ΠΑΝΕΠΙΣΤΗΜΙΟ
ΠΕΛΟΠΟΝΝΗΣΟΥ
UNIVERSITY *of the* PELOPONNESE

Ph.D. Thesis

---

# Scalable data-driven enrichment analysis of short RNAs

---

*Author:*
Konstantinos Zagganas

*Supervisor:*
Professor Spiros Skiadopoulos

23 Μαρτίου 2022

# Κλιμακώσιμη δεδομενοκεντρική ανάλυση εμπλουτισμού σε RNA μικρού μήκους

Διδακτορική Διατριβή
του
*Κωνσταντίνου Ν. Ζαγγανά*

Διπλωματούχου Ηλεκτρολόγου Μηχανικού και Μηχανικού Υπολογιστών Εθνικού Μετσοβίου Πολυτεχνείου (2014)

| | | |
|---|---|---|
| *Συμβουλευτική Επιτροπή:* | Σπύρος Σκιαδόπουλος | Επιβλέπων καθηγητής |
| | Χρήστος Τρυφωνόπουλος | |
| | Θεόδωρος Δαλαμάγκας | |

Εγκρίθηκε από την επταμελή εξεταστική επιτροπή την 24/03/2022

| Όνομα | Βαθμίδα | Ίδρυμα |
|---|---|---|
| Κωνσταντίνος Βασιλάκης | Καθηγητής | Παν. Πελοποννήσου |
| Θεόδωρος Δαλαμάγκας | Διευθυντής ερευνών | Ε.Κ. «ΑΘΗΝΑ» |
| Γεώργιος Δροσάτος | Εντεταλμένος Ερενυτής | Ε.Κ. «ΑΘΗΝΑ» |
| Νικόλαος Πλατής | Επίκουρος Καθηγητής | Παν. Πελοποννήσου |
| Σπύρος Σκιαδόπουλος | Καθηγητής | Παν. Πελοποννήσου |
| Χρήστος Τρυφωνόπουλος | Καθηγητής | Παν. Πελοποννήσου |
| Φώτης Ψωμόπουλος | Εντεταλμένος Ερευνητής | INEB/EKETA |

# Ευχαριστίες

Αρχικά Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου, Δρ. Σπύρο Σκιαδόπουλο, για την εμπιστοσύνη που μου έδειξε αναλαμβάνοντας την επίβλεψή μου. Ιδιαίτερα θέλω να ευχαριστήσω το Θανάση Βεργούλη και το Θοδωρή Δαλαμάγκα από το Ινστιτούτο Πληροφοριακών Συστημάτων (ΙΠΣΥ) του Ε.Κ. «ΑΘΗΝΑ» για την εμπιστοσύνη που μου έδειξαν από την πρώτη στιγμή, καθώς και για την πολύτιμη βοήθειά τους. Πέρα από εξαιρετικούς συνεργάτες, τους θεωρώ και σημαντικούς φίλους. Επίσης, θέλω να εκφράσω τις ευχαριστίες μου στο Ε.Κ. «ΑΘΗΝΑ» για την πάσης φύσεως υποστήριξη που μου έδωσε ώστε να διευκολυνθεί η εκπόνηση αυτής της διατριβής.

Παράλληλα είμαι ευγνώμων στους Γιώργο Γεωργακίλα, Μαρία Λιόλη, Ηλία Κανέλλο, Ιωάννη Βλάχο και Μαρία Παρασκευοπούλου με τους οποίους συνεργαστήκαμε σε εργασίες σχετικές με το ερευνητικό μου αντικείμενο. Ομοίως, θα ήθελα να ευχαριστήσω όλους τους συνεργάτες από το Ε.Κ. «Αθηνά» με τους οποίους συνεργάστηκα τα τελευταία χρόνια για την διεκπεραίωση των απαιτήσεων των πάσης φύσεως ερευνητικών έργων.

Τέλος, τίποτα δεν θα ήταν εφικτό χωρίς την ανιδιοτελή υποστήριξη των φίλων και της οικογένειάς μου όλα αυτά τα χρόνια, και τους ευχαριστώ θερμά.

# Abstract

*microRNAs* (or *miRNAs*) are short RNA molecules that play a crucial role in the regulation of *gene expression*, i.e. the production of proteins, which are important structural and functional components of a cell. miRNAs silence genes by binding to them, thus blocking the production of the respective protein. However, the mechanisms underlying miRNA function are often very complex; combined with the fact that the number of miRNAs in a given organism ranges in the order of thousands, it becomes evident that utilizing experiments in a wet lab is a lengthy, arduous and often expensive process. Additionally, scientists may be interested to compare differences in miRNA expression between diseased and healthy individuals and need to quantify them, using statistical methods. For this reason, Bioinformatics researchers developed *in-silico* methods and algorithms like *miRNA enrichment analysis*, which is a statistical technique to predict whether a set of miRNAs is likely to affect a certain biological function. One of the most recent such approaches is the *unbiased miRNA functional enrichment analysis*, which relies on a considerably large number of set operations and consequently, it leads to execution times in the order of hours or days.

In this thesis, we strive to make the unbiased miRNA enrichment, a computationally-intensive, data-driven analysis, more scalable by utilizing data management and other computer science techniques. Initially, we examine the performance of data structures called bit vectors in comparison to hash tables as a set representation technique and propose a new hybrid approach to reduce the execution time of the analysis. Moreover, we optimize miRNA enrichment by introducing two novel indices, utilized in order to reduce redundant set participation operations and filter out potentially insignificant associations between miRNAs and biological functions. Additionally, we showcase that the state-of-the art unbiased enrichment suffers from reduced sensitivity to false negatives and we propose a modification to its statistics engine in order to increase its quality. Furthermore, we propose an approach using supervised machine learning methods to predict approximate p-values in real time, instead of executing a full analysis. Finally, we introduce data management and processing techniques during the design of online miRNA analysis tools to achieve analyses in almost real-

time; also, we try to address the need for a platform facilitating reproducible and scalable execution of containerized software in a Cloud environment, consisting of heterogeneous machines.

# Περίληψη

Τα *microRNA* (ή *miRNA*) είναι μόρια RNA μικρού μήκους που παίζουν έναν πολύ σημαντικό ρόλο στη ρύθμιση της γονιδιακής έκφρασης, δηλαδή την παραγωγή πρωτεϊνών, οι οποίες αποτελούν σημαντικά δομικά και λειτουργικά τμήματα ενός κυττάρου. Τα miRNA «αποσιωπούν» τα γονίδια μέσω της πρόσδεσής τους με αυτά, σταματώντας την παραγωγή της αντίστοιχης πρωτεΐνης. Παρόλα αυτά, οι μηχανισμοί που διέπουν τη λειτουργία των miRNA είναι συνήθως αρκετά περίπλοκοι και σε συνδυασμό με το γεγονός ότι ο αριθμός των miRNA σε κάποιον οργανισμό μπορεί να φτάσει τις αρκετές χιλιάδες, γίνεται αντιληπτό ότι τα πειράματα σε ένα εργαστήριο μπορεί να είναι μια μακρά, δύσκολη και συχνά ακριβή διαδικασία. Επιπρόσθετα, κάποιοι επιστήμονες ενδιαφέρονται να συγκρίνουν τη γονιδιακή έκφραση ανάμεσα σε υγιή άτομα και άτομα που νοσούν από κάποια ασθένεια και χρειάζεται να ποσοτικοποιήσουν αυτή τη διαφορά μέσω στατιστικών μεθόδων. Για αυτό το λόγο, ερευνητές Βιοπληροφορικής έχουν αναπτύξει μεθόδους προσομοίωσης και αλγορίθμους σε υπολογιστή όπως η *Ανάλυση εμπλουτισμού miRNA*, η οποία αποτελεί μια στατιστική τεχνική πρόβλεψης του κατά πόσο ένα σύνολο από miRNA μπορεί να επηρεάζει μια βιολογική λειτουργία. Μία από τις πρόσφατες προσεγγίσεις τέτοιων αναλύσεων αποτελεί ο *αμερόληπτος λειτουργικός εμπλουτισμός για miRNA*, που βασίζεται σε έναν σημαντικά μεγάλο αριθμό πράξεων μεταξύ συνόλων και με αυτόν τον τρόπο, οδηγεί σε χρόνους εκτέλεσης που έχουν τάξη μεγέθους ώρες ή ακόμα και μέρες.

Στη διατριβή αυτή επιδιώκουμε να δώσουμε στον αμερόληπτο εμπλουτισμό miRNA, που αποτελεί μια υπολογιστικά εντατική δεδομενοκεντρική ανάλυση, μια πιο κλιμακώσιμη μορφή, χρησιμοποιώντας τεχνικές διαχείρισης δεδομένων και άλλες μεθόδους της επιστήμης υπολογιστών. Αρχικά εξετάζουμε την απόδοση μια δομής δεδομένων, που ονομάζεται διανύσμα από bit, σε σύγκριση με την απόδοση των πινάκων κατακερματισμού για αναπαράσταση συνόλων και προτείνουμε μία νέα, υβριδική προσέγγιση για τη μείωση του χρόνου εκτέλεσης. Παράλληλα, βελτιστοποιούμε την ανάλυση εισάγοντας δύο νέα ευρετήρια που χρησιμοποιούνται για την εξάλειψη πράξεων συνόλων που εκτελούνται περισσότερες από μία φορές καθώς και για να φιλτράρουν πιθανά στατιστικά ασήμαντες συσχετίσεις ανάμεσα σε miRNA και βιολογικές λειτουργίες. Επιπρόσθετα, δείχνουμε ότι η τεχνολογία αιχμής παρουσιάζει μειωμένη ευαισθησία στα ψευδώς αρνη-

τικά αποτελέσματα και επίσης προτείνουμε μία τροποποίηση στον στατιστικό πυρήνα της ανάλυσης προκειμένου να αυξήσουμε την ποιότητά της. Επιπλέον, προτείνουμε μία προσέγγιση πρόβλεψης p-values σε πραγματικό χρόνο αντί του πλήρους αναλυτικού υπολογισμού μέσω της χρήσης εποπτευόμενων τεχνικών μηχανικής μάθησης. Τέλος, εισάγουμε τεχνικές διαχείρισης και ανάλυσης δεδομένων κατά τη σχεδίαση διαδικτυακών εργαλείων, προκειμένου να επιτύχουμε αναλύσεις πραγματικού χρόνου· ταυτόχρονα προσπαθούμε να καλύψουμε την ανάγκη για μια πλατφόρμα που διευκολύνει την αναπαραγωγή και την κλιμακώσιμη εκτέλεση κιβωτιοποιημένου λογισμικού σε περιβάλλον Νέφους που αποτελείται από μηχανές με ετερογενή χαρακτηριστικά.

# Contents

# List of Figures

# Chapter 1

# Introduction

Since the discovery of the DNA helical structure in 1969, the study of nucleic acids (DNA and RNA) has advanced significantly with the aim of discovering the mechanisms of life as well as curing disease. The field of Biology that deals with nucleic acid research is known as Genomics. However, experiments in Genomics are mainly performed *in-vitro* or *in-vivo*, often requiring expensive materials and consent to be performed. Another issue is that the information contained in genetic material is vast and complex and often implicates more than one set of molecules concurrently, leading to a large network of interactions that often need to be explored. It becomes evident, that such an undertaking is complex bringing about the development of Bioinformatics. Bioinformatics usually involves *in-silico* techniques to model and simulate experiments in order to accurately predict their outcomes.

The field of Bioinformatics itself is divided in many fields; one of them deals with short RNA molecules, that do not have the capacity to produce proteins, called microRNAs (or miRNAs). Even though miRNAs are not responsible for producing proteins, their role in protein production regulation is significant. More specifically, miRNAs can influence whether a protein will be produced or not by binding to specific genes, thus preventing them from producing the respective proteins. This immediately implies that dys-regulation of miRNAs in tissues can lead to biological function disorders and indeed, many miRNAs have been linked to human diseases [LK12]. Moreover, different miRNAs can simultaneously affect one or more biological processes and simultaneously increase or decrease the combined impact on it.

Consequently, it is easy to understand why many methods have been developed, in order to predict the involvement of miRNAs in a variety of biological processes and the impact that miRNA function perturbation has on them. One such method is miRNA functional enrichment analysis that uses statistical methods to measure the effect of a group of miRNAs on a biological function, leveraging openly-accessible, published data sets. This method usually involves Fisher's exact test [Fis92], a statis-

tical test commonly used for association testing of two characteristics in a population of objects. The test utilizes the hypergeometric distribution and it is a very popular method with published research. However, in recent years, it was shown that it presents a bias stemming from a variety of factors that cannot be easily eliminated. This led to the development of the unbiased miRNA enrichment analysis, that takes advantage of the empirical (observed) distribution of the data through the utilization of a randomization test. The unbiased miRNA enrichment analysis is based on the calculation of a statistical measure called *left-sided* overlap, which relies mainly on union and intersection set operations. Given a group containing $n$ miRNAs under examination, the randomization test involves the production of the empirical distribution, i.e. randomly assembled miRNA groups with the same size as the initial one. Then the left-sided overlap is calculated for each permutation and the groups are sorted based on the result of this calculation. This is the empirical distribution and the sample of interest is compared against it in order to calculate an *empirical p-value*. It becomes evident that in order to produce the empirical distribution, all possible permutations of the data must be calculated. This means that such techniques can become impossible to approach in a reasonable time frame and thus, Monte Carlo simulations are usually preferred.

Still, even such simulations are very computationally intensive. This usually motivates software developers to scale their software based on methods like data-level parallelism, in order to reduce execution times. Another trend in Bioinformatics in the last few years is the use of containerized software, which means software that is packaged along with all libraries and dependencies on a simple image. The image then is used as a template to create a temporary virtual machine that executes the software and is then deleted. Combined with a cloud infrastructure and an orchestration software like Kubernetes[1], the execution of multiple analyses at the same time is facilitated. However, current approaches require programmatic access and a dedicated DevOps engineer that can schedule the execution of the software.

## 1.1 Motivation

As mentioned in the previous section, most software approaches to the unbiased miRNA enrichment analysis tend to leverage parallelism instead of other approaches in order to make the required execution time more manageable. This work initially aimed to introduce speedup of the unbiased miRNA enrichment analysis, by utilizing better performing data structures as well as techniques like indexing, to both accelerate set operations and reduce operations that are performed redundantly. These

[1] https://kubernetes.io/

methods apply to both single- as well as multi-processor settings. During the course of this work, however, it became evident that the left-sided overlap used by the analysis presents reduced sensitivity regarding false positives and a new statistical measure was introduced. Evidently, the vast majority of this work relies on computationally intensive experiments, that require a lot of resources for experiment execution. The use of these resources was facilitated through the utilization of containerized software, in conjunction with a Kubernetes cloud infrastructure. However, this highlighted the need for a platform that can help facilitate experiment execution, through an intuitive web interface, while taking advantage of the scalability of Kubernetes clouds. For this reason, we developed a user-friendly platform that allows researchers to execute containerized software, leveraging the underlying cloud infrastructure. Moreover, the system utilizes the RO-crate standard[2], which is an approach to packaging research data, along with their metadata to facilitate reproducibility of scientific experiments.

## 1.2 Contribution

The contribution of this work consists of the following points:

- We studied the computational requirements and examined the performance bottlenecks of the unbiased miRNA functional enrichment analysis.
- We investigated the performance of different data structures, namely hash tables and bitsets, in regards to their effectiveness in unblocking the identified bottlenecks. Moreover, we developed `BUFET`, a tool that utilises the results of the aforementioned investigation to boost the speed of the unbiased miRNA functional enrichment analysis. To achieve an even greater speed boost in the case of multi-core environments, we exploited multithreading to implement parallel execution of the analysis. Then, we performed an extensive evaluation of `BUFET` to demonstrate its efficiency. `BUFET` outperforms the state-of-the-art approach in all scenarios (in many cases by an order of magnitude).
- We introduced two novel indices to facilitate the efficient execution of randomization tests, the Frequent Itemset Index (FII) and the Significance Level Index (SLI). Furthermore, we implemented a novel approach that exploits the FII index to avoid redundant computations and significantly decrease execution times. In addition to that we developed a second approach that combines both indices (FII and SLI) to not only eliminate statistically insignificant associations but also vastly reduce the number of computations required. Moreover,

[2] https://www.researchobject.org/ro-crate/

we conducted comprehensive experiments showing that the use of our indices introduces significant speedup; more specifically, the approach combining both indices outperforms the state-of-the-art by an order of magnitude.

- We demonstrated that when the bias in miRNA functional enrichment analysis is removed, a new source of bias, related to gene annotations, is highlighted; this bias results in reduced sensitivity. In order to remove this bias, we modified the state-of-the-art statistical test and introduced a new statistical measure that increases the sensitivity of the test. Moreover, we designed BUFET2, a new version of BUFET, that implements the modified statistical approach. At the same time, we performed extensive experimental evaluation, which shows that BUFET2 is indeed more sensitive to false negative results in comparison to the state-of-the-art statistical approach.

- In order shorten to p-value production times even further, we attempt to predict approximate p-values by using supervised Machine Learning (ML) techniques. For this reason, we introduce a novel ML approach for miRNA enrichment analysis. To this end, we framed the problem, created an appropriate dataset and engineered several features. Furthermore, we determined a shortlist of promising machine learning problems, trained them using the cross-validation method and fine-tuned their parameters in to determine the best models for our case. Our approach shows that the best model demonstrates $MAE = 0.048$.

- We applied data management techniques to three online tools for miRNA analysis, in order facilitate data analysis by Bioinformatics researchers in almost real-time.

- Taking into account the need for experiment reproducibility, as well as the easy execution of containerized software, utilizing a Cloud infrastructure, we introduced a novel system that facilitates scalable execution of containerized software inside a Kubernetes cloud. The system promotes reproducibility through the use of RO-crates, a standard that is gaining popularity with each passing year. This system was also used to perform the experiments outlined in Chapter 5.

## 1.3 Structure of this thesis

The structure of this thesis consists is as follows: in Chapter 2 we introduce background knowledge related to the subject of this thesis, in order to assist the reader. In Chapter 3 we present the utilization of data-driven techniques in the unbiased miRNA enrichment; we also show how a speedup of an order of magnitude compared

to the state-of-the art was achieved. In Chapter 4 we introduce two novel indices for use by the unbiased miRNA enrichment and present an extensive evaluation which illustrates that the combined effect of the two indices leads to a further decrease in execution times by an order of magnitude. Moreover, Chapter 5 we recognize that the unbiased miRNA enrichment suffers from reduced sensitivity to false negatives and we establish a modification in the statistical engine of the analysis to amend this issue. In Chapter 6 we demonstrate the use of supervised machine learning techniques, exploiting the knowledge accumulated in the previous chapters, in order to select appropriate features and train a variety of models. Then, we evaluate the performance of these models and utilize the one with the best performance. Finally, in Chapter 7 we introduce three online tools that are useful in miRNA analysis, for which data management and indexing techniques were utilized; additionally, we introduce a new platform that takes advantage of containerization techniques and machine learning to facilitate reproducible research and elastic use of resources respectively. This platform was used to perform experiments for the work introduced in Chapter 5.

# Chapter 2

# Background and related work

*Proteins* are considered to be the fundamental building blocks of life, that are also responsible for a wide variety of biological functions. Examples include proteins that are used as construction materials for cells (e.g. their membrane), enzymes that catalyze specific chemical reactions (e.g. the metabolizing of alcohol) or even proteins that are utilized during the production of other proteins.

The information for protein production is encoded in a chemical polymer called *Deoxyribonucleic Acid* (DNA). This polymer does not consist of a signle molecule, but rather 4 almost identical chemical molecules, called *bases*. These 4 bases are called Thymine (denoted by `T`), Adenine (denoted by `A`), Guanine (denoted by `G`) and Cytocine (denoted by `C`). A sequence of interconnected bases constitutes a single *strand* of DNA. This strand is interconnected using the phosphoric acid part of the base molecule and it leaves the base end of the molecule dangling. This free part is able to create a bond with another base in a complementary fashion, i.e. `G` is only able to bind itself to the free end of `C` and `T` can only create a bond with `A`. This is called base complementarity and it leads to the creation of a sequence of bases, which is complementary to the first strand. The two strands of DNA are bound together using the base complementarity, thus creating a double-stranded DNA helix.This DNA helix is then organized in chromosomes and each chromosome is responsible for producing a very large number of proteins.

The information regarding protein production is in fact the DNA sequence itself and it is organized in segments of DNA called *genes*. Each gene is responsible for producing one or more proteins through the process of *transcription* and *translation* of that gene.

Gene transcription is the process where the double-stranded helix unfolds and one of the strands is "copied" using base complementarity into a *Ribonucleic Acid* (RNA) segment. RNA is a molecule with a similar structure as DNA, where Thymine is substituted with Uracile (denoted by `U`) and can complementarily bind to DNA where

`U` binds with `A`. For example, given a DNA sequence `GATTACCA` the RNA sequence produced (*transcript*) will be `CUAAUGGA`. After the process of gene transcription, the transcript undergoes a removal of certain segments, called *introns* and the final transcript, also known as messenger RNA (or mRNA), only consists of the remaining segments called *exons*. Depending on which protein will be produced, different segments of the same gene are regarded as introns and removed and this immediately implies that a single DNA gene can produce a multitude of different proteins.

After a gene is transcribed, the mRNA undergoes the process of translation where a molecular complex called *ribosome*, consisting of proteins and RNA, binds to it and begins assembling the protein. Based on the RNA sequence, the ribosome gathers different amino-acids and "glues" them together to produce the final product, which is the protein. Then the mRNA is either used a few times before it degrades, or it is disassembled immediately after translation and its bases are reused for new transcripts. The whole process from DNA unfolding to protein production is called *gene expression*.

## 2.1 Gene expression between healthy and diseased states

It has been observed that genes can present different expression between diseased and healthy tissues. Investigating such differences can assist researchers in understanding the pathology of diseases with the aim of treating them [REJ17]. In order to calculate this difference, tissues regarding a disease are first sampled and the level for the expression of each individual gene is quantified using methods like *microarrays* or *RNA-seq*. These methods essentially count the number of transcripts (mRNA) produced by each gene in the sample, which in turn translates to the amount of proteins that will be produced for that gene. This is why the terms gene, transcript and protein can be used interchangeably. The process is then repeated for the tissues of a healthy individual and the expression levels of each gene are estimated once more. Finally, the gap between the gene quantities in each state is bridged by using *Differential Expression Analysis* (or DEA in short). DEA utilizes statistical methods to calculate how large the change in the transcript count is between the two states by using algorithms like LIMMA [RPW+15], DESeq [AH10], edgeR [RMS09], etc.

Each of these algorithms produces relevant statistics like fold change, denoting the amount of change between the two states, and p-value, essentially denoting the probability that a gene presents a large enough change between the two states. Genes that demonstrate a p-value$\geq 0.05$ are called *differentially expressed*.

## 2.2 Linking differentially expressed genes with gene classifications

The explosion of compute resource capabilities and availability in the 1990's allowed several initiatives, which map genes to specific biological functions, to form. These initiatives *annotate* genes, either automatically or manually, and provide data sets which contain gene-to-biological-function data sets. Most prominent in this effort are the Gene Ontology [The18][ABB+00] and the KEGG pathway [KSK+16][KG00a] data sets. Other resources include DisGeNET [PRASP+19a], Medical subject Headings (MeSH) [ROG63], PsyGeNET [GSGV+15], OMIM [HSA+05] and others.

The Gene Ontology (GO) is a major bioinformatics initiative that attempts to unify the representation of genes and gene products across all species. The ontology consists of a set of classes (or terms, or concepts) that map each gene to one or more specific biological functions. In 2010 the majority of terms (about 98%) was automatically curated; however in order to increase accuracy, in 2019 only about 30% of the annotations are automatically inferred, while the rest are manually curated. The GO is covers three large domains:

- **cellular component:** the components of a cell or the environment outside it,
- **biological process:** operations or molecular events relevant to the function of units of life like cells, tissues, organs and organisms and
- **molecular function:** the interaction of a gene product at the molecular level pertinent to chemical reactions like protein binding or participation in a reaction as a catalyst.

The KEGG (Kyoto Encyclopedia of Genes and Genomes) pathway database is a collection of manually drawn pathway maps which portrays experimental knowledge regarding metabolism and other functions of Life. Each pathway map contains a molecular interaction and reaction network, linking genes in the genome to gene products, like proteins, RNA, chemical compounds, etc. The pathway maps are split into the following categories:

- Metabolism,
- Genetic information processing (like transcription, translation, DNA repair, etc.),
- Environmental information processing (like cell growth/death),
- Cellular processes (like signal transduction),

- Organismal systems (like the immune system, endocrine system, nervous system etc.),
- Human diseases (like Alzheimer's disease) and
- Drug development.

DisGeNET is one of the largest repositories containing human gene-to-disease associations currently available, while PsyGeNET is mostly focused on psychiatric disorders and their underlying genes. OMIM is a comprehensive, authoritative collection of gene-to-mendelian-disorders associations, focusing on the relationship between phenotype and genotype. Finally, MeSH is a controlled vocabulary thesaurus used for indexing articles from PubMed and there have been efforts [AS07] to associate these headings with specific genes.

### 2.2.1 Gene set enrichment analysis

Given a list of dysregulated genes between two states (e.g. healthy and Alzheimer's disease) the interest of researchers focuses on discovering whether this list of dysregulated genes can be attributed to the disease itself or to another cause. For this reason, statistical methods have been developed that "measure" the degree of association between the gene list and a particular GO term, pathway or disease (referenced as *gene class* from here on). This process is called *gene set enrichment analysis* and the two most popular methods are the *Overrepresentation analysis*, that utilizes Fisher's exact test [Fis92], and *Gene Set Enrichment Analysis (GSEA)* [STM+05] (not to be confused with the more general term), which relies on a randomization statistical test.

#### 2.2.1.1 Overrepresentation analysis

This analysis uses Fisher's exact test to "measure" the association between a set of dysregulated genes and a list of gene classes. This test relies on the hypergeometric distribution to calculate probabilities that constitute the p-value.

First, let $G$ be the set of dysregulated genes and $C$ a class of genes under examination (essentially a set of genes participating in that class). Given a null hypothesis that $G$ and $C$ are not associated, a $2 \times 2$ contingency table is constructed as follows: After the contingency table is constructed, the formula of the hypergeometric distribution is used to calculate the probabilities of all events at least as extreme as the one shown in the table:

$$p = \frac{\binom{a+b}{a}\binom{c+d}{c}}{\binom{n}{a+c}} = \frac{(a+b)!\ (c+d)!\ (a+c)!\ (b+d)!}{a!\ b!\ c!\ d!\ n!} \tag{2.1}$$

| | In G | Not in G | *Row total* |
|---|---|---|---|
| In $C$ | $a$ | $b$ | $a+b$ |
| Not in $C$ | $c$ | $d$ | $c+d$ |
| *Column total* | $a+c$ | $b+d$ | $n(=a+b+c+d)$ |

**Table 2.1:** Contingency table for Fisher's exact test

$$a = G \cap C,$$
$$b = C - G,$$
$$c = G - C \text{ and}$$
$$d = universe - G - C + (G \cap C)$$

Finally, the p-value indicates the probability to observe the numbers in the table given that the null hypothesis is true. This means that the smaller a p-value is, the stronger the association. In bibliography, a p-value of 0.05 or smaller is regarded as statistically significant.

Before this section is completed, it is worth mentioning that Fisher's exact test, makes the assumption that the genes in $G$ and $C$ are equally probable to be picked from the universe of genes in order to fulfil the requirement of the hypergeometric distribution. It is evident that such an assumption is not really feasible when dealing with real-world biological data and this lead to the development of new statistical methods that measure the level of association like *GSEA*, presented in the next section.

