Efficient Calculation of Empirical P-values for Association Testing of Binary Classifications

Konstantinos Zagganas
Univ. of the Peloponnese & IMSI, “Athena” RC
kzagganas@uop.gr
zagganas@athenarc.gr

Spiros Skiadopoulos
Univ. of the Peloponnese
spiros@uop.gr

Thanasis Vergoulis
IMSI, “Athena” RC
vergoulis@athenarc.gr

Theodore Dalamagas
IMSI, “Athena” RC
dalamag@athenarc.gr

ABSTRACT

Investigating whether two different classifications of a population are associated, is an interesting problem in many scientific fields. For this reason, various statistical tests to reveal this type of associations have been developed, with the most popular of them being Fisher’s exact test. However it has lately been shown that in some cases this test fails to produce accurate results. An alternative approach, known as randomization tests, was introduced to alleviate this issue, however, such tests are computationally intensive. In this paper, we introduce two novel indexing approaches that exploit frequently occurring patterns in classifications to avoid performing redundant computations during the analysis. We conduct a comprehensive set of experiments using real datasets and application scenarios to show that our approaches always outperform the state-of-the-art, with one approach being faster by an order of magnitude.

CSC CONCEPTS

• Applied computing → Bioinformatics; • General and reference → Performance.

KEYWORDS

Association testing, binary classifications, randomization tests, data structures, bitsets, indices, filtering

ACM Reference Format:
Konstantinos Zagganas, Thanasis Vergoulis, Spiros Skiadopoulos, and Theodore Dalamagas. 2020. Efficient Calculation of Empirical P-values for Association Testing of Binary Classifications. In 32nd International Conference on Scientific and Statistical Database Management (SSDBM 2020), July 7–9, 2020, Vienna, Austria. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3400903.3400923

1 INTRODUCTION

A common type of analysis performed by scientists in various disciplines, aims to reveal whether two binary classifications of a population (which divide it in two mutually exclusive classes: the positive and the negative one) are associated with each other. For instance, in medicine it is crucial to investigate if a biological gender-based classification of humans can be associated with their risk to develop a particular disease; in mechanical engineering, it is interesting to reveal if a particular type of vehicle is more prone to engine failures. Similar examples can be easily found in many other scientific fields.

Using this as motivation, several statistical tests have been developed, that measure the association between two binary classifications, usually producing a measure that quantifies the strength of the association between the classifications (called p-value). In fact, often, the objective is to find whether a set of “query” classifications (of arbitrary size) is associated with one or more classifications in a fixed set of “ground” classifications. Essentially, this translates into a series of association tests that need to be performed. The most widely-used association test is Fisher’s exact test [11]. When examining a query and a ground classification, the test takes into consideration the number of items that belong in the positive class of both classifications (i.e., their overlap) to decide whether they are associated or not. One of the assumptions made by this test is that the expected overlap between two independent classifications follows the hypergeometric distribution. However, it has been recently shown that in various cases, this is not a valid assumption [7]. In these cases, applying Fisher’s exact test may result in producing erroneous findings.

In these cases, scientists prefer to utilize randomization tests, which exploit a very large number of randomly generated query classifications in an attempt to estimate the real distribution of the expected overlap. The side-effect is that these tests require a large number of computations to be performed, resulting in significantly larger execution times. Since the performance of the association test is very important for some applications, methods to accelerate randomization tests attracted interest recently [22].

In this work, we introduce novel, indexing-based approaches that exploit frequently occurring patterns in the classifications of interest, in the span of a series of randomization tests to significantly accelerate their execution. More specifically:

• We introduce two novel indices to facilitate the efficient execution of randomization tests, the Frequent Itemset Index (FII) and the Significance Level Index (SLI). The former captures all overlaps that exist between the ground classifications, while the latter captures the minimum overlap that
a query classification should have to be a candidate for significant association with each of the ground classifications.

- We introduce a novel approach that exploits the FII index to avoid redundant computations occurring due to the overlaps that exist between the ground classifications.
- We also introduce a second approach that combines both indices (FII and SLI) to create an approach that can be used to eliminate statistically insignificant associations and vastly reduce the number of computations required.
- We conduct comprehensive experiments showing that our approaches introduce significant speedup; more specifically, the approach combining both indices outperforms the state-of-the-art by an order of magnitude (see Section 4).
- We provide open-source implementations\(^1\) of all described approaches.

2 BACKGROUND

2.1 Association testing for binary classifications

The binary classification of a given population is the result of classifying its items in two classes which are mutually exclusive. Usually, we refer to the one class as the “positive” and to the other as the “negative”. For instance, a possible binary classification for the items in a given bacteria population could be based on their pathogenicity for humans; based on this classification, there are two classes of bacteria, the pathogenic (positive class) and the non-pathogenic ones (negative class). Knowing all the items in a population, it is possible to use the set of items labeled as positive by a binary classification to represent it as a whole. In the remainder we adopt this convention and we use capital letters (e.g., A, B) to denote these item sets/classifications.

Given a population of items, investigating whether two different binary classifications are associated with each other, is a problem of great interest in many scientific applications (see also Section 1). Many statistical approaches that examine the significance of the association between two binary classifications based on a gathered sample from the population have been proposed in the literature (e.g., chi-squared tests [2], the Cochran–Mantel–Haenszel test [1], etc.). In the remainder of this paper, we refer to them as association testing methods.

The most popular of them, Fisher’s exact test [11], examines the number of items that belong to the positive class of both classifications (i.e., their overlap) to decide whether these classifications are associated or not. In this context, it assumes that the expected overlap between two independent classifications follows the hypergeometric distribution. Then, based on this assumption, the probability that an observed overlap between two binary classifications could have been observed by pure chance if they were independent (null hypothesis), can be used to calculate an indicator (p-value) that can help to decide if the two classifications are associated or not.

