UNIVERSITÉ DU QUÉBEC À MONTRÉAL

UNE APPROCHE DISTRIBUÉE DE PRÉDICTION ET D'ANALYSES DE MICROARNS: CAS D'APPLICATION DU BLÉ

MÉMOIRE

DE LA MAÎTRISE EN INFORMATIQUE

PAR

CHAO-JUNG WU

AOÛT 2018

RÉSUMÉ

L'organisation du génome du blé permet à des variétés de tolérer le froid durant l'hiver canadian. L'inhibition de la transcription par les microARNs (miARNs) est l'un des mécanismes de régulation les plus importants à la tolérance environnementale chez les plantes. Cependant, le groupe de miARNs du blé sensibles au froid et la manière dont ils coordonnent la tolérance par diverses voies biologiques pour divers besoins cellulaires et organismiques sont encore mal compris. L'une des raisons est l'absence d'outils bioinformatiques adéquats pour identifier les miARNs dans les donnés massives de séquençage. Dans ce travail, un pipeline efficace de prédiction de miARN, mirLibSpark, a été mis en œuvre pour effectuer une analyse plus rapide des librairies de petites ARNs en augmentant la précision de leur classification fonctionelle. Nous avons appliqué ce pipeline sur des données expérimentales issues du blé dans des conditions de stress provenant des champs. Nous avons découvert 134 miARNs parmi les souches du blé et les conditions dans le champ. Différents sous-ensembles de ces miARNs sont responsables de la tolérance dans plusieurs contextes génétiques. Les transcriptomes du blé ont été annotés avec les voies régulatrices KEGG. Pour reconnaître les molécules, une analyse de l'enrichissement fonctionnel des miARNs a montré que les voies régulatrices de transcription, de structure, de métabolisme et les systèmes organismiques peuvent être associés à la tolérance au froid. Les méthodes développées faciliteront les analyses massives des microARNs de blé. En plus, les résultats obtenus pour le pilote sont des ajouts dans la compréhension des mécanismes des miARNs.

ABSTRACT

Cold tolerance organizes wheat to survive Canadian winter. Translation inhibition carried out by microRNA (miRNA) is one of the most important regulatory mechanisms that lead to environmental tolerance in plants. However, the group of wheat cold-responsive miRNAs in field conditions and how they coordinate the tolerance through various biological pathways for cellular and organismal needs are still poorly understood. One of the reasons is the lack of an adequate bioinformatics tool to identify miRNAs from the massive sequencing data. To gain insight into miRNA functional effects in wheat cold tolerance, I sought to facilitate miRNA prediction and to elucidate the contributions of the cold-responsive miRNAs in wheat. From this work, an efficient miRNA prediction pipeline, mirLibSpark, was implemented to perform a faster analysis of short RNAs libraries while increasing the classification accuracy. We have applied this pipeline in the experimental data generated from wheat under field stresses. There were 134 miRNAs discovered among wheat strains and field conditions. Different subsets of these miRNAs are responsible for the cold tolerance in different genetic backgrounds. The wheat transcriptomes were annotated with KEGG pathways. To elucidate the molecules, functional enrichment analysis of these miRNAs has been conducted. The results showed that the pathways of transcription, structure, metabolism and organismal systems are associated with cold tolerance. This work not only contributes a tool to miRNA prediction, but also the functions of miRNAs in wheat cold tolerance. My discoveries will be an asset for breeding stable crops for climate changes.

ACKNOWLEDGEMENTS

The past two years were the beginning of an interesting journey of my career switch from biochemistry to bioinformatics and computer sciences. I could not have succeeded this master degree in computer sciences without the supports from friends and family. I am truly grateful to my supervisor, Dr. Abdoulaye Baniré Diallo, who believed in my potential and gave me a place to do what I did. I particularly thank Amine M. Remita, who accompanied me from the beginning of the program and helped me to learn and establish the essential knowledge and skills in bioinformatics. I thank Mr. Bruno Daigle, Dr. Ahmed Halioui and Dr. Abou Abdallah Malick Diouara for taking their time to discuss my questions in the course work. I thank Mr. Dylan Lebatteux, who created and shared a motivating and productive office with me. I thank Ms. Golrokh Kiani, who took care of all the daily details in our offices. I benefited substantially from the courses given by Dr. Alix Boc as well as by Dr. Beaudry, Dr. Blondin-Massé, Dr. Vandomme and Dr. Bouguessa. I also thank the teachers at the daycare *Fleur de Macadam*.

I dedicate this work to my supportive husband, Dr. Yi-Sheng Chao, and my wonderful children, Mr. Jeremie Chao of 7 years old and Miss Charlotte Chao of 4 years old.

Parts of this work have been recognized by the international conference RECOMB and by the committees granting me NSERC and FRQNT full scholarships for pursuing a PhD in computer sciences after this master program.

PUBLICATIONS

1. **Chao-Jung Wu**, Mohamed Amine Remita, and Abdoulaye Baniré Diallo. mirLibSpark: a scalable NGS microRNA prediction pipeline with data aggregation. In *The 22nd Annual International Conference on Research in Computational Molecular Biology, April 21-24, 2018, Paris*, poster. RECOMB, 2018.
Published.

2. **Chao-Jung Wu**, Mohamed Amine Remita, and Abdoulaye Baniré Diallo. mirLibSpark: a scalable NGS microRNA prediction pipeline with data aggregation.
In preparation for peer-reviewed journal.

3. **Chao-Jung Wu**, ..., and Abdoulaye Baniré Diallo. Identification and characterization of pathways in wheat transcriptome targeted by cold acclimation regulatory microRNAs.
In preparation.

TABLE OF CONTENTS

RÉSUMÉ ii

ABSTRACT iii

ACKNOWLEDGEMENTS iv

PUBLICATIONS v

LIST OF TABLES xi

LIST OF FIGURES xii

CHAPTER I
LITERATURE REVIEW 1

1.1 Molecular basis of miRNA abiotic stress response 2

1.1.1 MiRNA biogenesis 4

1.1.2 Regulation of miRNA expression 6

1.1.3 Functions of miRNAs: post-transcriptional regulation of target genes 8

1.2 MiRNA in agriculture 9

1.2.1 MiRNA in crop strain improvement 9

1.2.2 MiRNA-mediated plant defense 11

1.3 Current knowledge of cold-stress tolerance in wheat *Triticum aestivum* 12

1.3.1 Physiological adaptation of plants in cold acclimation 12

1.3.2 Genetic analysis of wheat cold tolerance 14

1.3.3 Current knowledge of wheat miRNAs 14

1.4 Annotation of plant miRNAs 16

1.4.1 Traditional criteria for plant functional miRNAs 17

1.4.2 TF-TG interaction graph of predicted targets 17

1.5 Bioinformatics tools for wheat miRNA analysis 18

1.5.1 Adapter trimming 19
1.5.2 Alignment 19
1.5.3 Secondary structure prediction 20
1.5.4 MiRNA prediction 21
1.5.5 MiRNA target prediction 22
1.6 Bioinformatics pipelines for wheat miRNA analysis 23
1.6.1 Available pipeline-like tools 24
1.6.2 Wheat chromosome walk 24
1.6.3 Wheat NGS transcriptome analysis 26
1.7 Databases to assist in wheat miRNA annotation 28
1.7.1 MiRNA databases 28
1.7.2 Reference genomes 28
1.7.3 Transcriptome repositories for sRNA 30
1.7.4 Network database 30
CHAPTER II
RESEARCH PROPOSAL 32
2.1 Introduction 33
2.2 Research problems 34
2.3 Specific aims: distributed miRNA prediction methodology and real world datasets application 35
CHAPTER III
MIRLIBSPARK: A SCALABLE NGS MICRORNA PREDICTION PIPELINE 38
ABSTRACT 39
3.1 Introduction 40
3.1.1 Annotation of miRNAs as a big data problem 40
3.1.2 Hadoop framework 41
3.1.3 Apache Spark 42
3.2 Implementation of mirLibSpark 43

3.2.1 Choice of bioinformatics tools 43
3.2.2 Passing data to the programs in a pipeline 45
3.2.3 Management of the project development 46
3.2.4 Spark RDD chaining in mirLibHadoop 46
3.3 Experimental methods 50
3.3.1 Arabidopsis thaliana sRNA data 51
3.3.2 Implement a sequential version 51
3.3.3 Implement a miRDeep-P wrapper 51
3.3.4 Evaluate the prediction performance 52
3.3.5 Supervised classification of the libraries 52
3.3.6 Unsupervised classification of the libraries 53
3.3.7 Dimension deduction: Manifold and PCA 53
3.4 Experimental results and conclusions 54
3.4.1 Comparing with miRDeep-P program 54
3.4.2 Scalability 54
3.4.3 Estimate the execution time to process multiple libraries . . . 56
3.4.4 Case study 57
3.5 Discussions 59

CHAPTER IV
TARGETED PATHWAYS BY COLD-RESPONSIVE MIRNAS IN FIELD WHEAT 61

4.1 Introduction 62
4.2 Materials and methods 64
4.2.1 Wheat sRNA libraries 64
4.2.2 Identification and extraction of potential pre-miRNA candidates from sequenced small RNA transcriptomes 64
4.2.3 Removing false-positive miRNAs 65
4.2.4 Differential expression analysis of miRNA abundances 65

4.2.5 Definition of indices and the regulation of miRNAs 66

4.2.6 GO term and KEGG pathway annotation 68

4.2.7 MiRNA targets identification and functional enrichments . . . 69

4.3 Results and discussions 70

4.3.1 KEGG pathway annotated 14% of the field wheat long RNAome 70

4.3.2 Wheat has 365 KEGG pathway 72

4.3.3 Discovery of wheat miRNAs expressed during cold acclimation in the field 75

4.3.4 Differential expressed field wheat sRNAs responsive to cold acclimation 76

4.3.5 Contribution of genetic backgrounds and Vrn-A1 genotypes to cold responsive sRNAs 78

4.3.6 Enriched GO terms and KEGG pathways targeted by differentially expressed sRNAs in cold acclimation 82

4.3.7 Classes of enriched biological pathways regulated by cold acclimation-responsive miRNAs in field wheat 84

CHAPTER V
DISCUSSIONS AND CONCLUSIONS 107

5.1 Network enrichment graph 108

5.2 Statistical methods 110

5.3 Parallelization scheme 111

5.4 User friendliness 111

APPENDIX A
SOURCE CODES 113

A.1 Contributors to mirLibSpark 114

A.2 Structure of the mirLibSpark project 115

A.3 paramfile.txt 116

A.4 submit_j.pbs 117

A.5 mirLibPipeline.py 118

A.6 mirLibRules.py 124

A.7 sequential.py 130

A.8 run_mirdeep_p.py 136

APPENDIX B
MISCELLANY 139

B.1 MirLibSpark and wheat miRNA availability 139

B.2 MirLibSpark manuscript first page 140

B.3 More resources for mirLibSpark 141

B.4 Description of wheat 365 KEGG pathways 143

REFERENCES 149

LIST OF TABLES

Table Page

1.1 Classification of sRNAs by functional roles 3

1.2 Known plant miRNAs responsive to abiotic stresses 16

1.3 Known miRNAs responsive to cold stress 16

1.4 Selected miRNA target predictors 23

1.5 Selected pipeline-like NGS miRNA predictors 24

3.1 Distributions of Hadoop framework 42

3.2 Time of execution of Bowtie with 0.25 million sequences 46

3.3 Comparing with miRDeep-P 54

3.4 Comparing the predicted miRNAs with miRBase-21 58

3.5 Scalability and number of executors presented as a table 60

4.1 Coverage rates of KEGG pathway annotations in plant genomes . 72

4.2 Number of KEGG pathways in selected plant, animal and microbian species 74

4.3 Coverage of wheat pathways by field transcriptomes 74

4.4 Cold responsive sRNAs regarding Vrn-A1 in different genotype backgrounds 81

4.5 Statistics of functional enrichment by annotation methods 83

4.6 Descriptions of predicted 220 sRNAs from wheat field transcriptomes 85

LIST OF FIGURES

Figure Page

1.1 Types of plant RNAs 3

1.2 MiRNA production and activity pathways in plants and humans . 6

1.3 Modes of regulation of MIR gene transcripts 8

1.4 Functional Relevance index 18

3.1 Overview of mirLibSpark workflow 47

3.2 mirLibSpark pipeline 49

3.3 Scalability and number of executors 56

3.4 Estimation of the number of predicted miRNA and the execution time 57

3.5 Manifold and PCA projections 58

3.6 Feature selection 59

4.1 KEGG pathway annotation workflow 71

4.2 Chromosomal distribution of 418 miRNA candidates 76

4.3 Time course of differential expression profiles for the 220 cold responsive sRNAs 77

4.4 Average trend of up or down regulation of the 220 cold responsive sRNAs in different genetic backgrounds 79

4.5 Enriched GO terms by precursor XLOCs 97

4.6 Enriched GO terms by top 1 target gene 98

4.6 Enriched GO terms by top 1 target gene, continued 99

4.7 Enriched GO Biological Process terms by top 5 scored target genes 100

4.7 Enriched GO Biological Process terms by top 5 scored target genes, continued 101

4.8 Enriched GO Cellular Component terms by top 5 scored target genes102

4.8 Enriched GO Cellular Component terms by top 5 scored target genes, continued 103

4.9 Enriched GO Molecular Function terms by top 5 scored target genes104

4.9 Enriched GO Molecular Function terms by top 5 scored target genes, continued 105

4.10 Enriched KEGG pathways by top 5 scored target genes 106

5.1 Enrichment graph for true positive prediction enhancement 110

A.1 Structure of the mirLibSpark project in Linux tree view 115

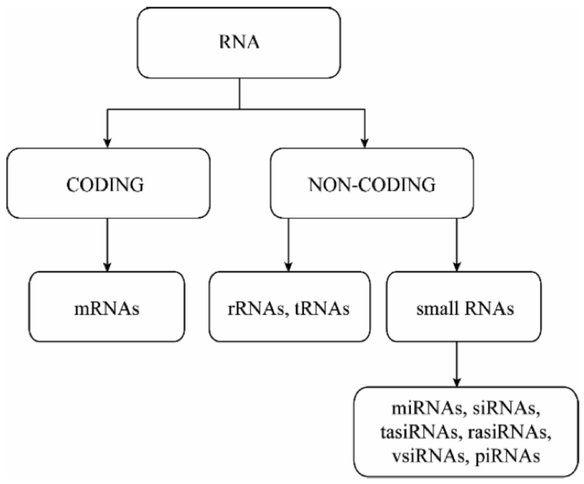
CHAPTER I

LITERATURE REVIEW

1.1 Molecular basis of miRNA abiotic stress response

Canada is subject to a variety of hydrometeorological hazards and severe weather. Abiotic stresses such as cold, drought, salt and aluminum cause reduction in crop yield and extensive economic losses. To tolerate, plants have evolved a range of physiological, metabolic and developmental adaptations, which are under the control of dynamic network of genetic regulatory mechanisms that involve transcription factors and genes in responsive pathways. Several types of small RNAs (sRNAs) exist in plants, summarized in Figure 1.1. Among these sRNAs, miRNA and siRNA exhibit regulatory functions to the expression of other genes, as summerized in Table 1.1.MiRNAs (microRNAs) have risen as an important regulator for plants to cope with environmental challenges [1]. Due to the global warming phenomenon, the plant hardiness zones are shifting, more unusual winter weather is seen, and many weeds, pests, and fungi thrive under warmer temperatures. In order to ensure the food safety and increase the biodiversity of crop strains, scientists are studying how the environment responsive mechanism led by miRNAs can be used to improve the crops survival in the global warming era. In this mémoire, focuses are addressed on the bioinformatics and statistical approaches in high-speed miRNA classification from massive transcriptomic data, and in the statistical analysis pipeline of wheat field experiment data to elucidate the roles of miRNAs in cold resistance.

Figure 1.1 Types of plant RNAs. RNAs are classified into coding and non-coding RNAs. Coding-RNAs constitute messenger RNAs (mRNAs). Non-coding RNAs are further divided into ribosomal RNAs (rRNAs), transfer RNAs (tRNAs) and small RNAs. Small RNAs comprise of microRNAs (miRNAs), short interfering RNAs (siRNAs) and piwi interacting RNAs (piRNAs)



Taken from Figure 1 in Guleria P. et al [2].

Table 1.1 Classification of sRNAs by functional roles

Basis of classification	Types
Origin, properties and functions	1. DNA markers 2. Gene regulators 3. Aboitic stress signals, synthesized/processed in response to abiotic stress 4. Biotic stress signals, induced by biologically active compound
Predicted functions	1. Cellular debris ncRNA, RNA with no specific function 2. Housekeeping ncRNA: tRNA, rRNA, small nucear RNA, small nucleolar RNA, signal recognition particle RNA 3. Regulatory ncRNA: miRNAs, siRNAs
Roles in RNA silencing	1. miRNA 2. siRNA

Taken from Table 1 in Guleria P. et al [2].

1.1.1 MiRNA biogenesis

MiRNAs are small single-stranded non-coding RNAs approximately 18-25 nucleotides long that play a crucial role in basic physiological and morphological processes and in response to various stresses in eukaryotic organisms by binding to their complementary target mRNAs [3].

MiRNA biogenesis varies between plants and animals, with the differences largely concerning the location of the process, the protein composition of the microprocessor, the mechanism of miRNA action on mRNA target, and the miRNA gene (MIR) structure [4]. A comparison of the production pathways and modes of functions between plants and animals is illustrated in Figure 1.2.The major steps in plant miRNA biogenesis are described here.

Organization of pri-miRNA transcripts [4]. Like protein-coding genes, MIR promoters contain TATA box and are transcribed by RNA Polymerase II (RNA Pol II). MIR genomic structures are as complex as protein-coding genes. An intron is a segment within a gene to be removed by RNA splicing during maturation of the final RNA product. In most cases, MIR genes contain only one or two exons, and the miRNAs are located in the first exons. Many known *Arabidopsis* MIRs contain introns, or are encoded within the introns of host genes. Because some protein components are shared in intron splicing and pre-miRNA formation, pre-miRNAs may be generated during splicing or in competition with splicing. It is possible to have more than one pair of miRNA/miRNA* duplexes in a single MIR gene, called polycistronic miRNA genes. Just like protein-coding genes, MIR gene transcriptions can be initiated at different start sites, and terminated at different polyadenylation sites. MIR genes can also undergo the transcription regulation of alternative splicing that skips some exons or includes some introns and thus produce different lengths of pri-miRNAs. The splice site sequences are conserved, including the AU/GC rule, and the average length of plant introns is 160 bp.

Pri-miRNA (primary structure). Plant miRNA genes are transcribed from MIR genes in the genome by RNA Pol II, producing primary transcripts (pri-miRNAs) that contain a cap structure at the 5' end and a poly-A tail at the 3' end.

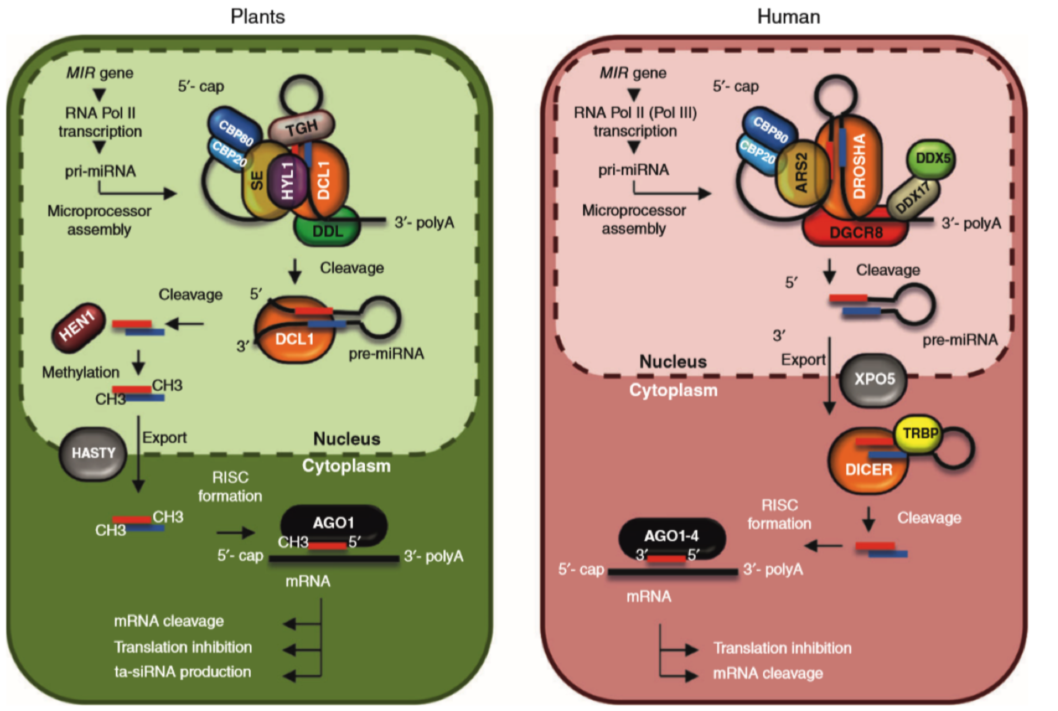
Pre-miRNA (secondary structure). Pre-miRNAs (precursor miRNAs) containing miRNA and complementary miRNA* sequences are produced from pri-miRNAs, which can fold back onto themselves to produce imperfectly double-stranded stem-loop precursor structures.

Duplex (endonuclease cleavage). The pre-miRNAs are then further processed into miRNA/miRNA* duplexes. RNase III-type endonucleases are involved in miRNA biogenesis: Drosha and Dicer in animals; Dicer Like 1-4 (DCL1, DCL2, DCL3 and DCL4) in plants in the microprocessor complex containing additional proteins, such as HYL1 and SE. The miRNA/miRNA* duplexes are methylated at 3' ends by the methylase HEN1 in plants, protecting the duplexes from degradation, then exported to the cytoplasm controlled by the exportin HASTY. The nomenclature of miRNA initially refers miRNA* as the imperfect complementary counterpart of the functional miRNA. Due to its shorter half life in cells, miRNA* has less expression counts than miRNA. However, microRNA arm-switching has been observed in many cases revealing that both miRNA and miRNA* can be functional depending on the situations. Therefore, nowadays scientists name them as miR-5p and miR-3p, referring to their relative positions rather than active roles

[5, 6].