#### 2.2.1.2 Gene Set Enrichment analysis (GSEA)

GSEA is an analytical method used to investigate gene expression data utilizing a class of genes with similar biological function. Given a set of genes retrieved from differential expression analysis, the set is sorted using the expression level of each gene, thus creating a *ranked list* $L$. Lset S be the set of genes that participate in a gene class (GO term, KEGG pathway, disease, etc.). The method then consists of the following steps:

- We calculate the *enrichment score* ($ES(S)$) by utilizing a running sum statistic while traversing the ranked list. This means that for each gene in $L$ we need to check whether the gene also exists in $S$. If it exists, we increase the running sum, depending on the correlation of the gene with the phenotype (i.e. its expression level or its rank in the list). If the gene is not found in $S$, then the running sum is decreased by a constant which depends on the number of genes in the L. The enrichment score $ES(S)$ is the maximum deviation of the running sum from zero. This process can be seen in Figure 2.1.

- We estimate an empirical (nominal) p-value for $ES(S)$ by shuffling the genes in the ranked list without keeping their expression level the same (i.e. permuting in the phenotype level) and calculate a $ES_1(S)$ as before. We repeat this process a large number of times in order to create a null empirical distribution of enrichment scores. Finally, the p-value is calculated relevant to its position on the empirical distribution (i.e. the proportion of enrichment scores $ES_j(S)$ that present a more meaningful score than $ES(S)$).

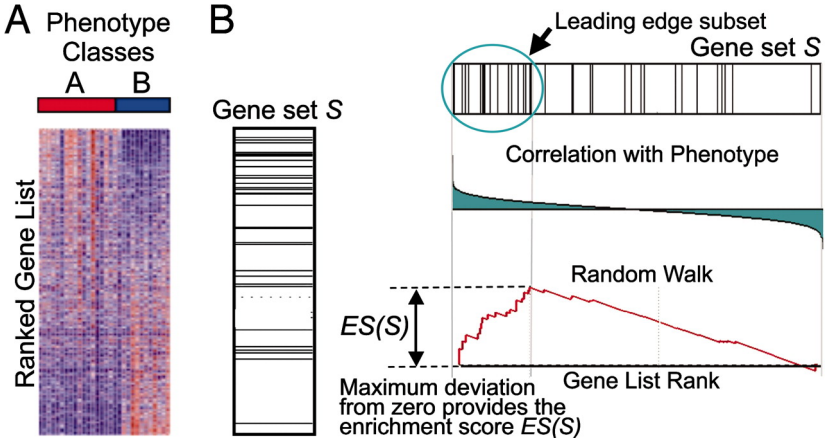



**Figure 2.1:** A single repetition of the GSEA method [STM+05]

## 2.3 miRNAs

microRNAs (or miRNAs) are short RNA molecules with a length of $\sim 23$ bases that are not utilized for protein production (non-coding RNA). However, miRNAs bind to mRNA produced by genes and either cause degradation of the mRNA or prevent ribosomes from binding with the mRNA. This makes them very potent gene expression regulators because they control the amount of a protein produced by *targeting* the respective gene. Since base complementarity (in a complete or incomplete way) is a prerequisite for RNA-to-RNA interactions, it becomes evident that specific miRNAs can only target a specific list of genes. The most reliable way of discovering the targets of a miRNA is through experimentation in a wet lab, where scientists can examine whether a miRNA can bind to the mRNA of a specific gene. There are numerous publications describing such miRNA-to-gene interactions and there have been efforts like TarBase [KPC+17a] or miRTarBase [CSY+17] to collect and curate these interactions aiming to compile databases of experimentally validated interactions. Some of those curated databases are used through the rest of this work. However, since about 25,000 genes and about 2,500 miRNAs have been discovered in humans,

it becomes immediately evident that the numbers of potential interactions that have to be validated is in the order of millions and this makes such a task infeasible.

To overcome this insurmountable task, scientists have developed algorithms that use characteristics of mRNAs like length of the 3'-UTR region, base complementarity and sequence conservation across species (sequences that have been preserved in a large degree through the process of evolution), in order to predict targets of a specific miRNA. Furthermore, each prediction result is accompanied by a prediction score which denotes the confidence that such an interaction is possible. Such miRNA target prediction algorithms include microT-CDS [PGK+13][RMA+12], TargetScan [LBB], miRanda [JEA+04] and others. A disadvantage of such algorithms is that predicting miRNA-to-gene interactions using such algorithms is computationally intensive and there have been efforts to reduce execution times to a few minutes using Cloud technologies [VAD+12] [KVS+14a]. On the other hand, prediction algorithms produce a very large number of false positive results, leading to an overestimation of the role of miRNAs in normal and pathological conditions [PLM+17a]. However, target prediction algorithms are still useful, because results with very low scores can be eliminated as potential interactions, thus saving time and resources in wet labs.

It should also be noted here that miRNAs are themselves product of translation of genes that do not participate in protein production (non-coding genes). Furthermore, it has been discovered that as many as 40% of miRNAs reside in the introns and exons of other protein coding genes [RGJAB04]. The sites where the transcription of miRNAs begins, are called *Transcription Start Sites* (TSSs), where a special protein called *RNA polymerase II* performs the process of transcription. Transcription itself is a very complex procedure and it is regulated, among others, by special proteins called *Transcription Factors* (TFs). TFs bind to certain DNA sequences and have the potential to block RNA polymerases from binding to a gene and this makes them very potent regulators of transcription. Finally, similar to miRNA target discovery, TSS identification and TF binding site discovery is a complicated task usually performed by *in silico* methods like microTSS [GVP+14] (for TSS identification) and Wellington [PEC+13] (for TF binding site discovery).

## 2.4 miRNA functional enrichment analysis

It has been observed that similarly to genes (see Section 2.1), miRNAs can be differentially expressed in tissues between a normal and diseased person. This means that the same algorithms, presented in Section 2.1, can be used to measure the level of differential expression between the two states. Utilization of DEA algorithms leads to the procurement of a group of miRNAs that are dysregulated in a diseased state.

However, this does not immediately imply that this group of miRNAs is dysregulated due to the disease or another reason. This provides the motivation to measure the association of this group of miRNAs to a disease using statistical methods. In order to achieve this, researchers need to procure (a) a data set containing gene targets for each miRNA (either predicted by an algorithm or experimentally validated) and (b) one of the data sets described in Section 2.2 that associates genes with a disease or pathway/biological function related to a disease. These two data sets are then utilized in order to statistically measure the association of a group of miRNAs with a biological function via the genes involved. In the following sections we are going to introduce the two most popular methods for miRNA functional enrichment analysis, which is the classic *overrepresentation analysis* as well as the *unbiased miRNA functional enrichment analysis*

### 2.4.1 Overrepresentation analysis

The miRNA overrepresentation analysis presents a large similarity to its gene counterpart wherein a $2 \times 2$ contingency table is created in order to use Fisher's exact test. The difference lies in the process of substituting the group of dysregulated genes with the union or intersection of the target genes of a group of dysregulated miRNAs. It should be noted here, that the intersection of targets of a group of miRNAs can very easily lead to an empty set, especially when a data set containing experimentally validated targets is used, since the overlap between miRNA targets may be very small. Consequently, the union of targets is preferred as the group's set of targets. Let M be the group of targets and C be the gene class (category/pathway/disease); the contingency table seen in see Table 2.2 is then created. After the table construction

| | In M | Not in M | *Row total* |
|---|---|---|---|
| In $C$ | $a$ | $b$ | $a+b$ |
| Not in $C$ | $c$ | $d$ | $c+d$ |
| *Column total* | $a+c$ | $b+d$ | $n(=a+b+c+d)$ |

**Table 2.2:** Contingency table for Fisher's exact test on miRNAs

$$a = M \cap C,$$
$$b = C - M,$$
$$c = M - C \text{ and}$$
$$d = universe - M - C + a$$

$a$,$b$,$c$ and $d$ are provided as input to Fisher's exact test and a p-value is produced denoting the level of association between the miRNA groups and a gene class.

It is also worth noting here that Fisher's exact test relies on the estimation of a large number of probabilities using formula 2.1. This number relies mainly on

$a$ in the contingency table and it becomes immediately evident that the number of calculations required is substantial and this means that this method was really not very practical before computers gained enough compute power to support such calculations. However, miRNAs are a special case, in that, they were discovered in the early 1990s [LFA93], when computers could support such intensive computations and consequently overrepresentation analysis became the standard method for miRNA functional enrichment analysis. On the other hand, regarding miRNAs, this method is known to suffer from specific biases that skew the results and make this method not suitable and this means that the results of hundreds of published studies are affected [BLGJ15]. In the next section we are going to introduce the definition of unbiased enrichment analysis, as was proposed by Bleazard *et al.*

### 2.4.2 Unbiased enrichment analysis

As mentioned in Section 2.2.1.1, the standard method makes the assumption that genes in sets $M$ and $G$ have an equal probability of being picked from the universe of genes. This essentially means, that each gene has an equal probability of being targeted by a miRNA. In 2015 Bleazard *et al.* described several sources of bias affecting miRNA target prediction algorithms, that immediately invalidate the requirement of the hypergeometric distribution (i.e. genes are not equally targeted by a miRNA). Moreover, in 2021, we demonstrated that another source of bias affects the standard method, namely the fact that genes are not equally likely to participate in a biological function [ZVG+21]. These works showcase that the standard method is not suitable for miRNA enrichment analysis while the latter of the two implies that this bias has the potential to affect gene overrepresentation enrichment analysis.

In regard to Bleazard *et al.*, the authors used data sets procured from data prediction algorithms, as well as data from GO (see Section 2.2 to demonstrate the issues that that arise when using the standard method. More specifically, they calculated the *empirical* (observed) distribution of the intersection between the targets of a query miRNA group (containing 39 miRNAs) and the "ion transport" biological process. Furthermore, they estimated the expected hypergeometric distribution of the overlap and they plotted the two distributions side-by-side (see Figure 2.2).

Figure 2.2) immediately suggests that the hypergeometric distribution and consequently Fisher's exact test is a poor choice of statistic method for miRNA functional enrichment analysis. In fact, it is really evident that the overlap between the two distributions demonstrates that significant results from Fisher's exact test are, in reality, insignificant, according to the empirical distribution.

For this reason, Bleazard *et al.* introduced a new statistical test also known as the *unbiased miRNA functional enrichment analysis*. This analysis belongs in a class

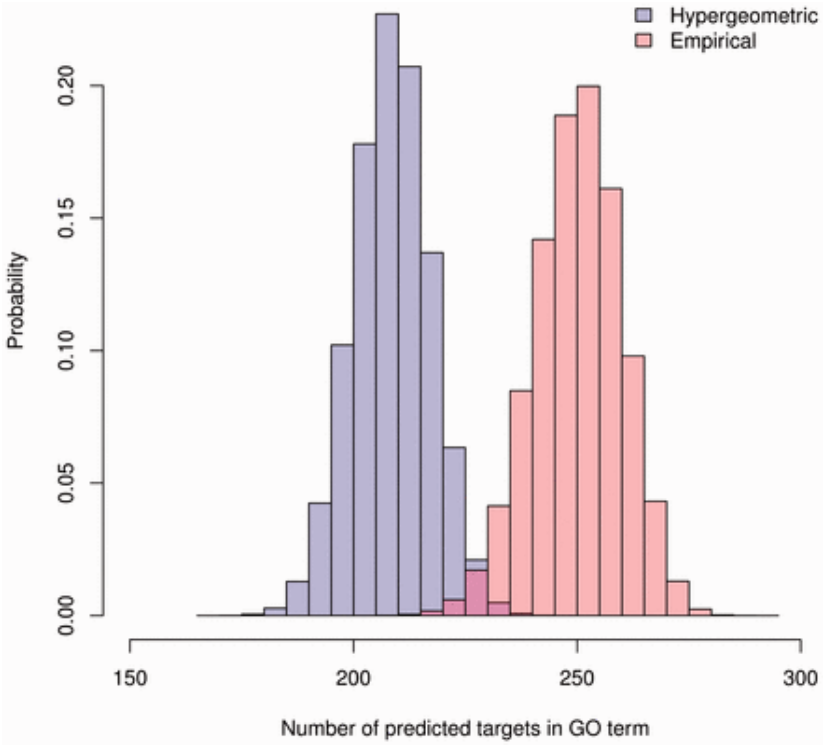



**Figure 2.2:** Mismatch between the hypergeometric and the empirical distribution [BLGJ15]

of problems called randomization tests [Sur11], wherein a statistical measure is calculated for a sample of interest and then compared against the same measure for a large number of randomly assembled samples with the same characteristics as the original sample (permutations). Ideally the number of permutations used should be equal to the total number of all possible permutations and this ensures the production of the observed (empirical distribution). Then the empirical p-value for the sample of interest is calculated relative to the position of the sample in the empirical distribution (i.e. the proportion of permutations that present a more favorable value for the statistic measure than the sample). If the sample belongs to the top 5% of the samples, then it has an empirical value of 0.05 and similarly to a normal p-value, the sample is considered statistically significant.

However, it becomes immediately apparent to the keen eye that the number of all possible permutations is usually substantially large, such that it is impossible to calculate in a sensible amount of time. This provides the motivation for scientists to utilize Monte Carlo simulations instead of exhaustive calculations. Thus a large number of randomly assembled permutations is usually selected. Unfortunately there is no foolproof way to theoretically calculate the randomly assembled permutations, since it depends mainly on the application. Generally, it should be large enough that it does not alter p-value significance between two repetitions of the same experiment (i.e.

the difference should be at least an order of magnitude smaller than 0.01) [ZVG+21].

Regarding the specific randomization proposed by Bleazard *et al.*, the statistic measure utilized is called the *GO term overlap* or *left-sided overlap* and it is defined as follows: given the set of targets of a miRNA group, denoted by $M$, and a gene class (category/pathway/disease), which is a set of gene participating in that class, denoted by $C$, the left-sided overlap is defined as

$$left - sided\ overlap = \frac{|M \cap C|}{|M|} \tag{2.2}$$

Given the left-sided overlap as measure, and a query which is the miRNA group of interest, the unbiased miRNA functional enrichment analysis consists of the following steps:

1. Calculate the left-sided overlap for the query.
2. Calculate the left-sided overlap for a large number (n) of miRNA randomly assembled miRNA groups, with the same size as the query.
3. Estimate the empirical p-value as the proportion of randomly assembled groups that present a larger overlap than the query group.

More formally:

$$empir.\ p - value = \frac{|\{M_j :\ overlap(M_j), C \geq overlap(M, C)\}|}{n} \tag{2.3}$$

where $M_j$, $j = 1, 2, \ldots, n$ are the random miRNA groups. Bleazard *et al.* proposed that $n$ should be set to 1 million in order to satisfy the required p-value accuracy.

It should finally be noted here, that it is more common for researches to test the significance of the query against multiple gene classes $C_i$ instead of only one class. This is done in order to discover other possible associations with the query or when there are multiple gene classes associated with the diseased state. The data sets described in Section 2.2 contain data for a large number of classes, ranging from a few hundred to more than 30K classes. This implies that this analysis is computationally intensive and indeed, such analyses require a few hours to complete [ZVP+17]. This fact provides the motivation for works that deal with the acceleration of this kind of analysis and such works are described in Chapters 3, 4 and 6.

# Chapter 3

# Data management techniques for miRNA enrichment

As mentioned in Chapter 2, the unbiased miRNA functional enrichment analysis is computationally intensive and as a result, execution times tend to range in the order of hours. This provides the motivation for the work presented in this chapter, namely our effort to accelerate the analysis, using more efficient data structures that achieve a significant speedup of up to an order of magnitude larger than the state-of-the-art.

Regarding the unbiased miRNA functional enrichment analysis introduced by Bleazard *et al.*, the number of random miRNA groups selected to perform the analysis is a parameter that controls the *accuracy* of the p-value to be produced. In particular, the higher the number of random miRNA groups selected, the more accurate the produced p-value will be. Usually, 1 million random groups are used to achieve sufficient accuracy [BLGJ15]. Unfortunately, using such a large number of groups results in unreasonably large execution times. For example, an execution of the state-of-the-art implementation [BLGJ15] for a group of 100 miRNAs as input, using 1 million random groups, on a single core of an Intel i7-3820 processor requires up to 17 hours of processing time.

In order to alleviate this issue, we introduce `BUFET` (Bitset-based Unbiased miRNA Functional Enrichment Tool). This approach exploits efficient data structures to significantly reduce the execution time of the unbiased enrichment analysis. `BUFET` also takes advantage of parallel computing techniques to achieve additional performance improvements in multi-core systems. The contribution of this work can be summarized in the following:

- We studied the computational requirements and examined the performance bottlenecks of the unbiased miRNA functional enrichment analysis.
- We investigated the performance of different data structures, namely hash ta-

bles and bitsets, in regards to their effectiveness in unblocking the identified bottlenecks.

- We developed `BUFET`, a tool that utilises the results of the aforementioned investigation to boost the speed of the unbiased miRNA functional enrichment analysis. To achieve an even greater speed boost in the case of multi-core environments, we exploited multithreading to implement parallel execution of the analysis.

- We performed an extensive evaluation of `BUFET` to demonstrate its efficiency. `BUFET` outperforms the state-of-the-art approach in all scenarios (in many cases by an order of magnitude).

- We provide `BUFET` as an open source implementation, which is freely available on GitHub (see the "Availability and requirements" section). `BUFET` is a powerful tool that provides flexible input file formats enabling many execution modes (e.g., execution using custom miRNA-gene interactions and gene annotations).

## 3.1 Introduction

As mentioned previously, the unbiased miRNA functional enrichment analysis involves the examination of a large number of biological processes (or, equivalently, annotation categories) to identify those, which are more likely to be affected by the gene-targets of a miRNA group. During this type of analysis, both biological processes and miRNAs are represented as gene sets: each biological process is represented by the genes involved in it, while each miRNA by its gene-targets.

It becomes evident that computing the biological process overlap of a miRNA group (see the Background section) involves the calculation of the *intersection* between the set of genes targeted by the miRNA group and the set of genes involved in the biological process. Moreover, the set of genes targeted by each miRNA group needs to be calculated "on the fly" by performing *union* operations on the gene sets of each miRNA in the group. Therefore, the unbiased miRNA functional enrichment analysis relies on performing a very large number of set unions and intersections. For instance, for a given query miRNA group of size 10, about 10 million unions and more than 8 billion intersections are required to produce a p-value.

The state-of-the-art implementation of the unbiased miRNA functional enrichment analysis [BLGJ15] uses hash tables (more specifically, Python sets[1]) to represent gene sets. The advantage of this data structure is that performing union and

[1] https://docs.python.org/3/tutorial/datastructures.html

intersection operations for small sets is usually very fast. Both operations are performed by executing a variant of the `hash-join` algorithm [FBY92]. On the other hand, `hash-join` becomes very inefficient when operating on large sets.

Unfortunately, in the case of the unbiased miRNA functional enrichment analysis, all union operations are performed on large gene sets. This is attributed to the fact that each of these gene sets corresponds to the predicted targets of a particular miRNA. Since miRNA target prediction algorithms usually produce hundreds or even thousands of results (interactions) for a single miRNA, it becomes evident that most of the performed union operations can be quite slow if `hash-join` is used.

To overcome this problem, the *bitset* (or *bit-vector*) [FBY92], an alternative data structure, which is more suitable for the representation of large sets, can be used. When sets of genes are implemented as bitsets, unions and intersections between them can be calculated by performing bitwise operations on bit blocks. In particular, `bitwise-or` can be used to get the union of two sets, while `bitwise-and` to get their intersection. Such operations are efficient for large sets, since their execution time is not affected by the size of the set[2]. Additionally, the representation of gene sets as bitsets is more efficient, memory-wise, in the case of relatively large sets of genes (like those produced by miRNA target prediction algorithms).

The calculation of the targets of each miRNA group would benefit greatly by the use of `bitwise-or`, since, as previously mentioned, it involves a large number of union operations on large gene sets represented by dense bitsets. In this case, bitsets also have a reduced memory footprint compared to hash-tables. On the other hand, gene sets related to biological processes, as provided by Gene Ontology annotations [ABB+00], usually consist of a small number of genes. Therefore, `hash-join` on these sets can be rather efficient[3].

The previous discussion suggests that a *hybrid solution*, using bitwise operations for unions and `hash-join` for intersections, seems more suitable than both of the aforementioned approaches. Unfortunately, this hybrid approach has a major drawback. The gene sets generated by `bitwise-or` for all miRNA groups must be provided as input to the `hash-join` algorithm for the calculation of the biological process overlaps. However, the `bitwise-or` algorithm produces gene sets represented as bitsets, while `hash-join` requires its input in the form of hash tables. Therefore, a data structure conversion must be performed, introducing an important execution overhead that counterbalances any gains in efficiency.

[2]In particular, the execution time of each bitwise operation depends on the number of bits it contains, i.e., on the cardinality of the set's domain.

[3]Regarding the calculation of the biological process overlap, an additional optimization is possible for the `hash-join` algorithm. In particular, the production of the output intersection set can be avoided, since only its size is required. However, a similar optimization is not feasible for `bitwise-and`.

## 3.2 The BUFET approach

Our approach, called `BUFET`, is demonstrated in Figure 3.1. It combines the best characteristics of bitset- and hash-table-based methods without suffering from the aforementioned shortcomings of a hybrid approach. It takes advantage of the efficiency of `bitwise-or` in calculating the union of large sets to produce the gene sets targeted by particular miRNA groups. These gene sets are represented as bitsets, called *miRNA group bitsets*.

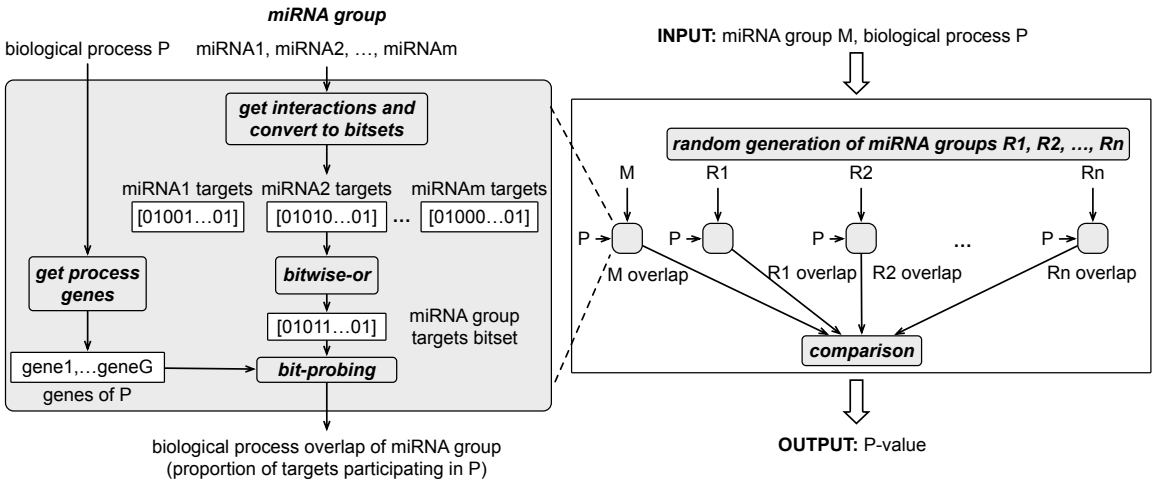



**Figure 3.1:** Flowchart summarizing the BUFET approach

Meanwhile, the biological process overlap of each miRNA group is calculated as follows: for each gene annotated as part of the biological process, the respective bit in the miRNA group bitset is examined. If the bit is set, then the value of a counter is increased by one (its value is initially zero). Otherwise, the value of the counter remains intact. After all genes related to the biological process have been considered, the value of the counter provides the size of the intersection and, subsequently, the biological process overlap. Since the genes are used to probe the miRNA group bitset, we refer to this method as `bit-probing`.

Further optimisations were introduced in order to achieve additional performance improvements. First, biological processes that have no common genes with the miRNA group under examination can be excluded from the analysis (since no interference by the miRNAs in the group with the process is recorded). Additionally, `BUFET` supports full utilization of multi-core computing systems by supporting parallelization at the biological process level.

It should be noted that parallel execution is also supported by the state-of-the-art approach presented in [BLGJ15]. However, in contrast to the use of multiprocessing

adopted by this approach to implement parallelization, BUFET uses multithreading. The advantage of multithreading over multiprocessing is that all processes running in parallel have access to the same part of the main memory. This eliminates the need to copy data across processes, thus reducing the execution time and memory footprint.

On the other hand, an issue with this approach is that the bitsets containing the targets of the random miRNA groups have to be calculated and stored in main memory. This step is necessary, so that every thread is able to access the data in order to calculate a p-value. Consequently, this increases the memory footprint, although, the amount of memory required does not pose a big challenge for contemporary computers. More specifically, none of the many real-world analysis scenarios examined during our experiments resulted in the allocation of more than 3.5 GB of RAM to our script.

BUFET is provided as a free, open source software licenced under GPL v3 (a download link is provided in the "Availability and requirements" section). Its core is implemented in C++ for greater efficiency, while a Python wrapper script facilitates its execution and its incorporation in existing bioinformatics workflows.

The input of the BUFET software consists mainly of two CSV files: one containing miRNA-to-gene interactions and another containing associations of biological functions with particular genes. The proper format of these files is described in the software download page. It should be noted that BUFET provides flexibility, enabling the users to upload miRNA-to-gene interactions based on the prediction algorithm of their choice (e.g., TargetScan [LBB], DIANA-microT [PGK+13][RMA+12], miRanda [JEA+04], etc.) and to use biological function annotations collected by their preferred source (e.g., GO [ABB+00], KEGG [KSK+16][KG00a] or PANTHER [TCK+03]).

Finally, BUFET also performs *Benjamini-Hochberg FDR correction* [YB95]. More specifically, following the method in [McD14], we assume that 5% (and 1%) of the produced p-values (under the 0.05 threshold) are false positives, while the rest are significant results. P-values significant at FDR 0.05 are marked with "*" while p-values significant at 0.01 are marked with "**" in the output file.