Often, during investigating associations of different classifications, we have a fixed set of \(k\) classifications of interest (let them be the ground classifications, denoted as \(B_1, \ldots, B_k\)) that we want to examine their association with a (maybe infinite) “family” \(A\) of related classifications (let them be the query classifications, denoted as \(A_1, A_2, \ldots \in A\)). The members of the family (i.e., the query classifications) usually share a similar mechanism that classifies objects of the population. For example, one possible binary classification for genes could be based on whether they are targeted (blocked) or not, by a particular set of biomolecules called microRNAs [7]. By selecting different sets of microRNAs we can determine different query classifications of this type. The classification mechanism behind them is similar to an extent (e.g., in regard to the principles of how microRNA groups target particular genes). Finally, examining the association of members of this family with the ground classifications of interest (e.g., genes being involved in particular biological processes or not) is of great interest and can be done using the aforementioned association testing methods.

2.2 Randomization tests

Although Fisher’s exact test is very widely used and has been very helpful in a wide range of applications, it has been shown that sometimes the expected overlap between the two classifications does not follow the hypergeometric distribution making the test unsuitable. This could be relevant to the fact that the one of the classifications under investigation belongs to a classification family (see also Section 2.1) something that modifies, among others, the way population items are being classified. For example, in [7] the authors, study this effect in microRNA functional enrichment analysis, which is used to indicate whether a group of biomolecules (microRNAs) can affect specific biological processes. The authors used a ground truth (formed based on laboratory experiments) to show that using Fisher’s exact test in this context could result in reporting known, strong associations as weak or the opposite.

Problems like this one provided the motivation for the introduction of randomization tests. These are statistical methods that, in the context described here, instead of assuming that the expected overlap of a random query classification \(A \in A\) with another, independent ground classification \(B\) follows the hypergeometric distribution, they estimate an empirical distribution based on calculating the exact overlap that \(n\) randomly selected query classifications \(A_1, \ldots, A_n \in A\) have with \(B\), for very large numbers of \(n\). According to that, when the association between two classifications \(A \in A\) and \(B\) is being investigated, an empirical p-value is calculated based on the proportion of random classifications \((A_1, \ldots, A_n)\) that present a larger overlap with \(B\) than the calculated overlap between \(A\) and \(B\). It should be noted that each randomization set provides its own definition of what consists an overlap. For example, in [7], the one-sided overlap between two classifications (let them be \(A\) and \(B\)) is used, that is defined as follows:

\[
\text{overlap}(A, B) = \frac{\text{sizeof}(A \cap B)}{\text{sizeof}(A)}
\]

For the remainder of this paper, we assume that the same definition of overlap is used. Under this assumption, based on the previous discussion, the empirical p-value could be formalised as follows:

\[
p = \frac{\text{sizeof}((A_j : \text{overlap}(A_j, B) \geq \text{overlap}(A, B)))}{n}
\]

Figure 1 illustrates the process of a such randomisation test.
2.3 Performance issues of randomization tests
Randomization test tend to be intensive, in regard to the computational resources required (CPU, RAM, execution time), especially when a very large $n$ is selected (e.g., $n \geq 1M$). Selecting values of such magnitude for $n$ is very common since, the larger the $n$ is, the higher the accuracy of the produced empirical $p$-value will be. As a result, randomization tests tend to have significant execution times.

To make matters worse, as already mentioned (see Section 2.1), in practice, researchers are interested to examine the association of a query classification $A \in A$ with a large set of different ground classifications $B_1, \ldots, B_k$ (with $k$ being a integer significantly larger than 1). It is evident that, this results in even larger execution times. Consequently, methods that could improve the performance of randomization tests received attention recently [22].

3 EFFICIENT CALCULATION OF EMPIRICAL P-VALUES

In this section we present two novel approaches for the efficient calculation of empirical $p$-values based on randomization tests. The first one (described in Section 3.1) exploits the fact that several sets of items (itemsets) appear in many of the ground classifications $B_1, \ldots, B_k$, resulting in redundant computations during the randomization test. The approach avoids these redundant computations using an index structure that identifies the overlaps of the ground classifications.

The second approach (described in Section 3.2) is just an extension of the first one that allows an even larger acceleration based on a second index. This index captures the minimum overlap a query classification should have with each of the ground classifications in order for the association between them to present as statistically significant.

3.1 The Frequent Itemset Index (FII) Approach

3.1.1 Basic approach. Calculating the (one-sided) overlap between two classifications is a core task performed multiple times during a randomization test (see Section 2.3). This task is based on applying the intersection operation on the corresponding sets. As a result, accelerating the intersection operations involved is expected to achieve significant speedups in the performance of randomization tests.

Currently, the best state-of-the-art approach for this is the one described in [22]. In brief, the positive items of each random query classification $A_j, j = 1, \ldots, n$ are kept as set bits in a bitset, that represents each element in the population with a bit; the positive items of each ground classification $B_i$ are kept in the form of lists of gene IDs (each gene ID is an integer that corresponds to the respective position of the gene in the bitsets). During the randomization test, for each ground classification $B_i$, its gene IDs are used to examine if the corresponding bit in the bitset of each of the query classifications $A_j$ is set (an operation called bit-probing). Based on that, an overlap counter that allows the calculation of the corresponding overlap is being updated.

However, it can be shown that the ground classifications contain overlapping items. In fact, some itemsets are very frequent, appearing in many ground classifications $B_i, i = 1, \ldots, k$. This means that during the execution of randomization tests, a large number of redundant bit-probing operations are taking place. To alleviate this issue, we could identify those frequent itemsets, compute their overlap with each of the query classifications $A_j$ beforehand, and store the results in a proper, easily accessible structure with counters. Then, each time a redundant calculation is about to happen (when the overlap of a query classification $A_j$ with a ground classification $B_i$ that contains a frequent itemset is required), instead of performing the calculation, a less expensive combination of an index probe...
and a subsequent addition to the corresponding counter will take place.

