalese et al. [1], to the case where a 2-strategy for at most one positive may contain at most one error. In our case, the weights assigned to the piece π_i is given by the following scheme:

- A piece gets weight $\frac{1}{2}$ if it may contain a positive only in the case an error already occurred.
- A piece gets weight $\frac{3}{2}$ if it may contain a positive, but it is not clear if an error already occurred.

Notice that when we assign the weights according to this scheme, we automatically adapt the following lemma to this case.

Lemma 1. *([1][Lemma 2]) Consider a multiset of k YES sets, and for each $i = 1, 2, \ldots, k$ and $j = 1, 2, \ldots, l$, let w_{ij} be the weight of the jth piece in the*

YES vector associated to the ith YES set. If there is $r > 0$ such that for all $j = 1, 2, \ldots, l$, it holds that $\sum_{i=1}^{k} \geq r$, then an adversary can force at least $\frac{nr}{k}$ queries in the second stage.

And the lemma above is the key for proving the following theorem:

Theorem 3. $\mathcal{N}(n, 1, 2, 1) \geq \sqrt{5n}$.

Proof. We show that we may achieve $r \geq 2.5$ using at most $t_1 + 2$ queries, where t_1 is the number of queries in the first stage. Then we always need at least

$$\min\left(t_1 + 2 + \frac{2.5n}{2t_1 + 4}\right) = \sqrt{5n}$$

queries.

To achieve $r \geq 2.5$, we create a specific YES set for each piece created by the t_1 queries in the first stage. This already guarantees $r \geq 1.5$. If we consider each consecutive pair of pieces, we may see that they may fall in one of the following cases, depending on how many boundaries separate them:

1 boundary: Consider the case where two pieces are separated by the boundary $(i, i + 1)$, which is the boundary of exactly one query. Even if a consistent YES set indicates the piece containing i as the one having a positive, there is the chance that exactly the query having the boundary $(i, i + 1)$ failed. In order to safely identify the positive, we need to investigate the piece containing $i + 1$. Notice that an error-free strategy may be used in this case. The case where the piece containing $i + 1$ has the positive is symmetric. As a consequence, each piece in a pair of neighbors separated by a single boundary automatically gets an extra weight $\frac{1}{2}$.

2 boundaries: If exactly two queries have boundary $(i, i + 1)$, a consistent YES set always indicates precisely one of the pieces as the one containing a positive. In these cases, we don't get the extra weight of $\frac{1}{2}$ for both neighbors. However, we can use the fact that, since there is no piece between these two boundaries, the number of pieces is at most $2t_1 - 1$, and so is the number of YES sets used so far. Therefore we may create an unspecific YES set involving both pieces. This gives us the desired extra weight $\frac{1}{2}$ to each piece.

3 boundaries or more: Using the same argument as in the previous case, if a pair of pieces is separated by more than 2 boundaries, then the number of pieces is at most $2t_1 - 2$. We may use two of this extra pieces to create a new specific YES set for each piece in the pair. At the end, each of the pieces gets an extra weight of $\frac{3}{2}$.

The analysis of these cases shows that it is possible to extend the previously suggested multiset of YES sets in such a way that each piece gets extra weight $\frac{1}{2}$ from each of its neighbors. As a result, all the pieces, but the ones on the extremities, surely have sum of weights at least 2.5. If it is necessary, we may always create two extra consistent YES sets so that the pieces on the extremities also get the desired total weight.

□

For the upper bound in this case, we show a 2-stage query scheme which is able to find a positive in a set of n elements using at most $\sqrt{5.5n}$. In this strategy, we use two groups of queries, as shown in Fig. 2:

A Composed by t_A overlapping queries that divide the set of objects in $2t_A$ pieces of the same size.

B The queries in group B are rt_A overlapping queries, for $0 < r < 1$.