RISC. One strand of either miRNA or miRNA* is stabilized in RNA-induced silencing complex (RISC). Only the miRNA molecules that are embedded in RISCs are able to control the expression levels of their target mRNA molecules via degradation or inhibition of translation.

Figure 1.2 MiRNA production and activity pathways in plants and humans



Circle shapes are proteins. Lines are RNA molecules. Red and blue lines are mature miRNA and miRNA, respectively. Arrows indicate the direction of the temporal events. Adapted from Figure 1 in Stepien A. et al [4]*

1.1.2 Regulation of miRNA expression

The miRNA species and their expression levels challenged by environmental stress agents of biotic and abiotic origin are regulated throughout the miRNA biogenesis process. A few modes of the regulations on the MIR genes are described here, and some are illustrated in Figure 1.3.

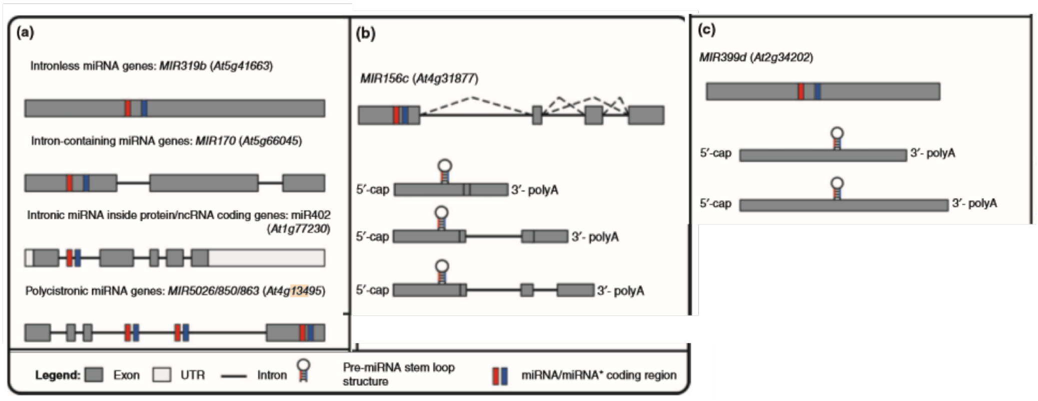
Promoter. The promoter regions of many miRNA genes contain transcription factor-binding sites, such as GATA, LFY, T-box, and other biotic and abiotic stress responsive elements [7].

Alternative splicing. The majority of plant miRNAs are located in intronless MIR genes or the first exons of their MIR genes. Transcriptomic analyses in plants have shown that the splicing profiles are altered upon stresses. In *Arabidopsis*, miR846 is expressed from pre-miR846 isoform 1. Exogenous application of abscisic acid to seedlings mediates the alternative splicing events by reducing the levels of functional isoform 1 and increasing those of nonfunctional isoform 3. As a result, the mature miR846 expression level is decreased [8].

PolyA. MIR163 expresses a longer transcript by selecting a more downstream polyadenylation signal when the plant *Arabidopsis* is challenged by bacterial infection [9]. The longer transcript has a longer half life and thus produces more miR163 that targets and inhibits the mRNA expression of gene At1g66690, which is a methyltransferase. The result leads to a proper response to biotic stress.

Protein machinery competition. There are at least four DCL proteins in plants having distinct but somewhat redundant expression patterns during the developmental course and the environmental stresses [10]. Furthermore, the components in the microprocessor for miRNA duplex formation are shared with the spliceosome machinery. It has been demonstrated that the competition induced by heat stress for the components to form either a microprocessor or a spliceosome plays a role in the maturation of miR402 [11].

Figure 1.3 Modes of regulation of MIR gene transcripts


(a)
Intronless miRNA genes: MIR319b (At5g41663)
Intron-containing miRNA genes: MIR170 (At5g66045)
Intronic miRNA inside protein/ncRNA coding genes: miR402 (At1g77230)
Polycistronic miRNA genes: MIR5026/850/863 (At4g13495)
(b)
MIR156c (At4g31877)
5′-cap
3′- polyA
(c)
MIR399d (At2g34202)
Legend: Exon
UTR
Intron
Pre-miRNA stem loop structure
miRNA/miRNA* coding region

(a) Localization of miRNAs within the introns or exons of MIR genes (b) Alternative splicings of a MIR gene (c) Alternative polyadenylation signals of a MIG gene. Adapted from Figure 3 in Stepien A. et al [4].

1.1.3 Functions of miRNAs: post-transcriptional regulation of target genes

The changes of miRNA expressions and their down-stream target genes are through the epigenetic mechanisms at the post-transcription level. MiRNAs mediate mRNA cleavage and translational repression by forming the RISC complex that contains the AGO protein. Slicing activity is the AGO-catalyzed cleavage of the targeted mRNA [12]. The RISC slicing mode leading to mRNA cleavage is predominant in plants, whereas miRNA-guided translation inhibition is a classical miRNA action in animal cells. Most of plant miRNAs are perfectly or near perfectly complementary to a single or a few target mRNAs [3], whereas animal miRNAs recognize their targets through partial base-pairing and have in average 100 target genes per miRNA [13]. The functions of the target genes confer to the cellular functions regulated by the miRNAs.

1.2 MiRNA in agriculture

Studies have shown that many miRNAs are evolutionarily conserved across all major lineages of plants [14]. Based on the conservation and diversification of miRNAs during plant kingdom evolution, miRNA families are grouped into ancient and young classes. The ancient miRNAs are often highly expressed and evolutionarily conserved. For example, many of the most abundant miRNA families in over 15 plant species have less than one nucleotide variation on average compared to the consensus sequence of miR157-7-5p or miR165/6-3p [15]. On the other hand, the young miRNAs are expressed at low levels or only induced under certain conditions and typically exist only in a few species [16]. New MIR genes may be evolved from small RNA-generating loci through aberrant replication, recombination or transposition events from expressed gene sequences [17]. To motivate our interest in studying miRNAs, this section will discuss how the knowledge of the miRNAs, their target genes and related advances in miRNA technology will benefit agriculture.

1.2.1 MiRNA in crop strain improvement

MiRNAs are involved in regulating essential plant metabolic processes by modulating the mRNA transcript abundance. Plant pre-miRNAs are processed by DCL proteins to generate diverse lengths of miRNAs. Several conserved miRNAs in crops have been identified to be involved in drought response, temperature stress, abiotic stress and disease resistance [18]. For example, miR395 that targets ATP sulfurylase genes, known for salt tolerance and energy generation, is up-regulated in response to abiotic stress [19].

Functional analysis of miRNAs revealed their importance in regulating plant growth, development and response to environmental stresses. Therefore, miRNA-based genetic modification technology is one of the most promising solutions to improve agricultural productivity. Gene therapy approaches can be implemented to manipulate either the miRNAs or the targets of miRNAs [20]. Because many miRNA targets are involved in several stress tolerances and phenotypes, instead of manipulating the miRNA targets, current trend is to manipulate the miRNAs.

Over-expression of miRNA. The design of miRNA precursor-based expression cassettes can be constructed to release artificial miRNAs into plants. This results in the over-expression of the miRNA and the down-regulations of specific target gene or sometimes a broad spectrum of target genes. Transgenic tobacco expressing an artificial miRNA sequence, designed to target cucumber mosaic virus suppressor 2b, has acquired resistance to this virus by effectively inhibiting the expression of 2b proteins [21]. In this study, the authors took the precursor of the *Arabidopsis* miR171 as backbone and replaced the miR171 sequence by their artificial miRNA while maintaining the same free energy of the original secondary structure [21]. Trangenic rice over-expressing Osa-miR820 exhibits salt tolerance and a dozen of genes are potentially silenced by this miRNA [22]. Another more intuitive delivering method is to deliver the miRNA molecules directly into the animal individual, using liposome-based delivery system [23]. However, approaches in this category are not economical for large scale crop production.

MiRNA mimicry. A phenomenon called *target mimics* has been described in plants in 2007 [24] that a short RNA resembles the miRNA but 1-3 nucleotides are inserted to produce central mismatches between the cleavage site or at other recognition site, providing subtle regulation to the inactivation of the target miRNA. Such methodology has been demonstrated in several dicot crops using plant viruses [25]. Recently this methodology has been extended to monocots such as wheat [26]. A note for the terminology: *miRNA targets* are the genes targeted by miRNAs through sequence complementary pairing. *target mimics* are the sequences mimicking that of the miRNAs.

The methods and cases described above seem effective, efficient and promising for crop improvement. However, these approaches could not avoid the modification of the plant genome. Often transgenic crops or genetically modified organisms (GMO) in general raise the concern of the consumers, either due to safety worries without valid ground or the disputable image and impact of the GMO crops brought by the existing GMO-promoting companies. Therefore, an potential approach is to incorporate the miRNAs as a biomarker for desired traits, such as cold tolerance or immunity to pathogen infection, in the traditional breeding procedures for new economical strains.

1.2.2 MiRNA-mediated plant defense

MiRNAs play an important role in plant defense. Flagellin is a surface structure for bacterial motility. Pre-stimulation of the plants with a synthetic flg22-peptide led to enhanced resistance against bacterial invaders. In *Arabidopsis*, a 22–amino acid peptide (flg22) from the N terminus of flagellin triggers rapid changes in transcription profile. Plant auxin-signaling pathway is essential for the growth of bacterium *Pseudomonas syringae* in plants. Navarro L. et al discovered that the presence of flg22 triggered the accumulation of miR393 in one hour. Further, transcripts of two receptors for the plant hormone auxin, TIR1 and AFB, were the targets of miR393 and their protein products were repressed in two hours [27]. Since then, studies have revealed that miRNAs are involved in the defense against not only bacteria, but also fungal and viral infections [28, 29].

Regarding another type of small RNAs, siRNAs, also known for their importance in plants against infections, siRNA-mediated gene silencing occurs only after the pathogens have started growth within the host cells. MiRNAs are induced by sensing the surface markers of pathogens prior to the invasive growth and sending signals to surrounding cells for plant immunity mechanism activation. Such property is ideal for field workers to identify the plants at early stages of disease infection and take proper measures for treatments and maintenance in the crop field.

1.3 Current knowledge of cold-stress tolerance in wheat *Triticum aestivum*

1.3.1 Physiological adaptation of plants in cold acclimation

Temperate plants are complex traits that have evolved mechanisms involving a number of genes in different gene families to withstand a period of low but non-freezing temperatures. It involves signal detection and transduction pathways, activation of low-temperature responsive genes, and finally induction of freezing tolerance. It may take a few days to weeks for plants to reach maximum levels of freezing tolerance. This dynamic and energy-demanding process is refereed as cold acclimation. In nature, cold acclimation is initiated by the decreasing temperatures in autumn or early winter. As temperature falls, ice forms in the intercellular spaces. There is a sudden drop of water potential outside the cell causing cellular dehydration and thus freezing injury. Freezing temperature, drought and salt all cause dehydration and therefore may share some responsive mechanisms, while there are still other mechanisms and genes under differential regulation [30, 31, 32].

To avoid the loss of water, cells of cold-tolerant organisms synthesize and accumulate low molecular weight solutes, such as sugars and proline, to counteract the osmotic pressure. Freeze-thaw cycle can further destabilize the shrunk cell during re-expansion that demands rapid change of the surface area and composition of its lipid membrane. Antifreezing proteins that can bind to small ice crystals are also expressed to protect the cell [31, 32]. Redox status of photosynthesis could be an important signalling mechanism during cold stress [33]. A highly conserved low-temperature responsive element, of which the consensus sequence is CCGAC, has been identified in plants and is recognized by transcription factors C-repeat-binding factor (CBP) [34].

Although more signalling mechanisms, responsive elements and trasncription factors remain to be found, it has been observed that upon the challenge of cold temperature, chromatin structure has remodeled, gene expression profile has modified by cis- and trans-regulators, protein profile has adjusted, and metabolism profile has shifted [31, 32]. The cold stress is in fact essential for the flowering pathway (vernalization) in the winter cultivars of wheat [35]. The studies of wheat abiotic stresses, including cold stress, investigate one to several aspects listed above. It is hypothesized that miRNAs play an important role coordinating the tolerance from sensing the environmental stimuli to proper response and resistance.

1.3.2 Genetic analysis of wheat cold tolerance

Several studies have conducted large scale of genetic analysis in wheat, searching for genes and loci related to cold tolerance. Other studies focused on the expression patterns of miRNAs in response to cold temperatures.

Genomic loci association analysis. Frost resistance loci FR1 and FR2 have been identified on wheat chromosome 5AL. An additional FT Quantitative Trait Locus (QLT) was described on 5B. Many of the genes identified in these regions are related to VRN (vernalization) and CBF proteins (review in [36]). A GWAS study using wheat Chinese Spring IWGSC RefSeq revealed that SNPs in CBF-A3, CBF-A15, VRN3 and PPD1 genes are associated with cold tolerance [36].

Genome-wide gene expression analysis. A genome-wide gene expression analysis in wheat identified 12901 genes that are differentially expressed during cold treatment [37].

MiRNA expression and target gene analysis. Since 2014, our team and 3 others groups reported miRNAs' involvement in cold stress response from the studies of wheat in controlled laboratory environment. Many target genes of differentially expressed miRNAs are associated with flowering and are enriched for signaling pathways, regulation of transcription, ion transport functions, photosynthesis and metabolism [30, 38, 39, 40].

1.3.3 Current knowledge of wheat miRNAs

Sequencing of small RNAs in the transcriptoms gives us a start point for understanding their structural characteristics, identities, diversity, temporal-spatial distribution, interaction with the environment and possible roles in plants. Deep sequencing technologies have recently uncovered an increasing number of miRNAs exhibiting transient or tissue-specific expression, whose target genes are involved in specialized functions.

Experimental studies in the model *Arabidopsis thaliana* and crop species, such as wheat, potato and cotton, have led to the discoveries of many miRNAs and their target genes [18], including a study by our team [38]. Wheat is one of the most planted cereal crops in Canada and throughout the world. Wheat *Triticum aestivum* (*tae*) contains 17 Gbp in its hexaploid genome, composed of three closely related and independently maintained genomes [41]. About 767 miRNAs are found in miRBase-22 for small-genome plant *A. thaliana* with a genome size of only 135 Mbp (mega basepairs). On the contrary, studies in the larger genome of wheat resulted in only 2707 miRNA candidates, including 534 registered in miRBase-22 for experimentally validated or 170 in PMRD (plant microRNA database) for identified *in silico*.

Some of the conserved miRNA candidates for many plant species are known to be responsive to certain types of aboitic stresses as summarized in Table 1.2.There is also a group of conserved miRNAs and a group of wheat-specific miRNAs that are responsive specifically to cold as summarized in Table 1.3.

Table 1.2 Known plant miRNAs responsive to abiotic stresses

Stress type	MiRNA
Water	miR1030, miR1035, miR1050, miR1088, miR1125, miR1126, miR170, miR172, miR395, miR397, miR408, miR474, miR529, miR845, miR851, miR854, miR896, miR901, miR903
Drought	miR408
UV-B	miR156/157, miR159/319, miR160, miR165/166, miR167, miR169, miR170/171, miR172, miR393, miR398, miR401
Sulfate	miR395
Salt	miR159a/b, miR164a/b/c/d, miR1661/m, miR169g, miR169n, miR395, miR396, miR397, miR474, miR399
Phosphate starvation	miR399
Mechanical	miR156, miR162, miR164, miR408, miR475, miR480, miR481
Hypoxia	miR156, miR157, miR159, miR172, miR391, miR393, miR775
Copper	miR398
Abscisic acid (ABA)	miR393, miR417

Summarized from contents in Guleria P. et al [2], which is not an updated exhaustive list.

Table 1.3 Known miRNAs responsive to cold stress

Cold stress	MiRNA
Conserved	miR156, miR159, miR164, miR165, miR167, miR168, miR169, miR172, miR319, miR369, miR389, miR393, miR396, miR397, miR398, miR400, miR402, and miR408
Wheat up-regulated	miR164, miR169, miR172, miR319, miR1029, miR1130, miR2118, miR9670, miR9672
Wheat down-regulated	miR168, miR393, miR397, miR444, miR827, miR9674, miR5048, miR5169, miR9663

Plant conserved miRNAs responsive to cold stress, up- or down-regulated, are listed in the conserved column.
Wheat-specific miRNAs are listed separately for up- and down-regulated expressions. in response to cold stress.
Summarized from multiple sources [30, 38, 39, 40]

1.4 Annotation of plant miRNAs

The traditional guidelines and criteria for plant miRNA prediction have been systematically established and documented. A brief summary of the criteria is presented in this section. However, predictions based on these criteria often contain a high proportion of functionally false-positive miRNAs that are considered valid miRNAs but not the functional cause of the phenotype. Therefore, one of the innovative approaches, based on network annotation, to boost the selection of functional relevant miRNs after they are predicted is described in this section.

1.4.1 Traditional criteria for plant functional miRNAs

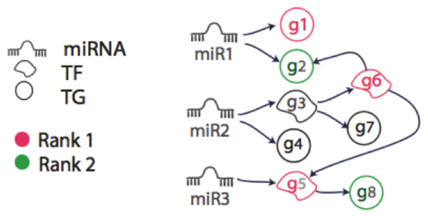
The features for a likely functional miRNA have been incorporated in the bioinformatics development of miRNA validation tools, which will be described in Section 1.5 for bioinformatics tools and 1.6 for prediction pipelines. These features include precise excision from the stem of a stem-loop precursor, DCL proteins dependence, RDRP (RNA-dependent RNA polymerase) and PolIV/PolV Independence (the dependence derives many siRNAs), belonged to a distinct miRNA family, containing multi-functional stem-loops and the genes of its potential targets [6, 42, 43].

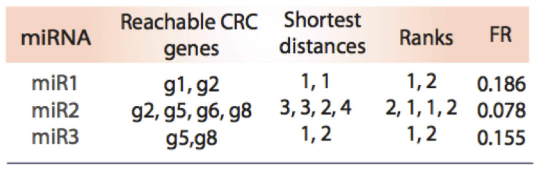
1.4.2 TF-TG interaction graph of predicted targets

Previous NGS approaches use the differential expressions of miRNAs to select for the miRNAs highly associated with the phenotypes. The target genes and enriched target functions are used only for further explanation of the cellular/organismal functions of the miRNAs.

In Vafaee et al [44], the TF-TG interaction network created by combining a few databases of transcription factor (TF) and regulated genes (TG) was used to create a Functional Relevance index (FR) based on the distance in the graph from the *a priori* knowledge of cancer-related genes to determine if a differential expressed miRNA is relevant for this cancer. As illustrated in Figure 1.4, the graph is drawn by connecting the miRNA with its target genes. If the genes in the nodes are transcription factors (TF), the genes regulated by these transcription factors (TG) are then added to this graph. Briefly, if a target gene G_1 of a miRNA is a TF, it creates an edge to another gene G_2 that is regulated by this TF. If G_2 is a TF, it creates another edge to another gene G_3, or this leaf terminates at G_2. The FR index was calculated based on the distance of a miRNA to each of the genes in the *a priori* functional relevant list, the importance ranking of the genes for this cancer. A model combing machine-learning and statistics is built using these features to predict the most relevant miRNAs from all the differential expressed ones. Therefore, the target functional network is used in addition to the targets to support the involvement of a candidate miRNA in cancer prognosis.

Figure 1.4 Functional Relevance index





miRNA	Reachable CRC genes	Shortest distances	Ranks	FR
miR1	g1, g2	1, 1	1, 2	0.186
miR2	g2, g5, g6, g8	3, 3, 2, 4	2, 1, 1, 2	0.078
miR3	g5,g8	1, 2	1, 2	0.155

Equation for FR calculation:

$$FR(m_i) = \varepsilon + \sum_{g_k \in TG} exp(-[d(m_i, g_k)] + r_{g_k}) \tag{1.1}$$

Reachable CRC genes are defined by the TF-TG network starting with the target genes of miRNAs and ending with the non-TF genes (TG) regulated by TF. Distance is defined by the number of edges in the graph from the miRNA to the gene. Rank is defined by a priori *knowledge of very relevant (Rank 1) and relevant (Rank 2) to CRC. CRC: Colorectal cancer. Taken from [44].*

1.5 Bioinformatics tools for wheat miRNA analysis

Due to the emergence of Next Generation Sequencing (NGS) technique since the early years of 2000 and the resulting large amount of RNA-seq data produced in the research projects using Illumina, Ion Torrent, Helicos and other NGS platforms [45], the trend for wheat miRNA analysis has become to assemble an analysis pipeline by choosing and incorporating the existing bioinformatics tools that are designed specifically for a particular step to process NGS data. Such pipelines will be discussed in Section 1.6. To improve our choice of the bioinformatics tools, this section will summarize and compare current bioinformatics tools that have been developed to pre-process NGS sequencing data, the tools to locate the genomic origin of miRNAs, the tools to learn and model the biogenensis process of miRNAs, and the tools to simulate the target gene recognition mechanism of miRNAs.

1.5.1 Adapter trimming

Due to the sequencing methods, an adapter is often added to either or both of the 3' and 5' ends of the reads. Many analysis procedures require the removal of the adapters before processing. Cutadapt [46] is one of the most popular tools. It could also be a simple in-house script that removes a given adapter sequence.

1.5.2 Alignment

Alignment is used to locate the reads to the reference sequences or the genome [47]. FASTA [48] and BLAST (Basic Local Alignment Search Tool) [49] are local alignment algorithms that are much faster than the full alignment Smith–Waterman algorithm [50]. Using seed sequence within the query to start off the alignment scoring around the seed, BLAST is still the most used alignment method since it was introduced.

As NGS data are emerging, BLAST is not fast enough to process the large data.

K-mer indexing (hashing) was then applied in tools such as BLAT [51] to speed up the searching for the genomic regions likely to be homologous to the query.