It should be noted that, in general, the FII approach comprises frequent itemsets of any size. However, in some cases where the fast creation of the FII index is crucial, a limited version of the FII approach that is based only on singular frequent itemsets (i.e., itemsets of size 1) can be used.

3.1.2 The index. Based on the previous discussion, we introduce the Frequent Itemset Index (FII). This index is designed to store all frequent itemsets among the ground classifications $B_i$ along with the counters that store the size of their overlap with each of the random query classifications $A_j$. Figure 3 illustrates this index. It consists of the following parts:

- **Inverted Index.** The inverted index containing frequent itemset definitions, as well as relations between the frequent itemsets and the ground classifications $B_i$ (i.e., which frequent itemsets appear in each ground classification). This part of the index can be calculated only once for each dataset and can be saved to disk to be used in subsequent tests involving it.

- **Array of counters.** A hybrid two-dimensional array of counters that contain the size of the intersection between all frequent itemsets and all random query classifications $A_j$. This array is hybrid in the sense that it contains rows both of char and integer type. In cases where the size of the intersection is potentially smaller than 255 we use a character row or else we use an integer row. This way, we can reduce the memory footprint of the array by using a smaller data type, in terms of memory space, where appropriate. Finally, it is created on-the-fly, since it depends on the query classifications $A_j$, which are computed during the analysis.

To create the index, we need to process the itemsets of all ground classifications $B_i$ to identify all their parts that occur frequently (i.e., in more than one $B_i$). Then, we need to transform them all so that they are expressed as sets containing both regular items and frequent itemsets (those identified from the preprocessing step). The latter task is not trivial since, often, each $B_i$ can be expressed in many different ways, each using different combinations of (singular or longer) frequent itemsets. Long frequent itemsets are, in general, preferable since they will replace a large number of redundant computations with only one index probe per element and one addition. The FII creation process should attempt to utilise characteristics like these to achieve better performance. In the following sections we elaborate on relevant implementation details.

3.1.3 Frequent Itemsets Identification. Regarding the first step (frequent itemsets identification), executing the Apriori algorithm [4] (with threshold $\text{sup}_\text{thr} = 2$) on the itemsets of all ground classifications $B_i$ can be used to perform this task. However, using Apriori with such a low support threshold is very computationally intensive. In particular, using Christian Borgelt’s implementation [8], we tried to produce the maximal itemsets with support threshold $\geq 2$ for the datasets in Section 4.1, which contain 15K-25K classifications, which we used as input transactions to the algorithm. However, after one hour into the execution of the program, we were forced to kill it, because it was using more than 100 GB of RAM.

To alleviate this issue, we introduce an alternative approach: let $F$ be a frequent itemset that would have been produced by executing Apriori with $\text{sup}_\text{thr} = 2$. Essentially, $F$ would fall under one of the following cases:

- **$F$ is subset of two ground classifications ($F \subseteq B_1 \& F \subseteq B_2$).** In this case, $F = B_1 \cap B_2$ or $F \subset B_1 \cap B_2$. Thus, any frequent itemset in this case will be dominated by $B_1 \cap B_2$ because our approach requires the largest of the itemsets possible. Thus, using only $B_1 \cap B_2$ for the classification transformation (see Section 3.1.4) of both $B_1$ and $B_2$ is an adequate solution for our approach.

- **$F$ is subset of more than two ground classifications.** Consider that $F$ appears in 3 classifications $B_1$, $B_2$, $B_3$ (but what is said here can be easily generalized for larger values). It holds that $B_1 \cap B_2 \cap B_3$ would always be smaller than any of $B_1 \cap B_2$, $B_1 \cap B_3$, and $B_2 \cap B_3$ (see also Venn diagrams in Figure 2). Thus, using $B_1 \cap B_2$, $B_1 \cap B_3$, and $B_2 \cap B_3$ for the classification transformation of both $B_1$, $B_2$, and $B_3$ is also an adequate solution for our approach.

It follows then, that we only need to calculate the intersections between all ground classification pairs and this eliminates a large number of intersection operations, making the complexity of this algorithm $O(k^3)$. Hence, for each ground classification $B_i$ we need to consider only the largest itemsets produced from its intersection with all of the other ground classifications. This results to a significantly reduced number of operations and, consequently, to better execution times from the creation of the FII.

3.1.4 Ground Classification Transformation. After we have procured all frequent itemsets, we need to decide which frequent itemsets better “cover” each ground classification $B_i$. Based on the decision made, each $B_i$ will be transformed to an equivalent set $B_i'$ that will contain both single items and some of the identified frequent itemsets.

As mentioned, ideally, for each ground classification $B_i$, large frequent itemsets should be selected to be included in $B_i'$, to minimize the number of addition operations performed. This is relevant
to the set cover problem, which is a very difficult problem, being one of Karp’s 21 NP-complete problems [14]. However, a greedy approach that can reveal a sub-optimal solution to our problem for each ground classification \( B_i \), is the following:

1. Sort all frequent itemsets by descending size in a list \( (\mathcal{F}) \).
2. Add the largest itemset \( F_{\text{max}} \), for which \( F_{\text{max}} \subseteq B_i \), to \( B'_i \) (initially empty).
3. For the rest of the itemsets \( F_p \in \mathcal{F} \), if \( F_p \subseteq B_i \) and none of its items is contained in any of the current itemsets in \( B'_i \), then \( F_p \) is also added to \( B'_i \).
4. Finally add all singular frequent itemsets of \( B_i \) that are not included in any of the current itemsets of \( B'_i \) to \( B'_i \).

It is worth noting here, that by following this approach, if a frequent itemset is dominant (i.e., it is frequent and none of its supersets are frequent) and it is selected as a cover, then all of its subsets (which are also frequent) will be discarded, since their elements have already been selected. Thus, we only need to consider dominant frequent itemsets as potential covers.