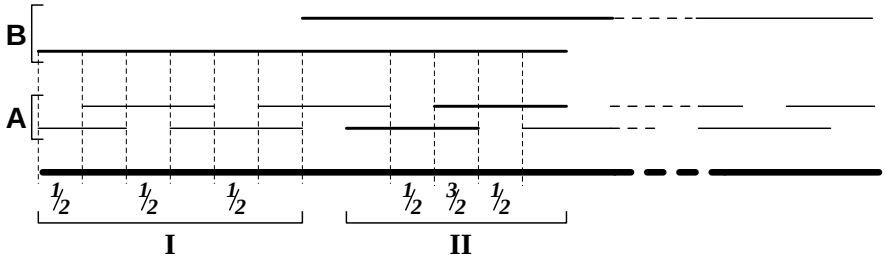

Fig. 2. Query scheme used in the first stage of a 2-stage strategy. The thicker line represents the set of objects, whereas all the other represent the a- and b-queries. In the figure we may see the two patterns that compete for the worst case. Darker lines indicate queries that answer YES.

An inspection of the possible YES sets gives us two situations that compete for the worst case:

I. When only one b-query answers YES, it may be the case that this answer is correct. This means that one of the a-queries lies. In any case, since at most one error is allowed, the positive must be in one of the single-covered pieces. As a consequence, all of these pieces covered by the b-query need to be checked in the second stage. The good news is that we may use the error-free approach to find the positive. The positive is surely in the portion covered by a single b-query, otherwise we already have 2 errors. This gives us $\frac{n}{8rt_A}$ queries in the second stage.

II. When two a-queries answer YES, together with the corresponding b-query, we must look for a positive in the piece corresponding to the overlapping part as if there was no error. We also need to consider the hypothesis that one of the a-queries gave the wrong answer. Therefore we the two pieces corresponding to the non-overlapping part must also be investigated. In the last case, we may take advantage of the fact that they are only possible in the presence of one error, and use the error-free strategy on the two pieces of size $\frac{n}{2t_A}$. This gives us a total of $\frac{5n}{4t_A}$ questions in the second stage.

The total number of queries used by this strategy is given by

$$\min\left(t_A(1+r) + \max\left(\frac{5n}{4t_A}, \frac{n}{8rt_A}\right)\right).$$

By choosing $r = 0.1$, we equalize both worst case candidates and get

$$\min\left(1.1t_A + \frac{5n}{4t_A}\right) = \sqrt{5.5n}.$$

5 Bounds for Two-Stage Strategies with More Positives

Lemma 2.

$$N(n, p, 2, 1) \geq \sqrt{6n(p-1)} - O(1).$$

Proof. Assume that the adversary accepts not to lie in the first phase. Moreover, she/he agrees to put the positives into the $p-1$ largest pieces defined by the first stage of queries.

Notice that this information, exchanged between the questioner andf the adversary, can only make the situation better for the questioner.

Let q be the number of questions in the first phase. These questions divide the search space into at most $2q+1$ pieces. Hence, the largest $p-1$ of these pieces have total size at least $(p-1)n/(2q+1)$.

Since each of these pieces might contain up to 2 positives, by Theorem 2 the questioner has to ask at least 3 questions per element in each of these pieces.

So we have that the number of questions asked by an algorithm that uses q queries in the first stage is at least $q + 3(p-1)n/(2q+1)$.

Thus, minimizing over all possible values of q we have the desired bound.

Lemma 3.

$$N(n, p, 2, 1) \leq 2\sqrt{6n(p-1)}.$$

Proof. We consider the following strategy, where q is a parameter to be decide later. In the first stage, we divide the search space into q non-overlapping intervals of equal size. We call them segments. We, then we ask twice one question coinciding with each segment.

Let $\mathcal{A}$ be the set of segments such that the two corresponding questions are answered YES. Let $\mathcal{B}$ the set of segments whose corresponding questions are answered NO. Finally, let $\mathcal{C}$ the set of segments such that one of the corresponding questions is answered YES and one is answered NO.

Since we are assuming at most one error, trivially, no question is necessary in the second stage in each segment in $\mathcal{B}$.

We also have $|\mathcal{C}| \leq 1$.

We can now have the following cases.