Instead of hashing the references, MAQ [52] and SHRiMP [53] hash the reads and work towards mapping short reads to the genome.

The collection of indice often consume a great portion of the available memory of a personal computer. Suffix/prefix tries with Burrow-Wheeler transform is one of the successful ways to store and search for an index. Example alignment method implementations are BWA [54], Bowtie [55] and SOAP2 [56].

NCBI BLAST group and Johns Hopkins Computer Science group are the most active development teams for alignment methods. More sophisticated alignment methods are developed for specific tasks. For example, both bowtie and bowtie2 are designed for short reads alignment, but bowtie2 works better for longer short reads [57]. HISAT [58] and HISAT2 [59] using graph-based indexing scheme are specialized for human genome alignment and genotyping; TopHat2 [60] for spliced read mapper for RNA-Seq; StringTie [61] for transcript assembly and quantification for RNA-Seq.

1.5.3 Secondary structure prediction

The most cited RNA structure prediction algorithm is Zuker's mfold [62]. Incorporating Turner's parameters [63], RNAstructure server also gained popularity [64]. Build on top of both Zuker's algorithm, Turner's parameters and other modifications, Vienna RNAfold [65] is probably the most used program because it is Unix-oriented and offers a user-friendly web server.

With RNAfold, the RNA secondary structures are predicted by dynamic programming that evaluates the minimal free energy for foldings conditional on containing certain base pairings chosen *a priori* and takes into account for the folding temperature (usually default at 37°C), ionic conditions ($[Na^{+}]$ and $[Mg^{++}]$), the window size of the number of foldings that are computed (default numbering is related to the length of sequence), and other parameters. The server hosting the program is a cluster of processors [62].

1.5.4 MiRNA prediction

Most core classifiers of miRNA prediction are in fact pre-miRNA classifiers that validate plausible RNA structures of pre-miRNAs by learning the shape features of pre-miRNAs of known and experimentally confirmed miRNAs. Examples include ProMir, TripletSVM, miRabela, miPred, SSCprofiler, microPred, HHMMiR, SplamiR, miRFinder and MiRenSVM (listed in [66]), and more than 30 tools are capable of processing plants (listed in Tools4miRs [67]).

MIRcheck, miRdup and other tools further consider the position of the miRNA within the hairpin structure. MIRcheck was developed initially for a study of plant miRNA predictions. It takes as input three pieces of information: **a.** the sequence of a putative pre-miRNA hairpin, **b.** a secondary structure of the putative hairpin, and **c.** a 20 mer sequence within the hairpin to be considered as a potential plant miRNA [68]. MIRcheck users can restrict the total number of unpaired nucleotides, the number of bulged or asymmetrically unpaired nucleotides, the number of consecutive unpaired nucleotides and the length of the hairpin.

On the other hand, miRdup can be run in two modes. In the first mode, miRdup takes as input a pre-miRNA sequence, a predicted secondary structure and a position of candidate miRNA, and assigns the likelihood score of the candidate being a real miRNA. In the second mode, miRdup evaluates every possible combination of miRNA position and length of a given pre-miRNA candidate, and reports the most likely configuration, which is a beneficial mode for *de novo* miRNA mapping [66]. MiRdup is based on a random forest classifier trained with experimentally validated miRNAs from miRBase, with features that characterize the miRNA-miRNA* duplex. The simplicity of miRdup program makes it faster than many pre-miRNA classifiers.

1.5.5 MiRNA target prediction

High-throughput miRNA target prediction is done either with experimental evidence or based on *in silico* algorithm. To reveal the identity of a small fragment of miRNA-targeted transcript, the length of the fragment of interest is extended by Rapid Amplification of cDNA Ends (RACE) before subsequent sequencing [69]. Degradome sequencing is a modified version of this technique using deep sequencing methods such as Illumina's SBS technology [70]. On the other hand, there are more than 20 bioinformatics tools available for miRNA target prediction for any plant (listed in [67]), such as miRanda, RNAhybrid, psRNAtarget, psRobot, WMD3 and TAPIR. For animal miRNAs, TargetScan and Diana-microT are two popular target predictors [67]. Selected miRNA target predictors are listed in Table 1.4. Among them, miRanda is an ideal tool for the prediction pipeline assembly, because it is the most cited target predictor and works locally.

Table 1.4 Selected miRNA target predictors

Program	Range	Local/online	Dependency	ML	Year	Citation*
miRanda	any	local	C, RNAlib (ViennaRNA)	no	2004	2902
RNAhybrid	any	local	na	no	2006	637
CleaveLand4	any	local	na	no	2008	208
UEA sRNA workbench	any	local	na	no	2012	127
IntaRNA	any	online	na	no	2008	264
MicroInspector	any	online	na	no	2005	241
psRobot	plant	local	Mfold-3.5	no	2012	82
WMD3	plant	local	SQLite or MySQL	no	2008	489
TAPIR	plant	local	bioperl, ViennaRNA with perl modules	no	2010	102
psRNAtarget	plant	online	na	no	2011	729
SVMicrO	mammal	local	na	yes	2010	83
TargetScan	animal	local	Perl	no	2015	418
Diana-microT	animal	online	na	yes	2013	229
mirMark	animal	na	na	no	2014	10

** : nb google scholar citation as of May 2017; na : not assessed; Range : working for plant, animal, or virus; ML : Users can provide their own training sets for building the machine learning model.*

1.6 Bioinformatics pipelines for wheat miRNA analysis

In order to understand the functions of miRNAs, we must predict conserved and novel miRNAs from the genome and/or transcriptomes; predict the target genes; imply the functions of the target genes as the functions of the upstream miRNAs.

In this section, the discussion will firstly be focused on the existing pipeline-like tools for miRNA prediction. Secondly, a summery will present the history of wheat miRNA prediction pipeline development before NGS when the chromosome walk approach was dominated. Finally, as the demand of the pipeline becomes more and more sophisticated, our team developed a comprehensive pipeline that also incorporated well-made bioinformatics tools to enhance the performance and analysis for NGS data processing.

1.6.1 Available pipeline-like tools

There are about 20 miRNA predictors for NGS data of any plant [67]. Selected pipeline-like tools are listed in Table 1.5. None of them are capable of processing wheat NGS data, due to wheat's huge genome and other limitations. MiRDeep-P [71], a pipeline specifically for plant, and miRDeep2 [72], an upgraded predictor for any species, are two of the most popular tools in this category. Moreover, take miRDeep-P for example, it requires manual operation to chain the steps in the pipeline, runs in one computer but not a cluster, and does not incorporate any memory management mechanism to handle large reference genome sequences.

Table 1.5 Selected pipeline-like NGS miRNA predictors

Program	Range	Local/online	Year	Citation
CAP-miRSeq	any	local	2014	67
miRanalyzer	any	local/online	2011	217
miRDeep-P	plants	local	2011	147
miRDeep2	any	local	2011	765
mirTools 2.0	any	local/online	2013	54
sRNAtoolBox	any	online	2015	78
UEA sRNA Workbench	any	local	2012	182
wapRNA	any	local/online	2011	43

*** : nb google scholar citation as of August 2018; Range : working for plant, animal, or virus;*

1.6.2 Wheat chromosome walk

In the past decade, several bioinformatics workflows have been constructed to predict plant or wheat miRNAs. A few classical examples are described in this section.

Sunkar R. and Jagadeeswaran G. [73] used BLASTN to search against miRBase-10 for conserved miRNAs among genome survey sequences (GSS), high-throughput genomics sequences (HTGS), expressed sequenced tags (ESTs) and nonredundant (NR) nucleotides databases of 155 diverse plant species, including wheat. The secondary structure of the pre-miRNA sequences of the candidates were verified with the assistance of an RNA secondary-structure prediction software, mfold [62]. They identified 682 miRNAs. A few of them were monocot-specific.

Wu Y. et al developed MiRPara [74], a prediction model learning miRNA and pre-miRNA features to predict miRNA coding regions in the genome with 80% of accuracy against experimentally verified mature miRNAs. They suggested to use this tool prior to high throughput miRNA prediction experiments.

As new genomic sequence of wheat became available, Lucas SJ. and Budak H. conducted a search for conserved miRNAs in GSS of wheat Chromosome 1AL [75]. They firstly used BLAST to screen for sub-sequences within the GSS reads that had less than 2 mismathces with a known mature miRNA sequence. The retained GSS sequences had an average length 532 bases harboring a putative miRNA. They implemented Zuker algorithm in their SUmirFold script and checked if these GSS sequences form pre-miRNA-like hairpin structures. In total, 42 sequences of conserved miRNAs were identified on 1AL.

These studies have made significant progress in advancing our knowledge of wheat miRNAs. However, a lot of information could be missing since they were done before the first almost-complete wheat genome assembly was released [76]. Their approaches were chromosome walk style based on **static** genomic or EST databases, without considering the dynamics in strains and conditions for differential MIR gene expression. And their prediction accuracies were either poor or not properly assessed. Moreover, there is a need to review the prediction algorithms and the tools incorporated to ensure the prediction complies with current scientific knowledge and demands by NGS data processing.

1.6.3 Wheat NGS transcriptome analysis

The prediction of wheat miRNAs has focused on conserved and highly expressed species derived from the orthologous sequences of known miRNAs, ignoring those that appeared recently in evolution or are expressed with low abundance. While NGS emerges, it has become feasible to screen for miRNAs that may be expressed with large quantity only in certain strains or conditions.

Therefore, a few years ago, our team proposed a conservation-independent approach [38] that took into account not only the criteria of plant miRNA annotation with the aid of machine learning techniques and recent bioinformatics tools but also the miRNA expression profiles. For the latter and to complete the previously identified miRNA repertoire associated with some abiotic stresses in wheat, they applied this new approach to analyze the development and tolerance to cold, salt and aluminum (Al) stresses in 10 wheat RNA expression libraries from different tissues and conditions.

With that approach in mind, they assembled a pipeline to predict wheat miRNAs and their targets [38]. It consists of three parts: Small RNAs sequencing, Bioinformatics predictions and Functional analysis. Bioinformatics tools take the small RNAs produced and sequenced under different treatments by NGS Sequencing for miRNA prediction, and use the wheat Expression Sequence Tags as reference for matched pre-miRNAs prediction (MAQ). The sRNA sequences are extracted according to the primary sequence features of small RNAs and pre-miRNAs and are subjected to RNAfold to predict secondary structure. Candidates are retained if they satisfy the pre-miRNA hairpin sequence and free energy features for secondary structure defined by MiPred or the typical miRNA hairpin topology features defined by HHMMiR, both trained for machine learning on pre-miRNAs from wheat and plants. Considering the localization of the candidate miRNA within the hairpin, the expression profiles of the residues within the hairpin, and/or other plant-miRNA characteristics, the authors apply MiRdup*. MiRdup*, a classifier developed recently by the authors, was trained on all miRBase, all plant or monocot only, so it increases species-specificity. miRNA candidates also undergo a general filtration to eliminate the sequences known not to be miRNAs, such as sequences with low complexity (RepeatMasker) and rRNAs.

Finally, 199 miRNAs were found. In plants, miRNAs usually silence their target transcripts through a high degree of complementary base pairing. They predicted the target genes of miRNA candidates using TAPIR, and further compute the functional enrichment of the GO Slim terms associated with these genes. Expression patterns of the predicted miRNAs were different among the libraries, and selected ones were confirmed by northern blot. The predicted miRNAs were clustered in 43 families, some were known in wheat, some were in plants, and some were not yet been reported. The putative functions of miRNAs in response to abiotic stresses and development were classified into 24 groups according to their differential expression in two wheat genotypes of different tolerance abilities. Their targets were enriched in the stress adaptation-related cell components, biological processes or molecular functions for gene expression, plant defense, cell growth and flowering.

1.7 Databases to assist in wheat miRNA annotation

1.7.1 MiRNA databases

MiRBase (`http://www.mirbase.org`) [77] is a miRNA repository, containing more than 24521 miRNA loci from 206 species to produce 30424 mature miRNAs (record of miRBase-20, 2013). Current version is miRBase-22.

PMRD (plant microRNA database, `http://bioinformatics.cau.edu.cn/PMRD/`) [78] collects miRNAs from 128 plant species, including wheat *Triticum aestivum*. The source of each miRNA collected is indicated either experimental or computational.

Wheat MicroRNA Portal (`http://wheat.bioinfo.uqam.ca`) [79] accumulates wheat miRNAs and associated annotations, and provides a suitable selection of recommended tools for miRNA analysis.

1.7.2 Reference genomes

The International Wheat Genome Sequencing Consortium (IWGSC) is responsible for wheat genome sequencing. Its repository server WHEAT@URGI PORTAL is hosted by INRA University, Paris. Bread wheat complete genome assembly, TGACv1 CS42 (published in 2017) [76], also known as Ensembl-39, was used in this study for target search. CS stands for Chinese Spring. Other names of this variant include Canadian hard winter wheat, *Triticum aestivam*, *Triticum aestivum8*, *Triticum vulgare* and common wheat. Cultivars *Manitou* and *Norstar* are varieties of this variant. The updated Ensembl-40 (accessible in January 2017, published in August 2018) has been named as IWGSC REFSEQ V1.0 [80].

TGACv1 represents 78% of the wheat genome [76]. And IWGSC RefSeq V1.0 represents 90% of the wheat genome [80].

Wheat is hexaploid, with an estimated genome size at about 17 Gbp, composed of three closely-related and independently maintained genomes, A, B and D. Detailed annotations and Relationships among the three genomes were recently systematically analyzed based on the newly assembled full genome [80]. The ancestral progenitor A-genome donor is considered to be *Triticum urartu*. The B-genome donor is related to *Aegilops speltoides*. The tetraploid emmer wheat then hybridized again with the D-genome donor *Aegilops tauschii* to produce modern bread wheat. The transcriptomes or open reading frames of several *Triticum* species are centralized in Ensemble database, including the wild winter wheat *T. monococcum ssp. aegilopoides* (accession G3116), the domesticated spring wheat *T. monococcum ssp. monococcum* (accession DV92), tetraploid *T. urartu* and *T. turgidum*. Their genetic variations, such as SNP (single nucleotide polymorphism), can be called by the alignments of A, B, D genomes and wheat subspecies.

The Ensembl plants website (`http://plants.ensembl.org`) provides access to more than 50 plant genomes.

Through PlantGDB (plant genome database, `http://www.plantgdb.org`), one can display plant genomes and annotations from several other databases, such as Genome Browsers.

The ongoing *The 1000 plants* project (`http://sites.google.com/a/ualberta.ca/onekp/`) will contribute to our understanding in crop improvement.

Medicinal Plant Genomics Website (`http://medicinalplantgenomics.msu.edu`) collecting a dozen of plants could provide interesting insights in plant metabolism study.

1.7.3 Transcriptome repositories for sRNA

The Meyers lab at the Donald Danforth Plant Science Center (`http://mpss.danforthcenter.org`) hosts a Next-Generation Database for *Arabidopsis*, wheat, and other 25 species [81], concentrating on plant small RNA libraries.

GEO (Gene Expression Omnibus, `http://www.ncbi.nlm.nih.gov/geo/`) is a database repository of high throughput gene expression data. Many experiments of plant full RNA or small RNA transcriptomes deposited in this database are not yet collected in the Danforth database.

1.7.4 Network database

The Gene Ontology (GO) Consortium (`http://www.geneontology.org`) [82] provides a comprehensive resource for computational model of biological systems regarding the functions of genes and gene products. It classifies functions in terms of three aspects: **molecular function** (GOMF) for molecular activities of gene products; **cellular component** (GOCC) for where gene products are active; **biological process** (GOBP) for pathways and larger processes made up of the activities of multiple gene products.

KEGG Pathway database (`http://www.genome.jp/kegg/pathway.html`) [83] is a collection of manually drawn pathway maps representing our knowledge on the molecular interaction, reaction and relation networks for metabolism, genetic information processing, environmental information processing, cellular processes, organismal systems, human diseases and drug development.

Protein-protein interaction or interactome databases include BioGRID (`http://thebiogrid.org`), EMBL-EBI IntAct (`http://www.ebi.ac.uk/intact/`) and others specifically for human, mammalians and yeast. Few of such databases are dedicated to plants.

CHAPTER II

RESEARCH PROPOSAL

2.1 Introduction

In the past five years, our laboratory has stayed at the forefront of computational miRNA prediction and analysis, and published a series of miRNA classification algorithms and tools [38, 79, 84]. As described in Section 1.6.3, a comprehensive wheat miRNA prediction pipeline that improves miRNA prediction by combining profiles and criteria of plant miRNAs, machine learning and bioinformatic tools, has led to the discovery of wheat abiotic-stress responsive miRNAs. It not only reduces the false prediction but also distinguishes conserved, clade and species-specific or young miRNAs. These discoveries have also contributed the miRNA knowledge to the regulatory mechanisms in different stresses. Other groups have also developed NGS pipelines for wheat miRNAs since 2015, varying in efficiency and accuracy due to the tools and methodologies [1, 40, 85, 86, 87]. Overall, these tools are semi-automated, not able to scale-up and lack of a user-friendly interface.

The emergence of the Next Generation Sequencing increases drastically the volume of transcriptome data. It is also hypothesized that studying sequences in a multi-library dynamics reduces false predictions and better predicts their roles in the studied libraries. However, the functional annotation of plant miRNAs has never been systematically achieved. One of the main obstacle is exactly the time requirement for processing the large volume of NGS sequencing data. For example, an average-sized *Arabidopsis* sRNA NGS library would take an automated version of miRDeep-P program almost 10 hours to process (accessed in this work). This does not even count for the time needed for optimization, manual chaining and processing several libraries in one experiment. In animal miRNA research, miRNA prediction can be scaled up by Pan et al's MapReduce method [88], and target prediction can be scaled up by MR-microT [89].

2.2 Research problems

Judging our experience and expertise in this field, I decide to focus my research objectives of this project in identifying the best approaches to develop an efficient platform that analyzes miRNA faster while boosting the classification accuracy by integrating the learning of massive data and functional classes of miRNAs at the kernel of the prediction. There are several challenge-associated steps involved in the identification of the biological roles and the functional classification of miRNAs:

Problem 1: miRNA prediction. It has become a trend for biology researchers to purchase service hours on a cloud cluster rather than invest in the hardware and maintenance to compute big data. Although many algorithms and workflows for novel miRNA prediction have been proposed, including one by our team specifically for wheat [38, 79], none of them is designed for efficiently processing large volume of sequence data. Improvement of an efficient algorithm has been made to facilitate the prediction at many individual bioinformatics tools. For example, the alignment tools have evolved more and more sophisticated and optimized for a specific type of sequence alignment. Since there are more than 30 computational steps and several bioinformatics tools required for a proper prediction workflow, it makes sense to parallelize the entire workflow rather than improve each of the individual steps that have been well made and updated.

Problem 2: Annotation of the target genes with pathway data. MiRNAs usually silence their target genes. Analyzing the functional annotations, such as pathways, of target genes helps to interpret the function of miRNAs. However, the pathway data are not readily available for many species that do not have a fully assembled and annotated genome. This fact is a major obstacle for conducting functional enrichment analysis.

Problem 3: Boosting prediction from functional analysis by miRNA-targeted pathways. The significant and functionally relevant targeted pathways of the miRNAs are identified through enrichment analysis. Hypergeometric distribution and permutation tests are two of the prevailing statistical enrichment measures of overly presented features in a dataset. Since miRNA mainly attenuates gene expression, an overexpressed miRNA should match with suppressed target genes. DIANA-microT-CDS, specific for a few animal species, is the only tool that provides the workflow to compute enriched pathways of target genes [90].

2.3 Specific aims: distributed miRNA prediction methodology and real world datasets application

Although wheat is an important crop, its genome is large, complex and not fully annotated [76, 80]. To improve our knowledge of wheat miRNAs, we need to not only improve the bioinformatics prediction methods but also design and collect proper biological samples for a meaningful analysis. We recently acquired new field experimental data containing 80 libraries of 4 different vernalization strains of wheat treated by different degrees of cold stress conditions: Four wheat cultivars that differ in their freezing tolerance and vernalization responses including winter habit Norstar, isogenic Norstar (Iso8S), spring habit Manitou and isogenic Manitou (Iso11W) were grown under field conditions in two different years, 2010 and

2013, and were sampled at five time points between the beginnings of September and November in Saskatoon for the construction of small RNA libraries.

In order to solve *Problem 1*, I will study how distributing and parallelization algorithm could be used to accelerate the miRNA prediction. To this end, I will study the architecture of the scalable methods, such as MapReduce and Spark, to propose an improved version of our wheat pipeline for a high volume data facility.

Aiming at *Problem 2*, I will study how data integration could improve the pathway annotation. For this, I will explore the databases of Kyoto Encyclopedia of Genes and Genomes (KEGG) pathways, Gene Ontology terms and protein-protein interactions as well as literature mining. Next, I will investigate how these approaches could complement the molecular function annotation.

To achieve *Problem 3*, I will study how enrichment analysis could provide a comprehensive functional classification, especially taking into account the expression profiles of both miRNAs and their target genes. I will also study the feasibility of a cloud service that integrates the mentioned miRNA prediction, target annotation, and functional analysis.

The following chapters are organized as follows.

In Chapter III, I will present the implementation of a **distributed miRNA prediction methodology**, mirLibSpark, established and standardized by experimental validated plant miRNAs.

In Chapter IV, I will apply the above methods on the **real world datasets of field wheat** to study towards the knowledge of the cold-responsive regulatory mechanisms among wheat strains and conditions from miRNA to function.

In Chapter V, I will conclude the contributions presented in this work, and discuss the limitations and future directions of this work.

In Appendix A, the contributors and the organization of the mirLibSpark project files in GitHub repository are illustrated, and the contents of highlighted files are displayed, including the file of parameter settings *paramfile.txt*; the example file to launch mirLibSpark in a Compute Canada server *submit_j.pbs*; and the source codes of the mirLibSpark core program *mirLibPipeline.py* and *mirLibRules.py*, the sequential version of mirLibSpark *sequential.py* and the automated version of miRDeep-P *run_mirdeep_p.py*.