3.1.5 Extra implementation details. We have implemented FII in such a way that it takes advantage of Single Instruction Multiple Data (SIMD) CPU instructions during counter additions. SIMD instructions can perform simultaneous mathematical operations on memory positions that exist alongside each other (i.e. array) inside a memory word. For example, on an 128-bit register, 16 elements of an array of type char or 4 elements of type int can be added simultaneously to another char or int array respectively. This approach resulted in improved performance for the FII.

Another interesting implementation detail is the following. The number of frequent itemsets to be used by the FII index can often become very large resulting in a very large array of counters. In cases where it is essential to reduce the memory footprint of the program, one option is to use \( \text{sup}_\text{thr} > 2 \). In order to do that, we first calculate the support for each of the itemsets produced by our approach for \( \text{sup}_\text{thr} = 2 \); then we discard all itemsets that have a support \( < \text{sup}_\text{thr} \) and use the rest. This means that the greater the value of \( \text{sup}_\text{thr} \) is, the smaller the memory footprint will be. Of course, as a side-effect, the performance is expected to degrade. In the experimental section, we investigate the effect of greater \( \text{sup}_\text{thr} \) values both to the memory footprint and to the execution time of the FII approach (see Section 4.3).

3.1.6 A toy example. The following example outlines how the FII is created and used to calculate the intersection size:

**Example 3.1.** Let \( B_1 \) a ground classification containing items 1, 5 and 6, \( B_2 \) another ground classification containing items 1, 5 and 7 and finally, \( B_3 \) containing items 3, 4 and 7. Also, let \( A_1 \) be a query classification represented as a bitset, containing items 1, 2, 5, 8 and 10 by having the appropriate bits set to 1. The \( \text{apriori} \) algorithm with \( B_1 \), \( B_2 \) and \( B_3 \) as transactions and support threshold 2 produces two frequent itemsets: \( \text{Freq}_1 \) containing elements 1 and 5 and \( \text{Freq}_2 \) containing item 7. The contents of \( \text{Freq}_1 \) and \( \text{Freq}_2 \) are removed from \( B_1 \), \( B_2 \) and \( B_3 \). Furthermore relations, which are stored in an inverted index, are also created: \( \text{Relations}[B_1] = \{ \text{Freq}_1 \}, \text{Relations}[B_2] = \{ \text{Freq}_1, \text{Freq}_2 \}, \text{Relations}[B_3] = \{ \text{Freq}_2 \} \). Moreover, two counters which contain the size of the intersection between \( A_1 \) and \( \text{Freq}_1 \) as well as the intersection between \( A_1 \) and \( \text{Freq}_2 \) respectively are created, by probing \( A_1 \) at the appropriate positions for 1, 5 and 7.

To calculate the size of the intersection between \( A_1 \) and \( B_1 \), we probe \( A_1 \) in position 6 and add the counter for \( \text{Freq}_1 \) to the intersection size. However, for \( B_2 \) we only need to add the counters for \( \text{Freq}_1 \) and \( \text{Freq}_2 \) and finally, for \( B_3 \) we add the counter for \( \text{Freq}_2 \) and probe \( A_1 \) two times in positions 3 and 4. Finally, the number of probes we perform without the FII is 9, while we probe \( A_1 \) only 6 times in total by using the index.

Figure 4 depicts the index described above in regard to the previous example.

---

**Figure 3: Structure of the FII**

| Classifications | Frequent itemsets | Counter array |
|-----------------|-------------------|--------------|
| \( B_1 \)       | \( F_1 \)         | char         |
| \( B_2 \)       | \( F_2 \)         | char         |
| \( B_k \)       | \( F_m \)         | int          |

Arrays with \( n \) elements

\[ \text{char*} \text{ char*} \text{ char*} \text{ char*} \text{ int*} \]
3.2 The Significance Level Index (SLI) Approach

3.2.1 Basic approach. During an association test, all associations having a p-value equal or smaller than a predefined threshold (usually 0.05) are considered to be strong and are those to be reported. Consider, for a while, that we are interested in performing a randomization test to examine if a given query classification $A \in \mathcal{A}$ is significantly associated with a particular ground classification $B$. In the context of the randomization tests described in Section 2.2, an empirical p-value smaller or equal to 0.05 essentially means that the overlap of the query classification $A$ with the ground classification $B$ is so large that it will be among the top $5\%$ of overlaps observed for all random query classifications $A_i$.

It is evident that the lowest overlap in the top $5\%$ of overlaps between ground classification $B$ and all query classifications $A_i$ can be used to define a threshold for the lowest possible overlap that $A$ should have in order to be significantly associated with $B$. For the remainder of the manuscript we will refer to this threshold as the overlap threshold ($ov\_thr$). The intuition behind the Significant Level Index (SLI) approach is to build an index that keeps these overlap thresholds for all ground classifications of interest $B_i$. Then, by simply calculating the overlap of query classification $A$ with ground classification $B_i$ and comparing it with the overlap threshold we will be able to know if the p-value of $A$ will be adequate to characterize the association of $A$ with $B_i$ as statistically significant.

The only problem with the previous approach is that each execution of the randomization test needs a new, on-the-fly created set of random query classifications $A_1, \ldots, A_n \in \mathcal{A}$. This means that the set of random query classifications to be used during the analysis cannot be known beforehand. Instead of that, during a pre-processing phase, we can create a similar set of random query classifications $A'_1, \ldots, A'_n \in \mathcal{A}$ and then create the thresholds for all ground classifications of interest $B$, based on them. This means that the calculated overlap thresholds could be slightly different than the exact thresholds that will be calculated for the final randomization tests. However, by definition, the randomized test assumes that these sets simulate the actual empirical overlap distribution of the data and, thus, we expect that the distribution will be similar between two runs of the same experiment (given a large enough number of random sets). This means that the SLI can be saved on disk and used for multiple subsequent tests.