## 3.3 Experimental evaluation

In this section, the efficiency of BUFET is evaluated against that of the state-of-the-art implementation (EmpiricalGO[4]), in both single- and multi-core environments. First, we examine the effect of the miRNA group size on the execution times of both

[4] http://sgjlab.org/empirical-go/

implementations. Next, we investigate their parallel behavior for a varying number of CPU cores. miRNA-to-gene interactions were collected from DIANA-microT-CDS (score threshold=0.8) and miRanda (score threshold=155 and free energy=−20), while GO annotation data were obtained from Ensembl. Statistics related to miRNA-to-gene-interactions data used are presented in Table 3.1. All experiments were executed on a machine powered by an Intel Core i7-3820 processor with 8 cores (4 physical) and 64 GB of main memory.

| | Number of genes/miRNA | | | | | Total miRNAs |
|---|---|---|---|---|---|---|
| | Min. | Max. | Avg. | Median | Std. Dev. | |
| microT | 1 | 4547 | 404 | 206 | 459 | 2580 |
| miRanda | 11 | 6977 | 1309 | 1096 | 932 | 2588 |

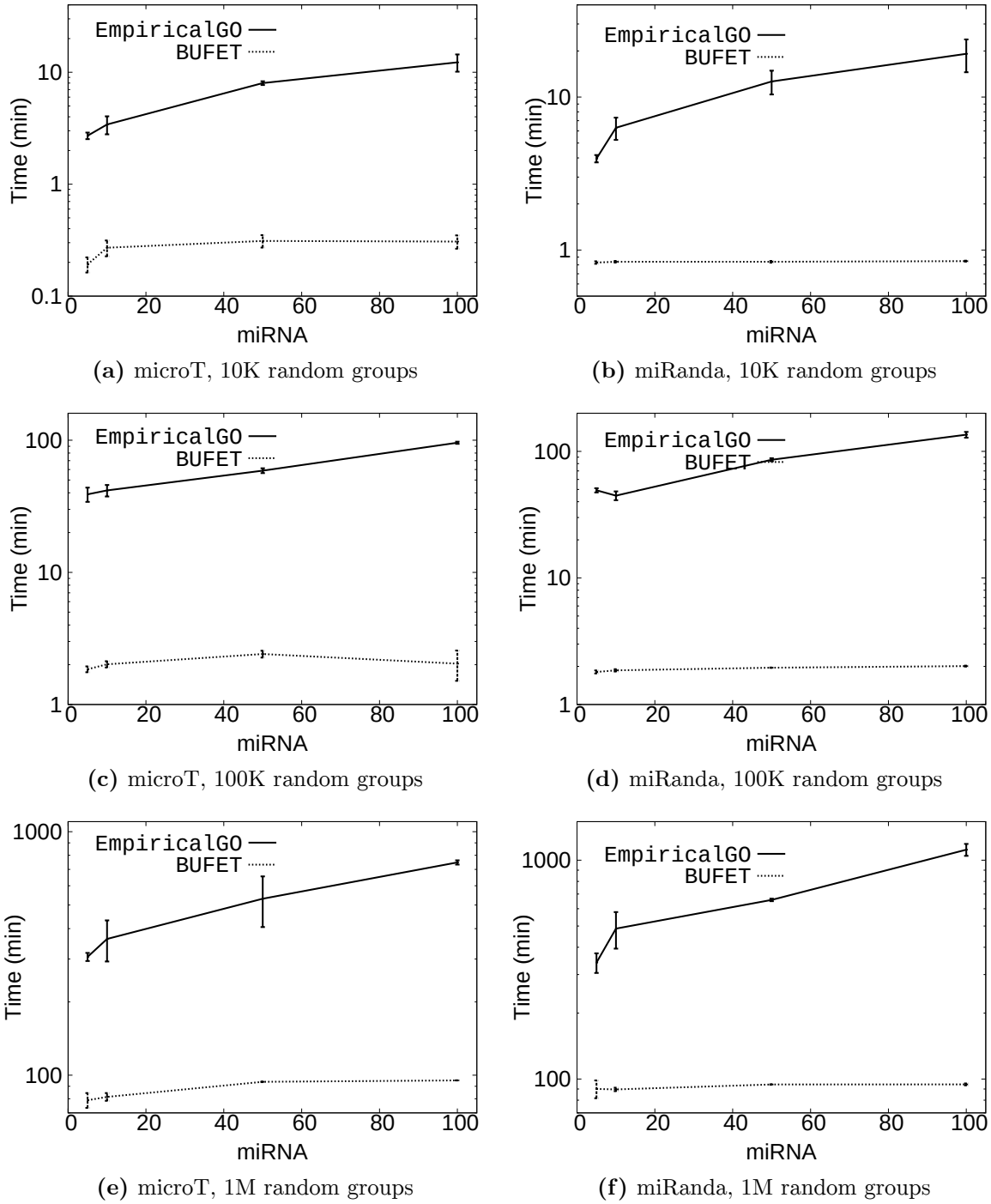
**Table 3.1:** Statistics related to the miRNA-to-gene interactions used

### 3.3.1 Varying the miRNA group size

Figure 3.2 presents (a) the average execution time of `BUFET` and `EmpiricalGO` and (b) its standard deviation (using error bars) for each measurement point and for varying miRNA group sizes (5, 10, 50 and 100 miRNAs) in a single-core environment. For each miRNA group size, 10 different groups were used as input to both implementations. Thus, every reported execution time is the average of 10 executions. The left column corresponds to the experiment performed using DIANA-microT-CDS interactions, while the right to the one using miRanda interactions. We performed each experiment by selecting the following, commonly-used settings: 10 thousand (10K), 100 thousand (100K), and 1 million (1M) random miRNA groups. Since the difference in the execution times between `EmpiricalGO` and `BUFET` are very large, all diagrams are presented in log scale for the y axis to enhance legibility.

It is clear that the execution time increases as the number of miRNAs in the group under examination increases for both approaches (due to the larger number of union operations that have to be performed). However, it is evident that the rate of the increase in the execution time is larger for `EmpiricalGO` than `BUFET`. This can be attributed to the fact that `BUFET` exploits the efficiency of `bitwise-or` in calculating unions on large gene sets. It also becomes evident that `BUFET` scales better than `EmpiricalGO` and in some cases, it is faster by at least an order of magnitude. Therefore, `BUFET` is a very efficient approach when high accuracy is needed for functional analysis of large miRNA groups.

Figure 3.3 shows the same experiments in a multi-core environment (7 cores were used). Note that the main trends observed in the single-core experiment continue to




**(a)** microT, 10K random groups

**(b)** miRanda, 10K random groups

**(c)** microT, 100K random groups

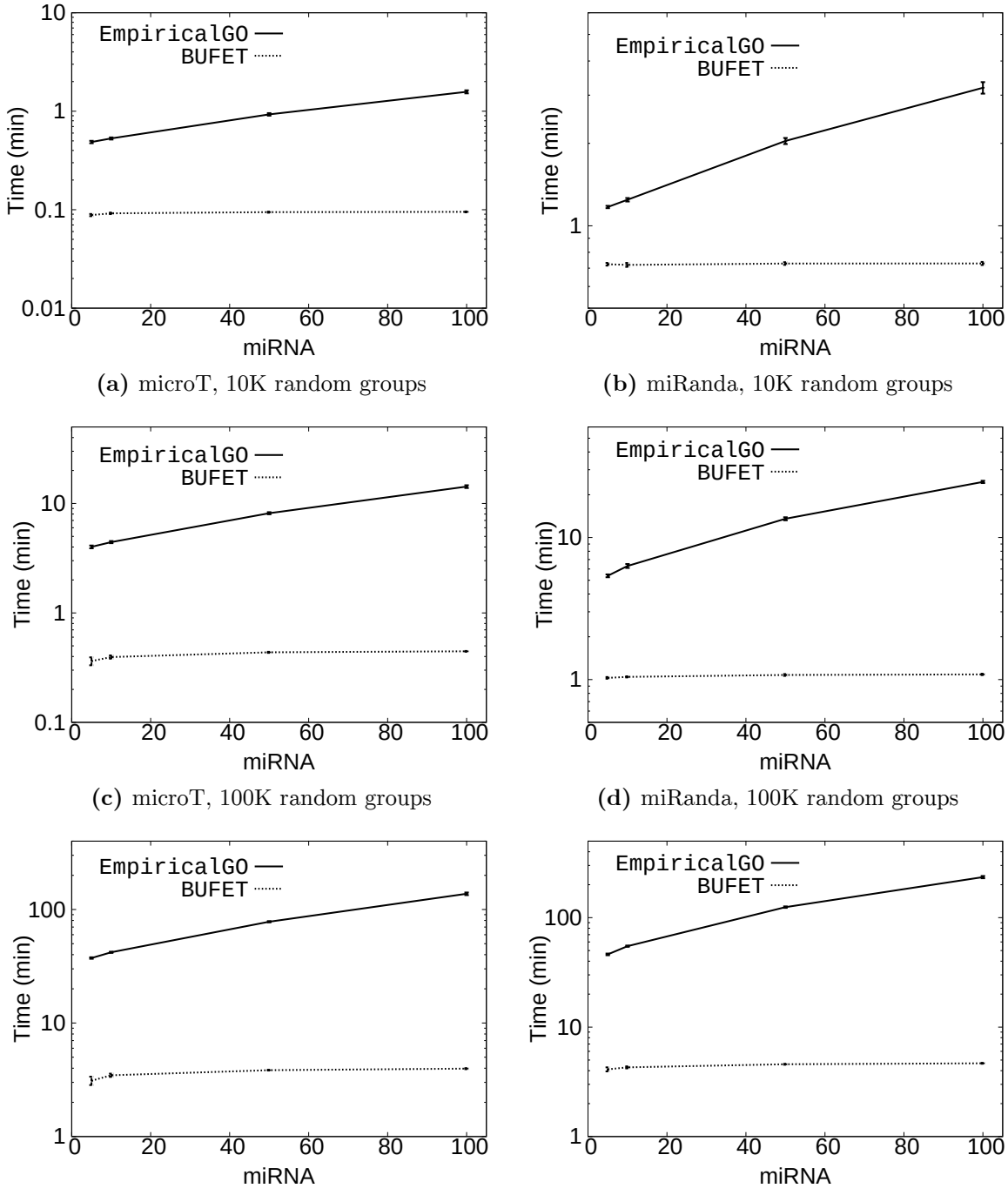
**(d)** miRanda, 100K random groups

**(e)** microT, 1M random groups

**(f)** miRanda, 1M random groups

**Figure 3.2:** Average execution times (log scale) on a single core with a varying number of miRNAs

occur: increasing the miRNA group size leads to increased execution times for both methods, while `BUFET` is significantly more efficient than `EmpiricalGO` in all cases. Note that, for the case of 5 miRNAs in low accuracy mode, the execution times tend




**(a)** microT, 10K random groups

**(b)** miRanda, 10K random groups

**(c)** microT, 100K random groups

**(d)** miRanda, 100K random groups

**(e)** microT, 1M random groups

**(f)** miRanda, 1M random groups

**Figure 3.3:** Average execution times (log scale) on 7 cores with a varying number of miRNAs

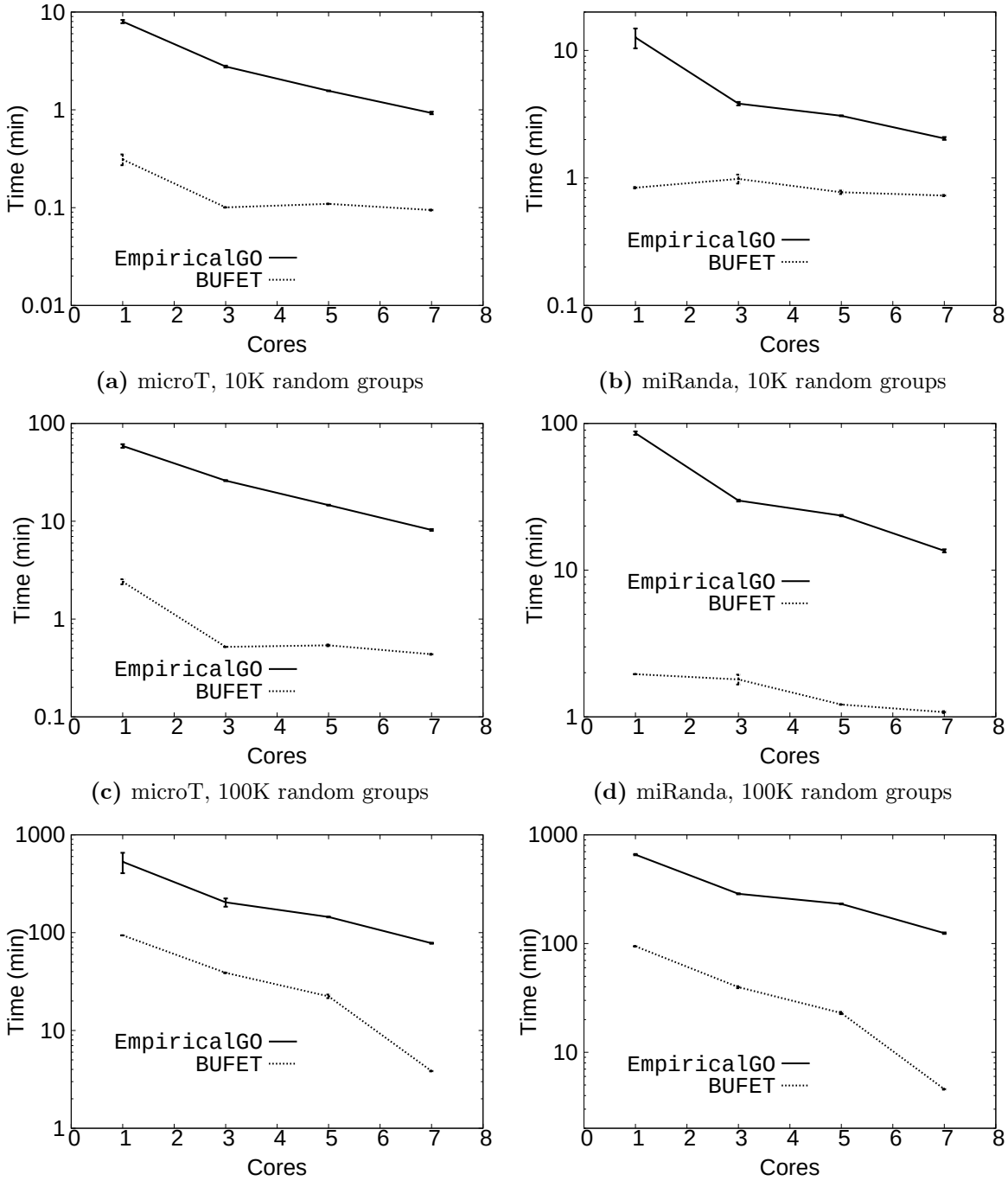
to converge to the time needed for serial operations (i.e. file reading, output writing, and FDR correction). Finally, it is worth mentioning that, in the case of 100 miRNAs using 7 cores, in high accuracy mode, `BUFET` can produce results in under 5 minutes, while `EmpiricalGO` needs more than 7 hours for the same task.

### 3.3.2 Varying the number of cores

Figure 3.4 shows the average time required by `BUFET` and `EmpiricalGO` to calculate the empirical p-values for 10 input groups of size 50 by using a varying number of CPU cores. It is clear that both approaches become faster as the number of cores increases. However, in every case `BUFET` requires significantly less time to execute.

## 3.4 Conclusions

In this chapter we dealt with the performance of the unbiased miRNA functional enrichment analysis. We showed that the state-of-the-art approach to perform this type of analysis (EmpiricalGO) is not practical in terms of computational efficiency, especially for large miRNA groups when high accuracy is required. To deal with this problem we introduced `BUFET`, an alternative bitset-based approach. Our experiments make evident that `BUFET` outperforms the state-of-the-art implementation in all scenarios (in many cases by orders of magnitude). Additionally, the better scalability of `BUFET` makes it a very appealing solution for the analysis of large miRNA groups when 1 million random groups are used for the analysis. Note that, `BUFET` is provided as an open source implementation which is freely available on GitHub (the download URL is provided in the Availability and requirements section).




**(a)** microT, 10K random groups

**(b)** miRanda, 10K random groups

**(c)** microT, 100K random groups

**(d)** miRanda, 100K random groups

**(e)** microT, 1M random groups

**(f)** miRanda, 1M random groups

**Figure 3.4:** Average execution times (log scale) varying the number of cores

# Chapter 4

# Data indexing and optimization for miRNA enrichment

In this chapter, we introduce another work that aims to speed up the unbiased miRNA functional enrichment analysis at an even greater rate. This is achieved by using an index called *Frequent Itemset Index* (FII) to remove redundant bit-probing operations (see Section 4.3.1; by utilizing another index called *Significance Level Index* (SLI) to predict whether an association is potentially statistically significant leads to an even further reduction in execution times (up to an order of magnitude).

## 4.1 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* [Fis92]. 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 [BLGJ15]. 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 [ZVP+17].

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 4.4).

- We provide open-source implementations[1] of all described approaches.

[1] https://github.com/diwis/fii-sli

## 4.2 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 4.1). 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 [chi11], the Cochran–Mantel–Haenszel test [cmh03], etc). In the remainder of this chapter, we refer to them as *association testing methods*.

The most popular of them, *Fisher's exact test* [Fis92], 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"* $\mathbb{A}$ of related classifications (let them be the *query classifications*, denoted as $A_1, A_2, \cdots \in \mathbb{A}$). 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 [BLGJ15]. 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.

### 4.2.1 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 than 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 4.2) something that modifies, among others, the way population items are being classified. For example, in [BLGJ15] 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 $A \in \mathbb{A}$ 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 $A_1, \ldots, A_n \in \mathbb{A}$ have with $B$, for very large numbers of $n$. According to that, when the association between two classifications $A \in \mathbb{A}$ 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 [BLGJ15], the *one-sided overlap* between two classifications (let them be $A$ and $B$) is used, that is defined as follows:

$$overlap(A, B) = \frac{sizeof(A \cap B)}{sizeof(A)}$$

For the remainder of this chapter, we assume that the same definition of overlap is used. Under this assumption, based on the previous discussion, the empirical p-value

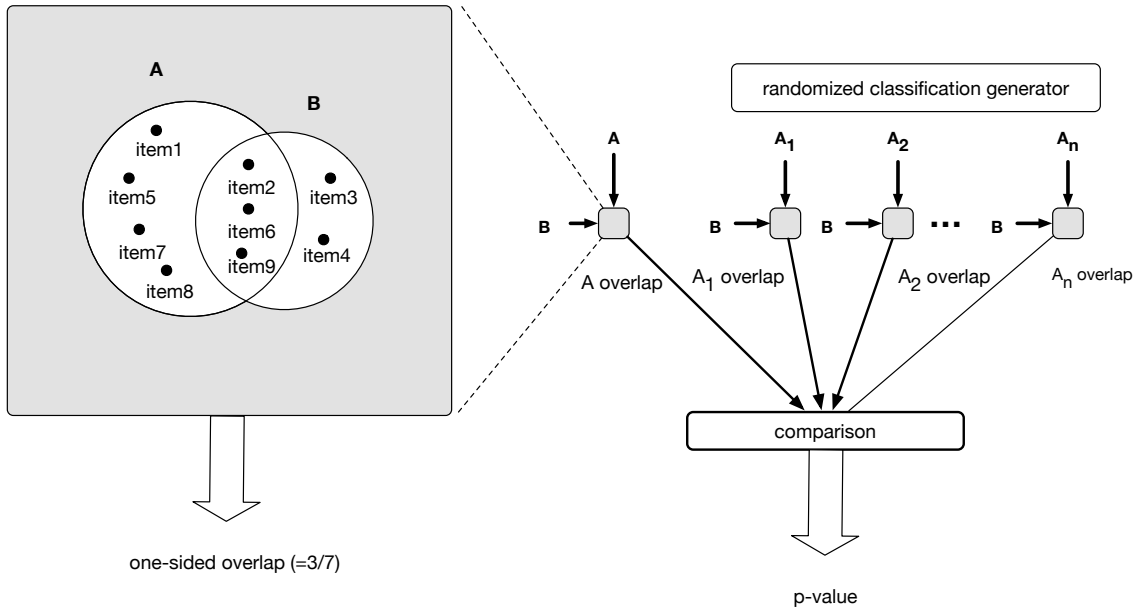



**Figure 4.1:** Association randomization test using one-sided overlap

could be formalised as follows:

$$p - value = \frac{sizeof(\{A_j : overlap(A_j, B) \geq overlap(A, B)\})}{n}$$

Figure 4.1 illustrates the process of a such randomisation test.

### 4.2.2 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 4.2), in practice, researchers are interested to examine the association of a query classification $A \in \mathbb{A}$ 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 [ZVP+17].

## 4.3 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 4.3.1) 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 4.3.2) 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.

### 4.3.1 The Frequent Itemset Index (FII) Approach

Calculating the (one-sided) overlap between two classifications is a core task performed multiple times during a randomization test (see Section 4.2.2). 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 [ZVP+17]. 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.

#### 4.3.1.1 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 4.3 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.

#### 4.3.1.2 Frequent Itemsets Identification

Regarding the first step (frequent itemsets identification), executing the Apriori algorithm [AMS+96] (with threshold $sup\_thr = 2$) 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 [Bor12], we tried to produce the maximal itemsets with support threshold$\geq 2$ for the datasets in Section 4.4.1, 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 $sup\_thres = 2$. 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 4.3.1.3) 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 4.2). 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 $O(k^2)$. 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.

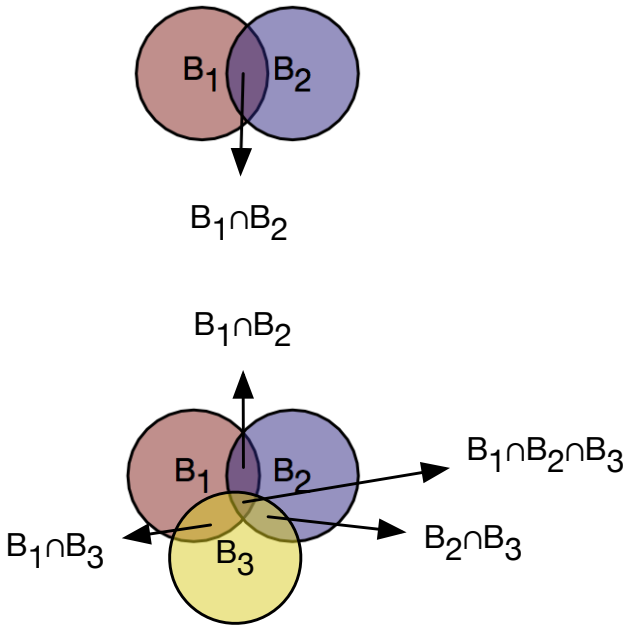



**Figure 4.2:** Venn diagrams showing the overlap between transactions

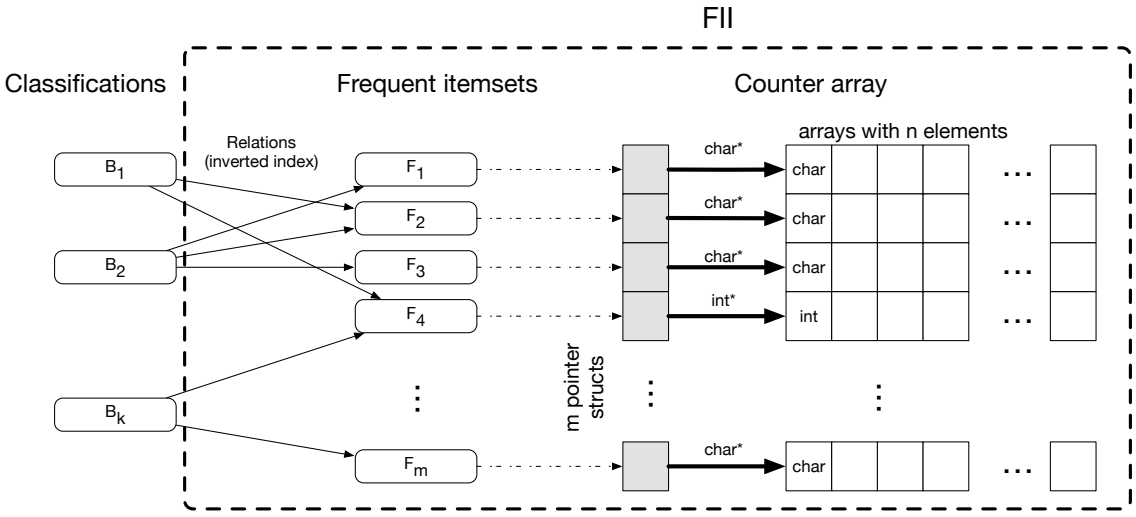



**Figure 4.3:** Structure of the FII

#### 4.3.1.3 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 [Kar72]. 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 $F_{max}$, for which $F_{max} \subseteq B_i$, 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 items 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.

#### 4.3.1.4 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 $sup\_thr > 2$. In order to do that, we first calculate the support for each of the itemsets produced by our approach for $sup\_thr = 2$; then we discard all itemsets that have a $support < sup\_thr$ and use the rest. This means that the greater the value of $sup\_thr$ 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 $sup\_thr$ values both to the memory footprint and to the execution time of the FII approach (see Section 4.4.3).

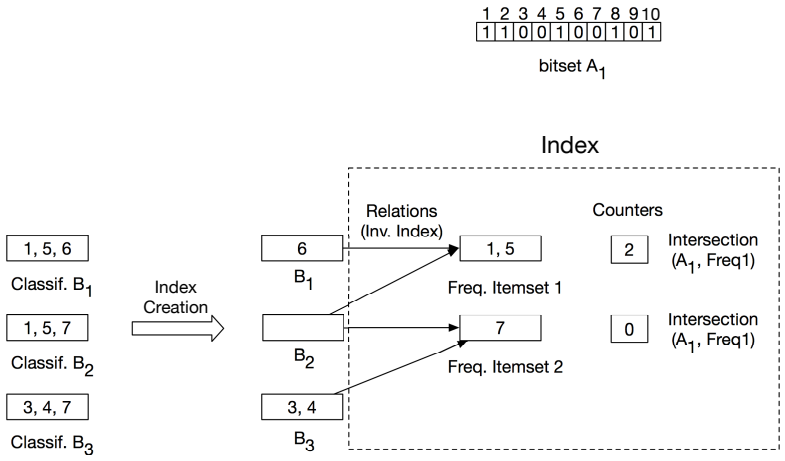



**Figure 4.4:** Example for the creation and use of the FII

#### 4.3.1.5 A toy example

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

**Example 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* apriori *algorithm with $B_1$, $B_2$ and $B_3$ as transactions and support threshold* 2 *produces two frequent itemsets: $Freq_1$ containing elements* 1 *and* 5 *and $Freq_2$ containing item* 7. *The contents of $Freq_1$ and $Freq_2$ are removed from $B_1$, $B_2$ and $B_3$. Furthermore relations, which are stored in an inverted index, are also created: $Relations[B_1] = \{Freq_1\}$, $Relations[B_2] = \{Freq_1, Freq_2\}$, $Relations[B_3] = \{Freq_2\}$. Moreover, two counters which contain the size of the intersection between $A_1$ and $Freq_1$ as well as the intersection between $A_1$ and $Freq_2$ 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 $Freq_1$ to the intersection size. However, for $B_2$ we only need to add the counters for $Freq_1$ and $Freq_2$ and finally, for $B_3$ we add the counter for $Freq_2$ 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 4.4 depicts the index described above in regard to the previous example.*

### 4.3.2 The Significance Level Index (SLI) 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 $A \in \mathbb{A}$ is significantly associated with a particular ground classification $B$. In the context of the randomization tests described in Section 4.2.1, 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 $A_j$.

It is evident that the lowest overlap in the top 5% of overlaps between ground classification $B$ and all query classifications $A_j$ 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 $A_1, \ldots, A_n \in \mathbb{A}$. 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 $A'_1, \ldots, A'_n \in \mathbb{A}$ 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 4.3.2.3.

#### 4.3.2.1 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 4.5 demonstrates an example of the use of the SLI index.

**Example 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.*

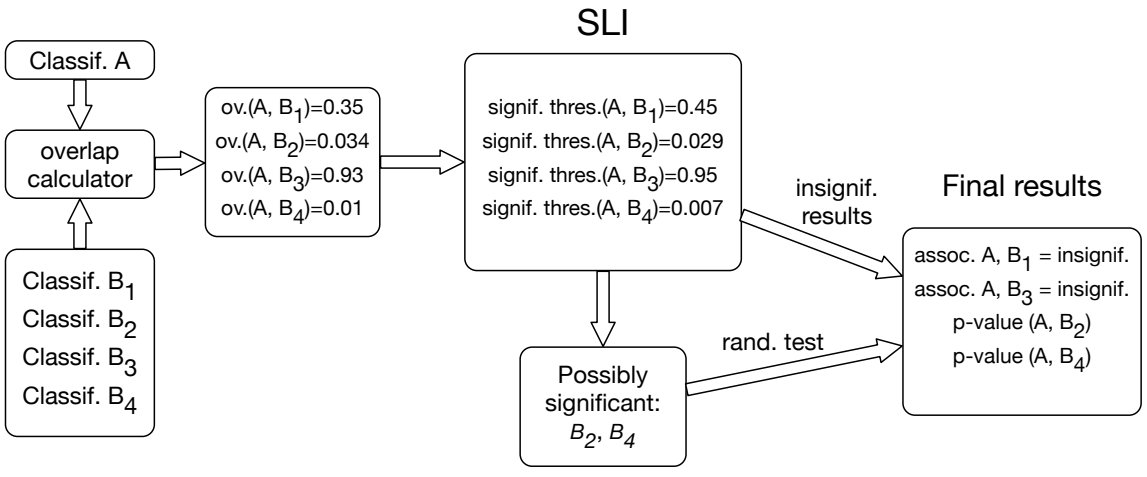



**Figure 4.5:** Example of the use of the SLI

In order to create the SLI, we use the approach described in Section 4.3.1 (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 $x * 100\%$ of overlaps (for a discussion on the proper selection of $x$, see Section 4.3.2.2), 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.