Case 1. $|\mathcal{A}| \leq p-1$, $|\mathcal{C}| = 0$. Then, since each segment π might contain more than 1 positive, and the adversary might still lie, by Theorem 2, $3|\pi|$ questions have to be asked in π in the second stage. SInce all segments are of the same size, in total we have $2q + 3|\mathcal{A}|n/q$ questions are asked in this case.

Case 2. $|\mathcal{A}| \leq p-1$, $|\mathcal{C}| = 1$. Again, each segment π might contain more than 1 positive. However, in this case the adversary has clearly already used a lie. Then,

by Theorem 2, for each segment $\pi \in \mathcal{A},$, $|\pi|$ questions have to be asked in π in the second stage. Moreover, for the only segment γ, in $\mathcal{C}$ either $|\gamma|/2$ or $3|\gamma|/2$ questions have to be asked, according as $\mathcal{A}$ contains $p-1$ or less segments. In fact, in the first case, γ can contain at most one positive, and Theorem 1 applies. Whilst in the second case, γ might contain more than one positive and then Theorem 1 applies. Since all segments are of the same size, in total we have $2q + 3(p-1)n/2q + n/2q$ questions in the first case and $2q + 3(|\mathcal{A}| + 1)n/2q$ in the second case ($|\mathcal{A}| \leq p-2$).

It is not hard to see that the worst situation for the questioner is given by *Case 1* with $|\mathcal{A}| = p - 1$.

Thus, the above strategy uses in total at most $2q + +3(p-1)n/q$ questions. Minimizing with respect to q we have the desired bound.

References

1. F. Cicalese, P. Damaschke, U. Vaccaro, *Optimal group testing strategies with interval queries and their application to splice site detection*, Int. Journal of Bioinformatics Research and Applications.
2. F. Cicalese, P. Damaschke, L. Tansini, S. Werth, *Overlaps Help: Improved Bounds for Group Testing with Interval Queries*, to appear in Discrete Applied Mathematics.
3. G. Cormode, S. Muthukrishnan, *What's hot and what's not: Tracking most frequent items dynamically*, in: ACM Principles of Database Systems, 2003
4. L. A. Cox, X. Sun, and Y. Qiu, *Optimal and Heuristic Search for a Hidden Object in one Dimension*, in: Proc. of IEEE Conf. on System, Man, and Cybernetics, 1252–1256, 1994.
5. D.Z. Du, F.K. Hwang, *Combinatorial Group Testing and its Applications*, World Scientific, Singapore, 2000.
6. M. Farach, S. Kannan, E.H. Knill, S. Muthukrishnan, *Group testing with sequences in experimental molecular biology*, in: Proc. of Compression and Complexity of Sequences 1997, B. Carpentieri, A. De Santis, U. Vaccaro, J. Storer (Eds.), IEEE CS Press, pp. 357-367, 1997.
7. M. Gelfand, A. Mironov, P.A. Pevzner, M. Roytberg, S.H. Sze, PROCRUSTES: Similarity-based gene recognition via spliced alignment, `http://www-hto.usc.edu/software/procrustes/`
8. E. H. Hong, R.E. Ladner, *Group testing for image compression*, IEEE Transactions on Image Processing, **11(8)**, pp. 901-911, 2002.
9. Y.W. Hong, A. Scaglione, *On multiple access for distributed dependent sources: A content-based group testing approach*, IEEE Information Theory Workshop ITW 2004.
10. Y.W. Hong, A. Scaglione, *Group testing for sensor networks: the value of asking the right answers*, Asilomar Conference 2004.
11. Y.W. Hong, A. Scaglione, *Generalized group testing for retrieving distributed information*, ICASSP 2005.
12. R. Karp, *ISIT'98 Plenary Lecture Report: Variations on the theme of 'Twenty Questions'*, IEEE Information Theory Society Newsletter, vol. **49**, No.1, March 1999.