In Appendix B, diverse supplemental information for this work is presented, including the GitHub address of mirLibSpark project repository; the Dropbox address of supplementary materials and data related to Chapter III and IV, respectively; the first page of the manuscript for the mirLibSpark pipeline intended to submit in a peer-reviewed journal; some resources that assisted in the development of mirLibSpark pipeline but were not cited elsewhere; and finally the description of the 365 KEGG pathways in wheat discovered in this study along with the coverage rates of pathway components by the field wheat transcriptomes.

CHAPTER III

MIRLIBSPARK: A SCALABLE NGS MICRORNA PREDICTION PIPELINE

ABSTRACT

The emergence of the Next Generation Sequencing increases drastically the volume of transcriptome data. Although many algorithms and workflows for novel microRNA (miRNA) prediction have been proposed, they are mostly semi-automated and few are designed for processing large volume of data. This work aims to develop an efficient pipeline that performs a faster analysis of short RNAs libraries while increasing the classification accuracy. We propose an improved pipeline for a high volume data facility, by implementing the mirLibSpark prediction pipeline based on the Apache Spark framework. It can process data 100 times faster than a program that executes the same steps without parallelization, and at least 568 times faster than miRDeep-P after we made a script to run all their manual steps automatically. Moreover, the performance measures are better than those of miRDeep-P, a predictor of plant miRNAs. We have tested it on computer clusters as well as local environments. We have also applied our pipeline on more than one hundred libraries of A. thaliana small RNAs and demonstrated that analyzing the expression patterns of the predicted miRNAs among the libraries enables the classification of functional miRNAs. We deliver here the first fully automated and distributed miRNA predictor. It is an efficient and accurate miRNA predictor with functional insight. Eventually this pipeline will be extended to analyze small RNAs of other organisms and conditions such as human diseases. It will benefit the miRNA research community for its applications.

3.1 Introduction

3.1.1 Annotation of miRNAs as a big data problem

MiRNA prediction. It has become a trend for researchers to purchase service hours on a cloud cluster rather than invest in the hardware and maintenance to compute big data of high volume, variety and velocity [91, 92]. Data integration across genomic and transcriptional sequence platforms allows meta-analysis of miRNAs and help to justify their biological relevance [93]. The needs to quickly predict plant miRNAs and analyze meta-data will soon become a burden [94]. Hadoop MapReduce framework is a parallel programming model for processing large data with a distributed architecture, and has recently benefited bioinformatics researchers to realize efficient, scalable and reliable computing performance on Linux clusters and on cloud computing services [95]. Scalable methods are revolutionizing traditional bioinformatics tools in RNA research [96, 97]. Although more than 25 algorithms and workflows for novel miRNA prediction in plants have been proposed [67], including the one by our team specifically for wheat [38], none of them is designed for parallel processing large volume of sequence data. There is only one miRNA predictor employing MapReduce by [88] but its algorithm is specific to animal genomes [88]. Being the first of its kind, it incorporates some bioinformatics tools that are not optimal for the task. Therefore, it is an urgent quest to similarly propose an improved version of our wheat pipeline to have a high volume data facility.

MiRNA target gene prediction. miRNAs usually silence their target transcripts in plants through a high degree of complementary base pairing. *In silico* miRNA target prediction is indispensable for the design of further experimental analyses. More than 23 bioinformatics tools are available for plant miRNA target prediction [67], including Tapir [98] and psRNAtarget [99] for plants. However, none of them has incorporated pathway data with the prediction. miRNA studies in animals are often more advanced. Diana tools not only predict the animal miRNA targets but also identify the significantly targeted pathways [100]. MR-microT, a module of DIANA-microT-CDS, is the only tool using MapReduce paradigm in Hadoop framework to predict the animal miRNAs targets [89].

3.1.2 Hadoop framework

The Apache Hadoop project is an open-source software that allows distributed processing of large data across clusters of computers using simple programming models. It is designed for reliable, scalable and distributed computing. It can scale up from single server to a cluster of machines, each offering local computation and storage, and handle failures at the application level. The project includes these modules: Hadoop Common, Hadoop Distributed File System (HDFS), Hadoop YARN and Hadoop MapReduce. In particular, MapReduce is a data processing programming framework that parallelizes the computation procedures across data sets. It is the main concept that we will apply to our pipeline to improve the speed by data and computation parallelism. However, MapReduce programming is hence restricted by the acyclic dataflow of mapper and reducer, denying the revisit of a certain work set.

Hadoop framework is provided by several distributors, including Apache, Coudera, MapR and Horton Works, as shown in the Table 3.1. They distribute various combination of Hadoop framework including MapReduce, HDFS, Spark, etc. Among them, is Cloudera, which is open source and the free version is supported by an active community.

Table 3.1 Distributions of Hadoop framework

	Code	Price	Support
Apache	open source	free	community
Cloudera	open source	free/premium	community/pro
MapR	open source/Proprietary	free/premium	community/pro
HortonWorks	open source	free/premium	community/pro

Cloudera QuickStart virtual machines (VMs) provide an equipped environment for Hadoop framework in a single-node cluster, allowing users to quickly engage in the implementation of big data processing. Our prototype pipeline portrayed in this report will require some elements conveniently provided by Cloudera, for example, an Unix environment, Python, Perl and Spark.

We downloaded and installed the *Cloudera distribution of Apache Hadoop* (CDH) in a virtual machine *Oracle VM VirtualBox*. Cloudera QuickStart VM (single-node cluster) uses package-based install and contains *Cloudera distribution of Apache Hadoop* (CDH). Python, Perl and Spark are in an Unix environment.

3.1.3 Apache Spark

Apache Spark is an engine and interface for processing large-scale data in parallel over a cluster of nodes. Spark exposes an abstraction of a distributed memory called *Resilient distributed dataset* (RDD) and a set of operations including transformations (*map*, *flatMap*, etc.) and actions (*reduce*, *collect*, etc.) [101]. The concept of RDDs improves the performance of data processing by in-memory saving intermediate data of iterative and interactive tasks. RDD is a read-only collection of distributed objects and built from data saved on disk or from other

RDDs with transformations. Transformations of RDDs are lazy, *i.e.* the computing and materialization of RDDs are done if only actions are applied on their RDD children. Each RDD keeps the information of all the transformation of its parents (*lineage*). This property confers also RDDs a tolerance to failures.

Spark is implemented in Scala, a functional and object-oriented programming language, and provides Java and Python interfaces. Users write a *driver* program that connects to a set of distributed processes. The driver program defines one or several RDDs, applies transformations and actions on them and tracks their lineages. Worker processes store RDD partitions on memory of each node. More resources can be found in Appendix B.3.

Deployment in cluster

The initial development of mirLibSpark was done in a local machine with Cloudera virtual clusters of 1 to 4 nodes. Then mirLibSpark was uploaded to a server in Compute Canada for further testing and generalization. The servers tested include Guillimin and Colosse.

3.2 Implementation of mirLibSpark

3.2.1 Choice of bioinformatics tools

We will incorporate improved algorithms, machine learning methods and open source bioinformatics tools with fewer dependencies that are optimal for the tasks and oriented toward the prediction in plants. Several criteria are imposed to the choice of the programs. Criteria and examples of the decisions are as follows:

(1) The performance of the program is equal or better to that of the

ones chosen in [38]. RepeatMasker took one month to process the data in 2015. There are more than 15 softwares available for finding repeats, including CENSOR, Datter, ReAS and Recon. DustMasker took only a fraction of an hour to process very large files in our current test. Same situation applies to alignment programs. MAQ took a few days to align data that Bowtie processed in a matter of hours.

(2) The program is currently maintained and widely accepted by the researchers as one of the best tools for its purpose. It exists Bowtie (or Bowtie 1) and Bowtie 2. However, Bowtie 2 has a different purpose of usage for reads longer than about 50 bp. Bowtie 1 is sometimes faster and/or more sensitive for relatively short reads (e.g. less than 50 bp), according to its manual. It also exists crossbow that integrates Bowtie, SoapSNP and Elastic MapReduce on Hadoop cluster. At first glance, it might suit our needs for the application of miRNA prediction on Hadoop framework. However, it actually forms an inseparable part with SoapSNP for resequencing analysis as genotyper. Therefore, we will develop our own procedures to integrate Bowtie in our framework.

(3) The program is open source to facilitate the pipeline service distribution. All the programs incorporated by [38] and mirLibHadoop are open source. Softwares with permissive licenses (MIT, BSD, Apache, LGPL) so can be distributed without restriction. Softwares with copy left license (GPLv2,3, AGPL) could be distributed only if it is called externally with system but their codes are not extracted or modified.

(4) The program requires less dependency. Dependency means more configuration of each nodes, and therefore does not favor service distribution. Bowtie pre-compiled binary file requires TBB library to perform multi-threading. However, TBB library is not readily available and can be replaced by built-in multi-threading function such as native Windows threads, which does not require extra library. Moreover, bowtie execution is managed by Spark, which takes care of multi-threading and task parallelization. Therefore, we did not choose the binary file. Instead, we used the bowtie version that we compiled from source code and disabled the TBB dependency. Although RNAfold requires gsl library to facilitate its mathematical routines, it is still the best program available for folding and possibly that no one can perform a proper folding calculation without importing a math library.

3.2.2 Passing data to the programs in a pipeline

There are three possible modes to pass data to a program in a Spark-coordinated pipeline: by-file in command line, by-element in command line and by-piping elements. By-file is the default mode for most of the native programs due to processing speed. Passing by-element in command line is also provided by the native programs. This mode is easy to achieve in Spark by creating child processes for each element. However, the creation of child process for each element decelerates significantly the speed by 120 folds as compared to by-file mode. Spark favors the data processing element by element. Its piping function creates only one child process for an RDD and pipes its elements one by one to the programs. Spark piping is easy to implement and is as fast as by-file mode. Therefore, we implemented this mode to execute the programs in the pipeline. An example of the time of execution for Bowtie is shown in Table 3.2.

Table 3.2 Time of execution of Bowtie with 0.25 million sequences

mode	Pass by file	Pass by element	Pipe element
time (secs)	5	600	5

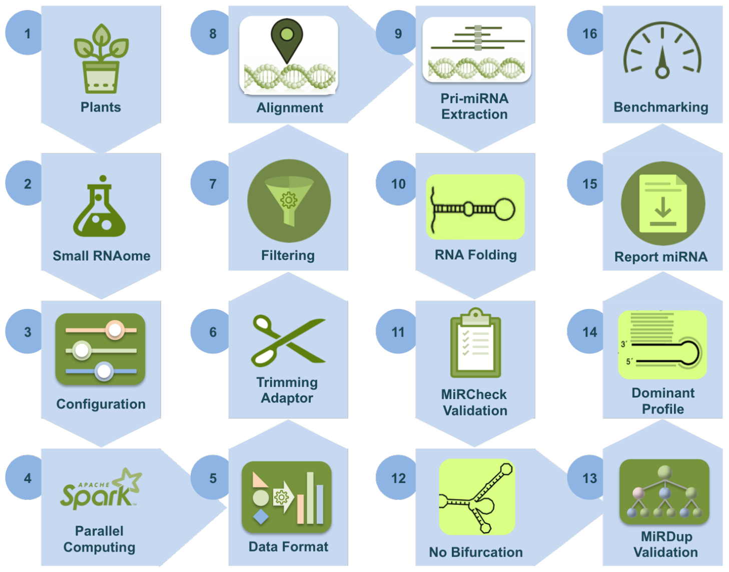
3.2.3 Management of the project development

To coordinate the work of two developers for this project, we used Git, a version control system for tracking changes in computer files and coordinating project development among multiple collaborators. The structure of the project is shown in Appendix Figure A.1. The **src/** folder contains the source code developed for this project, including the pipeline script (*mirLibPipeline.py*), the rule script defining the functions for the RDD operations (*mirLibRules.py*) and the wrapper script to execute MIRcheck program (*eval_mircheck.pl*). The **lib/** folder contains the dependencies of the pipeline in the format of source code. Spark and pipeline parameters are defined in *paramfile.txt*. Users can easily modify all the parameters according to their needs, including running the pipeline in a local mode or cluster mode, the resource allocated to Spark and the values of the cutoffs.

3.2.4 Spark RDD chaining in mirLibHadoop

The mirLibSpark platform uses four major computational steps for miRNA prediction, including filtering, alignment, folding and prediction. The proposed workflow that mirLibSpark will handle is shown in Figure 3.1. It uses Spark to orchestrate these steps. A NGS dataset contains a lot of sequences that are expressed at a very low level. These sequences are often the degradation by-product of miRNAs, mRNAs and other functional RNAs. Such low-expression small RNAs are not candidates for functional miRNAs and need to be excluded by introducing a general cutoff filter. In addition, wheat and other plant genomes contain a lot of repetitive sequences ($> 80\%$) which are unlikely pre-miRNA candidates. To predict pre-miRNA, the RNA sequences will undergo RNAfold software to compute their secondary structure for their likelihood being a pre-miRNA candidate. This folding step is very time consuming. Therefore, we are also motivated to remove the repetitive sequences before computing the folding.

Figure 3.1 Overview of mirLibSpark workflow



The mirLibSpark implementation covers the step 3 to the step 11. The step 16 (Benchmarking) is used to optimize the step 3 (Configuration) in terms of speed and prediction performance.

A simplified illustration of Spark RDD chaining in mirLibHadoop is shown in the right panel of Figure 3.2. For each submitted task, a *SparkContext* is initiated with the configuration defined in $paramfile.py$. The first RDD object is created by reading the input data.Thereafter, there are more than 30 operations applied on RDDs until the output is obtained and printed to a file. According to the need to process each element in the data frame, the RDD operations used in the pipeline include *map*, *filter*, *pipe*, *persist*, *groupByKey*, *join* and *collect*. In particular, two RDD objects are reused in later operations. One is being joined with another RDD to create a new one. The content of the other RDD object is stored in a dictionary and passed as parameter for another operation. Such reused RDD objects are persisted in the memory so that they can be used without re-computing.

During the implementation phase of mirLibSpark, several particular approaches were taken to speed up the processing:

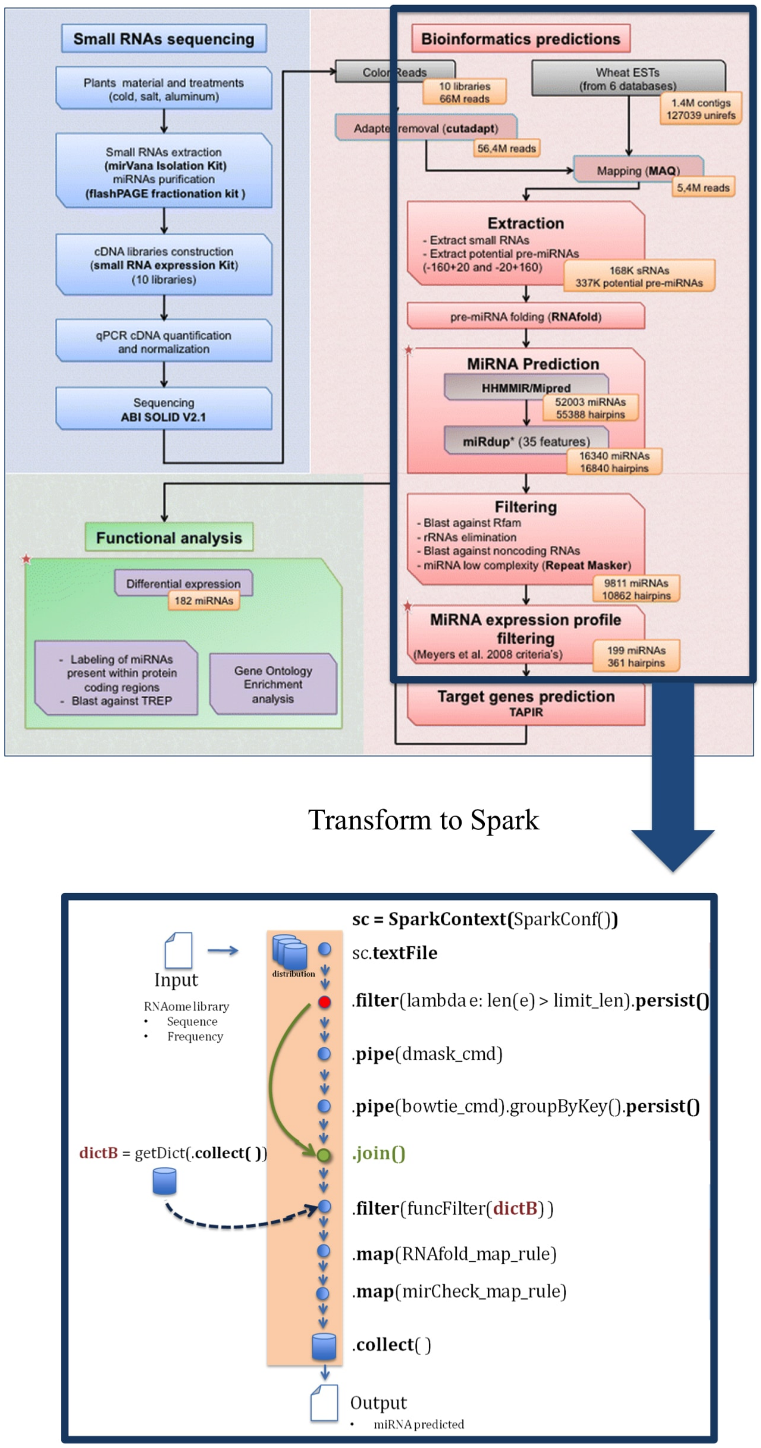
Input textfile RDD partitioning. In order to use every node in the cluster, there should be as many elements in the RDD as the number of available nodes. In practice, to saturate the nodes, the input textfile is partitioned twice as many as the number of nodes.

Piping element to external programs in the pipeline. If an external program is used for a particular step rather than an in-house script built within the pipeline, piping mode is much faster.

Accessibility of data. Certain reference files, scripts and RDD-derived data structures are required for every node. To save the communication between the salve nodes and the master node, a copy of these materials is either *added* or *broad casted* depending on the material types.

Temporary memory. Certain types of RDDs benefit from being *persisted* to prevent duplicated calculation and to increase the availability during piping for the next RDD operation.

Figure 3.2 mirLibSpark pipeline


Small RNAs sequencing
Plants material and treatments (cold, salt, aluminum)
Small RNAs extraction (mirVana Isolation Kit) miRNAs purification (flashPAGE fractionation kit)
cDNA libraries construction (small RNA expression Kit) (10 libraries)
qPCR cDNA quantification and normalization
Sequencing ABI SOLID V2.1
Bioinformatics predictions
Color Reads
10 libraries 66M reads
Wheat ESTs (from 6 databases)
1.4M contigs 127039 unirefs
Adapter removal (cutadapt)
56,4M reads
Mapping (MAQ)
5,4M reads
Extraction
- Extract small RNAs
- Extract potential pre-miRNAs (-160+20 and -20+160)
168K sRNAs 337K potential pre-miRNAs
pre-miRNA folding (RNAfold)
MiRNA Prediction
HHMMIR/Mipred
52003 miRNAs 55388 hairpins
miRdup* (35 features)
16340 miRNAs 16840 hairpins
Filtering
- Blast against Rfam
- rRNAs elimination
- Blast against noncoding RNAs
- miRNA low complexity (Repeat Masker)
9811 miRNAs 10862 hairpins
MiRNA expression profile filtering
(Meyers et al. 2008 criteria's)
199 miRNAs 361 hairpins
Target genes prediction TAPIR
Functional analysis
Differential expression
182 miRNAs
- Labeling of miRNAs present within protein coding regions
- Blast against TREP
Gene Ontology Enrichment analysis
Transform to Spark
Input
RNAome library
• Sequence
• Frequency
distribution
Output
• miRNA predicted

```
sc = SparkContext(SparkConf())
sc.textFile
.filter(lambda e: len(e) > limit_len).persist()
.pipe(dmask_cmd)
.pipe(bowtie_cmd).groupByKey().persist()
.join()
.filter(funcFilter(dictB) )
.map(RNAfold_map_rule)
.map(mirCheck_map_rule)
.collect( )

dictB = getDict(.collect( ))
```

Upper panel: miRNA prediction pipeline [2] consists of preprocessing, align-

ment, filtering, extraction and folding, and prediction. Our previous pipeline was semi-automated.

Lower panel: mirLibSpark uses the architecture of Spark to coordinate input data management and the prediction steps. Low-expressed sRNAs are excluded by filter() operations. By pipe() operations, sequences aligned within a small number of genomic loci by bowtie and high complexity determined by dustmasker are retained. By map() operations, pre-miRNA candidates surrounding the sRNA genomic loci are extracted and their secondary structures are computed using the RNAfold tool. The decisions of MIRcheck and miRdup are also rendered by map() operations. A dominantly expressed sRNA within the pre-miRNA candidate sequence is considered a true miRNA.

3.3 Experimental methods

In order to implement mirLibSpark and assess its performance and utility, the pipeline is firstly oriented to process the NGS sRNA libraries of *Arabidopsis thaliana* (*ath*). It is known that the processing time is affected by the size and the complexity of the genome. In contrast to *ath* having a genome size of only 135 Mbp (mega basepairs), wheat *Triticum aestivum*, for example, contains 16 Gbp in its hexaploid genome. Known miRNA data set is used to approve the prediction steps. A sequential version executing the same steps without parallelism is implemented to compare the time of execution with the pipeline. A miRDeep-P [71] wrapper is implemented to compare the prediction power and time with this miRNA predictor, which is also based on a probabilistic model of miRNA biogenesis but does not include a mechanism to exclude false positive predictions. Finally, we underline the importance of the ability to predict multiple libraries in one submission and incorporate a data aggregation module using k-means clustering method to improve the interpretation of the prediction results.

3.3.1 Arabidopsis thaliana sRNA data

The reference genome is TAIR10 and stored in bowtie1 index format in the mirLibSpark package. The 427 known *ath* miRNAs of *A. thaliana* were retrieved from miRBase-21. Certain miRNAs have same sequences but are derived from different pre-miRNA precursors. Distinct miRNA sequences were used as ID. Among known miRNAs, 349 distinct miRNA IDs were collected for this study, including 100 distinct high confidence miRNAs. For the case study applying this pipeline, 127 sRNA libraries in raw format were obtained from Danforth Center Arabidopsis Next-Generation Database [81]. One library was randomly chosen for the assessment of scalability of the pipeline. The size of this *standard library* is 50 megabytes consisting of 1.8 millions of sRNA sequence elements.