#### 4.3.2.2 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 $A_i$ 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 $ov_thr$ 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 $k = ((|dev - x|) * n)$ 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 de-

viation in order to guarantee that the SLI will not mark significant associations as insignificant. The effectiveness of this method is evaluated in Section 4.4.4.

#### 4.3.2.3 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.

## 4.4 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 [BLGJ15, ZVP+17], 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 [BLGJ15] 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.

### 4.4.1 Datasets

In experiments, we used three openly accessible life-sciences datasets as ground classifications $B_i$:

- *Gene Ontology (GO) [ABB+00, The18].* 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* [YAA+19] for version 84.

- *DisGeNET [PRASP+19b].* This dataset was retrieved from DisGeNET, which is one of the largest and comprehensive repositories of human gene-disease associations. We used *DisGeNET version 5* annotations.

- *MeSH.* This dataset maps genes to *Medical Subject Headings* [ROG63]. The gene mappings were retrieved from the REST API of Gene2Mesh [AS07].

Regarding the query classifications $A_j$ (miRNA-gene interactions) we used the microT dataset, with an interaction score threshold of 0.8, which we produced by using MR-microT [KVS+14a] for Ensembl version 84.

### 4.4.2 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 4.1. It is

| Method | Time (nsec) |
|---|---|
| bit-probing (sparse) | 1.884 |
| bit-probing (medium) | 6.254 |
| bit-probing (dense) | 7.016 |
| addition (character) | 0.3665 |
| addition (integer) | 1.461 |

**Table 4.1:** Average time required for addition operations and bit-probing operations on different data types and variable densities.

easy to see that in all cases, array additions are faster than bit-probing operations.

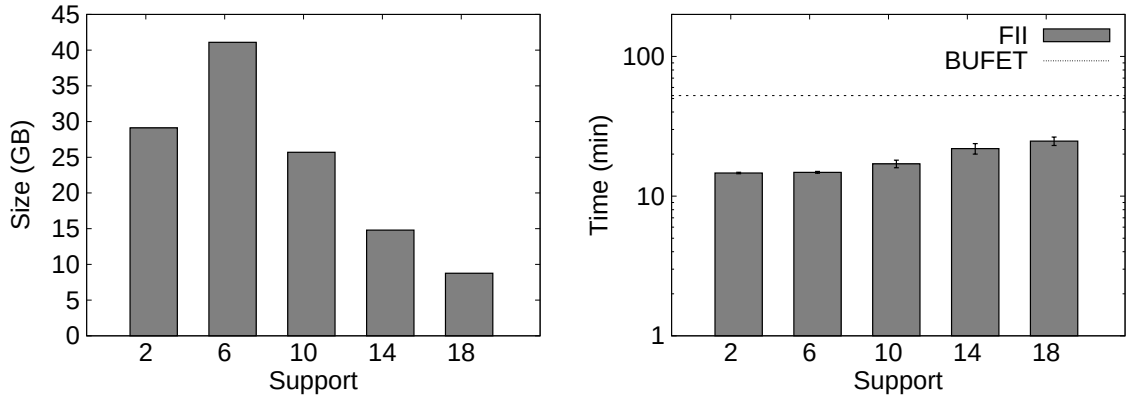
Furthermore, regarding the three datasets we are using for the evaluation of our approach, we found that the vast majority of itemsets produced by our method contain less than 255 elements.

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 4.4.5).

### 4.4.3 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 4.3.1.3 and 4.3.1.4. More specifically, we set the support threshold to 2, 6, 10, 14 and 16 and selected 10 different inputs of 39 miRNAs (query classification $A$) 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 4.4.5). The results can be seen in Figure 4.6.

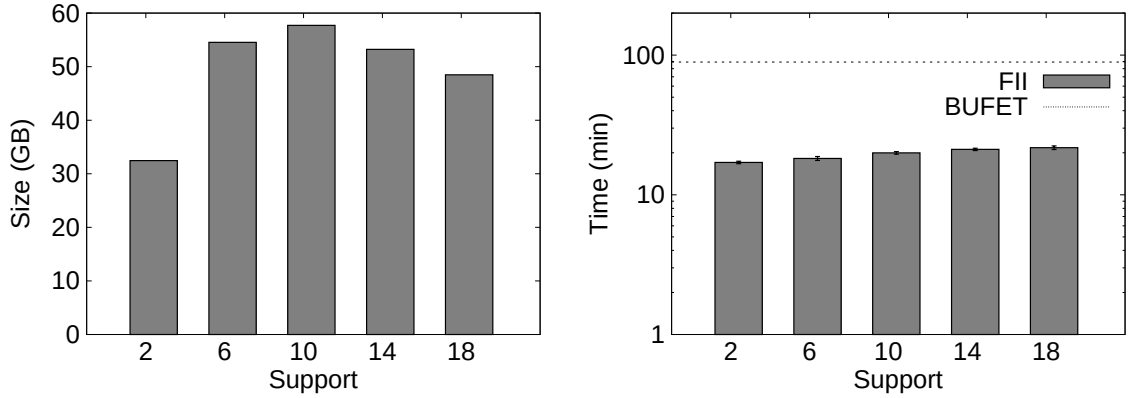
We can see that the performance of the index starts to decline as the support threshold increases, which is to be expected, since more operations that are slower (bit-probes) are performed. As we increase the support threshold even more, our approach should present with slightly higher execution times than the state-of-the-art.




**(a)** Index size (GO)



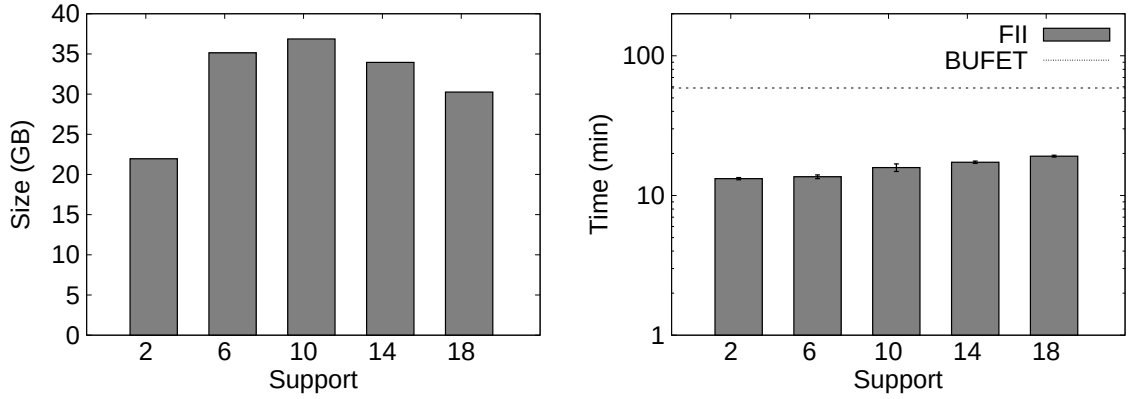

**(b)** Execution time (GO)




**(c)** Index size (DGN)




**(d)** Execution time (DGN)




**(e)** Index size (MESH)




**(f)** Execution time (MeSH)

**Figure 4.6:** Index size and execution time (logscale) vs the itemset support threshold

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.

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

In this experiment we use the method described in Section 4.3.2.2 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 4.2.

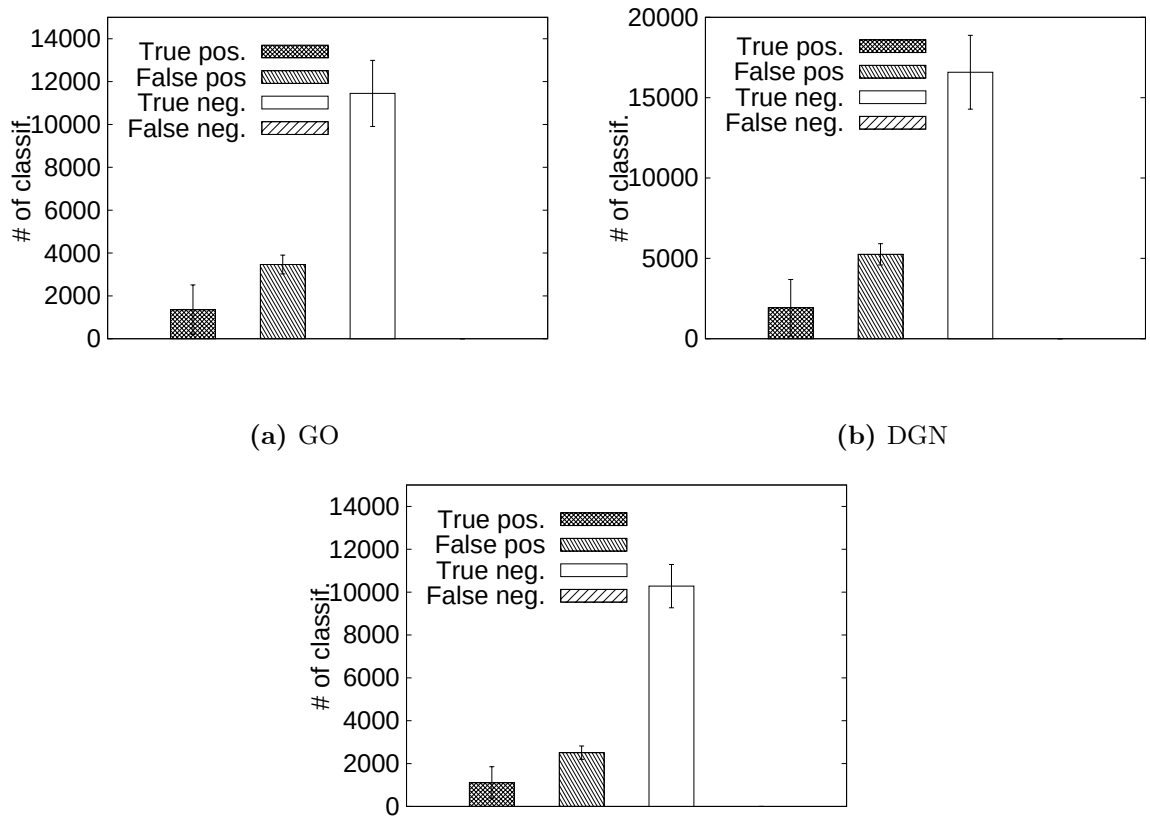
| Dataset | % Max. st. deviation |
|---|---|
| GO | 0.00105 |
| DisGeNET | 0.000997 |
| MeSH | 0.00105 |

**Table 4.2:** P-value maximum standard deviation for different inputs for the three datasets

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 4.7.



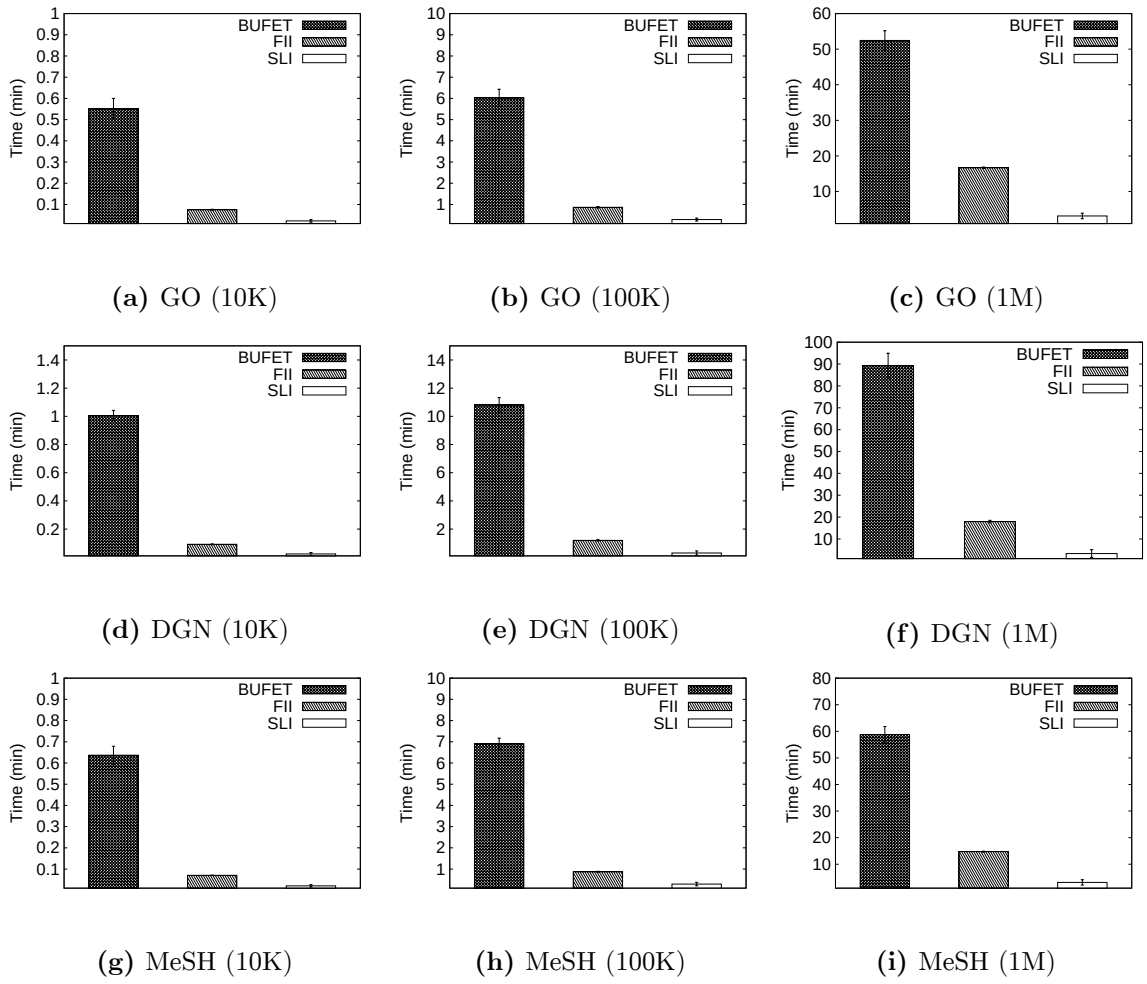

**(a)** GO

**(b)** DGN

**(c)** MeSH

**Figure 4.7:** Filtering performance of the SLI

It is easy to notice in Figure 4.7 that the standard deviation for all types of ground classifications datasets is very large, since the number of filtered and unfiltered categories depends on the one-sided overlap of the input set of microRNAs (query classification $A$). However we can see that the number of associations being filtered out by the SLI are more than double on average compared to those for which the randomization test is run. Additionally, we can see that the SLI produces zero false negatives results for all cases which is important, because it guarantees that we do




**(a)** GO (10K) **(b)** GO (100K) **(c)** GO (1M)

**(d)** DGN (10K) **(e)** DGN (100K) **(f)** DGN (1M)

**(g)** MeSH (10K) **(h)** MeSH (100K) **(i)** MeSH (1M)

**Figure 4.8:** Comparison of our two approaches with the state-of-the-art

not miss actually significant results.

### 4.4.5 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 [ZVP+17]. 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 $A_1, \ldots, A_n \in \mathbb{A}$), namely 10K, 100K and 1M. Moreover, as ground classifications, we used the 3 datasets outlined in Section 4.4.1.

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 4.4.4, it produces no false negatives and thus, the output p-values of the test will be reliable. The results can be seen in Figure 4.8.

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

## 4.5 Brief history of association testing and randomization tests

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 [Arb10] and Pierre-Simon Laplace [BJ07], 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 [chi11], William Sealy Gosset developed the Student's t-test [Stu08] and Ronald Fisher developed Fisher's exact test [Fis92].

Randomization tests received attention in the 1800s with the work of C. S. Peirce [PJ84] and the are very popular in clinical trials and life-sciences in general [Sur11, BLGJ15]. However, since randomization tests are computationally expensive there have been attempts to make them faster [GBIH+17, WRD+16, ZVP+17] in order to allow researchers to run more tests and gain more insight into the mechanisms of life in a shorter amount of time.

## 4.6 Conclusion

In this chapter 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 $B_i$ 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).

# Chapter 5

# Towards higher-quality unbiased miRNA enrichment

In this chapter, we demonstrate that when the bias in miRNA functional enrichment analysis is removed, by using experimentally validated miRNA targets, then a new bias, related to gene annotations, is made visible. Additionally, we illustrate that this occurs due to a bias affecting the gene-to-biological-function annotations and is not accounted for by the unbiased enrichment analysis proposed by Bleazard *et al.*, resulting in reduced sensitivity. To alleviate this issue, we introduce a new statistical measure that increases the sensitivity of the unbiased enrichment analysis. This measure accounts for both sources of bias (predicted interactions and gene annotations) and is more sensitive when predicted or experimentally validated targets are used. Finally, we introduce BUFET2, a new version of BUFET that calculates p-values using both statistical measures. BUFET2 is available as source code and Docker image. Additionally, it is also accessible through a REST API compute server.

## 5.1 Background

Regarding the bias introduced by miRNA target prediction algorithms, one contributing factor is the limited amount of validated positive miRNA:target interactions, and more importantly the virtually non-existent validated negative interactions, which severely affect the training of robust and highly accurate target prediction algorithms ([KPC+17b]). To make matters worse, most target prediction algorithms have been trained on seed-enriched data sets with features extracted from the sequence surrounding the seed, even though recent evidence shows that non-seed-based interactions are common in miRNA-mediated gene expression regulation ([LKC+12]). The inefficient selection of negative samples for the training process is often unavoidable due to the lack of validated negative miRNA:target interactions in

the literature. Additionally, the process of experimentally validating miRNA binding sites is frequently driven by target prediction algorithms. An experimentalist will often apply a target prediction algorithm on the under-study gene, select the top target and carry out the validation process. Negative results are usually not mentioned while the published positive interactions are inevitably enriched in seed-based binding types. The whole back and forth between miRNA target prediction results and experimentally-driven interaction validation results in a vicious circle that inflates miRNA functional enrichment analyses with false positive target genes. This, inevitably introduces a source of bias that invalidates the assumption made by the hypergeometric distribution, that genes are targeted by miRNAs in a uniform fashion. Moreover, Bleazard *et. al* illustrated that statistically significant results with the standard method did not remain so, after correcting for bias.

In order to eliminate the bias affecting the standard overrepresentation analysis, Bleazard *et al.* proposed a new method to perform miRNA functional enrichment analysis, that involves a randomization test. This test moves the analysis to the miRNA level instead of the gene level and it consists the following steps:

1. Given a miRNA group of interest (query) calculate a statistical measure relevant to the problem.

2. Create 1 million randomly assembled miRNA groups with the same size as the query and for each of them calculate the same statistical measure.

3. The empirical p-value is then defined as the proportion of randomly assembled miRNA groups that present a better statistical behaviour compared to the behaviour of the query (i.e. comparing the statistical measure of the miRNA groups to the group of interest).

The statistical measure introduced is called *GO term overlap* ([BLGJ15]) and it is defined as the proportion of genes targeted by a group of miRNAs, that are also members of a specific GO category. Let $A$ be the set of genes targeted by the group and $B$ be the set of genes that participate in the GO category. Then the GO term overlap, to which we are going to refer as *left-sided-overlap* from here on, is more formally defined as:

$$left\text{-}sided\text{-}overlap = \frac{|A \cap B|}{|A|} \tag{5.1}$$

Intuitively we expect that the left-sided overlap accounts for the bias introduced by target prediction algorithms, which is the main focus of [BLGJ15]. This is done by dividing the size of the overlap between the set of targets and the GO category by the number of the targets.

## 5.2 The BUFET2 approach

In this section we provide a detailed outline of each of the methodological changes proposed for the analysis as well as the rationale behind each of them.

### 5.2.1 Investigating experimentally validated miRNA targets

The aim of this study, is to show that a bias affecting miRNA functional enrichment analysis still persists even when experimentally validated miRNA targets are used, instead of predicted targets. To do this, we follow the same approach as Bleazard *et al.* to show that the empirical distribution of the overlaps does not match with the expected hypergeometric distribution. First, we selected miRNAs dysregulated in multiple sclerosis, from Table 5 in [Moh20] (see supplemental. material) and gene ontology (GO) annotations from Ensembl ([YAA+19]) version 100. Also we retrieved disease-to-gene associations from DisGeNET v7 ([PRASP+19c]), as well as pathway data from KEGG ([KG00b]). Moreover, we used the latest version of miRTarBase ([CSY+17]) at the time this article is written, in order to get a list of experimentally validated miRNA targets. We selected 14 of the dysregulated miRNAs that exist in the data set.

Initially, we randomized at the miRNA level, by creating 1 million randomly assembled miRNA groups containing 14 miRNAs each. For each of these groups, we calculated the gene members of each GO category in the data set, that are also targeted by the miRNAs in the group and created a histogram. On the same graph, we plotted the expected hypergeometric distribution for the overlaps, given the number of targeted/non-targeted genes and the number of genes belonging/not-belonging to the same GO term. The results indicate that the analysis still suffers from a bias related to the size of each gene class (see Section 5.3.1).

Based on the definition of the left-sided overlap, we can see that it essenially is the *percentage* (and not the absolute number) of targets that also belong to the gene class. Intuitively, this suggests that the left sided overlap is designed to remove the bias stemming from target prediction, because the intersection is normalized using the total number of genes being targeted by the miRNA group. In the next section we introduce a new metric, which is expected to account for both sources of bias.

### 5.2.2 Introducing the two-sided overlap

To eliminate the bias related to the size of each gene class, we introduced the Jaccard coefficient ([Han14]), to which we will refer as *two-sided overlap* from here on, as a metric for the randomization test:

$$two\text{-}sided\text{-}overlap = \frac{|A \cap B|}{|A \cup B|} \tag{5.2}$$

We argue that normalization taking into account the total number of genes involved between a gene class and a miRNA group will increase the overall sensitivity of the test and it should be the preferred method regarding miRNA functional enrichment analysis.

In Section 5.3.2 we present a series of experiments that showcase that the two-sided overlap indeed increases the sensitivity of the test. With this in mind, we used published collections of dysregulated miRNAs in diseases like Alzheimer's and several cancer types, in order to show that accounting only for the bias from miRNA target prediction leads to decreased sensitivity and consequently to false negative results. To prove this, we compared the p-values produced from using the left- and two-sided overlap for specific pathways related to Alzheimer's disease and cancer. Moreover, in Section 5.3.3, we performed the same experiment for the lists used in section 5.3.2, using miRNA targets from microT-CDS ([PGK+13]) and compare the p-values produced using the left- and two-sided overlaps. Finally, in the next section we outline the design of BUFET2, introduced in this chapter to increase the sensitivity of this test, while taking advantage of the scalable performance of BUFET ([ZVP+17]).

### 5.2.3 Introducing BUFET2

Motivated by the decreased sensitivity of the randomization test utilizing the left-sided overlap, we introduce BUFET2, which is a new version of BUFET. BUFET2 is designed to provide two p-values for the same randomization test that are produced using the left- and two-sided overlap as a statistical measure, respectively. BUFET2 harnesses the power of special data structures, called bit vectors to provide scalable performance and produce results in a matter of minutes for the specified accuracy of 1 million random miRNA groups.

At the same time, BUFET2 comes with a new, novel *reverse search module* that allows researchers to discover statistically significant microRNAs associated with a gene class (GO term, disease, pathway, etc), in almost real time. The module takes a gene class as input and randomizes at the miRNA level; however, the randomization is not a Monte Carlo approach, but rather an exhaustive one, since the number of known miRNAs is relatively small ($\sim 2500$). It should be noted here that the reverse search module also estimates both p-values. Moreover, the advantage of our software compared to other reverse search approaches, like miRPath v.3 ([VZP+15a]), is that it allows users to provide custom interaction and gene annotation data sets, making

our approach significantly more flexible.

Finally, both modules are also publicly available via a REST API[1] which facilitates programmatic access to the analysis. The API is implemented using Python[2] and it is documented using OpenAPI and Swagger[3]. Finally the source code has been packaged in a Docker image[4] to facilitate easier execution of the code without the need for compilation.

## 5.3 Experimental evaluation

In this section we demonstrate the experiments performed throughout this study as well as their results, including the experimental settings for each of them.

### 5.3.1 Comparison of Empirical and Hypergeometric distribution

Here we demonstrate that even when the bias from miRNA target prediction is removed, the hypergeometric test still presents decreased robustness, due to a bias related to gene class data. For this reason, we used gene annotation data from GO to illustrate that the expected hypergeometric distribution differs from the empirical distributions of the overlaps between a gene class and a miRNA group.

In order to compare the two distributions, we first calculated the number of targets for the 14 miRNAs in the group. We found that 3106 out of a total of 15064 genes are indicated as targets in the set of interactions. Then, given the size of each GO category and the genes not targeted by our original miRNA group, we calculated the expected hypergeometric distribution similar to the method followed by Bleazard *et al.* The next step was to estimate the empirical distribution. First, we created 1 million randomly assembled miRNA groups of size 14 and for each of those miRNA groups we calculated the size of the intersection between the targets of the group and each GO category. Finally, to facilitate comparison of the two distributions, two separate histograms were created and plotted on the same graph.

Considering that our data set for GO consists of 18686 categories, we selected indicative examples, for the diverse behaviour of the two distributions. These examples include the three domains of GO (biological process, molecular function, cellular component), ion transport (since it was also used by Bleazard *et al.*) and others. We plotted the results on Figure 5.1.