To alleviate this issue, we introduce a filtering approach: We first calculate a slightly looser threshold for each ground classification of interest $B$ (based on the top $x\%$ of classifications, where $x > 5\%$) and include this threshold in the SLI. Then, for each ground classification of interest $B_i$, we calculate the overlap of query classification $A$ with it and compare it with the corresponding overlap in the SLI. For all ground classifications $B_i$ for which the overlap of $A$ was found to satisfy the threshold we perform the full randomization test, since these pairs correspond to candidate significant associations. In fact, using SLI we can significantly filter out ground classifications $B_i$ that do not have any chance to provide significant results beforehand, resulting in significantly increased performance. More details about this process could be found in Section 3.2.4.

3.2.2 The Index. The SLI comprises a list of float numbers, one for each of ground classification of interest $B$. Each float number represents the one-sided overlap significance level (i.e., overlap threshold) that has been produced using a large number of random query classifications $A'_i$. The list is then saved in a file, which can be used for multiple subsequent tests. Figure 5 demonstrates an example of the use of the SLI index.

Example 3.2. Given a query classification $A$ and ground classifications $B_1, B_2, B_3$ and $B_4$, we first calculate the overlap of $A$ with each of the $B_i$. Then we compare these overlap values with the significance overlap values in the SLI for each of the ground classifications $B_i$ and decide which of them are potentially significant.
associations \((B_2, B_4)\). The rest \((B_1\) and \(B_3)\) are marked as insignificant and no \(p\)-values for them are produced. Then for \(B_2\) and \(B_4\) we run the full randomization test and produce the respective \(p\)-values.

In order to create the SLI, we use the approach described in Section 3.1 (FII) to calculate the overlaps between all query classifications \(A_j\) and all ground classifications \(B_i\). Then, for each \(B_i\) the list of overlaps is sorted and the overlap significance threshold is extracted based on the top \(x \times 100\%\) of overlaps (for a discussion on the proper selection of \(x\), see Section 3.2.3), where \(x\) is the \(p\)-value threshold of the SLI. Finally, the identifier of the ground classification \(B_i\) along with the one-sided overlap threshold is saved in a file.

3.2.3 SLI significance threshold. One possible issue with using the SLI is that, unless the \(p\)-value significance threshold \(x\) is set to a high enough value, the SLI might mark actually significant associations between a query classification \(A\) and ground classifications \(B_i\) as insignificant and discard them (false negatives). For example, if \(x = 0.05\) then the method is potentially approximate since the randomization test is an approximation process and if the set of all query classifications \(A\) that were used for the experiment, changes even slightly, then the SLI might produce false negatives.

On the other hand, if \(x\) is set to too high a value, then we lose some of the filtering power of the SLI because many insignificant associations can be marked as potentially significant (false positives). This will lead to calculation of empirical \(p\)-values for them, increasing the total execution time of the approach.

Given the fact that the one-sided overlap at the \(p\)-value significance threshold \(x\) is calculated using the same randomization test, it is easy to see that each time we run the experiment, the \(ov_i/hr\) at \(x\) is expected to change, since the set of query classifications \(A_j\) changes. More specifically, let \(N\) be the total number of all possible randomized query classifications \(A_j\) and \(n\) the number of query classifications that have been selected for the randomization experiment, while \(K\) is the number of query classifications that were not picked for the experiment. Then in order for \(x\) to have a deviation of \(dev\) between re-runs of the same experiment, a different set of query classifications must be selected for the re-run, namely \(k = ((dev - x) \times n)\) different query classifications. Then, given a deviation \(dev\), we could use the hypergeometric distribution to find the probability that \(k\) different query classifications could be selected when the experiment is repeated. However, as we mentioned earlier (see the Background section), the intersection sizes and consequently the one-sided overlaps do not follow the hypergeometric distribution.

Thus, we propose an empirical approach to set the \(p\)-value significance threshold based on the observed data. More specifically, we can repeat a randomization test a number of times with the same input. Then, for the \(p\)-values in the output of the multiple repetitions we calculate the maximum standard deviation from the mean. After that, we can use different inputs, repeat the experiments a number of times and calculate the total maximum standard deviation. Finally, we can arbitrarily set the significance threshold to a value that is larger than the maximum standard deviation in order to guarantee that the SLI will not mark significant associations as insignificant. The effectiveness of this method is evaluated in Section 4.4.

3.2.4 Calculating \(p\)-values using the SLI. In order to calculate \(p\)-values using the SLI, we first calculate the overlap of query classification \(A\) with each of ground classifications \(B_i\). Then, we use the SLI to compare these overlaps with the respective overlap thresholds for all \(B_i\). If an overlap is above the significance overlap threshold, we mark the association between query classification \(A\) and the respective ground classification \(B_i\) as potentially significant. In the case that the association is marked as insignificant, we also print the \(p\)-value that corresponds to the overlap threshold.

After we have collected all potentially significant associations, we use the FII version of our approach to calculate empirical \(p\)-values. However, in this case, the index consists only of singular itemsets with support \(\geq 2\) and it is created on-the-fly. The reason we do not re-use the FII that already exists from the creation of the SLI is that, since a lot of associations between \(A\) with ground classifications \(B_i\) have been eliminated from the analysis by the use of the SLI, a lot of itemsets that were frequent before (with support \(\geq 2\)) are not frequent any more. Moreover, since the collection of potentially significant associations changes based on the input query classification \(A\) and is calculated at run-time, we must also find frequent itemsets on-the-fly. Since the execution of the Apriori algorithm (or our approach) is computationally expensive, the speedup is not expected to overcome the overhead of the index creation. On the other hand, it is easy and fast to discover frequent singular itemsets (support \(\geq 2\)) among the collection of ground classifications potentially significantly associated with \(A\) using hash tables in \(O(n)\) time where \(n\) is the total number of significant associations. This FII method is then used as before to eliminate duplicate probes and calculate \(p\)-values.