3.3.2 Implement a sequential version

The python script, *sequential.py*, is a prediction pipeline that mimics mirLibSpark but without Spark data management or parallelism. It executes in memory the same prediction steps as in the mirLibSpark and renders the same results.

3.3.3 Implement a miRDeep-P wrapper

The miRDeep-P predictor is semi-automated and requires users to execute several scripts and commands one after another. We implemented the python script, *run_mirdeep_p.py*, to execute the prediction automatically with the same parameters used in mirLibSpark, such as the number of locations aligned to the genome, the length of pri-miRNA extraction, and TAI10 genome and annotations.

3.3.4 Evaluate the prediction performance

The prediction performance was measured by F1 score and accuracy. Their formula are as follows:

$$F_1 = \frac{2 \times \mathrm{TP}}{2 \times \mathrm{TP} + \mathrm{FP} + \mathrm{FN}} \tag{3.1}$$

$$ACC = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{3.2}$$

In the formula, TP is true positive, TN is true negative, FP is false positive and FN is false negative. The positive data set was constructed of the 100 distinct high confidence *ath* miRNAs. The negative data set was constructed of randomly selected short sequences (18-25 nt) from *ath* genome. Known miRNAs were excluded from the selection. One thousand distinct sequences were collected. An artificial frequency was assigned to each sequence to prevent it from being filtered by mirLibSpark prediction criteria.

3.3.5 Supervised classification of the libraries

The tissue class labels were obtained from the Arabidopsis Small RNA Library Information section in the Danforth database. There are in total 15 different tissues among the 127 libraries for this study. The following three tissues are the major types found in this collection: flower (35 libraries), inflorescence (42 libraries) and whole aerial (24 libraries).

3.3.6 Unsupervised classification of the libraries

Expectation-maximization (EM) is a distribution-based model where statistical analysis is used to find the maximum likelihood [102]. Three clusters were found and each was assigned a name as flower-like, inflorescence-like and whole-aerial-like reflecting the major tissue class it contained.

3.3.7 Dimension deduction: Manifold and PCA

Manifold and PCA (Principle Component Analysis) learning are an approach to reduce dimension non-linearly. Manifold methods used in this study include Isomap, MSD, and Spectral Embedding.

To aid visualization of the structure of a dataset, the dimension must be reduced in some way and be composed 2 or 3 components (axes) so that it can be visualized in a 2D or a 3D illustration. Each component is a function of all the attributes.

PCA is one of the methods to reduce dimension and extract the significant attributes. Once the clusters can be visualized in PCA and several manifold projections, we can conduct the next step, *feature selection* using one of the methods. PCA is then chosen for feature selection in this study.

The coefficients of the first PCA component are ranked and at certain point the values of the coefficients for each attributes in the later portion do not reduce as much as the former portion. The former portion, comprised of about 3% of total attributes, are collected as important attributes.

3.4 Experimental results and conclusions

3.4.1 Comparing with miRDeep-P program

The same positive data set and negative data set were applied to assess the performance of miRDeep-P as shown in Table 3.3. Although miRDeep-P predicted the true positive miRNAs better than mirLibSpark, it predicted a lot more false positive miRNAs than mirLibSpark. miRdup is a stringent machine-learning module that reduces the chance of false predictions [66]. As a result, the statistic measures indicate that mirLibSpark is much more robust than miRDeep-P. Moreover, the execution time of miRDeep-P is also longer than mirLibSpark.

Table 3.3 Comparing with miRDeep-P

Program	TP	TN	FP	FN	F1	Accuracy
mirLibSpark	81	997	3	19	0.880	0.980
miRDeep-P	86	954	46	14	0.741	0.947

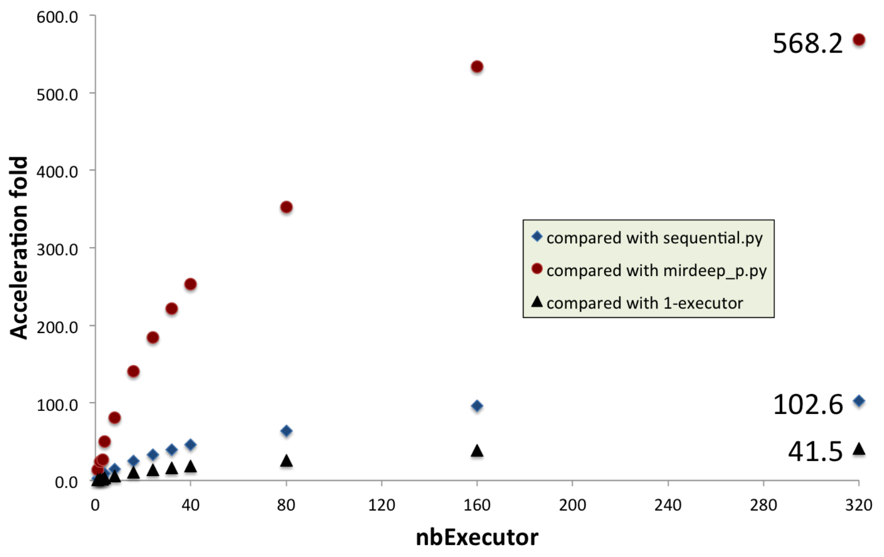
3.4.2 Scalability

To precess the standard library, mirLibSpark with 1-executor took 2579 seconds; the sequential version of the pipeline, *sequential.py*, took 6360 seconds; and *mirdeep_p.py*, the wrapper of miRDeep-P that we made to render it automated, took 35227 seconds.

As the number of executors were gradually increased to be allocated to the pipeline, the acceleration increased as shown in Figure 3.3. Using 320 executors, mirLibSpark accelerated 41.5 times from 1-executor. We have achieved 102.6 times of acceleration from the sequential version to the Spark version. Moreover, it is at least 568.2 times faster than the miRDeep-P wrapper, not including the human labor to operate the original semi-automated version.

Communication between executors is an important cause of the loss of efficiency. When the number of executors increased 2 folds, the time of execution reduced about 1.8 folds. The time of execution continued to reduce in this fashion until there were more than 40 executors. Because the gain of the parallelism was no longer sufficient to compensate the cost of communication and the pipeline attained less than 50% efficiency by each executor. A midpoint of 24 executors was chosen to conduct the estimation of time when processing multiple libraries. With 24 executors, mirLibSpark is 184.4 times faster than the miRDeep-P wrapper (Table 3.5).

Figure 3.3 Scalability and number of executors

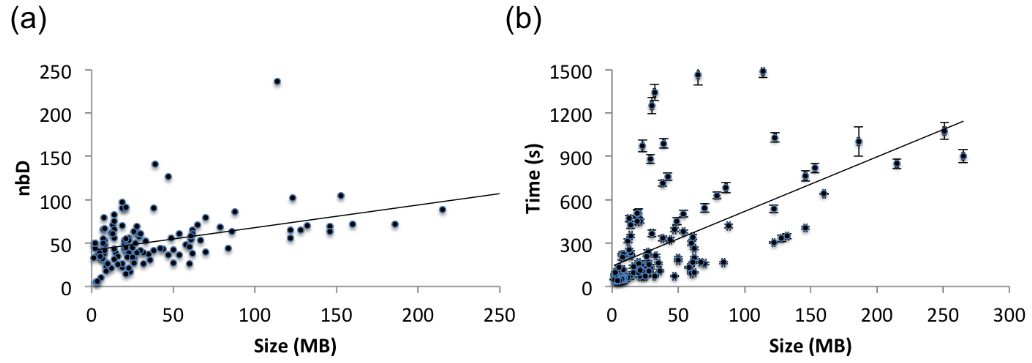


NOTES: nbExecutor: number of executors allocated to mirLibSpark.
compared with sequential.py: relative acceleration, the acceleration folds of mirLibSpark as compared to the time of the sequential version.
compared with mirdeep_p.py: comparative acceleration, the acceleration folds of mirLibSpark as compared to the time of the script that automates the prediction processing of mirDeep-P.
compared with 1-executor: absolute acceleration, the acceleration folds of mirLibSpark as compared to the time of one executor of itself.

3.4.3 Estimate the execution time to process multiple libraries

When submitting a job, we can only allocate server time for a submission based on the input size. The records of processing time for each of the 127 libraries could serve as a reference for future jobs. It took mirLibSpark 323.5 minutes to process 127 libraries using 24 executors. As estimated by the records of processing 127 libraries with 24 executors shown in Figure 3.4, one library of 42 MB on average takes 300 seconds to predict 53 miRNAs.

Figure 3.4 Estimation of the number of predicted miRNA and the execution time by the records of processing 127 libraries. One library of 42 MB on average takes 300 seconds to predict 53 miRNAs. (a) Number of predicted distinct miRNA (nbD) as a function of file size, (b) Execution time as a function of file size.



NOTES: Linear regressions are shown. Their functions are (a) $y = 0.26x + 42$, $R^2 = 0.20$ *(b)* $y = 3.8x + 141$, $R^2 = 0.33$

3.4.4 Case study

A total of 859 miRNAs were predicted from 127 Danforth *ath* sRNA libraries. Seventy among the 100 high confidence miRBase-21 miRNAs and 84 among other 249 validated (low confidence) miRBase-21 miRNAs are present in this data set, as summarized in Table 3.4. The remaining 705 are novel miRNAs. Since mirLibSpark has very low false positive rate, we believe that all the predicted miRNAs are true miRNAs.

Unsupervised clustering method, Expectation Maximization, was applied to regroup the libraries into three clusters based on the miRNA expression binary profile. Along with the supervised tissue class labels, manifold and PCA projection methods were applied to reveal if the classes or clusters were well segregated, as shown in Figure 3.5.With supervised tissue classification, the libraries of the same tissue class do not always locate in the proximity with each other.

Further, the strong discriminating miRNAs for each class or cluster are selected by feature selection method. As shown in Figure 3.6, 27 miRNAs were selected from the predicted 859 miRNAs as they were sufficient to segregate the three clusters of the 127 libraries.

Table 3.4 Comparing the 859 predicted miRNAs with the 349 *ath* distinct miRNAs in miRBase-21.

	miRBase-21	Predicted
nbMiRBaseHigh	100	70
nbMiRBaseLow	249	84
nbNovel	-	705
SUM	349	859

Figure 3.5 Manifold and PCA projections of the 127 libraries for (a) unsupervised clustering and (b) supervised tissue classification. Data points of the same color represent the libraries of the same class or cluster labels. There are 3 clusters by unsupervised clustering and 15 tissue classes by supervised classification.

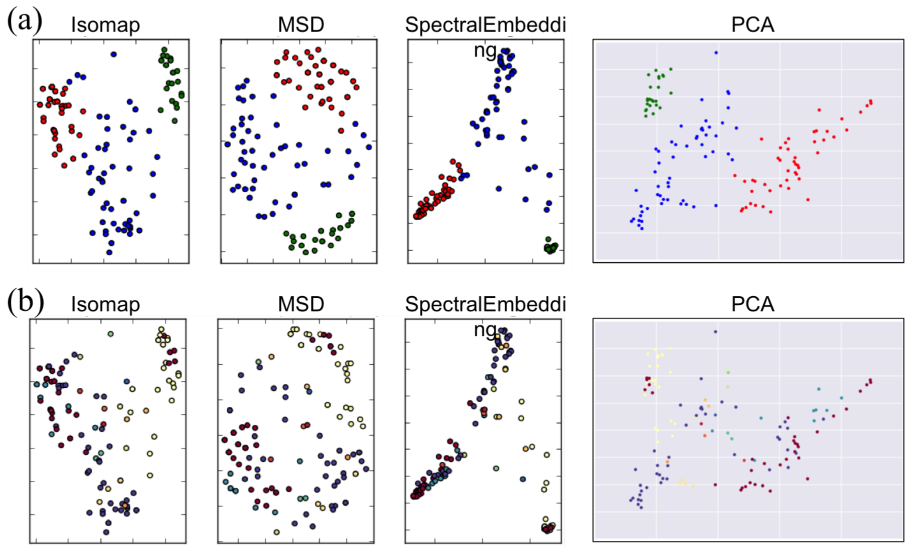
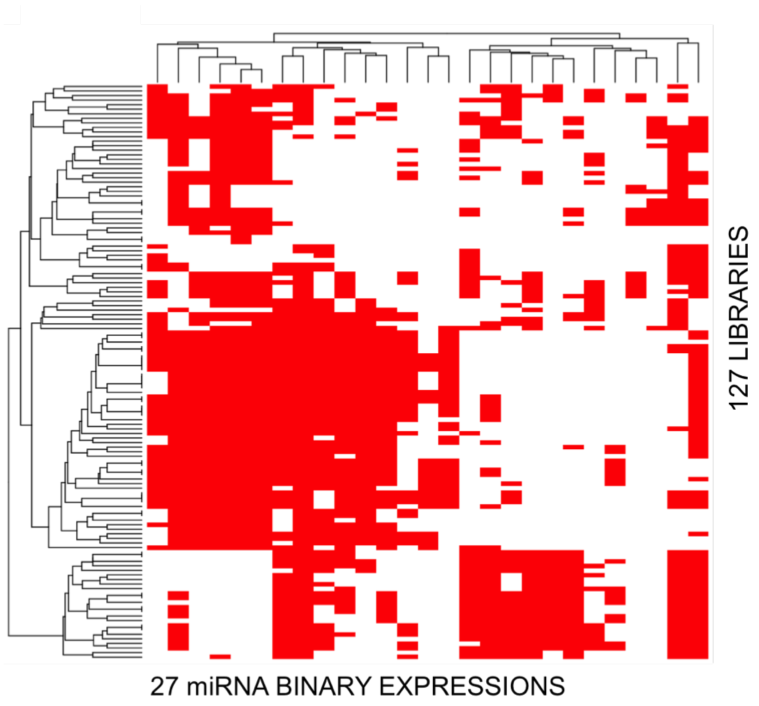


Figure 3.6 Feature selection. Top 27 miRNAs (3%) were chosen out of 859 miRNAs.



NOTES: Euclidean distances are shown in both axes.

3.5 Discussions

The pipeline still has potential to be improved to achieve faster speed, including partitioning RDD elements in relation to their genomic locations. Directed partitioning is critical for the speed because same genomic location can be processed in the same node. Furthermore, we will extend the prediction for more relaxed criteria so that it accepts different input formats including raw and fasta files and takes into account the mutations and isomers for prediction. Eventually, this pipeline will be deployed on *Compute Canada* and the *Wheat miRNA Portal*, and published as an open source program on Github. Current version of this pipeline also predicts the target genes of miRNAs, but this step is not used for benchmarking. We will complete this pipeline with more modules and perform functional analysis including the enrichment of GO terms and pathways. We will conduct a thorough evaluation analysis to make sure that our pipeline does not only run faster but also deliver reliable predictions. This all-in-one service will assist in uncovering in-depth knowledge in the miRNA roles of specific pathways, molecular markers and tolerant phenotypes.

Table 3.5 Scalability and number of executors presented as a table

-	nbExecutor	exec. time (s)	rel speed up	comp speed up	ab speed up	efficiency
sequential.py	-	6360	1	-	-	-
mirdeep_p.py	-	35227	-	1	-	-
mirLibSpark	1	2579	2.5	13.7	1.0	1.00
-	2	1462	4.4	24.1	1.8	0.88
-	3	1297	4.9	27.2	2.0	0.66
-	4	707	9.0	49.8	3.6	0.91
-	8	435	14.6	81.0	5.9	0.74
-	16	250	25.4	140.9	10.3	0.64
-	24	191	33.3	184.4	13.5	0.56
-	32	159	40.0	221.6	16.2	0.51
-	40	139	45.8	253.4	18.5	0.46
-	80	100	63.6	352.3	25.7	0.32
-	160	66	96.4	533.7	38.9	0.24
-	320	62	102.6	568.2	41.5	0.13

NOTES: rel speed up: the acceleration folds of mirLibSpark as compared to the time of sequential.py.
comp speed up: the acceleration folds of mirLibSpark as compared to the time of the script mirdeep_p.py.
ab speed up: the acceleration folds of mirLibSpark as compared to the time of 1-executor of itself.
$efficiency = \frac{abspeedup}{nbExecutor}$

CHAPTER IV

TARGETED PATHWAYS BY COLD-RESPONSIVE MIRNAS IN FIELD WHEAT

4.1 Introduction

The induction of cold acclimation and cold tolerance is a complex process of plants' interaction with the environment. From autumn to early winter, not only the temperature descends but also there are other changes: the hours of sun decrease; the humidity reduces; the wind change its strength and duration; the activities and the numbers of the microbiome in the soil around the plants decline. We know that rapid cellular responses engage the signal transduction mechanisms that activate or inactivate a protein's function by post-translational modifications such as phosphorylation and glycosylation, which do not require the much expensive *de novo* gene expression and protein production. Plants fail in fast environmental changes. Because it takes the plants days to weeks to adapt to the environmental changes, the changes must be steady, and the adaptation must hence involve transcription and translation. On the other hand, studies for plants' cold response are usually conducted in the controlled laboratory chambers, which minimize the thermal fluctuations throughout the day and the season and prevent other factors to occur such as variations of light hours across weeks. Few studies investigate plants' cold response in the open-air field, where the plants perceive all the biotic and abiotic changes in the natural course of seasonal transition.

In our previous study, a significant change was shown in the expression patterns of several miRNAs in response to cold acclimation in both winter and spring wheat cultivars [38]. Several cold responsive miRNAs targeted a set of key cold regulated genes known for their functions in freezing tolerance. These include miRNA target genes in the ICE1–CBF major pathway that regulates freezing tolerance in cold hardy plants.

Despite these efforts, many wheat miRNAs have still not been identified due to the complexity of the wheat genome and the limited experiment designs. To address this limitation, a large field experiment was designed to study how cold acclimation regulates miRNAs and their target genes under natural conditions in the strains of winter wheat, spring wheat and their near isogenic lines that differ in their VRN-A1 expression level and their capacity to develop freezing tolerance.

Using the established miRNA prediction workflow described in the previous chapter, a batch of miRNA candidates was predicted from 80 transcriptomic libraries of wheat field samples collected from autumn to early winter. Some of the candidates can be siRNAs and other false-positive miRNAs that share similar features of miRNAs. SiRNAs share similar gene silencing mechanisms with miRNAs and were retained for targeting analysis. Current evidence could not fully exclude some of the false-positive miRNAs being a true miRNA, and therefore they were also retained. After these candidates of true miRNAs and miRNA-like sRNAs were collected, functional analysis was performed to understand the roles of the miRNAs in the cold tolerance. A functional analysis framework was hence developed in this chapter to validate the identities of the miRNAs, provide cross-annotations from different miRNA-related database, and integrate target gene predictions and statistical functional annotations for the miRNAs, as well as to establish an analytical framework and to interpret the newly discovered knowledge.

4.2 Materials and methods

4.2.1 Wheat sRNA libraries

Four wheat cultivars that differ in their freezing tolerance and vernalization responses (regarding the genotypes of Vrn-A1) including winter habit Norstar (-/-), isogenic Norstar (Iso8S +/+), spring habit Manitou (+/+) and isogenic Manitou (Iso11W -/-) were grown under field conditions in two different years, 2010 and 2013. Crown tissues were sampled at five time points between the beginnings of September and November in Saskatoon and used for small RNA extraction. At least 5 crowns were used for each RNA extraction. Samples were ground to a powder in dry ice and total RNA was extracted using the mirVana^{TM} miRNA Isolation Kit (Life Technologies). RNA quantity and quality were analyzed using Nanodrop spectrophotometer and Agilent 2100 bioanalyzer system, respectively. Small RNA libraries were constructed using TruSeq Small RNA Sample Preparation Kit (Illumina) and the sequencing was performed on Illumina HiSeq 2000 Platform..

4.2.2 Identification and extraction of potential pre-miRNA candidates from sequenced small RNA transcriptomes

The analyses were performed on 730 million to 766 million reads collected respectively from sequencing 80 libraries representing small RNAs of two replicates of five time points from each of the four cultivars grown in two years, 2010 and 2013 (i.e. 2 replicates $\times 5$ timepoints $\times 4$ cultivars $\times 2$ years). Each of the 80 libraries was checked for quality control using FastQC [103]. The adapters were removed using cutadapt 0.9 [46] and the reads were mapped by Bowtie2 [57] on the reference genome. From the mapped small RNAs, 266338 unique small RNAs were extracted, with abundance of 10 raw reads in at least one library and with size between 18 and 25. About 107800 unique small RNAs with a total abundance of at least 100 raw reads were retained for prediction. For each mapped small RNA, four sequences were considered for they could include a potential pre-miRNA candidate as follows: 20 nt before the start and 160 nt after the end of the mapping; 160 nt before the start and 20 nt after the end of the mapping; 200 nt before the start and 500 nt after the end of the mapping; 500 nt before the start and 200 nt after the end of the mapping. The secondary structures of the extracted sequences (pri-miRNA) were folded with RNAfold from ViennaRNA v2.4.1 package [65] with temperature set at 25°C to identify those having a hairpin-like shape. These sequences were submitted to miRCheck [68] to validate an optimized hairpin (pre-miRNA) and repredicted again with miRCheck. The conserved miRNAs were identified by BLASTN for small RNAs against miRBase-22 [77] with a word size 7, maximum e-value 0.1, percentage identity 80, gap open penalty 5, gap extension penalty 4, match score 1, mismatch score -2, and filter low complexity.

4.2.3 Removing false-positive miRNAs

Certain false-positive miRNAs were removed by their annotations as originating from known RNAs such as tRNA, rRNA and snoRNA, from retrotransposons and from other non-annotated repeated sequences in the genome.