[1] https://bufet2.imsi.athenarc.gr/
[2] https://www.python.org/
[3] https://swagger.io/specification/
[4] https://hub.docker.com/r/diwis/bufet2

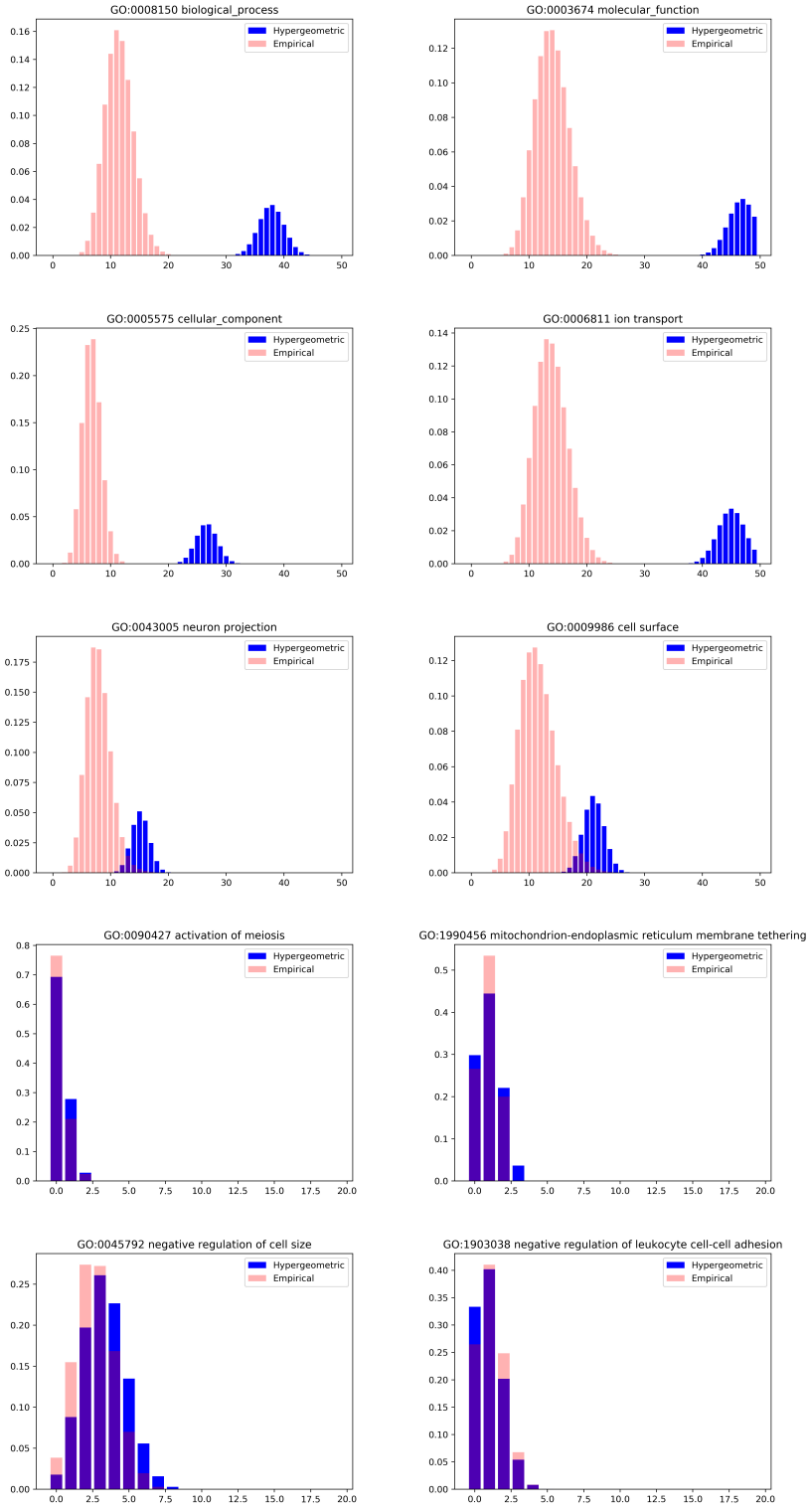



**Figure 5.1:** Hypergeometric vs empirical distributions for different GO categories

| Accession number | Name | Size |
|---|---|---|
| GO:0008150 | biological_process | 570 |
| GO:0003674 | molecular_function | 707 |
| GO:0005575 | cellular_component | 401 |
| GO:0006811 | ion transport | 679 |
| GO:0043005 | neuron projection | 453 |
| GO:0009986 | cell surface | 634 |
| GO:0090427 | activation of meiosis | 2 |
| GO:1990456 | mitoch. endopl. reticulum memb. tethering | 3 |
| GO:0045792 | neg. regulation of cell size | 10 |
| GO:1903038 | neg. reg. of leukocyte cell-cell adh. | 6 |

**Table 5.1:** Number of genes in each of the GO categories in Figure 5.1

Immediately, it becomes evident that there are categories for which the two distributions match and categories for which there is a clear mismatch between the distributions. Categories, which are higher in the GO hierarchy seem to have a larger distance between the distributions. On the other hand, categories, that describe more specific biological functions, tend to significantly overlap with the hypergeometric distribution. Moreover, the categories that present the larger mismatch seem to be those, that contain a large number of genes. This provides the intuition, that maybe, the size of the GO category is related to the mismatch between the distributions. In Table 5.1 we demonstrate the sizes of each of the GO categories shown in Figure 5.1.

Given the results in Table 5.1, it seems that the mismatch is indeed more prominent as the size of the category increases. In order to quantify and investigate this effect, we designed the following experiment. First, we define as *distance* between the distributions the horizontal distance (number of genes in common with the GO category) between the maximum values of each distribution. Then, using this definition, for all GO categories in the data set, we evaluate the distance between the two distributions in relation to the category size. The result of this experiment can be seen in Figure 5.2.

We can clearly observe that, barring a few outliers, in general, the greater the size of a GO category, the larger the distance is and this translates to a larger mismatch between the two distributions. Given the hierarchical structure of the Gene Ontology, it is easy to understand why this phenomenon exists: the genes in a GO category are also contained in a category which is higher in the hierarchy (e.g. neuron projection, cell projection and cellular component) and this means that the genes in a category higher in the tree are not picked uniformly and thus they do not satisfy the assumptions of the hypergeometric distribution. This, essentially introduces a bias that is more prominent, the higher up from the branches of the tree one goes. To check

whether this effect can also be observed with DisGeNET and KEGG, we performed the same experiment using these data sets. The results can be seen in Figures 5.3 and 5.4 respectively.

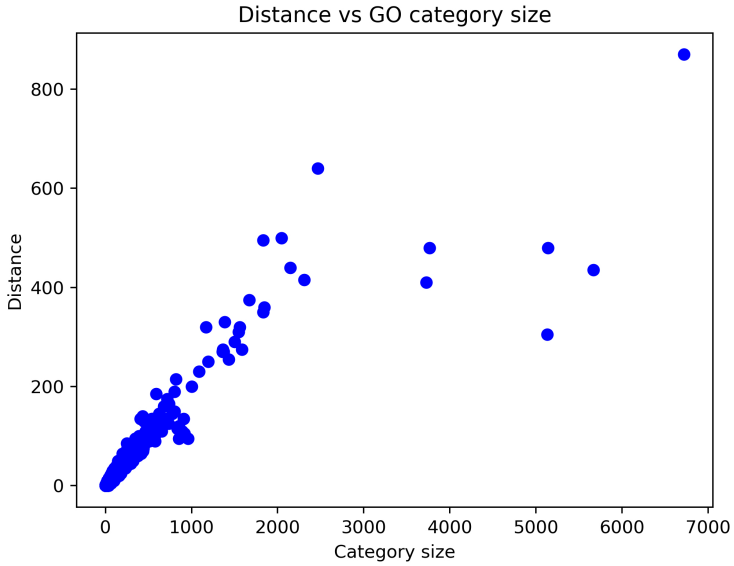



**Figure 5.2:** Distance between hypergeometric and empirical distribution vs GO category size

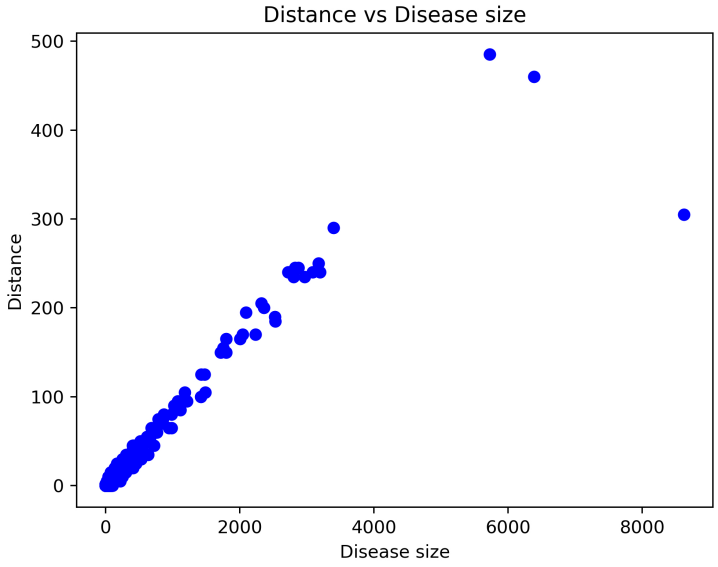



**Figure 5.3:** Distance between hypergeometric and empirical distribution vs disease size

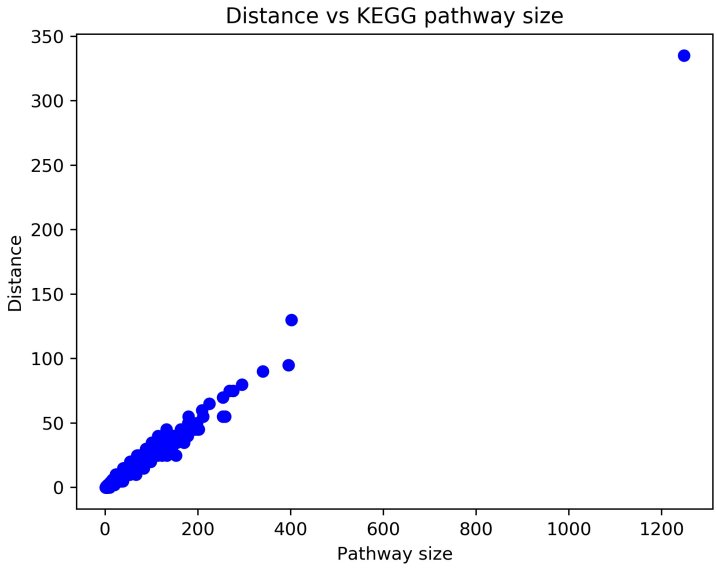



**Figure 5.4:** Distance between hypergeometric and empirical distribution vs size of KEGG pathway

Evidently, the aforementioned effect is even more pronounced for DisGeNET and the relationship between the disease size and the distance between the two distributions. This can be explained by the fact that the text mining tools used to compile the database, utilize structured vocabularies and ontologies ([PRASP+19c]). Thus, the hierarchy existing between the diseases introduces the same bias as seen for GO. Regarding KEGG, the same effect is also observed. This could be attributed to complex interactions between genes in pathways that are not specified in the data set. An example of this could be gene co-expression or other more complex effects that lead to genes being included in a pathway.

In conclusion, we showed that for all three data sets there is a bias, which is clearly a product of the gene annotation structure, affecting the miRNA functional enrichment analysis.

### 5.3.2 Left-sided vs two-sided overlap

In the previous section, we demonstrated that there is a clear bias related to the structure of the gene annotations. In this section we show that the left-sided overlap does not account for this bias, highlighting the need to use another, more sensitive metric, namely the two-sided overlap. For this reason, we compare the performance of the two metrics using four, published lists of miRNAs: one for Alzheimer's disease ([ACV+19]) one for non-small cell lung cancer ([ZKH+21]), one for breast cancer

([vSWV+15]) and one for gastric cancer ([HHL+17]). These lists are provided as supplementary data.

Furthermore, we utilized two modules of the KEGG database, namely PATHWAY and DISEASE, since DISEASE provides pathways associated with each of the four diseases. It stands to reason that if we provide the list of miRNAs for each disease to the randomization test as input, at least some (if not all) of the pathways related to each disease should be marked as significant. Apart from the pathways related specifically to each disease, we will present other pathways commonly associated with cancers (for the three cancer lists). Each experiment was performed 10 times and in Tables 5.2, 5.3, 5.4 and 5.5 we present the average of each p-value along with the standard deviation for the 10 experiments.

| Pathway ID | Name | Left-s. p-value | Left-s. St. dev. | Two-s. p-value | Two-s. st. dev. |
|---|---|---|---|---|---|
| hsa05010 | Alzheimer disease | 0.0863 | $2.3 \times 10^{-4}$ | 0.02412 | $9.9 \times 10^{-5}$ |
| hsa05022 | Pathways of neurodegeneration | 0.06637 | $1.7 \times 10^{-4}$ | 0.007467 | $7.4 \times 10^{-5}$ |

**Table 5.2:** Left- and two-sided p-values for Alzheimer's disease using miRTarBase

| Pathway ID | Name | Left-s. p-value | Left-s. St. dev. | Two-s. p-value | Two-s. st. dev. |
|---|---|---|---|---|---|
| hsa05223 | Non-small cell lung cancer | 0.2606 | $3.9 \times 10^{-4}$ | 0.229 | $3.5 \times 10^{-4}$ |
| hsa05200 | Pathways in cancer | 0.01745 | $2.1 \times 10^{-4}$ | 0.0005191 | $1.9 \times 10^{-5}$ |
| hsa05206 | MicroRNAs in cancer | 0.0867 | $2.6 \times 10^{-4}$ | 0.03432 | $1.5 \times 10^{-4}$ |
| hsa05205 | Proteoglycans in cancer | 0.06989 | $1.5 \times 10^{-4}$ | 0.03196 | $1.5 \times 10^{-4}$ |

**Table 5.3:** Left- and two-sided p-values for Non-small cell lung cancer using miRTarBase

| Pathway ID | Name | Left-s. p-value | Left-s. St. dev. | Two-s. p-value | Two-s. st. dev. |
|---|---|---|---|---|---|
| hsa05224 | Breast cancer | 0.00436 | $8.6 \times 10^{-5}$ | 0.002436 | $5 \times 10^{-5}$ |
| hsa05200 | Pathways in cancer | 0.000223 | $1.7 \times 10^{-5}$ | 0.0000164 | $3.7 \times 10^{-6}$ |
| hsa05206 | MicroRNAs in cancer | 0.0001501 | $1.6 \times 10^{-5}$ | 0.0000107 | $3.3 \times 10^{-6}$ |
| hsa04020 | Calcium signaling pathway | 0.07019 | $1.7 \times 10^{-4}$ | 0.04543 | $2.1 \times 10^{-4}$ |

**Table 5.4:** Left- and two-sided p-values for breast cancer using miRTarBase

| Pathway ID | Name | Left-s. p-value | Left-s. St. dev. | Two-s. p-value | Two-s. st. dev. |
|---|---|---|---|---|---|
| hsa05226 | Gastric cancer | 0.03668 | $2 \times 10^{-4}$ | 0.02293 | $1.3 \times 10^{-4}$ |
| hsa05200 | Pathways in cancer | 0.01216 | $1.5 \times 10^{-4}$ | 0.0005839 | $1.7 \times 10^{-5}$ |
| hsa05206 | MicroRNAs in cancer | 0.083 | $3.8 \times 10^{-4}$ | 0.03825 | $2.4 \times 10^{-4}$ |

**Table 5.5:** Left- and two-sided p-values for gastric cancer using miRTarBase

Tables 5.2, 5.3, 5.4 and 5.5 immediately suggest that using the two-sided overlap as a metric increases the sensitivity of the test and leads to more accurate results. This is particularly evident with Alzheimer's disease, where the "Alzheimer's disease" pathway is not marked as statistically significant by the left-sided p-value. Moreover, even when both methods suggest that a pathway is significant, the two-sided p-value

is smaller, denoting a stronger association of the miRNA group with the pathway. The complete list of results is provided as supplementary data.

### 5.3.3 Randomization test metric vs type of type of miRNA targets

In this section we perform the same experiment as in the previous section, but this time, instead of experimentally validated results, we utilize miRNA targets predicted using microT-CDS, retrieved from the MR-microT ([KVS+14b]) online application using a prediction score threshold of 0.8.

The results (see Supplementary data) paint a picture similar to the previous section. As an example we present in Table 5.6 some indicative results for non-small cell lung cancer. It becomes evident that the use of predicted targets leads to more false negative results than those procured from the utilization of validated targets for both metrics. However, even in this case the two-sided p-value seems to present a larger sensitivity.

| Pathway ID | Name | Left-s. p-value | Left-s. St. dev. | Two-s. p-value | Two-s. st. dev. |
|---|---|---|---|---|---|
| hsa05223 | Non-small cell lung cancer | 0.1685 | $4.2 \times 10^{-4}$ | 0.1646 | $3.7 \times 10^{-4}$ |
| hsa05200 | Pathways in cancer | 0.1935 | $3.8 \times 10^{-4}$ | 0.1511 | $3.3 \times 10^{-4}$ |
| hsa05206 | MicroRNAs in cancer | 0.05948 | $2.8 \times 10^{-4}$ | 0.04554 | $2.4 \times 10^{-4}$ |
| hsa05205 | Proteoglycans in cancer | 0.1727 | $5.2 \times 10^{-4}$ | 0.1569 | $3.4 \times 10^{-4}$ |

**Table 5.6:** Left- and two-sided p-values for Non-small cell lung cancer using microT

## 5.4 Discussion

Issues regarding overrepresentation analysis and the hierarchy of Gene Ontology annotations have been mentioned as far back as 2003 ([DKM+03, ZCKS10]). Bleazard *et al.* also point out that there may be underlying correlations between targeting of processes and the hierarchical GO structure. In Section 5.3.1 we demonstrated that there is a definite bias related to the hierarchical structure of the GO and that this bias also affects other gene annotation data sets like KEGG PATHWAY and DisGeNET. This suggests that Fisher's exact test does not present enough robustness for miRNA functional enrichment analyses, even when the bias from miRNA prediction algorithms is eliminated through the utilization of experimentally validated miRNA targets. Despite its drawbacks, however, the standard method is still very popular in published studies ([GMCS20]).

Furthermore, Sections 5.3.2 and 5.3.3 clearly demonstrate that the two-sided overlap leads to a more sensitive test that produces less false negative results. This denotes

that the same bias affects the analysis regardless of the type of interactions used. On the other hand, it becomes evident that the enrichment analysis using miRNA predicted targets is not as sensitive as the one using experimentally validated targets. This is expected and can be attributed to the fact that prediction algorithms produce hundreds or thousands of targets for each miRNA and the results contain a large number of false positive interactions ([PLM+17b]). This implies that genes in categories or diseases at the bottom of the hierarchy (more specific) are being targeted by a lot of miRNAs, even if these are not true interactions. This translates to many randomly assembled groups that often present a better overlap than the sample of interest and consequently, the sample of interest loses its status of being among the 5% of miRNA groups in the empirical distribution (i.e. p-value$\geq$0.05)

We should also note here an important observation regarding empirical p-values and randomization tests. More specifically, in our case, the standard deviation between experiments is at least two orders of magnitude smaller than the p-value significance threshold (0.05), for both types of overlaps. This means that the p-values produced by the 10 repetitions of the same experiment do not produce a variation in significance. In other words, if a gene class is marked as significant by one of the 10 repetitions, then the magnitude of the standard deviation implies that the statistical significance of the gene class will be preserved across every repetition of the same experiment. Consequently, no matter how many times one runs the experiment with the same input, the produced p-values are not expected to be qualitatively altered. In general, this shows that 1 million random groups are sufficient for this randomization test. It should be mentioned here that the aim of randomization tests is to model the empirical distribution of the data in order to reach conclusions. As a result, the accuracy of the empirical p-values produced depends mainly on the number of random samples used for the randomization test. A small number of random samples is not enough to model the empirical p-values and the results can present a high variation between two repetitions of the same experiment. On the other hand, a very large number of random samples is not necessary since it can consume a large number of compute resources and lead to significantly large execution times; this is because it does not really contribute much to the results, considering that a smaller number of samples can accurately predict whether the sample of interest presents a p-value smaller than 0.05. Thus, such randomization tests require a balance between too few and too many random samples and sufficient accuracy can be reached when the results between two repetitions of the same experiment do not vary in a way that changes whether a result is significant or not.

## 5.5 Conclusion

Concluding, in this chapter we demonstrated that when the bias from miRNA target prediction algorithms is removed, then another bias, affecting gene classes appears. This bias has a clear relation to the size of each class and it might be related to a hierarchical structure or other non-specific reasons. Additionally, we introduced a new metric, the two-sided overlap which seems to be more appropriate as a randomization test metric in all cases (predicted and experimentally validated targets). Furthermore, all data sets used in Section 5.3 have been uploaded to Zenodo (see Abstract) to facilitate experiment reproduction. Finally, we introduced BUFET2 that calculates p-values utilizing both the left- as well as the two-sided overlap as metrics to increase the accuracy of the analysis.

# Chapter 6

# Supervised methods for approximate miRNA enrichment

Motivated by the work in Chapter 4 and more specifically the SLI, in this chapter, we present a novel approach for miRNA enrichment analysis. This approach utilizes machine learning techniques, to predict p-values using features of miRNA groups, relevant to the problem, instead of calculating them using randomization experiments. This simplifies the work for bioinformatics data analysts, helping them to efficiently perform multiple enrichment analysis tasks. Our contributions are: (a) framing the problem, (b) data set creation and feature engineering, (c) determining a shortlist of promising machine learning models, using cross-validation, and (d) fine-tuning to determine the best models for our case. Our approach shows that the best model demonstrates an $R^2$ score above 90%, and $MAE = 0.048$.

## 6.1 Permutation test

Given a miRNA group as a query, the biological function permutation test introduced in [BLGJ15] consists of the following steps:

1. Calculate a statistical measure $S$ (e.g., left-sided overlap - see below), that captures a type of 'relevance' of the biological function with the query, according to the genes that are related to both of them.

2. Create a large number (e.g., 1 million) of randomly assembled miRNA groups, with each containing the same number of miRNAs, and calculate $S$ for each of these groups, as well.

3. Measure the proportion of randomly assembled groups that present more favourable values for $S$ than the query.

More formally, each miRNA is represented as the set of genes targeted by it. Consequently, a group of miRNAs is represented as a set, containing the union of all genes targeted by each miRNA in the group. Moreover, a biological function is also represented as the set of genes participating in that function.

Based on the previous, we can now describe one popular permutation test: Let $M$ be the set of genes containing the union of targets from all miRNAs in a query group, and also let $B$ the set of genes participating in a biological function. The statistical measure used to compare the query to the randomly assembled miRNA groups is the *left-sided overlap* and it is defined as follows:

$$left\text{-}sided\ overlap(M, B) = \frac{sizeof(M \cap B)}{sizeof(M)}$$

Essentially, the left-sided overlap is defined as the proportion of targeted genes that also participate in the biological function. Then, we create 1 million random miRNA groups $M_j$ with the same size as the query and calculate the left-sided overlap for each of them. The empirical p-value is defined as (where overlap is the left-sided overlap):

$$p - value = \frac{sizeof(\{M_j : overlap(M_j, B) \geq overlap(M, B)\})}{n}$$

which is the proportion of randomly assembled groups presenting a larger left-sided overlap than the query.

### 6.1.1 Performance issues

The above analysis relies on a very large number of union and intersection set operations. Given that this analysis is performed for more than one biological functions, and that more than 20K biological functions exist, a few million union and about 20 billion intersection operations are performed in the span of the analysis.

The software implemented by the authors in [BLGJ15] is written in Python, uses hash join set operations and a typical analysis on a single CPU core of an Intel Xeon CPU requires many hours. However, based on the fact that this analysis is very repetitive, even a small increase in operation speed is going to lead to a large total speedup. With this motivation, in [ZVP+17] we re-implemented the algorithm in C++. We also improved the analysis performance by exploiting bit vectors, as well as a hybrid version of hash join between sets of items and bit vectors. This made the analysis one order of magnitude faster requiring about 40 minutes to complete on a single core of the same Xeon computer.

Then in [ZVSD20], we introduced novel indexing techniques, that allowed us to remove a large number of redundant operations performed by our previous version

and thus managed to reduce the time to approximately half of what was required previously on a single core. Furthermore, we used a technique that allowed us to run the analysis on only a subset of the biological functions (which are predicted to be statistically significant) and managed to further reduce the time required for p-value calculation to about 3 minutes (on a single core of the Xeon processor). The downside to the latter approach, is that p-values are produced only for the functions expected to be significant and not all biological functions in the dataset. Consequently, we were motivated to use ML to train a model that will predict p-values very quickly and for every biological function in the data set.

## 6.2 Features selected for ML training

In order to train a machine learning model to predict p-values (`p_value`) as accurately as possible, we selected features from our dataset based on their biological meaning and their relevance to the analysis. The features are summarized below:
**miRNA group size (`mirna_group_size`):** The number of miRNAs in a miRNA group.
**Biological function ID (`biological_process`):** a unique string that identifies each biological function. This string consists of 2 letters (same for all biological functions) and a numerical part. We turned this ID into a numerical value, by stripping the letters from the string.
**Number of common genes (`number_of_common_genes`):** The number of genes targeted by a miRNA group that also belong to the biological function.
**Left-sided overlap (`left_sided_overlap`):** the left-sided overlap as defined in Section 6.1.
**Right-sided overlap (`right_sided_overlap`):** Given $M$ and $B$ from Section 6.1, the right-sided overlap is defined as:

$$right\text{-}sided\ overlap(M, B) = \frac{sizeof(M \cap B)}{sizeof(B)}$$

**Two-sided overlap (`overlap`):** Given $M$ and $B$ from Section 6.1, the two-sided overlap (or Jaccard coefficient) is defined as:

$$two\text{-}sided\ overlap(M, B) = \frac{sizeof(M \cap B)}{sizeof(M \cup B)}$$

**Common genes as a percentage of the universe of genes (`common_genes_proportion_to_total`):** The number of common genes between M and G as the proportion of the total number of genes in the universe of genes.

**Common gene list (`common_genes`):** The list of common genes, sorted by alphabetical order. We used label encoding to turn the values into categorical values.
**Chromosomes of common genes (`common_chr`):** The list of chromosomes on which the common genes are located, sorted by alphabetical order. We used label encoding to turn the values into categorical values.
**Number of chromosomes where common genes are located (`number_of_common_chr`):** The number of chromosomes on which the common genes are located.
**Left chromosome overlap (`chr_left_sided_overlap`) :** The number of chromosomes on which the common genes are located divided by the number of chromosomes of the genes targeted by the miRNA group.
**Right chromosome overlap (`chr_right_sided_overlap`):**
The number of chromosomes on which the common genes are located, divided by the number of chromosomes of the genes belonging in the biological function.
**Two sided chromosome overlap (`chr_overlap`):** The number of chromosomes on which the common genes are located, divided by the number of chromosomes for the union of the genes contained in $M$ and $B$.

Using these features, we produced a dataset for groups of miRNAs of size 10, 25, 50, 100 (containing 100 groups of each size) for all biological functions as described in the Gene Ontology [ABB+00][The18]. Finally, to handle categorical features (miRNA group size, biological function, common gene list, chromosomes of common genes) we used label encoding.

### 6.2.1 ML Algorithms

In order to find the best method to use for our case, we are going to use the following algorithms:

- **Linear/Ridge/Lasso Regressors:** traditional regression methods, fitting a linear equation that uses the least squares method (with several variants of regularization).
- **Decision Tree Regressor:** uses decision trees to make predictions.
- **Random Forest Regressor:** estimates target value by combining average estimation values of several individual prediction models based on classifying decision trees for a number of subsets of the data set.
- **Adaboost Regressor:** combines multiple weak decision trees into one.
- **Gradient Boost (XGBoost, LightGBM, CatBoost):** in XGBoost, the estimation of the target is done by combining estimates of many individual

prediction models based on decision trees. LightGBM is similar to XGBoost, but prediction is much faster than XGBoost. Regarding CatBoost, the key difference is that it builds decision symmetric trees. Both LightGBM and CatBoost can inherently handle categorical features.

- **MLPerceptron Regressor:** typical Neural Net configuration.

## 6.3 Evaluation

In this section, we present preliminary results on a dataset that has been created by only using 1000 random miRNA groups to calculate p-values. This was done due to time constraints and to reach some preliminary conclusions about the dataset, in order to compare the algorithms presented in Section 6.2.1. Given the definition of an empirical p-value in Section 6.1, we expect that the p-values in the dataset will present a small accuracy, since the number of random miRNA groups is three orders of magnitude smaller than the required number. This is expected to create larger errors than the ones that the permutation test produces. However, these preliminary results are a first indicator on which algorithms present the worst performance in terms of accuracy and thus, which algorithms will be featured in the final work.

### 6.3.1 Linear correlation

In Figure 6.1 we can see a heat map of the linear correlation between the features. Each cell represents the correlation measured by Pearson correlation coefficient $r$ between features in row x and column y. The values of $r$ range between -1 and 1. The larger the deviation from 0 the stronger the positive (for values closer to 1) or negative (for values closer to -1) correlation is. It is interesting to note here that the p-value, which is the feature we want to predict, does not seem to have a strong linear correlation (either positive or negative) with any of the features individually. Nevertheless, this does not mean that the p-value is not related with the features in a non-linear way. More specifically, it is expected to be related with the left-sided overlap (see p-value definition in Section 6.1).