13. H.Q. Ngo, D.Z. Du, *A survey on combinatorial group testing algorithms with applications to DNA library screening*, in: *Discrete Mathematical Problems with Medical Applications*, DIMACS Ser. Discrete Math. Theoret. Comput. Sci., **55**, Amer. Math. Soc., pp. 171-182, 2000.
14. P.A. Pevzner, *Computational Molecular Biology, An Algorithmic Approach*, MIT Press, 2000.
15. M. Sobel, P.A. Groll, *Group testing to eliminate efficiently all defectives in a binomial sample*, The Bell Systems Technical Journal, **38**, pp. 1179-1253, 1959.
16. J. Wolf, *Born again group testing: Multiaccess communications*, IEEE Trans. Information Theory, **IT-31**, pp. 185-191, 1985.
17. G. Xu, S.H. Sze, C.P. Liu, P.A. Pevzner, N. Arnheim, *Gene hunting without sequencing genomic clones: Finding exon boundaries in cDNAs*, Genomics, **47**, pp. 171-179, 1998.

Bisher erschienene Reports an der Technischen Fakultät
Stand: 2007-08-13

94-01 Modular Properties of Composable Term Rewriting Systems
(Enno Ohlebusch)

94-02 Analysis and Applications of the Direct Cascade Architecture
(Enno Littmann, Helge Ritter)

94-03 From Ukkonen to McCreight and Weiner: A Unifying View of Linear-Time Suffix Tree Construction
(Robert Giegerich, Stefan Kurtz)

94-04 Die Verwendung unscharfer Maße zur Korrespondenzanalyse in Stereo Farbbildern
(André Wolfram, Alois Knoll)

94-05 Searching Correspondences in Colour Stereo Images – Recent Results Using the Fuzzy Integral
(André Wolfram, Alois Knoll)

94-06 A Basic Semantics for Computer Arithmetic
(Markus Freericks, A. Fauth, Alois Knoll)

94-07 Reverse Restructuring: Another Method of Solving Algebraic Equations
(Bernd Bütow, Stephan Thesing)

95-01 PaNaMa User Manual V1.3
(Bernd Bütow, Stephan Thesing)

95-02 Computer Based Training-Software: ein interaktiver Sequenzierkurs
(Frank Meier, Garrit Skrock, Robert Giegerich)

95-03 Fundamental Algorithms for a Declarative Pattern Matching System
(Stefan Kurtz)

95-04 On the Equivalence of E-Pattern Languages
(Enno Ohlebusch, Esko Ukkonen)

96-01 Static and Dynamic Filtering Methods for Approximate String Matching
(Robert Giegerich, Frank Hischke, Stefan Kurtz, Enno Ohlebusch)

96-02 Instructing Cooperating Assembly Robots through Situated Dialogues in Natural Language
(Alois Knoll, Bernd Hildebrand, Jianwei Zhang)

96-03 Correctness in System Engineering
(Peter Ladkin)

96-04 An Algebraic Approach to General Boolean Constraint Problems
(Hans-Werner Güsgen, Peter Ladkin)

96-05 Future University Computing Resources
(Peter Ladkin)

96-06 Lazy Cache Implements Complete Cache
(Peter Ladkin)

96-07 Formal but Lively Buffers in TLA+
(Peter Ladkin)

96-08 The X-31 and A320 Warsaw Crashes: Whodunnit?
(Peter Ladkin)

96-09 Reasons and Causes
(Peter Ladkin)

96-10 Comments on Confusing Conversation at Cali
(Dafydd Gibbon, Peter Ladkin)

96-11 On Needing Models
(Peter Ladkin)

96-12 Formalism Helps in Describing Accidents
(Peter Ladkin)

96-13 Explaining Failure with Tense Logic
(Peter Ladkin)

96-14 Some Dubious Theses in the Tense Logic of Accidents
(Peter Ladkin)

96-15 A Note on a Note on a Lemma of Ladkin
(Peter Ladkin)

96-16 News and Comment on the AeroPeru B757 Accident
(Peter Ladkin)

97-01 Analysing the Cali Accident With a WB-Graph
(Peter Ladkin)

97-02 Divide-and-Conquer Multiple Sequence Alignment
(Jens Stoye)

97-03 A System for the Content-Based Retrieval of Textual and Non-Textual Documents Based on Natural Language Queries
(Alois Knoll, Ingo Glöckner, Hermann Helbig, Sven Hartrumpf)