4.2.4 Differential expression analysis of miRNA abundances

To quantify and compare sequence abundance across different libraries, raw read counts were normalized using rpm (reads per million). Sequences with read counts lower than 100 in all libraries were removed. Significance level of the difference of small RNA between two libraries (later time points versus time point one in this study) was analyzed using a corrected Z–score method [104].

$$Z = \frac{p_1 - p_2}{\sqrt{\frac{p_0(1-p_0)}{N_1} + \frac{p_0(1-p_0)}{N_2}}} \tag{4.1}$$

where p is for proportion, n for specific tags, N for total tags, $p = n/N$, p_1 for total tag proportion, p_2 for sample tag proportion, $p_0 = \frac{n_1+n_2}{N_1+N_2}$. An adjustment for multiple comparisons based on the false discovery rate (FDR) [105] was performed. Each of the miRNA patterns was classified by comparing its profile at time point 1 with the following four other time points. The small RNAs with fold change lower than 0.5 or higher than 2.0 and FDR $<$ 5% were retained and noted as differential expressed (DF) sRNAs that were down-regulated (DOWN) or up-regulated (UP), respectively. i.e.: for one miRNA in one strain at one time point,

Let differential expression DF be T.

$$T = \begin{cases} UP, & \text{if fold change} \geq 2.0 \text{ and FDR} < 5\%, \quad (4.2) \\ DOWN, & \text{if fold change} \leq 0.5 \text{ and FDR} < 5\%, \quad (4.3) \\ No, & \text{otherwise} \quad (4.4) \end{cases}$$

4.2.5 Definition of indices and the regulation of miRNAs

Definition of W index, W for average:

W index is calculated by averaging the differential expression up (1), down (-1) or no change (0) in one strain throughout the total 8 different time point comparisons in years 2010 and 2013. i.e.: for each sRNA candidate, Let strains be $S = \{Norstar, Iso8S, Manitou, Iso11W\}$.

For $i \in S$, let time points be $T_i = \{i_{210}, i_{310}, i_{410}, i_{510}, i_{213}, i_{313}, i_{413}, i_{513}\}$.

Let the size of T_i be $\#T_i$.

For $j \in T_i$, then $T_{ij} = DF$.

Let the differential expression of an individual miRNA be D_{ij} ,

$$D_{ij} = \begin{cases} 1, & \text{if } T_{ij} = \text{UP}, \quad (4.5) \\ -1, & \text{if } T_{ij} = \text{DOWN}, \quad (4.6) \\ 0, & \text{otherwise} \quad (4.7) \end{cases}$$

Let the average regulation index of an individual miRNA be W_i ,

$$W_i = \frac{\sum\limits_{j \in T_i} D_{ij}}{\#T_i} \tag{4.8}$$

Definition of R index, R for regulation:

The regulation of miNRAs, as recorded in Table 4.6, is determined by R index. If a sRNA is strongly differential expressed ($|W_i| > 0.3$) in any strain in the same fashion, then the decision for the regulation is either up- or down-regulation. But if it is strongly expressed but not in the same fashion among strains, then the decision is UP/DOWN. Otherwise, the decision for the regulation is No. i.e.
$|W_i|$ is the absolute value of W_i.

Let sRNA candidates be $K = \{New_{001}, New_{002}, ...\}$.

For $n \in K$, let the regulation conclusion of a sRNA be R_n .

$$R_n = \begin{cases} UP, & \text{for all } i \in S,\ W_i \geq 0 \text{ and any } \overline{W_i} > 0.3\ , \quad (4.9) \\ DOWN, & \text{for all } i \in S,\ W_i \leq 0 \text{ and any } \overline{W_i} < -0.3, \quad (4.10) \\ No, & \text{for all } i \in S,\ \left|\overline{W_i}\right| \leq 0.3 \quad (4.11) \\ UP/DOWN, & \text{otherwise} \quad (4.12) \end{cases}$$

Definition of absolute difference (AB) of W indices:

For an individual miRNA, W is calculated for each strain. For a, $b \in S$, $a \neq b$, an absolute difference of W_a and W_b is:

$$AB_{a,b} = |W_a - W_b| \tag{4.13}$$

For example, for miRNA Newxxx, its expression in terms of W index, say, in strain Iso8S is $W_{Iso8S} = -0.9$, and in strain Norstar is $W_{Norstar} = -0.2$, then for this miRNA, absolute difference $AB_{Iso8S,Norstar} = |W_{Iso8S} - W_{Norstar}| = 0.7$. Since $W_{Iso8S} < W_{Norstar}$, Newxxx would be recorded in the *Down* column in Table 4.4 to indicate that this miRNA is expressed less in Iso8S than in Norstar.

4.2.6 GO term and KEGG pathway annotation

Unique sequences in wheat field transcriptomes longer than 250 nt were collected and serial logged as XLOCs (ex. XLOC_000001). Based on sequence similarity scoring, each sequence was assigned an Uniprot best match id from the reservoirs of all species, from animals to plants and viruses. With the aid of *bioservices* in *python* [106], each XLOC was annotated with KEGG entry ids (also known as

K numbers) based on Uniprot ids. Genes that are conserved across species with preserved functions are called orthologs. Some genes work together in a well-defined biological or biochemical process that is termed as pathway. To facilitate the annotation of KEGG pathway that requires the genes of a pathway to be in the same species, the K numbers were converted to a reference id system, KEGG Orthology entry ids (KO), that unifies the orthologs in the same reference species. The KEGG pathway of each XLOC was retrieved based on KO ids.

4.2.7 MiRNA targets identification and functional enrichments

MiRNA target genes were identified using psRNATarget program [99] and TGACv1 reference genome. At the time of this work was conducted, the most recent reference genome IWGSC RefSeq V1.0 that represents 90% of the wheat genome was not yet published [80]. The version used as the target gene reference in this study, TGACv1, represents 78% of the wheat genome [76]. PsRNATarget was chosen because it had a major update recently and provided TGACv1 in their choices of reference genome [107].

Targets with the top 5 scores were retained. Their annotations in Gene Ontology (GO) and KEGG pathways were retrieved [82]. The GO and KEGG pathway enrichments were performed using the standard hypergeometric test for the targets of the differentially expressed miRNAs. The wheat genome background was constructed by the overall annotation frequencies of GO terms and KEGG pathways covered by the complete protein repertoire of the reference genome. For functional enrichment computation, the backgrounds were the specific subsets of the genomic background regarding precursor XLOC, top 1 target and top5 scored targets related to the 220 sRNAs in this study.

The chance of all the drawing combinations of the genes to obtain the sample frequency based on the background frequency is the p-value. The GO terms and KEGG pathways with p-value < 0.05 are considered as enriched and is calculated based on the hypergeometric test formula of the upper cumulative distribution Q as follows.

$$p = Q\left(k, n, M, N\right) = \sum_{t=k}^{M} f\left(t, n, M, N\right) \tag{4.14}$$

where M is for background population; N for sample population; n for total objects of type of interest in M; and k for sampled objects of type of interest in N. Background frequency $\frac{n}{M}$ is the number of genes annotated to a functional term in the entire background set, while sample frequency $\frac{k}{N}$ is the number of genes annotated to that functional term in the input list. For example, if the input list contains 10 genes and the enrichment is done for biological process containing 6000 genes, then if 5 out of the 10 input genes are annotated to the functional term: DNA replication, then the sample frequency for DNA replication will be $\frac{5}{10}$. Whereas if there are 100 genes annotated to DNA replication in all of the genome, then the background frequency will be $\frac{100}{6000}$.

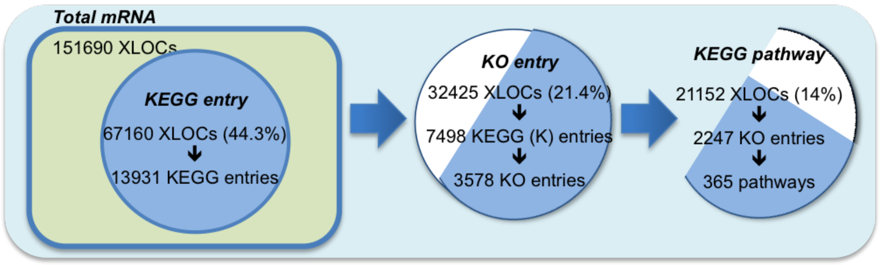
4.3 Results and discussions

4.3.1 KEGG pathway annotated 14% of the field wheat long RNAome

From the wheat field data, 151690 unique XLOC sequences longer than 250 nt were collected. These XLOCs were annotated with Uniprot. These 671690 XLOCs were annotated with KEGG entry ids (K numbers) based on Uniprot ids. There are 13931 unique K numbers in the wheat field data. To unify the orthologs, 32425 XLOCs were converted to 3578 unique KOs. There were

2247 KO entries working in 365 pathways. Because not all levels of the annotations were available for each of the genes, information attrition occurred along with each step of database-corresponding identification conversion. Only 14% of the initial 151690 XLOCs were annotated with KEGG pathway information for the final pathway analysis, as opposed to 63% for GO terms (Supplementary *field_ homolog_ annotations_ addPathways_ formatted.txt*). The KEGG pathway recovery rate of wheat was similar in other plant species, as shown in Table 4.1. The KEGG pathway annotation workflow and the number of annotation in each step is illustrated in Figure 4.1.

Figure 4.1 KEGG pathway annotation workflow



The collection of unique long transcripts from the 80 wheat field libraries contains 151690 XLOCs. Based on the best matching scores, the XLOCx were annotated by Unirprot ids from all species, including plants, animals, and others. Using Uniprot, 44.3% of the XLOCs were converted to KEGG ids (K numbers). Using K numbers, 21.4% of the XLOCS were further unified to a reference species by KEGG Ontology entry ids (KO ids). Using KO ids, 14% of the XLOCs found their KEGG pathway information and comprised 365 pathways.

Table 4.1 The coverage rates of KEGG pathway annotations in selected plant genomes available in KEGG ranges from 13% to 25%. The average is 17.90% among 67 plant genomes.

species	nbPathways	nbGenes having pathway Info	nbGenes in genome	percentage genome having pathway info(%)
ath	134	4890	32657	14.97
aly	134	5393	32533	16.58
bna	134	16442	101183	16.25
gra	132	6836	41605	16.43
gmx	134	9005	53327	16.89
var	133	5204	27843	18.69
fve	133	4548	26238	17.33
pxb	133	6977	38415	18.16
cmo	133	4124	21709	19.00
spen	132	5152	28367	18.16
osa	134	4450	31126	14.30
obr	133	4486	23100	19.42
ats	132	5336	33972	15.71
smo	131	6546	34814	18.80
vcn	121	1962	14434	13.59
ota	111	1403	8062	17.40
cme	116	1424	5616	25.36

There are in total 70 plant species in KEGG database, in which wheat tae is not curated.
KEGG data of three plant species are problematic and ignored for the calculation of plant average.
percentage = (nbGenes having pathway info) / (nbGenes in genome)

4.3.2 Wheat has 365 KEGG pathway

Wheat KEGG pathway is not curated in the KEGG database because researches in this area for wheat are in their infancy. In this wheat field study, the sum of the 80 transcriptomes represented 365 KEGG pathways. A detail list and description of the wheat pathways is included in Appendix B.4. This number of wheat pathways is probably saturated because in general, plant species have 130 pathways, microbians have 105 pathways, and animals have 300 pathways (Table 4.2). Therefore, we have likely uncovered all the known and potential wheat pathways in this study. The high pathway counts could be explained by the complexity of its genome and its highly developed status in evolution, diverse and highly specified pathways may accumulate in wheat. On the other hand, because wheat genes in this study were annotated using all species, including animals, inevitably the pathways uncovered also contained pathways that are seemed to be specific to animals, such as the pathway *ko05410: Hypertrophic cardiomyopathy*. Although the cutoff percentage to exclude the pathways unlikely belonging to wheat is still not clear, the coverage of required KOs for each of the pathways by the observed KOs tends to be smaller, as shown in Table 4.3 and Appendix B.4. For example, ko00073, the *cutin, suberine and wax biosynthesis* pathway that was firstly identified in *Arabidopsis* [108] has a coverage rate of 83%, whereas ko05410, a pathway working in the heart muscles, has a coverage rate of only 4%. But ko05231, a pathway in cancer, has a coverage rate of 23%. That is higher than many legitimate plant pathways. Further optimization may be performed based on species range and coverage rate to improve the quality of pathway prediction.

Table 4.2 Number of KEGG pathways in selected plant, animal and microbian species

abbreviation	ORGANISM	CATAGORY	nb of KEGG pathway
ath	Arabidopsis thaliana	plant	132
osa	Oryza sativa japonica (Japanese rice)	plant	128
zma	Zea mays (maize).	plant	132
lsl	Lactobacillus salivarius	microbian	97
chu	Cytophaga hutchinsonii	microbian	109
bay	Bacillus velezensis	microbian	115
spo	Schizosaccharomyces pombe (fission yeast)	microbian	112
ddi	Dictyostelium discoideum (cellular slime mold)	animal/microbian	104
mmu	Mus musculus (mouse)	animal	301
rno	Rattus norvegicus (rat)	animal	301
xla	Xenopus laevis (African clawed frog)	animal	157

Table 4.3 Coverage of required KO components to execute selected wheat pathways in the field transcriptomes. Coverage rates range from 90% to 1%. Coverage rates of pathways irrelevant to plants are essentially lower, but there is no clear cutoff percentage to exclude pathways.

pathway	nbXLOC	nbObserved	nbRequired	coverage
ko00010 Glycolysis / Gluconeogenesis	398	31	99	31.31%
ko00040 Pentose and glucuronate interconversions	116	9	66	13.64%
ko00051 Fructose and mannose metabolism	224	19	106	17.92%
ko00052 Galactose metabolism	238	16	75	21.33%
ko00053 Ascorbate and aldarate metabolism	151	15	45	33.33%
ko00062 Fatty acid elongation	206	7	23	30.43%
ko00072 Synthesis and degradation of ketone bodies	28	3	8	37.50%
ko00073 Cutin, suberine and wax biosynthesis	268	10	12	83.33%
ko00100 Steroid biosynthesis	137	17	30	56.67%
ko00254 Aflatoxin biosynthesis	19	1	13	7.69%
ko00281 Geraniol degradation	7	1	16	6.25%
ko05231 Choline metabolism in cancer	238	14	61	22.95%
ko05322 Systemic lupus erythematosus	142	8	45	17.78%
ko05323 Rheumatoid arthritis	72	12	65	18.46%
ko05340 Primary immunodeficiency	3	1	35	2.86%
ko05410 Hypertrophic cardiomyopathy	24	3	73	4.11%
ko05414 Dilated cardiomyopathy	2	1	82	1.22%
ko05416 Viral myocarditis	43	3	38	7.89%

nbXLOC: number of XLOCs annotated with this pathway. nbObserved: number of the KOs among the required one observed in a pathway. nbRequired: number of the KOs required to execute a pathway.
$Coverage = nbObserved/nbRequired$

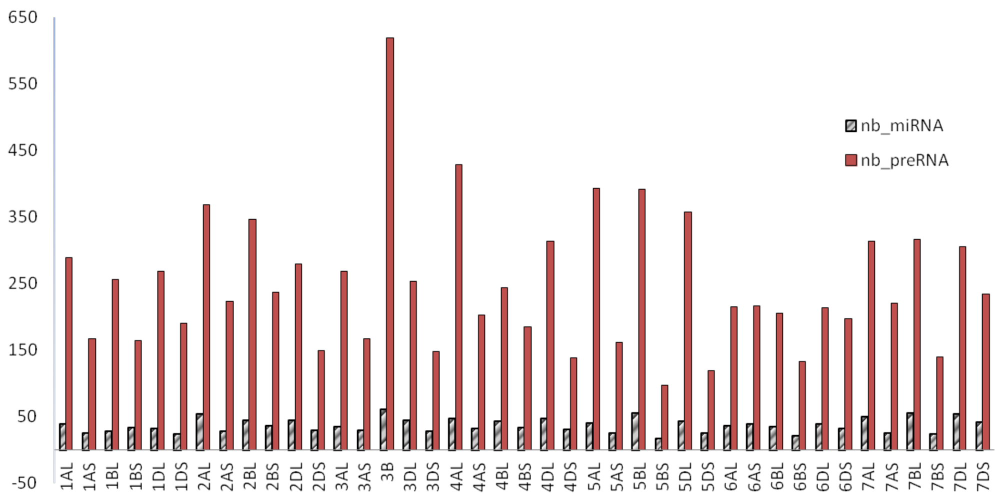
4.3.3 Discovery of wheat miRNAs expressed during cold acclimation in the field

There were 418 unique small RNAs ranging from 18 to 25 nt in the field transcriptomes. The chromosomal distribution of the 418 miRNA candidates and their miRCheck-validated precursors is shown in Figure 4.2 (also see details in Supplementary *all418report_v11.1.tsv*).

Based on their localization within the genome or XLOC transcripts and their presenting features of miRNAs, 220 among the 418 were included for further analysis and annotated with genomic loci, miRBase families and target gene functions, as shown in Table 4.6. Based on the validation and characteristics of their precursor foldings, the 220 small RNAs were further classified into 3 categories: 134 miRNAs, 60 siRNAs and 26 false RNAs. There were insufficient evidence to conclude the class of the false RNAs being miRNA, siRNA or non-functional RNA. Based on their sequence conservation (matching with miRBase miRNAs with less than 2 mismatches), miRNAs were further categorized as 58 conserved/known or 76 non-conserved/novel. Novel miRNAs could be wheat-specific, monocot-specific or new that response to certain conditions.

On the other hand, one candidate may have more than one validated precursor. First, because the A, B and D genomes have high degree of similarities, same miRNA candidate may appear on the same chromosomes in some or all of the three sub-genomes. Second, a candidate may originate from distinct precursor sequences that share low similarities. Further analysis of the miRNAs during evolution regarding the acquisition of the sub-genomes into modern wheat will help us to understand the origins of the functional miRNAs.

Figure 4.2 Chromosomal distribution of 418 miRNA candidates

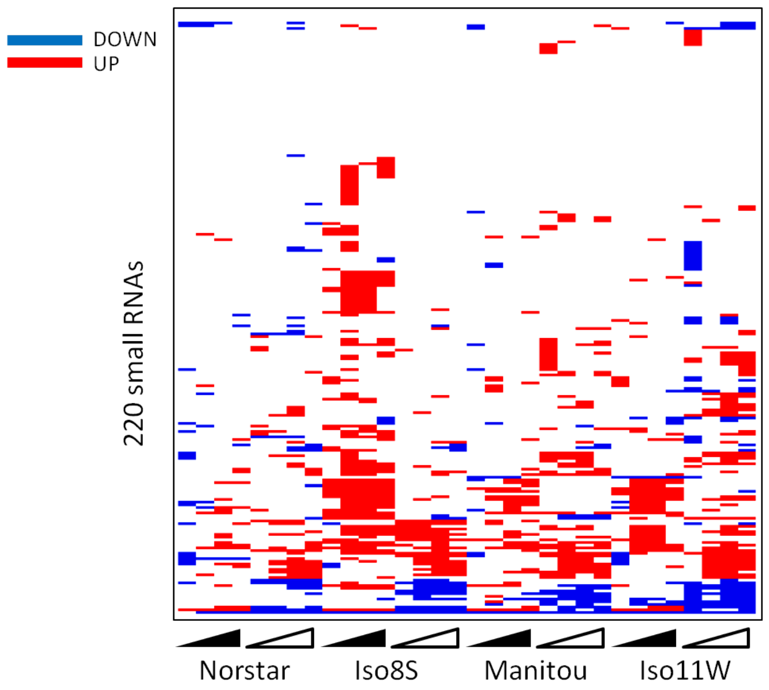


There are 7 chromosomes in each of the A, B and D genomes. The chromosomes are further divided into L and S, each standing for long arm and short arm portions, respectively, of the chromosomes regarding the centromere loci. Mitochondria and choloroplast genomes were included in the analysis but few miRNAs were validated from these regions.

4.3.4 Differential expressed field wheat sRNAs responsive to cold acclimation

During the time course of cold acclimation, there were 32 comparisons for small RNA differential expressions among strains. Based on Formulas 4.2 to 4.4 in Section 4.2.4, statistical significantly up-regulated sRNA expression was noted as UP and color red, and significantly down-regulated sRNA expression was noted as DOWN and color blue, shown in Figure 4.3.

Figure 4.3 Time course of 32 differential expression DF profiles for the 220 cold responsive sRNAs



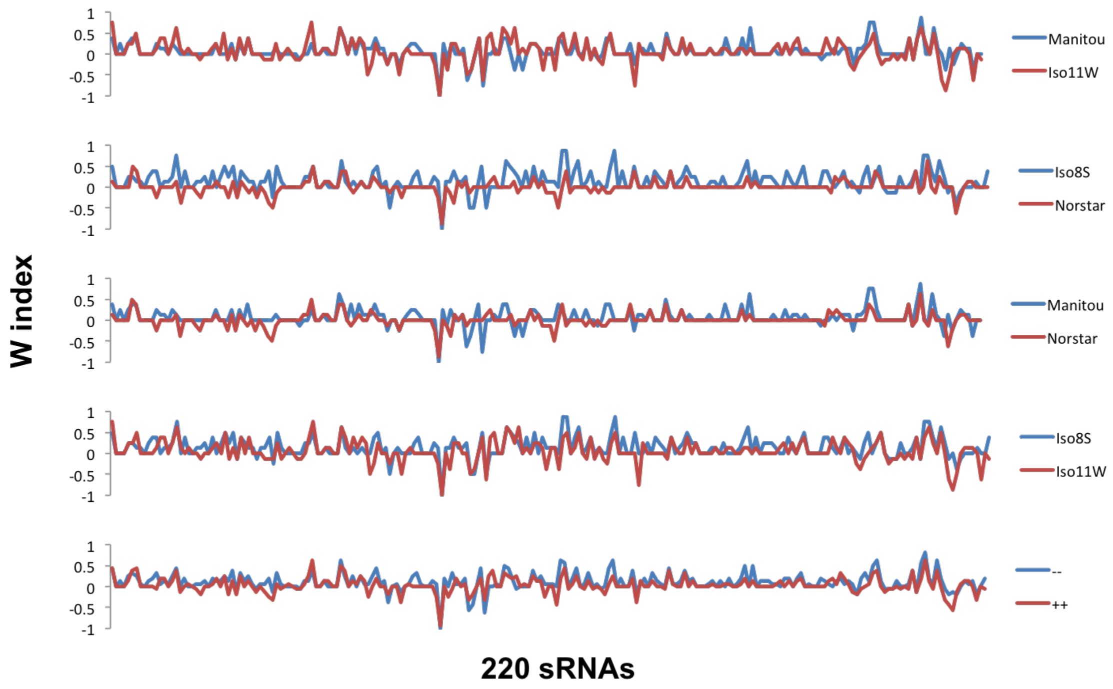
Each triangle indicates the comparisons of four time points from autumn to winter. Solid triangle are for year 2010 and hollow one are for year 2013.