### 6.3.2 Preliminary results

In this section we are going to perform a preliminary evaluation of the algorithms we outlined in Section 6.2.1. In order to evaluate and compare the algorithms, we are going to use the following statistical measures [Gér19]:

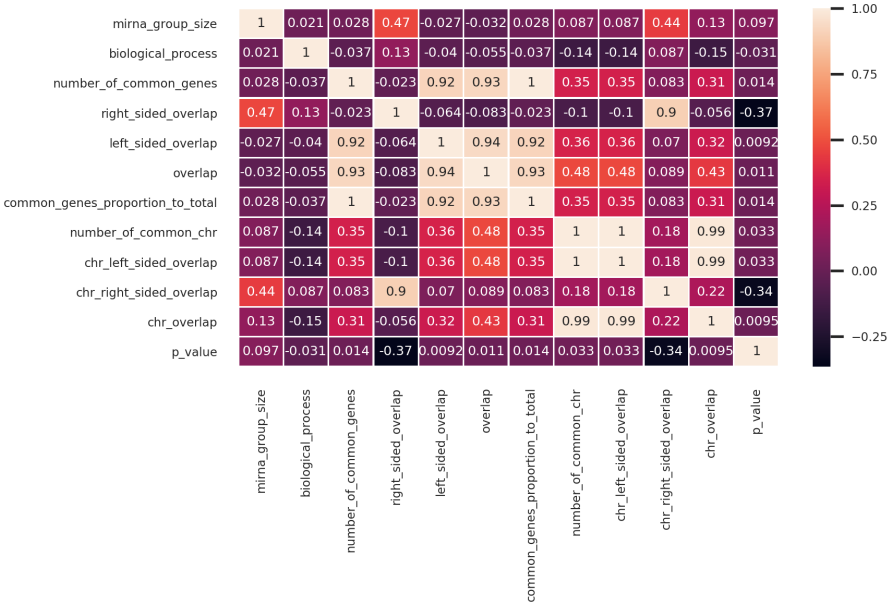



**Figure 6.1:** Linear correlation between the selected features

1. Mean Absolute Error ($MAE$): the $MAE$ is the mean absolute difference between observed and predicted values. Essentially, it is used to compare predictions to actual observed values.

2. Coefficient of determination ($R^2$): the $R^2$ score is a statistical measure of how close are the predictions to the real data points. It is essentially a goodness-of-fit measurement.

Model selection and implementation was performed with the method of k-Fold cross-validation. We first split the dataset into the *training* and *testing* dataset. The training dataset is split into k folds (groups). Then k-1 of those groups are used to train the model and the group left is used for *validation*. The score of the validation is recorded. Then, the process is repeated, but this time another group is used for validation and the rest for re-training, until all folds have been exhausted. The average of all validation scores is the final validation score. Finally, the *test score* is retrieved by testing the model against the testing data set.

Final validation scores for each algorithm are shown in Table 6.1 and the testing scores in Table 6.2. Model parameters were selected via Grid Search for the Linear Regressors and the rest via Random Search, due to the large number of parameters and memory constraints.

It should be noted that the algorithms based on linear regression models do not perform as well as the other algorithms. Our approach shows that the best model is LightGBM, demonstrating a $MAE = 0.048$.

| | $MAE$ | $R^2$ | Model parameters |
|---|---|---|---|
| **Linear Regressor** | 0.242 | 0.246 | copy_x = True, fit_intercept = False, normalize = True |
| **Ridge Regressor** | 0.242 | 0.2462 | alpha=1, copy_x = True, fit_intercept = False, normalize = True, solver = auto, tol = 0.001 |
| **Lasso Regressor** | 0.4364 | $4 \times e^{-6}$ | default |
| **Decision Tree Regressor** | 0.1027 | N/A | min_samples_leaf = 2, min_samples_split = 5,max_depth = None,max_features = None |
| **Random Forest Regressor** | 0.0992 | N/A | n_estimators = 50, n_jobs = -1, min_samples_leaf = 2, min_samples_split = 2, verbose = 10 |
| **Adaboost Regressor** | 0.13 | N/A | n_estimators = 20, loss = square, base_estimator = RandomForestRegressor, (n_jobs =-1, verbose=2, max_depth=25, n_estimators=4) |
| **LightGBM Regressor** | 0.038 | N/A | objective = poisson, boosting_type = gbdt, learning_rate = 0.5, n_estimators = 900, n_jobs = -1, num_leaves = 40, max_depth=20, reg_alpha = 1, reg_lambda = 1, force_col_wise = True |
| **XGBoost Regressor** | 0.0745 | N/A | verbosity = 2, max_depth = 20, n_estimators=200 |
| **CatBoost Regressor** | 0.13 | N/A | n_estimators=1000, learning_rate=0.5, bootstrap_type=Bernoulli, |
| **ML Perceptron Regressor** | 0.1817 | N/A | hidden_layer_sizes = (24,24,24), activation = "relu", max_iter = 40, learning_rate='adaptive' |

**Table 6.1:** Algorithm evaluation: validation set

| | $MAE$ | $R^2$ | Model parameters |
|---|---|---|---|
| **Random Forest Regressor** | 0.1443 | N/A | n_estimators = 50, n_jobs = -1, min_samples_leaf = 2, min_samples_split = 2, verbose = 10 |
| **LightGBM Regressor** | 0.0477 | N/A | objective = poisson, boosting_type = gbdt, learning_rate = 0.5, n_estimators = 900, n_jobs = -1, num_leaves = 40, max_depth=20, reg_alpha = 1, reg_lambda = 1, force_col_wise = True |
| **XGBoost Regressor** | 0.136 | N/A | objective = poisson, boosting_type = gbdt, learning_rate = 0.5, n_estimators = 900, n_jobs = -1, num_leaves = 40, max_depth=20, reg_alpha = 1, reg_lambda = 1, force_col_wise = True |

**Table 6.2:** Algorithm evaluation: test set

## 6.4 Conclusion & Future work

In this chapter, we presented an approach for miRNA enrichment analysis using machine learning techniques, in order to predict p-values instead of calculating them using randomization experiments. The goal is to simplify the work for bioinformatics data analysts, facilitating multiple enrichment analysis. Preliminary results showed

that the best model demonstrates $MAE = 0.048$. As next steps, we plan to expand our dataset to include p-values estimated using 1 million random groups in order increase their accuracy as well as the accuracy of the prediction.

# Chapter 7

# Data services for miRNA enrichment analysis

This chapter outlines supplementary work regarding miRNAs analysis The first part deals with the design, data management and implementation of online miRNA analysis tools while the latter part illustrates the design of a cloud-based system developed in order to facilitate scalable and elastic execution of containerized software. The motivation for this work is to perform the experiments for the work described in Chapter 5.

## 7.1 Online miRNA data management and processing

In this section we describe the work we performed in the course of developing scientific web applications targeted towards Bioinformatics researches. This work deals with the management and online processing of Bioinformatics data and enables researchers to perform analyses in real-time through intuitive web interfaces. The web interfaces provide functionality and features that facilitate the execution of scientific data analyses while removing the need to write and execute code through the command line. In the next sections we outline the data management methods utilized during the design of these applications.

### 7.1.1 Data management in online miRNA functional enrichment

miRPath v.3 [VZP+15b] is the newest version of miRPath that builds on the old version, while preserving the usability and functionality of the older version. miRPath

is a tool that performs the standard method of miRNA functional enrichment analysis using Fisher's exact test as outlined in section 2.4.1. The application contains data regarding miRNA-to-gene interactions as well as gene classification data, while the input query is provided by the user either as a file containing the list of miRNA or by adding each individual miRNA through the interface. A screenshot of the application can be seen in Figure 7.1.

The original version of the web application used interaction data from microT-CDS [PGK+13] and TarBase [KPC+17a] and classifications retrieved from KEGG (see Section 2.2). However, the new version introduces the TargetScan target prediction data set (see Section 2.3) as well as gene classifications from the Gene Ontology (see Section 2.2). Moreover, the new version includes the option to use unbiased enrichment analysis (see Section 2.4.2) through the combination of pre-calculated results using *Fisher's method* (also known as *Fisher's Fisher's combined probability test*) [Fis92]. The introduction of the new data sets, allow researchers to explore the effect of a group of miRNAs, while combining different methods of target prediction at the same time.

#### 7.1.1.0.1 Enrichment using Fisher's exact test

First, the user selects whether they want to perform an analysis using data from KEGG or GO. Then they upload a file containing the list of miRNAs which will be used as input for the analysis or they have the option to add each miRNA separately. At the same time, the users are able to change the default source of interactions (microT-CDS) for each individual miRNA from the drop-down menu and this significantly enhances the versatility of the system. Every time a miRNA is added or one of the options changes, the results are being re-calculated and new p-values are estimated.

Given a miRNA list of interest, the system calculates p-values using Fisher's exact test or Fisher's method. The first case is applied whenever a researcher selects the "genes union" or "genes intersection" option, which controls whether the set of genes targeted by the miRNA group is the union or intersection of the targets of each individual miRNAs. On the other hand, when the "pathways union" or "pathways intersection" are selected the following method is applied:

1. For each individual miRNA and for each gene class (category/pathway), calculate a p-value using Fisher's exact test.

2. Estimate the union/intersection of the set of gene classes by combining the individual p-values for each class, using Fisher's method.

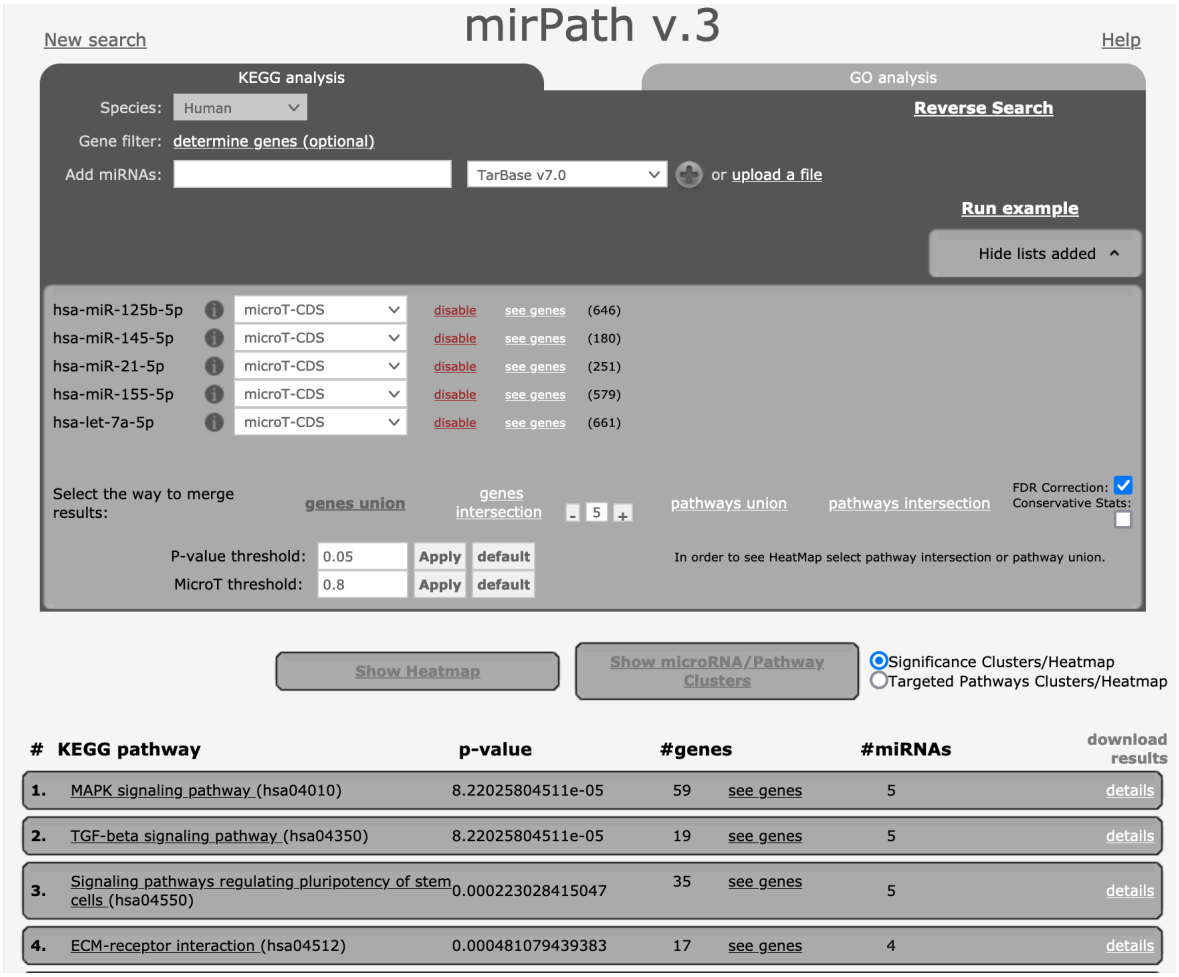



**Figure 7.1:** miRPath v.3 application

In the case that "pathways union" or "pathways intersection" is used, a drop-down menu appears, allowing users to utilize pre-calculated unbiased p-values (see Section 2.4.2) for step 1. The reason that pre-calculated results are used is because this analysis is computationally intensive and therefore it is impossible to calculate in real time. Additionally, this difficulty in real-time execution has the implication that it cannot be used for the "genes union"/"genes intersection" options. Moreover, this inability to calculate results in real time is is the reason that motivated future works that aim to accelerate this analysis (see Chapters 3, 4, 6). However, this application is the first online application to use unbiased statistics, adding to its overall power.

Furthermore, the user can filter results based on parameters like the p-value threshold or they filter miRNA predicted targets using parameters like the score produced by the respective algorithm.

#### 7.1.1.1 Reverse search module

In addition to the enrichment analysis module, miRPath v.3 supports reverse querying (i.e. finding which miRNAs are related to a gene class) accompanied by a p-value to showcase the strength of the association. Figure 7.2 demonstrates the reverse search module of the application. The user enters a gene class, as well as the type of miRNA targets to be used and the system calculates the results, which appear sorted based on the p-value (smaller first).

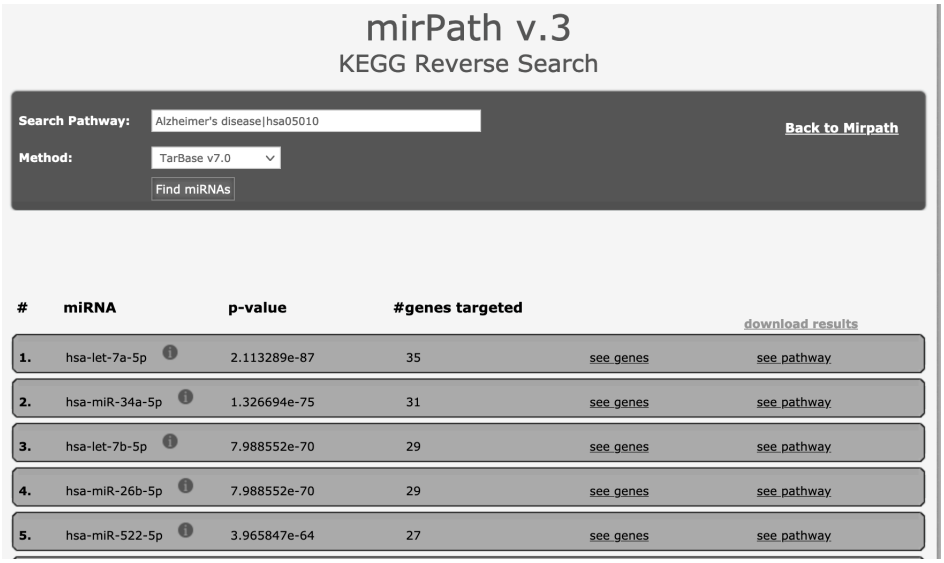



**Figure 7.2:** Reverse search module of miRPath v.3

It is worth noting here that, to avoid the large number of repeated set operations necessary for Fisher's exact test and since the universe of all known human miRNAs contains $\sim$ 2500 miRNAs, we pre-calculated the p-values for each result. This leads to the fact that querying is translated only to database queries, thus increasing the speed and avoiding redundant set operations.

### 7.1.2 Online exploration of miRNA transcription regulation

Since the discovery of miRNAs in the early 1990's, research on them has been fueled by the need to understand the mechanisms underlying their biogenesis, function and role in disease. However, knowledge regarding the regulation of genes being transcribed to miRNAs still remains limited. This has been largely due to the lack of experimental and/or computational methodologies capable of detecting transcription start sites (TSSs) with high accuracy and resolution.

miRGen v.3 [GVZ+15] was introduced to enhance the pool of knowledge regarding regulation of miRNA transcription. miRGen v.3 is the result of analyzing more

than 7.3 billion RNA-, ChIP- and DNase-Seq next generation sequencing reads with state-of-the-art miRNA TSS prediction and TF binding site identification algorithms. More specifically, TSS identification was performed using the microTSS [GVP+14] algorithm, which is a machine-learning algorithm that provides highly accurate predictions, in conjuction with the miRBase database [KGJ13]. On the other hand, Transcription Factor (TF) binding site discovery was carried out using the Wellington algorithm [PEC+13]. The identified miRNA TSSs as well as TF binding sites were stored in a PostgreSQL relational database. Indices in the database were created to guarantee the efficiency of the system and foreign keys were added to avoid integrity violations in the data.

miRGen v.3 was developed to allow the discovery of which TFs regulate the production of a miRNA or alternatively, which miRNAs are regulated by a TF, thus bridging the gap between miRNAs and TFs. This is done by enabling users to find overlapping regions between TF binding sites and miRNA TSSs, using different constraints, specified by relevant filters. To facilitate this, miRGen's intuitive web interface was designed around the new database schema and effort was made to be adaptable to a wide variety of screen formats and devices (PCs, tablets, smartphones, etc.). Moreover, the interface was developed using the Yii 2.0 PHP framework and the miRNA and transcription factor search fields were designed as auto-complete search boxes to assist users in selecting the proper search keywords. Finally, useful filters were implemented to enable researchers to focus on particular data that match the interests of the user. A screenshot of the miRGen v.3 system can be seen in Figure 7.3.

The interface contains two main search boxes, one for miRNAs and one for TFs, which are not mutually exclusive, meaning that the users have the ability to also input both at the same time. This allows the user to search whether there the transcription of a specific miRNA is regulated by a specific TF. To further facilitate the exploration of the data, the boxes have an auto-complete function for both miRNAs and TFs. After the user enters the input, the details for each miRNA appear collapsed, in order to optimize the visibility of the system. The user has the ability to show the details for each miRNA, which include the coordinates of the TSS, as well as links to external databases, like miRBase [KGJ13], TarBase [VPK+14] or lncBase [PVK+15]. At the second level, the user can see the TFs associated with the miRNA and information about the TF itself as well as the number of binding sites overlapping with the TSS of the corresponding miRNA. The third level reveals more information about each binding site.

Regarding the filters, the users have the ability to expand or reduce the allowed search region by moving its edges, using the upstream/downstream values. Moreover, they can filter out TSSs and TF binding sites, by using the P-value and TPM thresh-

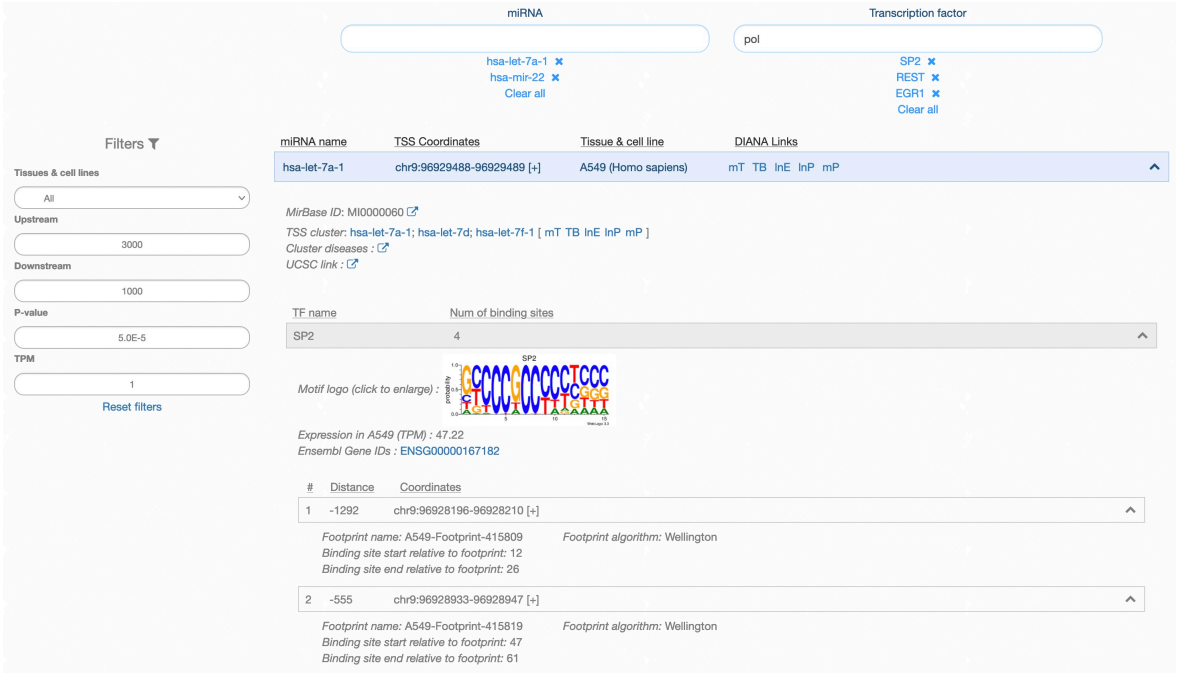



**Figure 7.3:** The web interface of miRGen v.3

old input boxes. Finally, there is also an option to limit the search for overlapping regions in specific tissues and cell lines.

### 7.1.3 Indexing interaction data between miRNAs and lncRNAs

Long non-coding RNAs (lncRNAs) are RNA molecules with a length greater than 200 nucleotides, whose regulatory role has been uncovered by Next Generation Sequencing (NGS) analyses in recent years [DKA+12]. lncRNAs act as regulators of gene expression in various levels of transcription and have also been shown to interact with miRNAs [HWB+11]. This provides the motivation investigate the potential interactions between miRNAs and lncRNAs and collect all of them in a single place. Even though there is an abundance of databases specifying interactions between mRNA and miRNAs, there are very few databases containing interactions between miRNAs and lncRNAs (e.g. LncBase v1 [PGK+12], miRcode [JML12]). LncBase v2 is the newer version of LncBase, containing both *in-silico* as well as experimentally validated interactions between miRNAs and lncRNAs. It was introduced to provide a single, comprehensive source of miRNA-lncRNA interaction data concerning two organisms: human and mouse.

The applications is split into two distinct modules, namely the *Experimental* and the *Predicted* modules. Each of the two modules is responsible for visualizing

different data procured using different methods. The retrieval of experimentally validated interaction data was performed by manually curating an extensive collection of manuscripts, leveraging an in-house semi-automated text mining pipeline, similar to TarBase v7 [VPK+14]. On the other hand, the data for the "predicted module", were collected *in-silico* through target prediction with an adjusted DIANA-microT algorithm [RMA+12]. Another interesting aspect is the collection of expression data for each lncRNA through RNA-seq. More specifically, we utilized several genome databases and analyzed them, in order to discover on which tissues the lncRNA may be found either in abundance or depleted.

The wealth of knowledge produced by the previous techniques was stored in a relational database using the PostgreSQL database management system using indices to facilitate the efficiency of the system. The interface of the application was developed using the PHP Yii 2.0 framework while the miRNA and lncRNA fields support an auto-complete feature the predicts and suggests the most appropriate values based on the user's input. Moreover, appropriate input fields for filtering the output data were designed, in order for user's to be able to customize their queries in a fashion relevant to their use-case. Screenshots of the web interface of LncBase v2 can be seen in Figures 7.4 and 7.5.

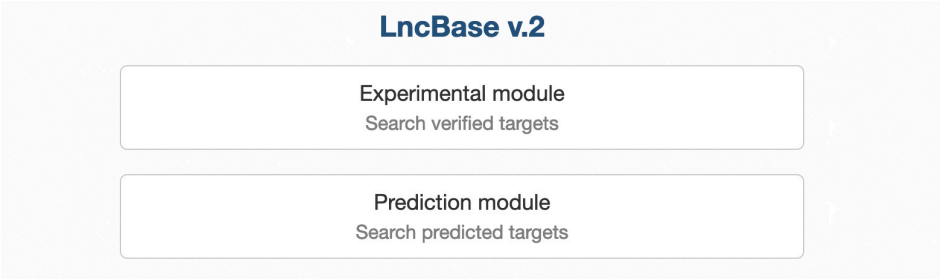



**Figure 7.4:** The web interface of LncBase v.2

As can be seen in Figure 7.4 the first step for a user is to select the module that they would like to use. The interfaces of the two modules are almost identical, barring the data filters available to the user.

After a user selects one of the modules, they can see the interface shown in Figure 7.5. From hereon until the end of the section, the term "lncRNA" is used interchangeably with "gene", meaning the gene from which the lncRNA is transcribed. The user inputs the miRNAs and/or the genes under examination, in order to query whether any interactions between the miRNAs and the lncRNAs a exist. Next, the output is retrieved and organized in three levels:

1. **Both modules:** information about the gene

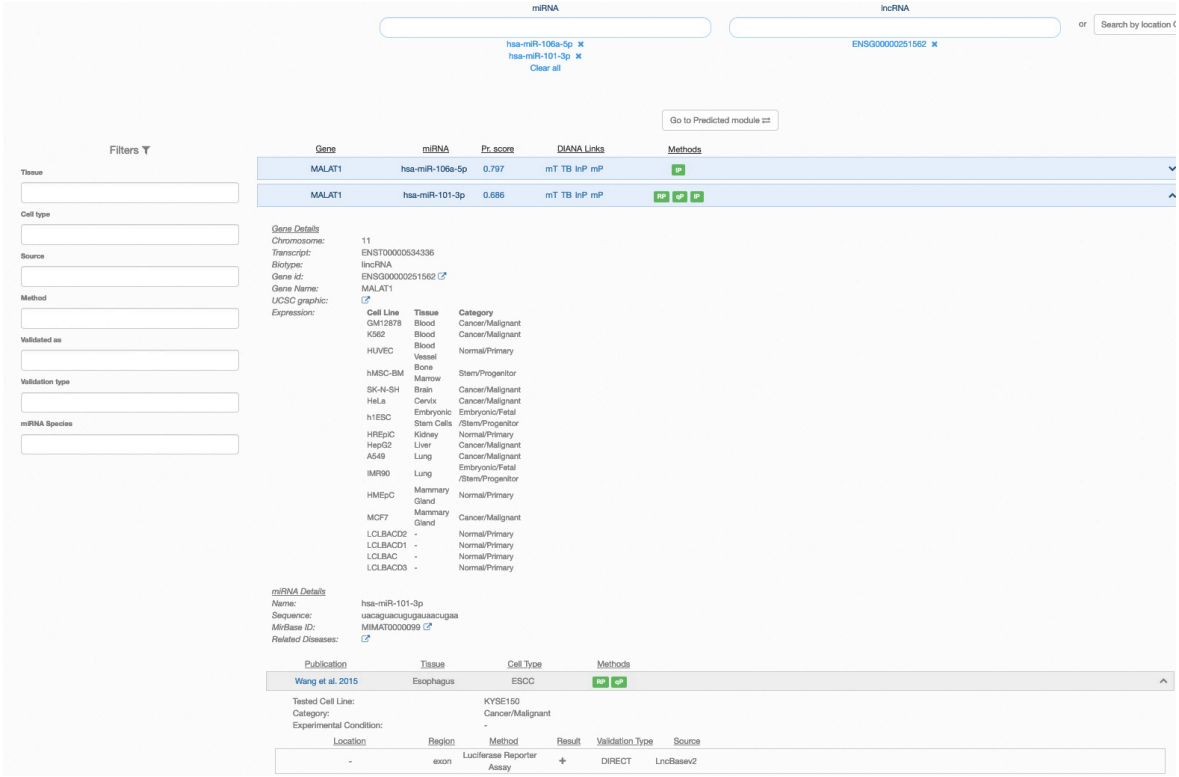



**Figure 7.5:** The web interface of LncBase v.2

2. **Experimental module**: information about the publication, from which the interaction was retrieved.
   **Predicted module:** binding site details

3. **Experimental module only:** information about the binding site specified on the publication

The filters for the "Predicted" module allow users to filter results based on the tissues, the cell types and other categories relevant to the cells themselves. On the other hand, the filters of the "Experimental" module are mostly focused around information of the publications on which the interactions appear, like the validation method that was followed in the publication, whether it was validated as positive or negative, directly or indirectly. Finally filters for species and tissues also exist in this module.