97-04 Rose: Generating Sequence Families
(Jens Stoye, Dirk Evers, Folker Meyer)

97-05 Fuzzy Quantifiers for Processing Natural Language Queries in Content-Based Multimedia Retrieval Systems
(Ingo Glöckner, Alois Knoll)

97-06 DFS – An Axiomatic Approach to Fuzzy Quantification
(Ingo Glöckner)

98-01 Kognitive Aspekte bei der Realisierung eines robusten Robotersystems für Konstruktionsaufgaben
(Alois Knoll, Bernd Hildebrandt)

98-02 A Declarative Approach to the Development of Dynamic Programming Algorithms, applied to RNA Folding
(Robert Giegerich)

98-03 Reducing the Space Requirement of Suffix Trees
(Stefan Kurtz)

99-01 Entscheidungskalküle
(Axel Saalbach, Christian Lange, Sascha Wendt, Mathias Katzer, Guillaume Dubois, Michael Höhl, Oliver Kuhn, Sven Wachsmuth, Gerhard Sagerer)

99-02 Transforming Conditional Rewrite Systems with Extra Variables into Unconditional Systems
(Enno Ohlebusch)

99-03 A Framework for Evaluating Approaches to Fuzzy Quantification
(Ingo Glöckner)

99-04 Towards Evaluation of Docking Hypotheses using elastic Matching
(Steffen Neumann, Stefan Posch, Gerhard Sagerer)

99-05 A Systematic Approach to Dynamic Programming in Bioinformatics. Part 1 and 2: Sequence Comparison and RNA Folding
(Robert Giegerich)

99-06 Autonomie für situierte Robotersysteme – Stand und Entwicklungslinien
(Alois Knoll)

2000-01 Advances in DFS Theory
(Ingo Glöckner)

2000-02 A Broad Class of DFS Models
(Ingo Glöckner)

2000-03 An Axiomatic Theory of Fuzzy Quantifiers in Natural Languages
(Ingo Glöckner)

2000-04 Affix Trees
(Jens Stoye)

2000-05 Computergestützte Auswertung von Spektren organischer Verbindungen
(Annika Büscher, Michaela Hohenner, Sascha Wendt, Markus Wiesecke, Frank Zöllner, Arne Wegener, Frank Bettenworth, Thorsten Twellmann, Jan Kleinlützum, Mathias Katzer, Sven Wachsmuth, Gerhard Sagerer)

2000-06 The Syntax and Semantics of a Language for Describing Complex Patterns in Biological Sequences
(Dirk Strothmann, Stefan Kurtz, Stefan Gräf, Gerhard Steger)

2000-07 Systematic Dynamic Programming in Bioinformatics (ISMB 2000 Tutorial Notes)
(Dirk J. Evers, Robert Giegerich)

2000-08 Difficulties when Aligning Structure Based RNAs with the Standard Edit Distance Method
(Christian Büschking)

2001-01 Standard Models of Fuzzy Quantification
(Ingo Glöckner)

2001-02 Causal System Analysis
(Peter B. Ladkin)

2001-03 A Rotamer Library for Protein-Protein Docking Using Energy Calculations and Statistics
(Kerstin Koch, Frank Zöllner, Gerhard Sagerer)

2001-04 Eine asynchrone Implementierung eines Microprozessors auf einem FPGA
(Marco Balke, Thomas Dettbarn, Robert Homann, Sebastian Jaenicke, Tim Köhler, Henning Mersch, Holger Weiss)

2001-05 Hierarchical Termination Revisited
(Enno Ohlebusch)

2002-01 Persistent Objects with O2DBI
(Jörn Clausen)

2002-02 Simulation von Phasenübergängen in Proteinmonoschichten
(Johanna Alichniewicz, Gabriele Holzschneider, Morris Michael, Ulf Schiller, Jan Stallkamp)

2002-03 Lecture Notes on Algebraic Dynamic Programming 2002
(Robert Giegerich)