In order to conclude the regulation of each sRNA during cold acclimation, the regulation index R of each sRNA in the four strains at all 8 time points (Formula 4.9 to 4.12 in Section 4.2.5) was calculated. Based on R, the conclusion of each sRNA's regulation, up- or down-regulated, during cold acclimation in the field is recorded in Table 4.6. Among the 220 sRNAs expressed in the field wheat during acclimation and analyzed in this study, 79 sRNAs showed differential expression in response to cold: 50 were up-regulated: 13 were down-regulated, and 16 were depending on the stains. Moreover, fifty-two of the 79 sRNAs were classified as miRNAs. Among which, 17 have been shown in previous studies to be related to cold tolerance, a strong support for our experimental design and methodology.

4.3.5 Contribution of genetic backgrounds and Vrn-A1 genotypes to cold responsive sRNAs

To understand the effect of genetic backgrounds to the sRNA expression during cold acclimation, an average index, W, was calculated for each of the four strains by average the sum of the time course with counting UP as 1, DOWN as -1, no change as 0 (Formula 4.8 in Section 4.2.5). The average trends of the 220 sRNAs were compared with its relevant genetic counterpart strains. In the illustrations in Figure 4.4, when comparing two related strains, the W index of most of the sRNAs were similar. And it was clear to see that certain sRNAs were more up- or down-regulated in one strain than the other.

Figure 4.4 Average trend of up or down regulation of the 220 cold responsive sRNAs in different genetic backgrounds. Comparisons of $W_{Manitou}$ vs W_{Iso11W}, W_{Iso8S} vs $W_{Norstar}$, $W_{Manitou}$ vs $W_{Norstar}$, W_{Iso8S} vs W_{Iso11W}, $W_{vrn-A1}(-/-)$ vs $W_{Vrn-A1}(+/+)$ are shown.



Blue and red lines are for vrn-A1 and Vrn-A1 genotypes, respectively. X-axis is the 220 sRNAs. Y-axis is the average index W. W greater than 0 indicates the trend of this sRNA to be up-regulated, smaller than 0 the trend to be down-regulated during the acclimation time course.
$W_{vrn-A1}(-/-)$ and $W_{Vrn-A1}(+/+)$ are the average of the W indices in strains with vrn-A1 or Vrn-A1 genotype, respectively

To select for the comparatively up- or down-regulated sRNAs regarding genetic backgrounds, the absolute difference of W of the same sRNA between two strains ($|W_{s1} - W_{s2}| \geq 0.5$, Formula 4.13 in Section 4.2.5), was calculated from the illustration of Figure 4.4. The selection results are shown in Table 4.4. For example, when the spring Manitou vrn-A1 (-/-) was replaced by Vrn-A1 (Iso11W +/+), the expressions of several small RNAs changed. Some in Iso11W were strongly up-regulated as compared to Manitou, such as New103. Some were strongly down-regulated, such as New212. When the winter Norstar Vrn-A1 (+/+) was replaced by vrn-A1 (Iso8S -/-), the expressions of several small RNAs also changed. Some in Norstar were strongly up-regulated as compared to Iso8S, such as New095. Some were strongly down-regulated, such as New114. The genotype of Vrn-A1 (+/+) in average was correlated with the up-regulation of New097, while vrn-A1 (-/-) was correlated with the down-regulation of New212, regardless the genetic background. The complete list of the comparatively regulated sRNAs among strains and genotypes is shown in Table 4.4

In summary, it is revealed that during cold acclimation different sRNAs are regulated according to its genetic context. Those known and novel miRNAs that are not involved in cold response may be related to the growth and development of wheat, and contribute to the overall fitness of the plant in cold acclimation. Several of the detected miRNAs were reported in wheat cold response previously and agreed with our observations. For example, New001 and New114 belong to miRNA families miR9670 and miR9674, respectively, and behaved the same way as reported in other studies (comparing Table 1.3 and Table 4.6). On the contrary, New020 (miR5048), New100 (miR393), New101 (miR393), New117 (miR9674, miR5064) and New118 (miR9674) were up-regulated in this study but were reported to be down-regulated in cold acclimation. And New113 (miR9672) was the other way around. Interestingly, New110 (miR827), New121 (miR9674) and New086 (miR319,miR159) partially agreed with the literature because their expressions were dependent to the genetic backgrounds they were presented. Other miRNAs that belong to miR319 included New084, New089 and New090 were also not regulated. Therefore, it implies that sophisticated regulations are required for miR319 and might involve the genomic origins and loci of these miR319 species. Similarly, New080 (miR444), New107 (miR397) and New116 (miR9674) were reported to be regulated during cold acclimation but were static in this study. These above observations emphasize the importance of experimental design. Although these miRNAs are sensitive to the environmental changes, their behaviors in the open-air field are affected by more than just temperature factor.

Table 4.4 Cold responsive sRNAs regarding Vrn-A1 in different genotype backgrounds, with the absolute difference of W indices greater than or equal to 0.5.

Genotype	Up	Down
Iso11W(+/+) / Manitou (-/-)	New103, New097, New051, New105, New085, New017	New212, New065, New210, New162, New211, New133, New192
Norstar (+/+) / Iso8S(-/-)	New095, New092	New114, New127, New040, New204, New087, New126, New012, New042, New118, New100, New017, New113, New190, New030, New094, New121, New101, New174, New115, New085
Norstar (+/+) / Manitou (-/-)	New095, New091	New212, New087, New126, New012, New113, New162, New193, New041, New061, New035
IsoW11W (+/+) / Iso8S (-/-)	New097	New211, New133, New218, New066, New002, New210, New190, New160, New132, New114, New110, New086, New073, New065, New040
Vrn-A1(+/+) / vrn-A1 (-/-) *	(New097, New095, New103)	(New212, New113, New114, New040, New087, New126, New012, New162, New127, New204)

**: None of the difference of W between Vrn-A1(+/+) and vrn-A1 (-/-) is more than* 0.5

For miRNA Newxxx, its expression in terms of W index, say, in strain Iso8S is $W_{Iso8S} = -0.9$, and in strain Norstar is $W_{Norstar} = -0.2$, then for this miRNA, absolute difference $AB_{Iso8S,Norstar} = |W_{Iso8S} - W_{Norstar}| = 0.7$. Since $W_{Iso8S} < W_{Norstar}$, Newxxx would be recorded in the Down column to indicate that this miRNA is expressed less in Iso8S than in Norstar.

4.3.6 Enriched GO terms and KEGG pathways targeted by differentially expressed sRNAs in cold acclimation

The GO term and KEGG pathway enrichment analysis was performed on three types of data: the precursor XLOCs, the top 1 target genes, and the top 5 scored target genes. When performing enrichment analysis, several GO term categories were used, including overall GO, GOBP, GOCC and GOMF, in addition to KEGG pathways. Therefore, a total of 15 enrichment analyses were conducted (details in Supplementary Chapter_wheat/Enrichment_analysis/output). Only the 6 enrichment analyses that contained significantly enriched functions were presented in Table 4.5 for the statistics of the functional enrichment results. It showed that the annotation rate of the sRNAs by the functions of their top 5 scored target genes was as high as 90% for GOBP terms, and the annotation rate by KEGG pathway was 49% despite the genomic annotation was only 14%. The KEGG pathway enrichment was accessed for the reduction of false positive predictions. Only 108 out of 220 sRNAs were able to be annotated with KEGG pathway by their top 5 scored target genes (Table 4.6). Among these 108 sRNAs, only 66 sRNAs could be considered as true positive predictions for their contributions to the enriched pathway targeting in the cold response mechanisms. However, this reduction method needs to be mathematically formalized and adjusted for better functional classification performance.

Table 4.5 Statistics of functional enrichment by annotation methods

Annotation method	nb sRNA annotated	Annotation rate of 220 sRNAs	nb annotated functions	nb enriched functions *	nb contributing sRNAs
PreXLOC GO	29	13,2%	29	5	TBD
top1TG GO	144	65,5%	255	34	TBD
top5scored GOBP	200	90,9%	624	56	TBD
top5scored GOCC	207	94,1%	182	36	TBD
top5scored GOMF	209	95,0%	321	40	TBD
top5scored KEGG pathway	108	49,1%	130	20	66

** : The sets of analysis containing significant enriched terms were plotted in the following paragraphs. TBD: to be determined.*

The distribution of the enriched functions targeted by the 79 differential expressed sRNAs in the field wheat libraries are shown in the following 6 tables. The numbers in the cells indicated the occurrence of this functional term at the particular time point in a strain. The dark coloration of the cell indicated that such occurrence conferred to a statistical significantly enriched frequency as compared to its proper background. The conclusions of each analysis are described as follows.

Enriched GO terms by precursor XLOCs include *hydrolase activity* and *ADP binding* throughout the time course of acclimation in all strains, as shown in Figure 4.5.

Enriched GO terms by top 1 scored target gene include *vacuolar membrane*, *transcription*, *development*, *response to abscisic acid* and *protein dimerization*, as shown in Figure 4.6.

Enriched GO terms by top 5 scored target genes include *cold acclimation*, *response to salt stress*, *cellulose biosynthetic process*, *mitochondrial membrane*, *apoplast*, *protein serine/threonine kinase activity*, *heme binding*, *protein dimerization* and *sequence-specific DNA-binding*, as shown in Figures 4.7, 4.8 and 4.9 for GOBP, GOCC and GOMF, respectively.

Enriched KEGG pathways by top 5 scored target genes include *photosynthesis*, *tryptophan metabolism*, *carotenoid biosynthesis*, *spliceosome*, *endocytosis*, and *longevity*, as shown in Figure 4.10.

4.3.7 Classes of enriched biological pathways regulated by cold acclimation-responsive miRNAs in field wheat

The enriched GO terms and KEGG pathways by the target genes of differentially expressed miRNA (referring to the 220 miRNA-like sRNAs) and their precursors can be categorized into the following classes.

Transcription. Several transcription related GO terms and a pathway were enriched in the wheat miRNA targets, such as transcription, protein dimerization, sequence-specific DNA-binding and spliceosome pathway.

Post-translational modification (PTM). Hydrolase activity, ADP binding and protein serine/threonine kinase activity are related to PTMs that modulate the activities and functions of other proteins.

Membrane. Regulations of the membrane composition and stability include vacuolar membrane, mitochondrial membrane, apoplast, cellulose biosynthetic process (cell wall) and endocytosis pathway.

Metabolism. Three KEGG metabolism pathways are enriched, including tryptophan metabolism and carotenoid biosynthesis. Tryptophan is a precursor of auxin [109], a plant hormone important for mediating abiotic stress response.

Carotenoid [110] is a product of the synthesis pathway of abscisic acid [111], a plant hormone important for plant dormancy in response to environmental stresses, including cold stress.

Stress. Several stress-related terms are also enriched, including responses to abscisic acid, salt stress, UV-B, hot, and cold acclimation. It is known that the heme binding activity is also part of the stress responsive activity [112].

Organismal systems. Development under the control of growth hormones [113] and photosynthesis [114] are known as general stress-responsive strategies. Staying alive by longevity pathway [115] is also regulated under cold stress.

Table 4.6 Descriptions of predicted 220 sRNAs from wheat field transcriptomes. Panel 1.1 Conserved miRNA, Panel 1.2 Non-conserved miRNA, Panel 2. siRNA, Panel 3. False RNA. ID and RNA sequence are the 220 small RNAs discovered in wheat in this study. MiRBase best hit is the best matching family in miRBase-22 with less or equal to 2 mismatches; Pre-miRNA genomic loci are the origins of validated precursors. Exact positions and sequences in Supplementary. Precursor functions are the functions of the corresponding XLOC Uniprot annotations. Target gene (with psRNATarget expectation) is the corresponding XLOC ID of the TGACv1 accession number that has meaningful Uniprot annotations. Target gene functions are the Uniprot annotations of the target gene in the previous column. Regulation is the cold acclimation response concluded in this study from 80 field wheat libraries of strains and conditions. Literature is a quick reference to Table 1.3

1.1 Conserved miRNA

ID	RNA sequence	miRBase best hit	Pre-miRNA genomic loci	Precursor functions	Target gene (Expectation)	Target gene functions	Regulation	Literature
New001	AGGTGGAATACTTGAAGAAGA	miR9670	6AS,6DS	unknown	XLOC_128592 (0.5)	Transcription termination factor MTERF8,chloroplastic	UP	up
New005	TTGAACATCCCAGAGCCACCG	miR9662	6AS,6DS	unknown	XLOC_115987 (0.5)	Transcription termination factor MTERF15,mitochondrial	No	
New020	TTGCAGGTTTGAGGTCTAAGT	miR5048	4AL,4BS,4DS,7AL,7BL,7DL	unknown	XLOC_071855 (1.0)	Serine/threonine-protein kinase mos	UP	down
New036	TCAAGCATCATATCATGGACA	miR5071	1AL,1BS,2BL,3AL,3B,3DL,3DS,4AS,4BL,5AS,5BL,5BS,5DS,6AS,6BL,6BS,6DL,6DS,7BL	NBS-LRR protein	XLOC_119887 (0.0)	Disease resistance proteins RPM	No	
New037	TCAAGCATCATGTCATGGACA	miR5071	1AL,1AS,1BL,1BS,1DL,2BS,3AL,3B,3DL,4AL,5BL,5DL,6AL,6AS,6BL,6DL,6DS,7AS,7BL,7DS	unknown	XLOC_015670 (0.0)	Disease resistance protein RPM1	No	
New039	ACAAAACCTTCAGCTATCCATC	miR7757	2DS,3AL,3B,3DL,7DL	unknown	XLOC_057563 (1.5)	Putative late blight resistance protein homolog R1B-14	No	
New050	TCGGACCAGGCTTCAATCCCT	miR165,miR5168,miR166	5AL,5BL,5DL,6AL,6BL,6DL	unknown	unknown	unknown	UP	
New051	GGGTTGTTGTCTGGTTCAAGG	miR5168	2DS,5AL,5BL,5DL	unknown	unknown	unknown	UP	
New052	TCGGACCAGGCTTCATTCCCC	miR5168,miR165,miR166	1AL,1BL,1DL,4AS,4BL,4DL,5BL,5DL,6AL,6BL,6DL,7AL,7BL,7DL	unknown	XLOC_110868 (0.5)	Homeobox-leucine zipper protein HOX33	No	
New053	TCGGACCAGGCTCCATTCCCC	miR165,miR166	1AL,1BL,1DL,4AS,4BL,4DL,5BL,5DL,7AL,7BL,7DL	unknown	unknown	unknown	No	
New054	TCGGACCAGGCTTCACTCCCC	miR5168,miR165,miR166	1AL,1BL,1DL,2BL,4AS,4BL,4DL,5BL,5DL,7AL,7BL,7DL	unknown	unknown	unknown	No	
New055	GGAATGTTGTCTGGCTCGGGG	miR166	7AL,7BL,7DL	unknown	XLOC_113807 (0.5)	Putative ribonuclease H protein At1g65750	No	
New056	TCGGACCAGGCTTCATTTCCC	miR166	1AL,1BL,1DL,4AS,4BL,4DL,5BL,5DL,6AL,6BL,6DL,7AL,7BL,7DL	unknown	XLOC_110868 (1.0)	Homeobox-leucine zipper protein HOX33	No	
New058	GGAATGTTGTCTGGTTCAAGG	miR165,miR166	1AL,1BL,1DL	unknown	unknown	unknown	UP	
New062	TACTGTGGGCACTTATTTGACA	miR9669	1AS,1DL,2AL,2BL,2DL,2DS,3AS,3B,3DL,3DS,4AL,4BL,4BS,4DL,5DL,6DL,6DS,7AS,7BS,7DL	unknown	unknown	unknown	No	
New067	TGCATCATCTCGAACTCGTCG	miR9778	1AS,2DL,7AL,7DL	unknown	XLOC_004785 (0.5)	Disease resistance protein RPP13	UP	
New071	TGAGAAGGTAGATCATAATAGC	miR9863	1BS,1DL,1DS,2AL,3AL,3B,3DL,4AL,5BL,5DL,6AS,6DS,7AS,7BL,7DS	CNL6	XLOC_057515 (0.5)	Disease resistance protein RPM1	No	
New072	TGAGAAGGCAGATCATAATAGC	miR9863	1AS,1BS,1DS,2AL,3DL,4AL,6DS,7AS	CNL6	unknown	unknown	No	

New080	TGTTGTCTCAAGCTTGCTGCC	miR444	2AL,2BL,2DL	MADS-box transcription factor 27	XLOC_020605 (0.0)	MADS-box transcription factor 27	No	down
New082	ATCAGGAGAGATGACACCGAC	miR1432	1AL,1BL,1DL,2AS,2BS,2DS,4DL,5AL,5BL,5DL	Probable calcium-binding protein CML21	XLOC_102168 (0.0)	Probable calcium-binding protein CML21	No	
New083	TGACAGAAGAGAGTGAGCAC	miR157,miR156	2AL,2BL,2BS,2DL,3AS,3B,3DS,4AS,4DL,5AL,5BL,5DL,6AS,6BS,6DL,6DS,7AL,7AS,7BS,7DS	unknown	XLOC_121975 (1.0)	Squamosa promoter-binding-like protein 11	DOWN	
New084	TTGGACTGAAGGGTGCTCCC	miR319,miR159	3AL,3AS,3B,3DL,3DS,4AL,4BL,4BS,4DL,4DS	Os01g0222001 protein	XLOC_060579 (0.0)	Os01g0222001 protein	No	up
New085	TTTGGATTGAAGGGAGCTCTG	miR159	1AS,2AS,2BS,3AS,3B,3DS,4AL,4BL,5DL,6AL,7DL	unknown	unknown	unknown	UP	
New086	CTTGGATTGAAGGGAGCTCT	miR319,miR159	3AS,3B,3DS,7AL,7AS,7DL	unknown	XLOC_046151 (1.5)	Transcription factor GAMYB	up/DOWN	up
New087	GAGCTCCTATCATTCCAATGA	miR159	3AS,3B,3DS,4AL,5BL,5DL	unknown	unknown	unknown	up/DOWN	
New088	AGCTGCTCGTTCATGGTTCC	miR159	3AS,3B,3DS,4AL,5BL,5DL	unknown	XLOC_146060 (1.5)	Basic leucine zipper 23 and 19	No	
New089	TTGGACTGAAGGGTGCTCCCT	miR319	3AL,3AS,3B,3DL,3DS,4AL,4BL,4BS,4DL,4DS	Os01g0222001 protein	XLOC_060579 (0.0)	Os01g0222001 protein	No	up
New090	AGAGCGTCCTTCAGTCCACTC	miR319	3AS,3B,3DS	Os01g0222001 protein	XLOC_060579 (0.0)	Os01g0222001 protein	No	up
New091	AGGTCATGCTGGAGTTTCATC	miR167	1AS,5AS,5BS,5DS	unknown	unknown	unknown	DOWN	
New092	TGAAGCTGCCAGCATGATCTGA	miR167	1AS,4AS,4BL,4DL,5AL,5AS,5BL,5BS,5DL,5DS,6AS,6BS,6DS	unknown	unknown	unknown	DOWN	
New093	CCCGCCTTGCACCAAGTGAAT	miR168	6DS	unknown	unknown	unknown	No	
New094	TCGCTTGGTGCAGATCGGGAC	miR168	6DS	unknown	unknown	unknown	UP	
New095	TGAGCCGAACCAATATCACTC	miR479,miR171	2AL,2BL,2DL,4AS,4BL,4BS,4DL	unknown	XLOC_079087 (0.0)	Nodulation-signaling pathway 2 protein	DOWN	
New096	TGTTGGCTCGACTCACTCAGA	miR171	4BL,4BS,4DL,5AL	unknown	unknown	unknown	UP	
New098	ACCTGCAGTTGGGCCAATGAC	miR1847	2AL,2BS,3B,4AS,4BS,5BL,7AS,7BL	unknown	unknown	unknown	No	
New099	AAGCTCAGGAGGGATAGCGCC	miR390	5AL,5BL,5DL	unknown	unknown	unknown	No	
New100	TCAGTGCAATCCCTCTGGAAT	miR393	2AL,2BL,2DL	unknown	unknown	unknown	UP	down
New101	TTCCAAAGGGATCGCATTGAT	miR393	2AL,2BL,2DL,3AS,3B	unknown	XLOC_037090 (1.0)	Transport inhibitor response 1-like protein Os04g0395600	UP	down
New102	TTGGCATTCTGTCCACCTCC	miR384,miR394	3AL,3B,3DL,6AL,6BL,6DL	unknown	XLOC_059789 (0.0)	F-box only protein 6	No	
New103	GGTCAAGAAAGCTGTGGGAAG	miR396	6AL,6BL,6DL	unknown	unknown	unknown	UP/down	
New104	TTCCACAGCTTTCTTGAACTG	miR396	6AL,6BL,6DL,7AS,7BS,7DS	unknown	unknown	unknown	No	
New105	TTCCACAGCTTTCTTGAACTT	miR396	6AL,6BL,6DL,7AS,7BS,7DS	unknown	unknown	unknown	UP/down	
New106	TCCACAGGCTTTCTTGAACTG	miR396	2AL,2BL,2DL,4AL,4BS,4DS,6AL,6BL,6DL,7AS,7DS	unknown	XLOC_030908 (0.0)	Growth-regulating factor 3	No	