## 7.2 Cloud-based, scalable miRNA enrichment

In this section, we introduce a new cloud-based system, that was developed with the need for elastic and scalable resource allocation in mind. This platform leverages containers for software execution and it was tested using a containerized version of the software in Chapter 3. Finally, since this platform promotes reproducibility of experiments as well as fast and scalable execution of software, we decided to perform all experiments of the work outlined in Chapter 5 on it.

### 7.2.1 Containerization technologies

The computational analysis of large datasets has been established as an important part of the daily routine for scientists in many disciplines, shaping the field of *data-driven science*. Due to the large size of the datasets, such computations are assigned to the nodes of computational clusters owned by the academic or research institution to which the scientists belong. It is very common that such computational clusters are heterogeneous, consisting of machines of very diverse specifications (CPUs, memory, disk, etc.) or capabilities (e.g., support for FPGAs and other accelerators). This *heterogeneity* is due to the fact that these infrastructures (a) have to serve a variety of analysis tasks, each having its own special needs (e.g., to exploit accelerators), and (b) are usually built incrementally, with equipment units being procured at different (and maybe significantly distant) time periods, based on the availability of funds.

Similarly to any other scientific experiment, replicating and reproducing the results of a computational analysis is an important guarantee for its credibility. This is especially important nowadays, since there is an increasing concern in the research and academic community about the existence of a large number of scientific works that cannot be reproduced [Bak16]. Although it may be an exaggeration that we experience an ongoing *reproducibility crisis*, it is unarguable that this is an important phenomenon that needs to be addressed [Fan18]. This is why facilitating reproducibility has become an important topic of many research and academic disciplines.

In the context of data-driven science, facilitating reproducibility can be translated into making the datasets, the code, and the configurations used for the analysis openly available. Motivated by this need, approaches to pack up scientific computational experiments (e.g., RO-crate [CGSSR19]) have been introduced. Although such packages are really useful, their true potential is not easily unleashed due to the fact that computer environments (e.g., software libraries, packages, programming languages) are complex and rapidly evolving, making the reproducibility and extension of computational analyses challenging and tedious [Boe14]. For instance, although the code of a computational experiment may be openly available (e.g., on GitHub or other similar

repositories), in many cases its installation may require searching for old or, even, deprecated versions of third-party software, resolving conflicts between different versions of particular dependencies required by different software units, adapting the code to work on a different operating system, and so on. Given the fact that, usually, scientific software lacks comprehensive documentation, tasks like the previous may require significant technical skills that most scientists do not possess.

The missing piece in this puzzle is the use of *containerization technologies* (e.g., Docker, Singularity), along with experiment packaging, which has the potential to alleviate issues like these [Boe14, CS14, JMM+15]. Such technologies allow the code of a complex software unit to be packed up with its dependencies so it can be easily and reliably executed in a variety of computing environments (laptops, PCs, Cloud or HPC nodes, etc). These packaged software units are known as *container images*, and can further facilitate the reproducibility of computational analysis tasks since they are already configured and ready-to-use without requiring advanced technical skills for their installation. In addition, in the last years, a large number of scientific containers have been released in public repositories (e.g., in the time of writing, Biocontainers [dVLGAA+17] currently contain more than 2 076 containers).

It is evident, from the previous discussion, that various technologies, that could facilitate the work of scientists in the direction of data-driven science, have been introduced in the recent years. However, there is still a lack of open source implementations to combine these technologies into a platform to offer concrete services to this end. In particular, the most relevant platforms are EOSC Life's WorkflowHub[1] and Galaxy Europe[2]. The former is an under-development, federated repository of workflows that is based on the SEEK platform [WOK+15], however it does not support workflow execution by itself. The latter supports job scheduling, however its implementation is not designed to schedule computations based on the different capabilities of heterogeneous nodes that may reside inside the same cluster, thus it is unable to unleash the full potential of a heterogeneous cluster.

In this work, we attempt to alleviate the aforementioned issues by developing SCHeMa (Scheduler for scientific Containers on clusters of Heterogeneous Machines) an open-source[3] platform to facilitate the execution and reproducibility of computational experiments on heterogeneous clusters. The platform exploits containerization, experiment packaging, and workflow management technologies to ease reproducibility, while it leverages machine learning technologies to automatically identify the type of node that is more suitable to undertake each submitted computational task.

It is worth mentioning that a deployment of SCHeMa powers the on-demand com-

[1] WorkflowHub: https://workflowhub.eu/
[2] Galaxy Europe: https://usegalaxy.eu/
[3] SCHeMa's code repository: https://github.com/athenarc/schema (GNU/GPL license)

putations performed on the Cloud infrastructure[4] of the ELIXIR-GR community[5], which is a network of research and academic institutions in Greece, specialised in life-science research and its translation to medicine, biological sciences and society. The infrastructure consists of 45 physical nodes with 2 600 CPU cores, 24 TBs RAM memory and 1 PB of storage, in total. The cluster, apart from various infrastructure nodes that are used to host the cloud management services, comprises 32 "regular" nodes (28 cores, 512GB RAM), 2 "memory-enhanced" nodes (48 cores, 1TB RAM), 3 "GPU-enhanced" nodes (28 cores, 768GB RAM, 2 GPUs), and 8 "SSD-enhanced" nodes (28 cores, 512GB RAM). This infrastructure went into production in May 2021 and it is expected to host the majority of compute demands made by tools and services of the ELIXIR-GR community; before that, it was running in beta mode facilitating the execution of more than 820 computational tasks during that period.

### 7.2.2 Design objectives and system overview

SCHeMa, our open-source platform, has been designed and implemented with the aim to assist the work of scientists in the era of data-driven and reproducible science. In this context, our design had two main objectives: (a) to make the reproduction of any computational experiment performed in the platform as easy as possible, and (b) to allocate the resources of the underlying heterogeneous cluster as wisely as possible.

Regarding the first objective, as was mentioned in Section 7.2.1, a set of technologies should be combined together to achieve the desired goal. In addition, we had to identify the most appropriate technologies to be used from a multitude of available options. Our selection was made taking into consideration the maturity of the technologies to be used, their compatibility to each other, and the level of their dissemination in the scientific community. Based on these criteria, we selected to adopt CWL[6] to describe software interfaces and workflows. We selected RO-crate [CGSSR19] to create packages that represent computational experiments by storing the CWL description and the respective configuration of the software used along with the input and output datasets involved. Finally, we used containerization (Docker in particular) as a technology to enable the easy software execution, without requiring technical knowledge about building the involved software packages.

Regarding the second objective, we approached the problem of selecting the most appropriate type of machine in the cluster as a classification problem where the input features are metadata relevant to the job to be executed (i.e., characteristics of the

[4]HYPATIA, ELIXIR-GR Cloud Infrastructure: https://hypatia.athenarc.gr/
[5]The ELIXIR-GR website: https://elixir-greece.org/
[6]CWL website: https://www.commonwl.org/

inputs used), while each class represents a particular type of machine (e.g., regular-memory machine, large-memory machine, slow-disk machine, etc). In particular, we implemented a profiler that, after a user request, is able to analyse the execution behavior of a software of interest by monitoring its execution on a wide range of different inputs and configurations. The profiler trains and evaluates the accuracy of various classification approaches in different hyperparameter configurations and selects the one presenting the best performance as the prediction model to be used for any execution of the particular software in the future.

#### 7.2.2.1 Architecture

SCHeMa implements a wide range of functionalities to assist scientists in the data-driven and reproducible science era. Most notable are (a) the option to upload custom-made scientific containers or container-based workflows, (b) a wizard and an API that facilitate the execution of individual containers or workflows, (c) a machine-learning-based classifier that (after a required training phase) identifies the type of cluster node which is more appropriate to undertake a particular computational task, (d) a monitoring component that oversees the execution process and informs the users about the consumption of computational resources, (e) a wizard to transform executed analyses into RO-crate-based "experiment packages", and (f) a wizard to facilitate interconnection with open data repository services. Figure 7.6 summarises SCHeMa's architecture, which supports these (and some extra) functionalities. In the following sections we discuss SCHeMa's external dependencies, as well as the implementation details of its internal software components.

##### 7.2.2.1.1 External dependencies

SCHeMa's function depends on the existence of a couple of external installations, the most important of which are the following:

- A *Kubernetes*[7] *installation* should be deployed on the computational cluster to be used. Kubernetes undertakes the low-level orchestration and monitoring of the computational jobs and interacts with the *Job Classifier* component that provides the feed of requested jobs along with recommendations about the most suitable types of node for each job.
- A *distributed file system* should be installed on the hard disks of the machines of the cluster. This file system is used to store input/output data required/produced by the computational tasks. Currently the implementation supports NFS volumes.

[7]Kubernetes website: https://kubernetes.io/

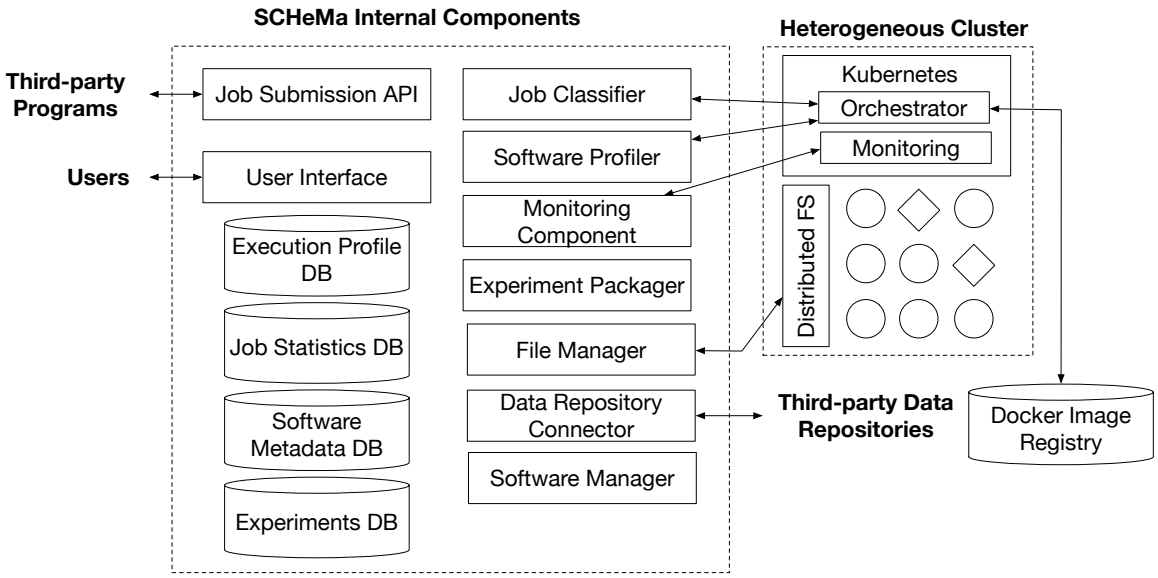



**Figure 7.6:** The architecture of SCHeMa.

- A private *Docker image registry* supporting TLS security and user authentication is required. SCHeMa uses this registry to upload container images. This ensures that user-uploaded images remain isolated from the outside world (especially those meant to be private).

#### 7.2.2.2 User Interface

A Web-based user interface has been developed using the Yii2 PHP framework[8]. It comprises various wizards that offer execution, reproducibility, and monitoring functionalities for computational experiments (see also Section 7.2.4). Of course, the function of all these wizards heavily relies on the functionalities provided by the rest of the components, on which we elaborate in the next sections.

#### 7.2.2.3 Software Manager

This component implements functionalities to upload (or update) container images and workflows. First of all, any involved container should be loaded in the *Docker Image Registry*. Additionally, in both cases (i.e., individual container or workflow), a CWL description is required and the corresponding metadata, which describe the required inputs and dependencies of the involved software packages (the latter only for workflows), are loaded in the *Software Metadata DB*. These data are used by various components, e.g., they are used by the *User Interface* wizard to automatically

[8]Yii2 website: https://www.yiiframework.com/

display a form containing one input field for each input parameter of the software (see also the example of Section 7.2.2.2).

#### 7.2.2.4 Job Submission API

Apart from manually submitting computational jobs through the UI wizards, the users are able to also submit batches of jobs programmatically using an implemented API. This API is based on GA4GH's Task Execution Schemas (TES)[9] and Workflow Execution Service (WES)[10] API specifications. The API supports batch execution and monitoring of computational tasks and can be used by user-implemented scripts and programs.

#### 7.2.2.5 Software Profiler

This component leverages machine learning to produce (after user request) "execution profiles" for software that has been uploaded on SCHeMa. For each software, this profiler builds a classification model that attempts to map candidate jobs of this software to one class of nodes, which corresponds to the type of node that is appropriate to undertake the computations of the job. As an indicative example, a cluster could have two types of nodes, one with regular-sized main memory and another one with large memory (ideally, to be used by memory-intensive tasks). In this case, given a particular software of interest, the objective of the *Software Profiler* would be to train a (binary) classification model to assign each job of this software to a regular-memory node or to a large-memory one.

The profiling process goes as follows: first, the uploader of the software provides a set of alternative values/files for its input parameters (a relevant *User Interface* wizard exists to collect this information)[11]. Then, the system runs the software for all input combinations, collects the resource consumption of each run, and creates a dataset of samples, where each sample is the combination of provided inputs and the recorded consumption of resources. Based on this dataset, the system trains and optimizes different models based on various classifiers (e.g., logistic regression, random forest) using the grid search approach[12]. The model presenting the best accuracy is selected and stored in the *Execution Profile DB*. All the stored models are exploited by the *Job Classifier* component, when the execution of the corresponding software is requested.

[9]TES specification: https://github.com/ga4gh/task-execution-schemas

[10]WES specification: https://github.com/ga4gh/workflow-execution-service-schemas

[11]Since each software has a different set of required inputs, it follows that the size of feature vectors is different for tasks related to different software packages.

[12]The grid for each classification approach is build according to its main hyperparameters; for each hyperparameter, a set of commonly used values are examined.

All examined models are implemented using the scikit-learn[13] Python library.

#### 7.2.2.6 Job Classifier

This component receives as input job submissions from the corresponding wizard of the *User Interface* and the *Job Submission API*. As a first step, it searches if there are any trained (by the *Software Profiler*) models for the involved software packages. If so, it exploits this model to create a suggestion for the most appropriate node type to undertake the job and then propagates it to the Kubernetes scheduler through the Kubernetes API. The scheduler takes into consideration this suggestion and schedules the corresponding job ensuring that the determined node type will be used. If there is no trained model, then the job request is propagated to the Kubernetes scheduler without any indication for the node type to be used; the Kubernetes scheduler decides the most appropriate node, based on the available resources of each computational node.

It is worth mentioning that job profiling and classification functionalities in SCHeMa are currently experimental and, thus, subject to various limitations. In fact, the evaluation of the classification accuracy needs an extensive experimentation that requires a large number of diverse software packages to be examined. Although we have started working in this direction, this type of evaluation is planned to be included in a future work and it is out of scope for the current publication. However, preliminary experiments showed encouraging results for some of the classification tasks (e.g., classify according to the memory consumption) based on software packages supported by the ELIXIR-GR Cloud.

#### 7.2.2.7 Data Repository Connector.

This component implements an interconnection with various open data repositories. Currently, two repository services are supported: Zenodo[14] and HELIX[15]. *Data Repository Connector* takes advantage of the APIs provided by the repository services to download/upload datasets from/to them. A relevant *User Interface* wizard exploits this functionality to enable users to download existing datasets from one of the supported repositories and then use them for their analyses or to directly upload the output of a particular analysis on a selected data repository.

[13]https://scikit-learn.org/stable/
[14]Zenodo open repository: https://zenodo.org/
[15]Hellenic Data Service (HELIX) repository: https://hellenicdataservice.gr

#### 7.2.2.8 Experiment Packager.

This component is responsible to create "experiment packages" (according to the RO-crate [CGSSR19] specification) from previously executed computational jobs, after user request. A relevant wizard is implemented in the *User Interface* and, based on it, the users can easily create packages that incorporate several metadata for the selected experiment, such as the software used, its configuration, the input and output dataset, or even the DOI of a relevant publication. To collect the required information, the *Experiment Packager* component communicates with the *Job Statistics* and the *Software Metadata* databases. The resulting packages are stored inside an *Experiments DB* but it is also possible for the users to download the packages to their local computer storage. The easy creation of RO-crate packages is one of the main SCHeMa functionalities that facilitate computational experiment reproducibility. An additional, powerful option for users is the ability to make the RO-crate packages openly available to other users of the platform, who are interested in reproducing the experiment described by the package. Thus, other users of SCHeMa have access to all publicly available RO-crates existing on the platform. Of course, making the packages publicly available is optional.

#### 7.2.2.9 Monitoring Component

This component aggregates data coming from the low-level logging and monitoring mechanisms of Kubernetes to create insightful reports about the jobs being executed in the cluster. All jobs, both those submitted through the UI and those submitted through the API are being considered. It also provides statistics about the load of the cluster. The component constantly communicates with the *Job Statistics DB* to update its recorded information or to use it for the production of the aggregated statistics. Finally, it propagates data to the job execution wizard so the user can monitor each job's output and status.

### 7.2.3 Availability & Installation

As mentioned, the code of SCHeMa is publicly available on GitHub (see in Section 7.2.1 for details). During its development, special care was given to support easy ways to install SCHeMa. Currently, cluster administrators who are interested in installing SCHeMa to their infrastructure have two options available: (a) to deploy the application in a physical or virtual machine with access to all external dependencies (which is more time consuming, since all dependencies must be installed and configured separately, but provides better data persistence) or (b) to deploy the prepackaged version of SCHeMa, which is deployed inside a Kubernetes cluster. The

advantage of the latter approach is that it uses Helm[16] to deploy the application as well as all dependencies outlined in Section 7.2.2.1.1 as containers inside the Kubernetes cluster. Data storage is facilitated by leveraging Kubernetes volume storage options, but it is important to note here that, depending on the storage option provided, data persistence is not guaranteed in the case of downtime in the Kubernetes cluster. However, deployment utilizing the second approach is a really streamlined process that only requires an existing Kubernetes cluster and removes the need for pre-configured components.

### 7.2.4 Quick Tour of the Interface

For the interest of space, in this section we indicatively describe only the main functionalities around the execution of scientific containers (however, the same functionalities for workflows are very similar). It should be highlighted that it is practically impossible for all the offered functionalities to be described.

A list of all containers in the *Docker Image Registry*, to which the connected user has access, can be displayed after clicking on the "Software" top menu item of SCHeMa. In Figure 7.7, a screenshot of SCHeMa's interface after this action took place is illustrated. By hitting the arrow-shaped button of any entry, the user is redirected to the job submission wizard of the corresponding software[17]. This wizard consists of a form that contains one input field for each input parameter of the software; the form is automatically generated based on the CWL description of the selected software (which is stored in the *Software Metadata DB*). After providing input values for all required fields and hitting "Run" the execution starts and the progress can be monitored through the UI. Closing the browser tab is possible without interrupting the execution; the user may revisit the execution page for this job by selecting the corresponding entry in the "Job History" page (accessible again through a top menu item). In the same page, the user can also select to rerun a previously completed execution or to create an RO-crate object based on it. Finally, any output files, which are stored in the distributed storage space, can be found in the "Data" page.

## 7.3 Demonstration

During the conference, we will explain the concept of reproducible and data-driven science and its requirements to the audience and we will demonstrate SCHeMa's rele-

[16] https://helm.sh/

[17] Hitting the diagram-shaped button the user can start the machine learning profiling of the same software.

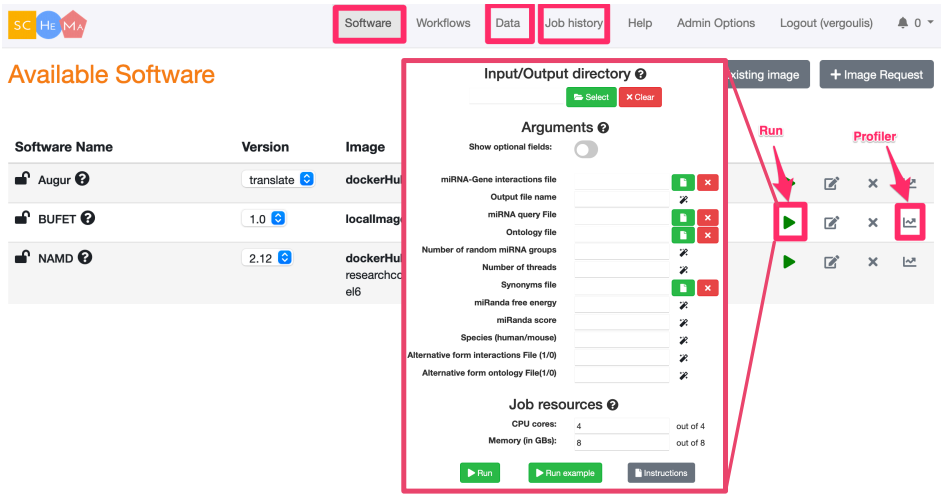



**Figure 7.7:** A screenshot of SCHeMa's interface.

vant functionalities elaborating on how they help in this context. For this demonstration, we will exploit SCHeMa's deployment for the ELIXIR-GR Cloud Infrastructure, which is based on a relatively large computational cluster (see Section 7.2.1). We will examine SCHeMa's capabilities in real-time by following any audience-defined scenario, however we will also demonstrate some interesting scenarios we have identified.

The main scenario, is based on executing a pre-loaded scientific container using the corresponding wizard. We will run the container twice: once without exploiting its pre-trained execution profile and once leveraging it and we will prompt the users to observe any differences (e.g., without using the profile a node with unnecessary large resources may be selected, thus its "precious" CPU time may be wasted instead of being used for a more demanding job). We will also navigate the user through the steps required to create the execution profile for a particular tool. After the execution, we will also guide the audience through the process of uploading the output files on an open data repository and packaging the whole experiment into an RO-crate object. Finally, we will also use the created object to recreate the respective computational experiment in another system, showing how the RO-crate support facilitates reproducibility.

## 7.4 Conclusion

In this chapter we outlined data-management techniques, applied to three online miRNA analysis tools with the aim of assisting Bioinformatics researchers interested in performing analyses in a real-time online manner. In the second part of this chapter, we introduced and showcased SCHeMa, an open-source platform that aims to

assist the work of scientists in the data-driven science era through facilitating the execution, reproducibility, and monitoring of computational experiments on heterogeneous computational clusters. To this end, it leverages various technologies like containerization, experiment packaging, workflow management, and machine learning.

# Chapter 8

# Conclusions and future work

In this thesis, we focused on scalable data-driven analysis for short RNA molecules, also known as miRNAs. In Chapter 2 we introduced the background of this work, aw well as relevant work regarding genes, miRNAs and enrichment analyses.

Then, in Chapter 3 we introduce a data management approach to the miRNA enrichment analysis by using data structures relevant to the data, after a thorough examination; we manage to reduce execution times of the analysis and we illustrate that through extensive evaluation based on a variety of different data.

Moreover, in Chapter 4 we generalize the problem of association testing through unbiased statistics and we perform an inspection of the data, trying to find overlaps that lead to redundant operations. Then, we introduce two novel indices, one that eliminates redundant operations and one that predicts which associations will be insignificant. Using the former, we manage to cut the required execution times by half, while, through the utilization of the latter, the execution times were further reduced by an order of magnitude.

In Chapter 5 we attempt to increase the quality of the established enrichment test methods. More specifically, we first show that the first of the tests (Fisher's exact test) is not suitable for association testing involving miRNAs, genes and biological functions due to certain biases that render the assumptions of the hypergeometric distribution invalid. Then we modify the test metric of the unbiased statistical test and illustrate through extensive evaluation, that the modified metric reduced the number of false negative results when compared to the established test; this holds for all types of relevant data.

In Chapter 6 we recognize the need to run the enrichment analysis in almost real time (especially regarding web applications) and this lead us to use supervised machine learning techniques to train a model that can predict p-values.First, we created a data set containing features relevant to the statistical procedure and then we used these features to train a variety of models and measured their performance using

the cross-validation method. Furthermore, we calculated the optimal parameters for each model using grid search.

Finally, in Chapter 7 we describe three web tools that facilitate online miRNA data management and processing. Moreover, we introduce a new platform, that allows easy and reproducible research through the use of containerization technologies like Kubernetes. This platform was utilized to execute the experiments of Chapter 5.

## 8.1 Future Work

Possible future directions for the work presented in this thesis include:

- Discovery of other statistical tests in Bioninformatics that use unbiased statistics, in order to see if the techniques introduced in this work can also be applied in them, in order to achieve execution speedup.
- Application of the techniques introduced in Chapter 4 to other real-world association testing analyses to try and accelerate them.
- Performing a survey to find whether the biases described in this work, in regard to the hypergeometric distribution affect other association tests in Bioinformatics.

# Bibliography


[ABB+00] 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. Gene ontology: tool for the unification of biology. *Nat Genet*, 25(1):25–29, 05 2000.

[ACV+19] Francesco Angelucci, Katerina Cechova, Martin Valis, Kamil Kuca, Bing Zhang, and Jakub Hort. Micrornas in alzheimer's disease: Diagnostic markers or therapeutic agents? *Frontiers in Pharmacology*, 10:665, 2019.

[AH10] Simon Anders and Wolfgang Huber. Differential expression analysis for sequence count data. *Genome Biology*, 11(10):R106, Oct 2010.

[AMS+96] Rakesh Agrawal, Heikki Mannila, Ramakrishnan Srikant, Hannu Toivonen, A Inkeri Verkamo, et al. Fast discovery of association rules. *Advances in knowledge discovery and data mining*, 12(1):307–328, 1996.

[Arb10] John Arbuthnot. Ii. an argument for divine providence, taken from the constant regularity observ'd in the births of both sexes. by dr. john arbuthnott, physitian in ordinary to her majesty, and fellow of the college of physitians and the royal society. *Philosophical Transactions of the Royal Society of London*, 27(328):186–190, 1710.

[AS07] ZC Ade, AS anhd Wright and DJ States. Gene2mesh [internet], 03 2007. Ann Arbor (MI): National Center for Integrative Biomedical Informatics.

[Bak16] Monya Baker. 1,500 scientists lift the lid on reproducibility. *Nature*, 533:452–454, 2016.

[BJ07] Éric Brian and Marie Jaisson. *Physico-Theology and Mathematics (1710–1794)*, pages 1–25. Springer Netherlands, Dordrecht, 2007.

[BLGJ15] Thomas Bleazard, Janine A Lamb, and Sam Griffiths-Jones. Bias in microrna functional enrichment analysis. *Bioinformatics*, 2015.

[Boe14] Carl Boettiger. An introduction to docker for reproducible research, with examples from the R environment. *CoRR*, abs/1410.0846, 2014.

[Bor12] Christian Borgelt. Frequent item set mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(6):437–456, 2012.

[CGSSR19] Eoghan Ó Carragáin, Carole Goble, Peter Sefton, and Stian Soiland-Reyes. A lightweight approach to research object data packaging. In *Bioinformatics Open Source Conference (BOSC) 2019*, 2019.