4 EVALUATION
In this section we evaluate the performance of our method against competitor methods using a real randomization experiment as use case. In particular, we use the case of microRNA functional enrichment analysis [7, 22], where we are interested to investigate the association between genes targeted by a particular group of microRNAs (query classification \(A\)) and genes involved in a particular biological process or diseases (ground classification \(B\)). We have selected this scenario for the experiments since (a) this is a known example where Fisher’s exact test has been shown to be inadequate [7] and (b) from a previous work, we know there are relevant open datasets that we could utilise. However, our approach can be used in other domains that use the overlap-based randomization, with very minor changes to the code.

All of the experiments were performed on a single CPU core on a server with a Xeon E7- 4830 CPU and 256GB of RAM.

4.1 Datasets
In experiments, we used three openly accessible life-sciences datasets as ground classifications \(B_i\):
- Gene Ontology (GO) [6, 20]. This dataset contains three structured controlled vocabularies (ontologies) that categorize genes according to their function. Each gene can belong to multiple categories. The dataset used was retrieved from the Ensembl Biomart [10] for version 84.
4.2 Performance of addition operations vs. bit-probes

In this section, we describe the experiment we conducted to compare the performance of bit-probes vs the one of addition operations. This experiment is designed to show that the latter are more efficient in the context of the FII approach. We used bitsets 25,000 bits long, since the universe of human genes has a size of about 25,000. Also the bitsets we are using, are of three different densities: sparse (100 bits set), medium (10,000 bits set) and dense (20,000 bits set). We used these bitsets to calculate the size of the intersection with bit-probing. We performed 10,000 probes for 10,000 different bitsets in each setting and calculated the average amount of time required.

On the other hand, we created arrays of numbers which have a length of 10,000. We designed the following two versions: one array of characters, that can store numbers from 0 to 255 and one for an array of integers, which can store numbers from 0 to \(2^{32} - 1\), since the FII uses both of these data types. Moreover, we created 10,000 arrays for each data type and measured the total times required for addition operations. It should be noted here, that we enabled SIMD instructions for the addition operations, since they are also used by the FII. Each experiment was repeated 100 times and the average execution times per operation are shown in Table 1. It is easy to see that in all cases, array additions are faster than bit-probing operations.

Furthermore, regarding the three datasets we are using for the evaluation of our approach, we found that the vast majority of itemsets produced by our method contain less than 255 elements. It is clear that the addition operations performed are mostly of char type and this means that even if bitsets are sparse, addition operations still are an order of magnitude faster than bit-probing. Thus, we expect a large speedup when we use the FII for all three of the datasets compared to the state-of-the-art (see Section 4.5).

4.3 Performance & memory footprint of FII varying the itemset support threshold

In this section, we calculate the time required for the execution of the analysis, as well as the memory footprint required for the counters of the FII for variable support thresholds in regard to our approach in Sections 3.1.4 and 3.1.5. More specifically, we set the support threshold to 2, 6, 10, 14 and 16 and selected 10 different inputs of 39 miRNAs (query classification \(A_j\)) as well as 10 different sets of 1,000,000 miRNA groups (query classifications \(A_{j+}\)) and calculated the average execution time for each support threshold. We have also added a horizontal line demonstrating the performance of BUFET, indicatively (full comparison experiments are presented in Section 4.5). The results can be seen in Figure 6.

---

**Figure 5: Example of the use of the SLI**

**Table 1: Average time required for addition operations and bit-probing operations on different data types and variable densities.**
We can see that the performance of the index starts to decline as the support threshold increases, which is to be expected, since more operations that are slower (bit-probes) are performed. As we increase the support threshold even more, our approach should present with slightly higher execution times than the state-of-the-art. The reason for this is that our approach (FII) adds an overhead, like the allocation of memory, which will not be balanced by the speedup after a point in regard to support threshold.

It is noteworthy, however, that the size of the FII counters almost doubles as the support threshold increases from 2 to 6 for GO and 2 to 10 for the other datasets, before it starts dwindling. This can be attributed to the fact that as the support threshold increases, large itemsets with support = 2 are discarded. The reason why larger itemsets have a support of 2 is that usually, the permutations of elements in such itemsets are not found in more than 2 or 3 ground classifications in each dataset. This means that their support generally tends to be low. Instead, a large number of smaller itemsets with generally larger support are used to "cover" the ground classifications $B_i$ and this leads to a significant increase in memory footprint since the size of the footprint depends on the number of frequent itemsets.

Finally, it is also easy to observe that large support thresholds have a greater negative impact on the execution times in case of the GO dataset when compared to DGN or MeSH. This can be attributed to the fact that 90% of the itemsets, produced by our method, in GO had a support threshold $\leq$ 10, compared to 45% for DGN and 50% for MeSH. This is further corroborated by the fact that the size of the index (which depends on the number of itemsets) has a more significant decrease as the support threshold increases. It is evident that the larger the number of frequent itemsets, the better the performance of the FII is and since GO has fewer itemsets for large support thresholds its performance degrades faster as the support threshold increases compared to DGN and MeSH.

### 4.4 Setting the SLI significance threshold and evaluating the filtering effectiveness

In this experiment we use the method described in Section 3.2.3 to set the SLI significance threshold $x$. Each experiment was repeated 10 times with different random query classifications $A_1, \ldots, A_n \in \mathbb{A}$ and the outputs for each input were compared with each other. For each dataset, the maximum standard deviation of all p-values can be seen in Table 2.

![Histograms showing index size and execution time for GO, DGN, and MeSH datasets.](image)

**Figure 6: Index size and execution time (logscale) vs the itemset support threshold**

We can see that the standard deviation for all datasets is an order of magnitude smaller than 0.05. Thus if we arbitrarily set the p-value significance threshold to 0.075 (50% greater than 0.05) we expect that the SLI will produce no false negatives. It also means that the results of the randomization test (p-values) are not changing significantly between multiple runs of the same experiment.