2002-04 Side chain flexibility for 1:n protein-protein docking
(Kerstin Koch, Steffen Neumann, Frank Zöllner, Gerhard Sagerer)

2002-05 ElMaR: A Protein Docking System using Flexibility Information
(Frank Zöllner, Steffen Neumann, Kerstin Koch, Franz Kummert, Gerhard Sagerer)

2002-06 Calculating Residue Flexibility Information from Statistics and Energy based Prediction
(Frank Zöllner, Steffen Neumann, Kerstin Koch, Franz Kummert, Gerhard Sagerer)

2002-07 Fundamentals of Fuzzy Quantification: Plausible Models, Constructive Principles, and Efficient Implementation
(Ingo Glöckner)

2002-08 Branching of Fuzzy Quantifiers and Multiple Variable Binding: An Extension of DFS Theory
(Ingo Glöckner)

2003-01 On the Similarity of Sets of Permutations and its Applications to Genome Comparison
(Anne Bergeron, Jens Stoye)

2003-02 SNP and mutation discovery using base-specific cleavage and MALDI-TOF mass spectrometry
(Sebastian Böcker)

2003-03 From RNA Folding to Thermodynamic Matching, including Pseudoknots
(Robert Giegerich, Jens Reeder)

2003-04 Sequencing from compomers: Using mass spectrometry for DNA de-novo sequencing of 200+ nt
(Sebastian Böcker)

2003-05 Systematic Investigation of Jumping Alignments
(Constantin Bannert)

2003-06 Suffix Tree Construction and Storage with Limited Main Memory
(Klaus-Bernd Schürmann, Jens Stoye)

2003-07 Sequencing from compomers in thepresence of false negative peaks
(Sebastian Böcker)

2003-08 Genalyzer: An Interactive Visualisation Tool for Large-Scale Sequence Matching – Biological Applications and User Manual
(Jomuna V. Choudhuri, Chris Schleiermacher)

2004-01 Sequencing From Compomers is NP-hard
(Sebastian Böcker)

2004-02 The Money Changing Problem revisited: Computing the Frobenius number in time

$O(k\,a_1)$

(Sebastian Böcker, Zsuzsanna Lipták)

2004-03 Accelerating the Evaluation of Profile HMMs by Pruning Techniques
(Thomas Plötz, Gernot A. Fink)

2004-04 Optimal Group Testing Strategies with Interval Queries and Their Application to Splice Site Detection
(Ferdinando Cicalese, Peter Damaschke, Ugo Vaccaro)

2004-05 Compressed Representation of Sequences and Full-Text Indexes
(Paolo Ferragina, Giovanni Manzini, Veli Mäkinen, Gonzalo Navarro)

2005-01 Overlaps Help: Improved Bounds for Group Testing with Interval Queries
(Ferdinando Cicalese, Peter Damaschke, Libertad Tansini, Sören Werth)

2005-02 Two batch Fault-tolerant search with error cost constraints: An application to learning
(Ferdinando Cicalese)

2005-03 Searching for the Shortest Common Supersequence
(Sergio A. de Carvalho Jr., Sven Rahmann)

2005-04 Counting Suffix Arrays and Strings
(Klaus-Bernd Schürmann, Jens Stoye)

2005-05 Alignment of Tandem Repeats with Excision, Duplication, Substitution and Indels (EDSI)
(Michael Sammeth, Jens Stoye)

2005-06 Statistics of Cleavage Fragments in Random Weighted Strings
(Hans-Michael Kaltenbach, Henner Sudek, Sebastian Böcker, Sven Rahmann)

2006-01 Decomposing metabolomic isotope patterns
(Sebastian Böcker, Zsuzsanna Lipták, Anton Pervukhin)

2006-02 On Common Intervals with Errors
(Cedric Chauve, Yoan Diekmann, Steffen Heber, Julia Mixtacki, Sven Rahmann, Jens Stoye)

2007-01 Identifying metabolites with integer decomposition techniques, using only their mass spectrometric isotope patterns
(Sebastian Böcker, Matthias C. Letzel, Zsuzsanna Lipták, Anton Pervukhin)