[chi11] *Tests for Nominal Scale Data: Chi-Square and Fisher Exact Test*, chapter 8, pages 155–191. John Wiley & Sons, Ltd, 2011.

[cmh03] *Building and Applying Logistic Regression Models*, chapter 6, pages 211–266. John Wiley & Sons, Ltd, 2003.

[CS14] Ryan Chamberlain and Jennifer Schommer. Using docker to support reproducible research. *DOI: https://doi. org/10.6084/m9. figshare*, 1101910:44, 2014.

[CSY+17] Chih-Hung Chou, Sirjana Shrestha, Chi-Dung Yang, Nai-Wen Chang, Yu-Ling Lin, Kuang-Wen Liao, Wei-Chi Huang, Ting-Hsuan Sun, Siang-Jyun Tu, Wei-Hsiang Lee, Men-Yee Chiew, Chun-San Tai, Ting-Yen Wei, Tzi-Ren Tsai, Hsin-Tzu Huang, Chung-Yu Wang, Hsin-Yi Wu, Shu-Yi Ho, Pin-Rong Chen, Cheng-Hsun Chuang, Pei-Jung Hsieh, Yi-Shin Wu, Wen-Liang Chen, Meng-Ju Li, Yu-Chun Wu, Xin-Yi Huang, Fung Ling Ng, Waradee Buddhakosai, Pei-Chun Huang, Kuan-Chun Lan, Chia-Yen Huang, Shun-Long Weng, Yeong-Nan Cheng, Chao Liang, Wen-Lian Hsu, and Hsien-Da Huang. miRTarBase update 2018: a resource for experimentally validated microRNA-target interactions. *Nucleic Acids Research*, 46(D1):D296–D302, 11 2017.

[DKA+12] Ian Dunham, Anshul Kundaje, Shelley F. Aldred, Patrick J. Collins, Carrie A. Davis, Francis Doyle, Charles B. Epstein, Seth Frietze, Jennifer Harrow, Rajinder Kaul, Jainab Khatun, Bryan R. Lajoie,

Stephen G. Landt, Bum-Kyu Lee, Florencia Pauli, Kate R. Rosenbloom, Peter Sabo, Alexias Safi, Amartya Sanyal, Noam Shoresh, Jeremy M. Simon, Lingyun Song, Nathan D. Trinklein, Robert C. Altshuler, Ewan Birney, James B. Brown, Chao Cheng, Sarah Djebali, Xianjun Dong, Jason Ernst, Terrence S. Furey, Mark Gerstein, Belinda Giardine, Melissa Greven, Ross C. Hardison, Robert S. Harris, Javier Herrero, Michael M. Hoffman, Sowmya Iyer, Manolis Kellis, Pouya Kheradpour, Timo Lassmann, Qunhua Li, Xinying Lin, Georgi K. Marinov, Angelika Merkel, Ali Mortazavi, Stephen C. J. Parker, Timothy E. Reddy, Joel Rozowsky, Felix Schlesinger, Robert E. Thurman, Jie Wang, Lucas D. Ward, Troy W. Whitfield, Steven P. Wilder, Weisheng Wu, Hualin S. Xi, Kevin Y. Yip, Jiali Zhuang, Bradley E. Bernstein, Eric D. Green, Chris Gunter, Michael Snyder, Michael J. Pazin, Rebecca F. Lowdon, Laura A. L. Dillon, Leslie B. Adams, Caroline J. Kelly, Julia Zhang, Judith R. Wexler, Peter J. Good, Elise A. Feingold, Gregory E. Crawford, Job Dekker, Laura Elnitski, Peggy J. Farnham, Morgan C. Giddings, Thomas R. Gingeras, Roderic Guigó, Timothy J. Hubbard, W. James Kent, Jason D. Lieb, Elliott H. Margulies, Richard M. Myers, John A. Stamatoyannopoulos, Scott A. Tenenbaum, Zhiping Weng, Kevin P. White, Barbara Wold, Yanbao Yu, John Wrobel, Brian A. Risk, Harsha P. Gunawardena, Heather C. Kuiper, Christopher W. Maier, Ling Xie, Xian Chen, Tarjei S. Mikkelsen, Shawn Gillespie, Alon Goren, Oren Ram, Xiaolan Zhang, Li Wang, Robbyn Issner, Michael J. Coyne, Timothy Durham, Manching Ku, Thanh Truong, Matthew L. Eaton, Alex Dobin, Andrea Tanzer, Julien Lagarde, Wei Lin, Chenghai Xue, Brian A. Williams, Chris Zaleski, Maik Röder, Felix Kokocinski, Rehab F. Abdelhamid, Tyler Alioto, Igor Antoshechkin, Michael T. Baer, Philippe Batut, Ian Bell, Kimberly Bell, Sudipto Chakrabortty, Jacqueline Chrast, Joao Curado, Thomas Derrien, Jorg Drenkow, Erica Dumais, Jackie Dumais, Radha Duttagupta, Megan Fastuca, Kata Fejes-Toth, Pedro Ferreira, Sylvain Foissac, Melissa J. Fullwood, Hui Gao, David Gonzalez, Assaf Gordon, Cédric Howald, Sonali Jha, Rory Johnson, Philipp Kapranov, Brandon King, Colin Kingswood, Guoliang Li, Oscar J. Luo, Eddie Park, Jonathan B. Preall, Kimberly Presaud, Paolo Ribeca, Daniel Robyr, Xiaoan Ruan, Michael Sammeth, Kuljeet Singh Sandhu, Lorain Schaeffer, Lei-Hoon See, Atif Shahab, Jorgen Skancke, Ana Maria Suzuki, Hazuki Takahashi, Hagen Tilgner, Diane Trout, Nathalie Walters, Huaien Wang, Yoshi-

hide Hayashizaki, Alexandre Reymond, Stylianos E. Antonarakis, Gregory J. Hannon, Yijun Ruan, Piero Carninci, Cricket A. Sloan, Katrina Learned, Venkat S. Malladi, Matthew C. Wong, Galt P. Barber, Melissa S. Cline, Timothy R. Dreszer, Steven G. Heitner, Donna Karolchik, Vanessa M. Kirkup, Laurence R. Meyer, Jeffrey C. Long, Morgan Maddren, Brian J. Raney, Linda L. Grasfeder, Paul G. Giresi, Anna Battenhouse, Nathan C. Sheffield, Kimberly A. Showers, Darin London, Akshay A. Bhinge, Christopher Shestak, Matthew R. Schaner, Seul Ki Kim, Zhuzhu Z. Zhang, Piotr A. Mieczkowski, Joanna O. Mieczkowska, Zheng Liu, Ryan M. McDaniell, Yunyun Ni, Naim U. Rashid, Min Jae Kim, Sheera Adar, Zhancheng Zhang, Tianyuan Wang, Deborah Winter, Damian Keefe, Vishwanath R. Iyer, Meizhen Zheng, Ping Wang, Jason Gertz, Jost Vielmetter, E. Partridge, Katherine E. Varley, and Clarke Gasper. An integrated encyclopedia of dna elements in the human genome. *Nature*, 489(7414):57–74, Sep 2012.

[DKM+03] Sorin Drăghici, Purvesh Khatri, Rui P. Martins, G.Charles Ostermeier, and Stephen A. Krawetz. Global functional profiling of gene expression. *Genomics*, 81(2):98 – 104, 2003.

[dVLGAA+17] Felipe da Veiga Leprevost, Björn A Grüning, Saulo Alves Aflitos, Hannes L Röst, Julian Uszkoreit, Harald Barsnes, Marc Vaudel, Pablo Moreno, Laurent Gatto, Jonas Weber, et al. Biocontainers: an open-source and community-driven framework for software standardization. *Bioinformatics*, 33(16):2580–2582, 2017.

[Fan18] Daniele Fanelli. Opinion: Is science really facing a reproducibility crisis, and do we need it to? *Proceedings of the National Academy of Sciences (PNAS)*, 115(11):2628–2631, 2018.

[FBY92] William B Frakes and Ricardo Baeza-Yates. Information retrieval: data structures and algorithms. 1992.

[Fis92] R. A. Fisher. *Statistical Methods for Research Workers*, pages 66–70. Springer New York, New York, NY, 1992.

[GBIH+17] Felipe Gutierrez-Barragan, Vamsi K. Ithapu, Chris Hinrichs, Camille Maumet, Sterling C. Johnson, Thomas E. Nichols, and Vikas Singh. Accelerating permutation testing in voxel-wise analysis through subspace tracking: A new plugin for snpm. *NeuroImage*, 159:79 – 98, 2017.

[Gér19] A. Géron. *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, Incorporated, 2019.

[GMCS20] Adrian Garcia-Moreno and Pedro Carmona-Saez. Computational methods and software tools for functional analysis of mirna data. *Biomolecules*, 10(9):1252, Aug 2020.

[GSGV+15] Alba Gutiérrez-Sacristán, Solène Grosdidier, Olga Valverde, Marta Torrens, Àlex Bravo, Janet Piñero, Ferran Sanz, and Laura I. Furlong. PsyGeNET: a knowledge platform on psychiatric disorders and their genes. *Bioinformatics*, 31(18):3075–3077, 05 2015.

[GVP+14] Georgios Georgakilas, Ioannis S. Vlachos, Maria D. Paraskevopoulou, Peter Yang, Yuhong Zhang, Aris N. Economides, and Artemis G. Hatzigeorgiou. microtss: accurate microrna transcription start site identification reveals a significant number of divergent pri-mirnas. *Nature Communications*, 5(1):5700, Dec 2014.

[GVZ+15] Georgios Georgakilas, Ioannis S. Vlachos, Konstantinos Zagganas, Thanasis Vergoulis, Maria D. Paraskevopoulou, Ilias Kanellos, Panayiotis Tsanakas, Dimitris Dellis, Athanasios Fevgas, Theodore Dalamagas, and Artemis G. Hatzigeorgiou. DIANA-miRGen v3.0: accurate characterization of microRNA promoters and their regulators. *Nucleic Acids Research*, 44(D1):D190–D195, 11 2015.

[Han14] John M. Hancock. *Jaccard Distance (Jaccard Index, Jaccard Similarity Coefficient)*. American Cancer Society, 2014.

[HHL+17] Ning-Bo Hao, Ya-Fei He, Xiao-Qin Li, Kai Wang, and Rui-Ling Wang. The role of mirna and lncrna in gastric cancer. *Oncotarget*, 8(46):81572–81582, 2017.

[HSA+05] Ada Hamosh, Alan F. Scott, Joanna S. Amberger, Carol A. Bocchini, and Victor A. McKusick. Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders. *Nucleic Acids Research*, 33($suppl_1$):D514–D517, 01 2005.

[HWB+11] Thomas B Hansen, Erik D Wiklund, Jesper B Bramsen, Sune B Villadsen, Aaron L Statham, Susan J Clark, and Jørgen Kjems. mirna-dependent gene silencing involving ago2-mediated cleavage of a circular antisense rna. *The EMBO Journal*, 30(21):4414–4422, 2011.

[JEA+04] Bino John, Anton J Enright, Alexei Aravin, Thomas Tuschl, Chris Sander, and Debora S Marks. Human microrna targets. *PLoS Biol*, 2(11), 10 2004.

[JML12] Ashwini Jeggari, Debora S Marks, and Erik Larsson. miRcode: a map of putative microRNA target sites in the long non-coding transcriptome. *Bioinformatics*, 28(15):2062–2063, 05 2012.

[JMM+15] Ivo Jimenez, Carlos Maltzahn, Adam Moody, Kathryn Mohror, Jay Lofstead, Remzi Arpaci-Dusseau, and Andrea Arpaci-Dusseau. The role of container technology in reproducible computer systems research. In *2015 IEEE International Conference on Cloud Engineering*, pages 379–385. IEEE, 2015.

[Kar72] Richard M. Karp. *Reducibility among Combinatorial Problems*, pages 85–103. Springer US, Boston, MA, 1972.

[KG00a] Minoru Kanehisa and Susumu Goto. Kegg: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Research*, 28(1):27–30, 2000.

[KG00b] Minoru Kanehisa and Susumu Goto. KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Research*, 28(1):27–30, 01 2000.

[KGJ13] Ana Kozomara and Sam Griffiths-Jones. miRBase: annotating high confidence microRNAs using deep sequencing data. *Nucleic Acids Research*, 42(D1):D68–D73, 11 2013.

[KPC+17a] Dimitra Karagkouni, Maria D Paraskevopoulou, Serafeim Chatzopoulos, Ioannis S Vlachos, Spyros Tastsoglou, Ilias Kanellos, Dimitris Papadimitriou, Ioannis Kavakiotis, Sofia Maniou, Giorgos Skoufos, Thanasis Vergoulis, Theodore Dalamagas, and Artemis G Hatzigeorgiou. DIANA-TarBase v8: a decade-long collection of experimentally supported miRNA–gene interactions. *Nucleic Acids Research*, 46(D1):D239–D245, 11 2017.

[KPC+17b] Dimitra Karagkouni, Maria D Paraskevopoulou, Serafeim Chatzopoulos, Ioannis S Vlachos, Spyros Tastsoglou, Ilias Kanellos, Dimitris Papadimitriou, Ioannis Kavakiotis, Sofia Maniou, Giorgos Skoufos, Thanasis Vergoulis, Theodore Dalamagas, and Artemis G Hatzigeorgiou. DIANA-TarBase v8: a decade-long collection of experimentally supported miRNA–gene interactions. *Nucleic Acids Research*, 46(D1):D239–D245, 11 2017.

[KSK+16] Minoru Kanehisa, Yoko Sato, Masayuki Kawashima, Miho Furumichi, and Mao Tanabe. Kegg as a reference resource for gene and protein annotation. *Nucleic Acids Research*, 44(D1):D457–D462, 2016.

[KVS+14a] Ilias Kanellos, Thanasis Vergoulis, Dimitris Sacharidis, Theodore Dalamagas, Artemis Hatzigeorgiou, Stelios Sartzetakis, and Timos Sellis. Mr-microt: A mapreduce-based microrna target prediction method. In *Proceedings of the 26th International Conference on Scientific and Statistical Database Management*, SSDBM '14, New York, NY, USA, 2014. Association for Computing Machinery.

[KVS+14b] Ilias Kanellos, Thanasis Vergoulis, Dimitris Sacharidis, Theodore Dalamagas, Artemis Hatzigeorgiou, Stelios Sartzetakis, and Timos Sellis. Mr-microt: A mapreduce-based microrna target prediction method. In *Proceedings of the 26th International Conference on Scientific and Statistical Database Management*, SSDBM '14, New York, NY, USA, 2014. Association for Computing Machinery.

[LBB] Benjamin P. Lewis, Christopher B. Burge, and David P. Bartel. Conserved seed pairing, often flanked by adenosines, indicates that thousands of human genes are microrna targets. *Cell*, 120(1):15–20, 2016/12/28.

[LFA93] R. C. Lee, R. L. Feinbaum, and V. Ambros. CellThe C. elegans heterochronic gene lin-4 encodes small RNAs with antisense complementarity to lin-14. *Cell*, 75(5):843–854, Dec 1993.

[LK12] Yu Li and Kris V. Kowdley. Micrornas in common human diseases. *Genomics, Proteomics & Bioinformatics*, 10(5):246–253, 2012.

[LKC+12] Gabriel B. Loeb, Aly A. Khan, David Canner, Joseph B. Hiatt, Jay Shendure, Robert B. Darnell, Christina S. Leslie, and Alexander Y. Rudensky. Transcriptome-wide mir-155 binding map reveals widespread noncanonical microrna targeting. *Molecular Cell*, 48(5):760–770, Dec 2012.

[McD14] JH McDonald. *Multiple comparisons: Controlling the false discovery rate: Benjamini–Hochberg procedure*. 2014.

[Moh20] Eiman MA Mohammed. Environmental influencers, microrna, and multiple sclerosis. *Journal of Central Nervous System Disease*, 12:1179573519894955, 2020.

[PEC+13] Jason Piper, Markus C. Elze, Pierre Cauchy, Peter N. Cockerill, Constanze Bonifer, and Sascha Ott. Wellington: a novel method for the accurate identification of digital genomic footprints from DNase-seq data. *Nucleic Acids Research*, 41(21):e201–e201, 09 2013.

[PGK+12] Maria D. Paraskevopoulou, Georgios Georgakilas, Nikos Kostoulas, Martin Reczko, Manolis Maragkakis, Theodore M. Dalamagas, and Artemis G. Hatzigeorgiou. DIANA-LncBase: experimentally verified and computationally predicted microRNA targets on long non-coding RNAs. *Nucleic Acids Research*, 41(D1):D239–D245, 11 2012.

[PGK+13] Maria D. Paraskevopoulou, Georgios Georgakilas, Nikos Kostoulas, Ioannis S. Vlachos, Thanasis Vergoulis, Martin Reczko, Christos Filippidis, Theodore Dalamagas, and A.G. Hatzigeorgiou. Diana-microt web server v5.0: service integration into mirna functional analysis workflows. *Nucleic Acids Research*, 41(W1):W169–W173, 2013.

[PJ84] C. S. Peirce and Joseph Jastrow. On small differences in sensation. *Memoirs of the National Academy of Sciences*, 3:75–83, 1884.

[PLM+17a] Natalia Pinzón, Blaise Li, Laura Martinez, Anna Sergeeva, Jessy Presumey, Florence Apparailly, and Hervé Seitz. microrna target prediction programs predict many false positives. *Genome Research*, 27(2):234–245, 2017.

[PLM+17b] Natalia Pinzón, Blaise Li, Laura Martinez, Anna Sergeeva, Jessy Presumey, Florence Apparailly, and Hervé Seitz. microrna target prediction programs predict many false positives. *Genome Research*, 27(2):234–245, 2017.

[PRASP+19a] Janet Piñero, Juan Manuel Ramírez-Anguita, Josep Saüch-Pitarch, Francesco Ronzano, Emilio Centeno, Ferran Sanz, and Laura I Furlong. The DisGeNET knowledge platform for disease genomics: 2019 update. *Nucleic Acids Research*, 48(D1):D845–D855, 11 2019.

[PRASP+19b] Janet Piñero, Juan Manuel Ramírez-Anguita, Josep Saüch-Pitarch, Francesco Ronzano, Emilio Centeno, Ferran Sanz, and Laura I Furlong. The DisGeNET knowledge platform for disease genomics: 2019 update. *Nucleic Acids Research*, 48(D1):D845–D855, 11 2019.

[PRASP+19c] Janet Piñero, Juan Manuel Ramírez-Anguita, Josep Saüch-Pitarch, Francesco Ronzano, Emilio Centeno, Ferran Sanz, and Laura I Fur-

long. The DisGeNET knowledge platform for disease genomics: 2019 update. *Nucleic Acids Research*, 48(D1):D845–D855, 11 2019.

[PVK+15] Maria D. Paraskevopoulou, Ioannis S. Vlachos, Dimitra Karagkouni, Georgios Georgakilas, Ilias Kanellos, Thanasis Vergoulis, Konstantinos Zagganas, Panayiotis Tsanakas, Evangelos Floros, Theodore Dalamagas, and Artemis G. Hatzigeorgiou. DIANA-LncBase v2: indexing microRNA targets on non-coding transcripts. *Nucleic Acids Research*, 44(D1):D231–D238, 11 2015.

[REJ17] Raul Rodriguez-Esteban and Xiaoyu Jiang. Differential gene expression in disease: a comparison between high-throughput studies and the literature. *BMC Medical Genomics*, 10(1):59, Oct 2017.

[RGJAB04] Antony Rodriguez, Sam Griffiths-Jones, Jennifer L. Ashurst, and Allan Bradley. Identification of mammalian microrna host genes and transcription units. *Genome Research*, 14(10a):1902–1910, 2004.

[RMA+12] Martin Reczko, Manolis Maragkakis, Panagiotis Alexiou, Ivo Grosse, and Artemis G. Hatzigeorgiou. Functional microrna targets in protein coding sequences. *Bioinformatics*, 28(6):771–776, 2012.

[RMS09] Mark D. Robinson, Davis J. McCarthy, and Gordon K. Smyth. edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics*, 26(1):139–140, 11 2009.

[ROG63] F. B. ROGERS. Medical subject headings. *Bulletin of the Medical Library Association*, 51(1):114–116, Jan 1963. 13982385[pmid].

[RPW+15] Matthew E. Ritchie, Belinda Phipson, Di Wu, Yifang Hu, Charity W. Law, Wei Shi, and Gordon K. Smyth. limma powers differential expression analyses for RNA-sequencing and microarray studies. *Nucleic Acids Research*, 43(7):e47–e47, 01 2015.

[STM+05] Aravind Subramanian, Pablo Tamayo, Vamsi K. Mootha, Sayan Mukherjee, Benjamin L. Ebert, Michael A. Gillette, Amanda Paulovich, Scott L. Pomeroy, Todd R. Golub, Eric S. Lander, and Jill P. Mesirov. Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences*, 102(43):15545–15550, 2005.

[Stu08] Student. The probable error of a mean. *Biometrika*, 6(1):1–25, 1908.

[Sur11] K. Suresh. An overview of randomization techniques: An unbiased assessment of outcome in clinical research. *Journal of Human Reproductive Sciences*, 4(1):8–11, 2011.

[TCK+03] Paul D Thomas, Michael J Campbell, Anish Kejariwal, Huaiyu Mi, Brian Karlak, Robin Daverman, Karen Diemer, Anushya Muruganujan, and Apurva Narechania. Panther: a library of protein families and subfamilies indexed by function. *Genome research*, 13(9):2129–2141, 2003.

[The18] The Gene Ontology Consortium. The Gene Ontology Resource: 20 years and still GOing strong. *Nucleic Acids Research*, 47(D1):D330–D338, 11 2018.

[VAD+12] Thanasis Vergoulis, Michail Alexakis, Theodore Dalamagas, Manolis Maragkakis, Artemis G. Hatzigeorgiou, and Timos Sellis. Tarcloud: A cloud-based platform to support mirna target prediction. In Anastasia Ailamaki and Shawn Bowers, editors, *Scientific and Statistical Database Management*, pages 628–633, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[VPK+14] Ioannis S. Vlachos, Maria D. Paraskevopoulou, Dimitra Karagkouni, Georgios Georgakilas, Thanasis Vergoulis, Ilias Kanellos, Ioannis-Laertis Anastasopoulos, Sofia Maniou, Konstantina Karathanou, Despina Kalfakakou, Athanasios Fevgas, Theodore Dalamagas, and Artemis G. Hatzigeorgiou. DIANA-TarBase v7.0: indexing more than half a million experimentally supported miRNA:mRNA interactions. *Nucleic Acids Research*, 43(D1):D153–D159, 11 2014.

[vSWV+15] Eleni van Schooneveld, Hans Wildiers, Ignace Vergote, Peter B. Vermeulen, Luc Y. Dirix, and Steven J. Van Laere. Dysregulation of microrna in breast cancer and their potential role as prognostic and predictive biomarkers in patient management. *Breast Cancer Research*, 17(1):21, Feb 2015.

[VZP+15a] Ioannis S. Vlachos, Konstantinos Zagganas, Maria D. Paraskevopoulou, Georgios Georgakilas, Dimitra Karagkouni, Thanasis Vergoulis, Theodore Dalamagas, and Artemis G. Hatzigeorgiou. DIANA-miRPath v3.0: deciphering microRNA function with experimental support. *Nucleic Acids Research*, 43(W1):W460–W466, 05 2015.

[VZP+15b] Ioannis S. Vlachos, Konstantinos Zagganas, Maria D. Paraskevopoulou, Georgios Georgakilas, Dimitra Karagkouni, Thanasis Vergoulis, Theodore Dalamagas, and Artemis G. Hatzigeorgiou. Diana-mirpath v3.0: deciphering microrna function with experimental support. *Nucleic Acids Research*, 2015.

[WOK+15] Katherine Wolstencroft, Stuart Owen, Olga Krebs, Quyen Nguyen, Natalie J. Stanford, Martin Golebiewski, Andreas Weidemann, Meik Bittkowski, Lihua An, David Shockley, Jacky L. Snoep, Wolfgang Müller, and Carole Goble. Seek: a systems biology data and model management platform. *BMC Systems Biology*, 9(1):33, Jul 2015.

[WRD+16] Anderson M. Winkler, Gerard R. Ridgway, Gwenaëlle Douaud, Thomas E. Nichols, and Stephen M. Smith. Faster permutation inference in brain imaging. *NeuroImage*, 141:502 – 516, 2016.

[YAA+19] Andrew D Yates, Premanand Achuthan, Wasiu Akanni, James Allen, Jamie Allen, Jorge Alvarez-Jarreta, M Ridwan Amode, Irina M Armean, Andrey G Azov, Ruth Bennett, Jyothish Bhai, Konstantinos Billis, Sanjay Boddu, José Carlos Marugán, Carla Cummins, Claire Davidson, Kamalkumar Dodiya, Reham Fatima, Astrid Gall, Carlos Garcia Giron, Laurent Gil, Tiago Grego, Leanne Haggerty, Erin Haskell, Thibaut Hourlier, Osagie G Izuogu, Sophie H Janacek, Thomas Juettemann, Mike Kay, Ilias Lavidas, Tuan Le, Diana Lemos, Jose Gonzalez Martinez, Thomas Maurel, Mark McDowall, Aoife McMahon, Shamika Mohanan, Benjamin Moore, Michael Nuhn, Denye N Oheh, Anne Parker, Andrew Parton, Mateus Patricio, Manoj Pandian Sakthivel, Ahamed Imran Abdul Salam, Bianca M Schmitt, Helen Schuilenburg, Dan Sheppard, Mira Sycheva, Marek Szuba, Kieron Taylor, Anja Thormann, Glen Threadgold, Alessandro Vullo, Brandon Walts, Andrea Winterbottom, Amonida Zadissa, Marc Chakiachvili, Bethany Flint, Adam Frankish, Sarah E Hunt, Garth IIsley, Myrto Kostadima, Nick Langridge, Jane E Loveland, Fergal J Martin, Joannella Morales, Jonathan M Mudge, Matthieu Muffato, Emily Perry, Magali Ruffier, Stephen J Trevanion, Fiona Cunningham, Kevin L Howe, Daniel R Zerbino, and Paul Flicek. Ensembl 2020. *Nucleic Acids Research*, 48(D1):D682–D688, 11 2019.

[YB95] Yosef Hochberg Yoav Benjamini. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the*

*Royal Statistical Society. Series B (Methodological)*, 57(1):289–300, 1995.

[ZCKS10] Song Zhang, Jing Cao, Y. Megan Kong, and Richard H. Scheuermann. GO-Bayes: Gene Ontology-based overrepresentation analysis using a Bayesian approach. *Bioinformatics*, 26(7):905–911, 02 2010.

[ZKH+21] Xinping Zhu, Masahisa Kudo, Xiangjie Huang, Hehuan Sui, Haishan Tian, Carlo M. Croce, and Ri Cui. Frontiers of microrna signature in non-small cell lung cancer. *Frontiers in Cell and Developmental Biology*, 9:771, 2021.

[ZVG+21] Konstantinos Zagganas, Thanasis Vergoulis, Georgios K. Georgakilas, Skiadopoulos Spiros, and Theodore Dalamagas. Bias in mirna enrichment analysis related to gene functional annotations. *bioRxiv*, 2021.

[ZVP+17] Konstantinos Zagganas, Thanasis Vergoulis, Maria D. Paraskevopoulou, Ioannis S. Vlachos, Spiros Skiadopoulos, and Theodore Dalamagas. BUFET: boosting the unbiased mirna functional enrichment analysis using bitsets. *BMC Bioinformatics*, 18(1):399:1–399:8, 2017.

[ZVSD20] K. Zagganas, T. Vergoulis, S. Skiadopoulos, and T. Dalamagas. Efficient calculation of empirical p-values for association testing of binary classifications. In *32nd International Conference on Scientific and Statistical Database Management*, SSDBM 2020. Association for Computing Machinery (ACM), 2020.