Furthermore, we used the same experimental setting with the SLI index (20 inputs, 10 repetitions). Based on the outputs of the previous and the current experiment, for each input query classification, we calculated the number of actually significant p-values that were marked by the SLI as potentially significant (true positives) or insignificant (false negatives) as well as the number of actually insignificant p-values that were marked as insignificant (true negatives) or significant (false positives). The average results for each dataset can be seen in Figure 7.

![Bar chart showing comparison of state-of-the-art methods.](image)

**Figure 7: Comparison of state-of-the-art methods.**

We can see that the standard deviation for all datasets is an order of magnitude smaller than 0.05. Thus if we arbitrarily set the p-value significance threshold to 0.075 (50% greater than 0.05) we expect that the SLI will produce no false negatives. It also means that the results of the randomization test (p-values) are not changing significantly between multiple runs of the same experiment.

### 4.5 Comparison of state-of-the-art with our two approaches

In this section we compare the two versions of our approach along with the with the state-of-the-art (BUFET) in [22]. For this reason, we used 10 inputs of 39 miRNAs (query classifications $A$) and we...
also used a varying number of miRNA groups (query classifications $A_1, \ldots, A_n \in \mathcal{A}$), namely 10K, 100K and 1M. Moreover, as ground classifications, we used the 3 datasets outlined in Section 4.1.

Finally, we configured the FII approach to use a support threshold of 2 for frequent itemsets, since it leads to the best performance for this approach. Regarding the SLI, we set the p-value significance threshold $x$ to 0.075, because as we showed in Section 4.4, it produces no false negatives and thus, the output p-values of the test will be reliable. The results can be seen in Figure 8.

It is clear that both our approaches significantly outperform the state-of-the-art (BUFET) and in the case of SLI, the execution times are faster by almost an order of magnitude.

5 RELATED WORK

Association testing is a very old problem and it belongs to a larger class of statistical problems called hypothesis testing. Even though hypothesis testing became popular in the 20th century, the first references of statistical hypothesis testing started with the works of John Arbuthnot [5] and Pierre-Simon Laplace [9], who tested whether the gender-ratio of humans at birth is equally distributed. Then, in the 1900 Karl Pearson introduced the Pearson’s chi-squared test [2], William Sealy Gosset developed the Student’s t-test [18] and Ronald Fisher developed Fisher’s exact test [11].

Randomization tests received attention in the 1800s with the work of C. S. Peirce [15] and are very popular in clinical trials and life-sciences in general [7, 19]. However, since randomization tests are computationally expensive there have been attempts to make them faster [12, 21, 22] in order to allow researchers to run more tests and gain more insight into the mechanisms of life in a shorter amount of time.

6 CONCLUSION

In this paper we introduced two novel indices in regard to randomization tests and we applied these in a real-world randomization test. The first index (FII) leveraged the overlap that exists in the data regarding ground classifications $B_j$ to reduce computations and also eliminate redundant operations. We also introduced a novel approach to discovering frequent itemsets with $\sup_{thres} = 2$ among ground classifications, in order to use them for the FII. Furthermore, we demonstrated that the second index (SLI) accurately predicts whether the association between a query and an independent, ground classification is potentially significant. Also, the SLI successfully eliminated the vast majority of associations to be tested, thus leading to even smaller execution times. Finally, we performed experiments that clearly show that both of our approaches are faster than the state-of-the-art (BUFET) and the approach with the SLI is even faster (by an order of magnitude).

In the future, we plan to apply the techniques presented in this paper to problems not only to other analyses from the domain of life-sciences, but also to randomization experiments from other scientific disciplines.

ACKNOWLEDGMENTS

This work was partially funded by the “ELIXIR-GR: Managing and Analysing Life Sciences Data (MIS: 5002780)” project (co-financed by Greece and the European Union - European Regional Development Fund).

REFERENCES

[1] 2003. Building and Applying Logistic Regression Models. John Wiley & Sons, Ltd, Chapter 6, 211–266. https://doi.org/10.1002/0471249688.ch6

[2] 2011. Tests for Nominal Scale Data: Chi-Square and Fisher Exact Test. John Wiley & Sons, Ltd, Chapter 8, 155–191. https://doi.org/10.1002/9781118165881.ch8

[3] ZC; States DJ; Ade, AS; Wright. 2007. Gene2MeSH [Internet]. http://gene2mesh.ncbi.org Ann Arbor (MI): National Center for Integrative Biomedical Informatics.

[4] Rakesh Agrawal, Heikki Mannila, Ramakrishnan Srikant, Hannu Toivonen, A Inkeri Verkamo, et al. 1996. Fast discovery of association rules. Advances in knowledge discovery and data mining 12, 1 (1996), 307–328.

[5] John Arbuthnot. 1710. II. An argument for divine providence, taken from the constant regularity observe’d in the births of both sexes. By Dr. John Arbuthnot, Physician in Ordinary to Her Majesty, and Fellow of the College of Physicians and the Royal Society. Philosophical Transactions of the Royal Society of London 27, 328 (1710), 166–190. https://doi.org/10.1098/rszt.1710.0011

[6] Michael Ashburner, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, J. Michael Cherry, Allan P. Davis, Kara Dolinski, Selina S. Dwight, Janan T. Eppig, Midori A. Harris, David P. Hill, Laurie Issel-Tarver, Andrew Kasarskis, Suzanna Lewis, John C. Matese, Joel E. Richardson, Martin Ringwald, Gerald M. Rubin, and Gavin Sherlock. 2000. Gene Ontology: tool for the unification of biology. Nat Genet 25, 1 (05 2000), 25–29. http://dx.doi.org/10.1038/75556

[7] Thomas Bleazard, Janine A. Lamb, and Sam Griffiths-Jones. 2015. Bias in microRNA functional enrichment analysis. Bioinformatics 31, 10 (01 2015), 1592–1598. https://doi.org/10.1093/bioinformatics/btv023

[8] A. Lamb, and Sam Griffiths-Jones. 2015. Bias in microRNA functional enrichment analysis. Bioinformatics 31, 10 (01 2015), 1592–1598. https://doi.org/10.1093/bioinformatics/btv023

Figure 7: Filtering performance of the SLI
|   |   |   |
|---|---|---|
|   |   |   |
|   |   |   |
|   |   |   |
|   |   |   |
|   |   |   |
|   |   |   |
|   |   |   |
|   |   |   |
|   |   |   |

**Figure 8:** Comparison of our two approaches with the state-of-the-art

[8] Christian Borgelt. 2012. Frequent item set mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 2, 6 (2012), 437–456. https://doi.org/10.1002/widm.1074 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/widm.1074

[9] Éric Brian and Marie Jaisson. 2007. *Physico-Theology and Mathematics* (1710–1794). Springer Netherlands, Dordrecht, 1–25. https://doi.org/10.1007/978-1-4020-6036-6_3

[10] Fiona Cunningham, Premanand Achuthan, Wasu Akanni, James Allen, M Ridian Amode, Irina M Armean, Ruth Bennett, Jyothish Bhai, Konstantinos Billis, Sanjay Boddu, Carla Cummins, Claire Davidson, Kamalikumar Jayantilal Dodyya, Astrid Gall, Carlos Garcia Girón, Laurent Gil, Tiago Grego, Leanne Haggerty, Helen Schuilenburg, Dan Sheppard, Helen Sparrow, Eloise Stapleton, Marek Walts, Andrea Winterbottom, Amonida Zadissa, Marc Chakouvardia, Adam Frankish, Sarah E Hunt, Myrto Kostadima, Nick Langridge, Fergal J Martin, Matthias Muffato, Emily Perry, Magali Ruffier, Daniel M Staines, Stephen J Trevanion, Bronwen L Aken, Andrew D Yates, Daniel R Zerbino, and Paul Flicek. 2018. *Nucleic Acids Research* 47, D1 (2018), D4735–D4751. https://doi.org/10.1093/nar/gky1113 arXiv:https://academic.oup.com/nar/article-pdf/47/D1/D4735/27437646/gky1113.pdf

[11] R. A. Fisher. 1992. *Statistical Methods for Research Workers*. Springer New York, New York, NY, 66–70. https://doi.org/10.1007/978-1-4612-4380-9_6

[12] Felipe Gutierrez-Barragan, Vamsi K. Ithapu, Chris Hinzrichs, Camille Maumet, Sterling C. Johnson, Thomas E. Nichols, and Vikas Singh. 2017. Accelerating permutation testing in voxel-wise analysis through subspace tracking: A new plugin for SnPM. *NeuroImage* 159 (2017), 79 – 98. https://doi.org/10.1016/j.neuroimage.2017.07.025

[13] Ilias Kanellos, Thanasis Vergoulis, Dimitris Sacharidis, Theodore Dalamagas, Artemis Hatzigeorgiou, Stelios Sartzetakis, and Timos Sellis. 2014. MR-MicroT: A MapReduce-Based MicroRNA Target Prediction Method. In *Proceedings of the 26th International Conference on Scientific and Statistical Database Management* (SSDBM ’14). Association for Computing Machinery, New York, NY, USA, Article 47, 4 pages. https://doi.org/10.1145/2618243.2618299

[14] Richard M. Karp. 1972. *Reducibility among Combinatorial Problems*. Springer US, Boston, MA, 85–103. https://doi.org/10.1007/978-1-4684-2001-2_9

[15] C. S. Peirce and Joseph Jastrow. 1884. On Small Differences in Sensation. *Memoirs of the National Academy of Sciences* 3 (1884), 75–83.

[16] Janet Piñero, Juan Manuel Ramírez-Anguita, Josep Sastre-Pitarch, Francesco Ronzoni, Emilio Centeno, Ferran Sanz, and Laura I Furlong. 2019. The DisGeNET knowledge platform for disease genomics: 2019 update. *Nucleic Acids Research* 48, D1 (2019), D845–D855.
[17] F. B. ROGERS. 1963. Medical subject headings. Bulletin of the Medical Library Association 51, 1 (Jan 1963), 114–116. https://pubmed.ncbi.nlm.nih.gov/13982385
[18] Student. 1908. The Probable Error of a Mean. Biometrika 6, 1 (1908), 1–25. http://www.jstor.org/stable/2331554
[19] K. Suresh. 2011. An overview of randomization techniques: An unbiased assessment of outcome in clinical research. Journal of Human Reproductive Sciences 4, 1 (2011), 8–11. https://doi.org/10.4103/0974-1208.82352
[20] The Gene Ontology Consortium. 2018. The Gene Ontology Resource: 20 years and still GOing strong. Nucleic Acids Research 47, D1 (11 2018), D330–D338. https://doi.org/10.1093/nar/gky1055
[21] Anderson M. Winkler, Gerard R. Ridgway, Gwenaëlle Douaud, Thomas E. Nichols, and Stephen M. Smith. 2016. Faster permutation inference in brain imaging. NeuroImage 141 (2016), 502 – 516. https://doi.org/10.1016/j.neuroimage.2016.05.068
[22] Konstantinos Zagganas, Thanasis Vergoulis, Maria D. Paraskevopoulou, Ioannis S. Vlachos, Spiros Skiadopoulos, and Theodore Dalamagas. 2017. BUFET: boosting the unbiased miRNA functional enrichment analysis using bitsets. BMC Bioinformatics 18, 1 (2017), 399:1–399:8. https://doi.org/10.1186/s12859-017-1812-8