New107	TTGAGTGCAGCGTTGATGAAC	miR397	3AL,6AS	unknown	XLOC_012268 (0.5)	Laccase-25	No	down
New108	TGCCAAAGGAGAGTTGCCCTG	miR399	1AL,1BL,1DL,2AL,2BL,3AL,3B,4AS, 4BL,4DL,5BL,6BL,7AL,7BL,7DL	unknown	XLOC_063641 (1.0)	Pentatricopeptide repeat-containing protein	No	
New109	TGAACCTTGGGGAAAAGCCGCAT	miR5062	1BL,5AL,5DL	Heat shock cognate 70 kDa protein 2	XLOC_106053 (0.5)	Heat shock 70 kDa protein 3	UP	
New110	TTTTGTTGGTTGTCATCTAACC	miR827	2AL,2BL,2DL,5AL,6AS,6BS	unknown	XLOC_057939 (1.5)	Protein HYPER-SENSITIVITY-RELATED 4	up/DOWN	down
New111	CCTGTTTGTCATTAAGTTTCTT	miR9652	2AL,2BL,2DL	unknown	XLOC_096836 (1.5)	Protein argonaute MEL1	No	
New112	ATCGTTCTGGGTGAATAGGCC	miR9658	6BS	unknown	unknown	unknown	No	
New113	TACCACGACTGTCATTAAGCA	miR9672	5AL,5BL,5DL	unknown	unknown	unknown	DOWN	up
New114	GGTGCTATGGATAAATTCAAC	miR9674	2DL,3AL,3B,3DL,4BL,4BS,4DL,5AL, 6AS,6BS,6DS	unknown	unknown	unknown	DOWN	down
New116	ATAGCATCATCCATCCTACCC	miR9674	1DS,2BS,2DL,4BL,4BS,4DL,5AL,5B L,5DL,6AS,6BS,6DS	unknown	XLOC_123488 (1.0)	Protein Rf1,mitochondrial	No	down
New117	TGAATTTGTCCATAGCATCAG	miR9674,miR 5064	2BS,2DL,3AL,3B,4BL,4BS,4DL,5AL, 5BL,5DL,6AS,6BS,6DS,7AL,7AS	unknown	XLOC_016763 (0.5)	Protein Rf1,mitochondrial	UP	down
New118	GTGCTATGGATAAATTCAACC	miR9674	2DL,3AL,3B,3DL,4BL,4BS,4DL,5AL, 6AS,6BS,6DS	unknown	unknown	unknown	UP	down
New121	GTAGGATGGCTGGTGCTATGG	miR9674	4BL,4DL,6AS,6DS	unknown	unknown	unknown	UP/down	downs
New122	CAGAACCAGAATGAGTAGCTC	miR9679	1BS,2DL,4BL	unknown	unknown	unknown	No	
New123	TGAGATGAGATTACCCCATAC	miR9772	4AL,5BL,5DL,7AL	unknown	XLOC_122610 (1.0)	Nontranslating CDS	No	
New124	TGGACGAGGATGTGCAACTGC	miR9776	2AL,2BL,2DL,5BL	unknown	XLOC_128171 (1.5)	BTB/POZ and MATH domain-containing protein 5	No	

1.2 Non-conserved miRNA

ID	**RNA sequence**	**Pre-miRNA genomic loci**	**Precursor functions**	**Target gene (Expectation)**	**Target gene functions**	**Regulation**
New002	TTCTTCAAGTACTCCACTTTT	6AS,6DS	unknown	XLOC_115479 (0.5)	M7ZLP4: ortholog mitochondrial/chloroplastic transcription termination factor	UP/down
New003	GCGGCTCTCTGGTGTTCAAGC	na	unknown	unknown	unknown	No
New004	GCGGCTCTGTGGTGTTCAAGC	6AS	unknown	unknown	unknown	No
New006	AGGCAGTAAACTGGAGGCAGC	6AS,6BS,6DS	unknown	XLOC_115875 (1.5)	Ribosome biogenesis regulatory protein homolog	UP

New007	CAAGTTATGCAGTTGCTGCCT	1AS,1BS,1DS,6AS,6BL,6BS,6DL,6DS	unknown	XLOC_118633 (0.5)	MADS-box transcription factor 57	UP
New008	TATATTTGCAGGTTTTAGGTCT	4AL,4BS,4DS,7AL,7BL,7DL	ABC transporter G family member 26;G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	XLOC_146342 (0.0)	G-type lectin S-receptor-like serine/threonine-protein kinase	No
New009	ATATATTTGCAGGTTTTAGGTC	4AL,4BS,4DS,7AL,7BL,7DL	ABC transporter G family member 26;G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	XLOC_146342 (0.0)	G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	No
New010	TCAAAGCATCTTGGCAGACCA	7AL,7BL,7DL	G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	XLOC_146342 (0.5)	G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	No
New011	TTGTCTCTCAAGTTGCTTGGA	7AL,7BL,7DL	G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	XLOC_146342 (0.0)	G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	UP
New012	TCTTGACAGACCAAAATCCGTA	3B,4BL,7AL,7BL,7DL	ABC transporter G family member 26;G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	XLOC_144146 (0.0)	Cysteine-rich receptor-like protein kinase 25	No
New013	ATGTTTTCGTCTAGCAATATA	7AL,7BL,7DL	ABC transporter G family member 26;G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	XLOC_146342 (0.0)	G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	UP
New014	CCATGTTTTCGTCTAGCAATA	7AL,7BL,7DL	ABC transporter G family member 26;G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	XLOC_146342 (0.0)	G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	UP
New015	TCTAGGCGAAAACATAATGTT	7AL,7BL	G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	XLOC_017845 (1.5)	Chromatin structure-remodeling complex protein BSH	No
New016	TTAGACCTAGACATGCAAGTA	7AL,7BL,7DL	ABC transporter G family member 26;G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	XLOC_146342 (0.0)	G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	No
New017	GGATGGTTTTGAAGTGATAGC	7AL,7BL,7DL	unknown	XLOC_146342 (0.0)	G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	UP
New018	TAAACCTTCACAAATTCCCTTG	6AS,6BS,7AL,7BL,7DL	ABC transporter G family member 26;G-type lectin S-receptor-like	XLOC_146342 (0.0)	G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	up/DOWN

			serine/threonine-protein kinase At1g11305			
New019	GCTATGGAATGATCAAGGAAT	7AL,7BL,7DL	ABC transporter G family member 26	XLOC_074081 (1.5)	Acetate/butyrate--CoA ligase AAE7 peroxisomal	UP
New021	TATATTTGCAGGTTTGAGGTC	2DS,4AL,4BS,4DS,7AL,7BL,7DL	Serine/threonine-protein kinase mos	XLOC_110271 (1.0)	Putative disease resistance protein RGA4	No
New022	TCAAAGCATCTTGAGAGACC	3AS,3B,3DS,4AL,4BS,4DL,4DS,7BL,7DL	unknown	XLOC_146342 (1.0)	G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	No
New023	TCAAAGCATCTTGAGAGACT	3AS,3DS,4AL,4BS,4DL,4DS,7AS,7BS,7DS	unknown	XLOC_146342 (1.0)	G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	No
New024	TCTTGAGAGACCAAAATCTGC	4AL,4BS,4DS	Serine/threonine-protein kinase mos	XLOC_057426 (1.5)	Cysteine-rich receptor-like protein kinases 29	No
New025	TTGTGGAGATAATGCAAACCC	3B,4AL,4BS,4DS	Serine/threonine-protein kinase mos	XLOC_071855 (1.5)	Serine/threonine-protein kinase mos	UP
New026	TGACAGTTTTGAAGTAATAGC	2AL,4AL,4BS,4DS	unknown	unknown	unknown	UP
New027	TTTGAGGTCTAAGTGGAGAAT	4AL,4BS,4DS	unknown	XLOC_105582 (1.5)	Probable disease resistance protein At1g61190	No
New028	TTTGAGGTCTAAGTAGAGAAT	4BS,4DS	unknown	unknown	unknown	No
New029	TTTGAGGTCTAATTGGAGAAT	4AL,4DS	Serine/threonine-protein kinase mos	XLOC_130924 (1.5)	Probable cadmium/zinc-transporting ATPase HMA1 chloroplastic	No
New030	TTAGACCTGCAAACATATTTC	4AL,4BS,4DS	Serine/threonine-protein kinase mos	unknown	unknown	No
New031	TTAGACCTTAGACCTGCAAAC	4AL,4BS,4DS	Serine/threonine-protein kinase mos	XLOC_071855 (1.0)	Serine/threonine-protein kinase mos	No
New032	TGTTCGCTTAGACCTTAGACC	4AL,4BS,4DS	Serine/threonine-protein kinase mos	unknown	unknown	No
New033	TTAGACCTTAGACCTTCAAAC	4AL,4BS,4DS	unknown	unknown	unknown	UP
New034	TCTAGATGACAATATGATGCT	4AL,4BS,4DS	Serine/threonine-protein kinase mos	unknown	unknown	No
New035	TACATGATATGATGATTGATC	2DL,3AL,3B,3DL,4AL,4AS,5BL,6BS,6DS,7DS	CNL9	XLOC_014097 (1.5)	Phospholipid-transporting ATPase 2	UP/down
New038	TCAAGCATCATGTCATGGACAC	1AL,1AS,1BL,1BS,1DL,1DS,3AL,3B,3DL,4AL,5BL,5DL,6AL,6BL,6BS,6DL,6DS,7AS,7BL,7DS	unknown	XLOC_015670 (0.0)	Disease resistance protein RPM1	No
New040	TAATCTTCTGGAAATATGCTTA	3AL,3B,3DL,7AS	unknown	XLOC_004306 (1.5)	Putative disease resistance RPP13-like protein 3 and 4	up/DOWN
New041	ACTGGTTGGATCATGCTTCTG	3AL,3B,3DL	unknown	XLOC_127572 (1.0)	Disease resistance protein RPM1	DOWN
New042	GGAGCATGATTCAACCAATTG	3AL,3B,3DL,5AL,5DL	unknown	unknown	unknown	No
New043	TATCATGGACACCATAAGACT	3AL,3B,3DL,5BL,7BL	NBS-LRR protein	unknown	unknown	No
New044	ATGAGATCGTGAATAGGAGCA	3AL,3B,3DL	unknown	unknown	unknown	No
New045	TTTATGAGCTCATTGAAGTAACT	1BS,1DS,2AL,3AL,3B,3DL,4AL,4BL,5BL,7BL	unknown	XLOC_150402 (0.5)	Proteasome subunit beta type-5-A	No

New046	TTTATGAGCTCATTGAAGTAAC	1BS,1DS,2AL,3AL,3B,3DL,4AL,4BL,5BL,7BL	unknown	XLOC_150402 (0.5)	Proteasome subunit beta type-5-A	No
New047	TTTACGAGCTCATTGAAGTAAA	3B	unknown	XLOC_024162 (1.0)	Disease resistance protein RPP13	No
New048	TAGAAGGTTAGAAGATTAAGC	3AL,3B,3DL	unknown	unknown	unknown	No
New049	ACATCAATGTTGATAAAGGCATG	3AL,3B,3DL	unknown	unknown	unknown	No
New057	TCAGACCAGGCTTCATTCCCC	1AL,1BL,1BS,1DL,4AS,4BL,4DL,5BL,5DL,7AL,7BL,7DL	unknown	unknown	unknown	No
New059	AGTAATGCTGTCAGAGTTGGAACC	7AL,7BL,7DL	Ubiquitin carboxyl-terminal hydrolase 21	unknown	unknown	UP
New060	AGAAAGCAATGTCAGAGGAGT	7AL,7BL,7DL	Ubiquitin carboxyl-terminal hydrolase 21	XLOC_137230 (1.5)	Ubiquitin carboxyl-terminal hydrolase 21	No
New061	TGAGATTTCCATACTGTGGGC	3DL,4DL	unknown	unknown	unknown	UP/down
New063	CCTTAGTATGAAATTCTGATC	3DL	unknown	unknown	unknown	UP
New064	CTTAGTATGAAATTCTGATC	3DL	unknown	unknown	unknown	No
New065	TGAGAAAGGACTGCATCATC	1AL,1AS,1BL,1BS,1DL,1DS,2AL,2AS,2BL,2BS,2DL,2DS,3AL,3AS,3B,3DL,3DS,4AL,4AS,4BL,4BS,4DL,5AL,5AS,5BL,5BS,5DL,6AL,6AS,6BL,6BS,6DL,6DS,7AL,7AS,7BL,7BS,7DL,7DS	Probable disease resistance RPP8-like protein 2	XLOC_039460 (0.0)	Probable disease resistance RPP8-like protein 2	No
New066	AGTTCCAGACGATGCAGGCCT	2AS,2BS,2DS,7AL,7BL,7DL	unknown	unknown	unknown	UP/down
New068	CCATGATGAGGTCGTTCAACC	3B,4AL,7AS,7DS	Linalool synthase, chloroplastic	unknown	unknown	No
New069	TTTAGGCACTCGTCAGTGACC	4AL,7AS,7DS	Alpha-1,2-galactosyltransferase gmh3 ;Linalool synthase, chloroplastic	unknown	unknown	No
New070	TTGAGGCACTCGTCAGTGACG	7AS,7DL	unknown	unknown	unknown	No
New073	TGAGAAGGTAGATCATAA	1AL,1BL,1BS,1DL,1DS,2AL,2AS,2BL,2BS,3AL,3B,3DL,4AL,4BS,5BL,5DL,5DS,6AL,6BS,6DL,6DS,7AS,7BL,7DL,7DS	CNL6	unknown	unknown	No
New074	TAATAAACAAGTCTTCAGATG	1AL,1AS,1BS,1DS,2AL,2DL,3B,3DL,4AL,5BL,5BS,5DL,6DS,7AS,7BS,7DL,7DS	unknown	XLOC_148918 (0.0)	Putative disease resistance RPP13-like proteins	No
New075	TAATAAACAAGTCTTTAGATG	1AS,1BS,1DS,2AL,2AS,2BL,2BS,2DL,3B,3DS,4AL,4BL,4BS,5BL,6BL,6BS,7AS,7BL,7BS,7DS	unknown	XLOC_135172 (0.5)	Putative disease resistance RPP13-like protein 3	No
New076	TAATAAACATGTCTTCAGATG	1AS,1DS,2AS,2BS,2DS,4AL,5BL,5BS,5DL,6BS,6DS,7AS,7BL,7BS,7DS	unknown	XLOC_024162 (0.0)	Disease resistance protein RPP13	No
New077	CCATTTGAAGACTAGTTTATT	1AS,1BS	unknown	XLOC_129421 (1.5)	Disease resistance protein RPM1	UP

New078	TCTGAAGACTAGTTTATTACA	1AS,1BS,1DS,2AL,4AL	unknown	unknown	unknown	No
New079	CGGCAAGCTAGAGACAGCAAC	2AL,2BL,2DL	unknown	XLOC_030780 (0.0)	MADS-box transcription factor 27	No
New081	TTTTCTTGCAAGTTGTGCAGT	2AL,2BL,2DL,7DS	unknown	XLOC_020605 (0.0)	MADS-box transcription factor 27	No
New097	TATTGGCTCGGCTCACTCAGG	1AL,1BL	unknown	unknown	unknown	No
New115	GTAGGATGGCCGGTGCTATGG	4BL,4DL,6DS	unknown	unknown	unknown	UP
New119	CTTGAACTTCTCCATAGCATC	4DL	unknown	XLOC_058416 (1.5)	Probable glucuronosyltransferase Os01g0926700	No
New120	TTAATTTGTCCATAGCATCCG	3B,3DL	unknown	XLOC_017227 (1.0)	Protein Rf1	No
New125	TGGACGAGGATGTGCAGCTGC	2AL,2BL,2DL,3B,5BL	unknown	unknown	unknown	No
New126	AGCCAACAACCTCCTAGTTCC	1DL,5BS	Disease resistance protein RPM1	XLOC_016489 (0.5)	Probable mixed-linked glucan synthase 6	UP/down
New127	AGGTGGGAGGTTGTTGGCTGG	1BS,5AS,5BL,5BS,5DS	Disease resistance protein RPM1	XLOC_094640 (1.5)	Trafficking protein particle complex II-specific subunit 130 homolog	UP
New128	TTTAAAACTTTGACTTGGAAC	1AL,1AS,1BS,1DL,1DS,2AL,2AS,2BL,2BS,2DL,2DS,3AL,3AS,3B,3DL,3DS,4AL,4AS,4BL,4BS,4DL,4DS,5AL,5AS,5BL,5DL,6AL,6AS,6BL,7AL,7AS,7BL,7BS,7DL,7DS	unknown	XLOC_096043 (1.0)	Peroxisomal	No
New129	CTTGCTGCGTTGTTTGGGTCT	4DL	Neutral/alkaline invertase 1, mitochondrial	XLOC_082991 (0.0)	Neutral/alkaline invertase 1 mitochondrial	UP
New130	TCGCCGCGATTGTTTGGGTCT	4BL	unknown	XLOC_078297 (0.0)	Neutral/alkaline invertase 1 mitochondrial	No
New131	TTCCTTTGATTATTTGATAGCG	4AL,4DS,5AS,6AL	G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	XLOC_095239 (0.0)	G-type lectin S-receptor-like serine/threonine-protein kinase At1g11305	No
New132	TTTGTCCGGAAATACTTGTTC	1AS,1BL,1BS,1DL,2AL,2AS,2BL,2BS,2DL,2DS,3AL,3AS,3B,4AL,4AS,4BL,4DL,5AL,5AS,5BL,5BS,6AL,6AS,6BL,6BS,6DL,7AL,7BL,7DL,7DS	unknown	XLOC_101164 (0.5)	Early nodulin-93	UP
New133	CGCGGCTCCGTCGACTGGTGC	1DS,4BS	unknown	XLOC_111386 (1.5)	Zinc finger protein ZAT12	DOWN
New134	TCTTCCGCGATCATCATGACC	2AL,2BL	unknown	unknown	unknown	No

2. siRNA

ID	RNA sequence	Pre-miRNA genomic loci	Precursor functions	Target gene (Expectation)	Target gene functions	Regulation
New135	TATATTATCAGGTTTTAGGTCC	1BL,1DS,6AL,6BL,6DL	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	No
New136	TGATAATCCGGCACGAAAGTAC	6AL,6BL,6DL	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	No

New137	TTTTGAAATGATAATCCGGCAC	6AL,6BL,6DL	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	UP
New138	TTTCGTGCCGGATTATCATTTC	6AL,6BL,6DL	Cysteine-rich receptor-like protein kinase 25	XLOC_064557 (1.5)	Putative glutaredoxin-C2	No
New139	GAATTCTACGGTGGAAAGCTC	6AL,6BL,6DL,7AS	unknown	unknown	unknown	No
New140	GAATTCTACGGTGGAAAACTC	6AL,6BL,6DL,7AS	unknown	unknown	unknown	No
New141	TTTCCACCGTAGAATTCTGGT	6AL,6BL,6DL,7AS	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	No
New142	ACCGTAGAATTCTGGTGCCAAA	6AL,6BL,6DL,7AS	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	No
New143	AGACATATACAGTTTGGGTGTT	2AS,2DL,2DS,4AL,6AL,6BL,6DL,7AL,7BL,7DL	unknown	XLOC_093728 (1.5)	Putative ALA-interacting subunit 4	No
New144	ATTCAAGTCAGACATATACAGT	2AS,3AL,4DS,5BL,6AL,6BL,6BS,6DL,6DS	unknown	unknown	unknown	No
New145	ACACCCAAACTGTATATGTCTG	2AL,2AS,2BS,2DL,2DS,3AS,3B,4AL,6AL,6BL,6BS,6DL,7AL,7BL,7BS,7DL	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	UP
New146	ATGATTATAACACCCAAACTGT	2AS,2BS,2DL,4AL,5AL,6AL,6BL,6DL,6DS,7AL,7BL	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	No
New147	ACCCAAACTGTATATGTCTGAC	2AL,2AS,2BS,2DL,2DS,3AL,3B,4AL,4BS,4DL,4DS,5BL,6AL,6BL,6BS,6DL,7AL,7BL,7BS,7DL	LEAF RUST 10 DISEASE-RESISTANCE LOCUS RECEPTOR-LIKE PROTEIN KINASE-like 2.4	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinases	No
New148	TATCGGTGGACAACAATATGT	6AL,6DL	unknown	unknown	unknown	No
New149	CATATTATTCACTTGGACCTA	6BL,6DL	unknown	unknown	unknown	No
New150	CATATTGTTCACTTGGACCTA	6AL,6BL,6DL,7DL	unknown	unknown	unknown	No
New151	TTATTCACTTGGACCTAAAACC	1DS,2AL,2AS,2BL,2BS,2DS,3B,4AL,5AL,5BL,5DL,6BL,6DL,7DL	unknown	unknown	unknown	No
New152	TTTGAGAGACCAAAGTCAGCT	6AL,6BL,6DL	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	No
New153	CCGGTCACATTCAAGTCAGAC	2AS,6AL,6BL,6DL	unknown	unknown	unknown	No
New154	TATATCACCGGTCACATTCAAG	1BL,2AS,6AL,6BL,6DL	unknown	unknown	unknown	No
New155	ACCGGTCACATTCAAGTCAGAC	2AS,6AL,6BL,6DL	unknown	unknown	unknown	No
New156	TATGTCTGACTTGAATGTGAGT	2AS,2BS,6AL,6BL,6DL	Cysteine-rich receptor-like protein kinase 25	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	UP
New157	TCTGACTTGAATGTGACCGGT	2AS,6AL,6BL,6DL	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	No
New158	TATAAGACATGTTGGAAAGCT	6BL,6DL	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	No
New159	AAAGTCAGCTATTTTCGGTACC	6AL,6BL,6DL	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	UP
New160	TATTTTCGGTACCATATTATC	2DL,6AL,6BL,6BS,6DL,7BS	Cysteine-rich receptor-like protein kinase 25 ;LEAF RUST 10 DISEASE-RESISTANCE LOCUS RECEPTOR-LIKE PROTEIN KINASE-like 2.4	XLOC_143073 (0.0)	Putative cysteine-rich receptor-like protein kinase 32	UP

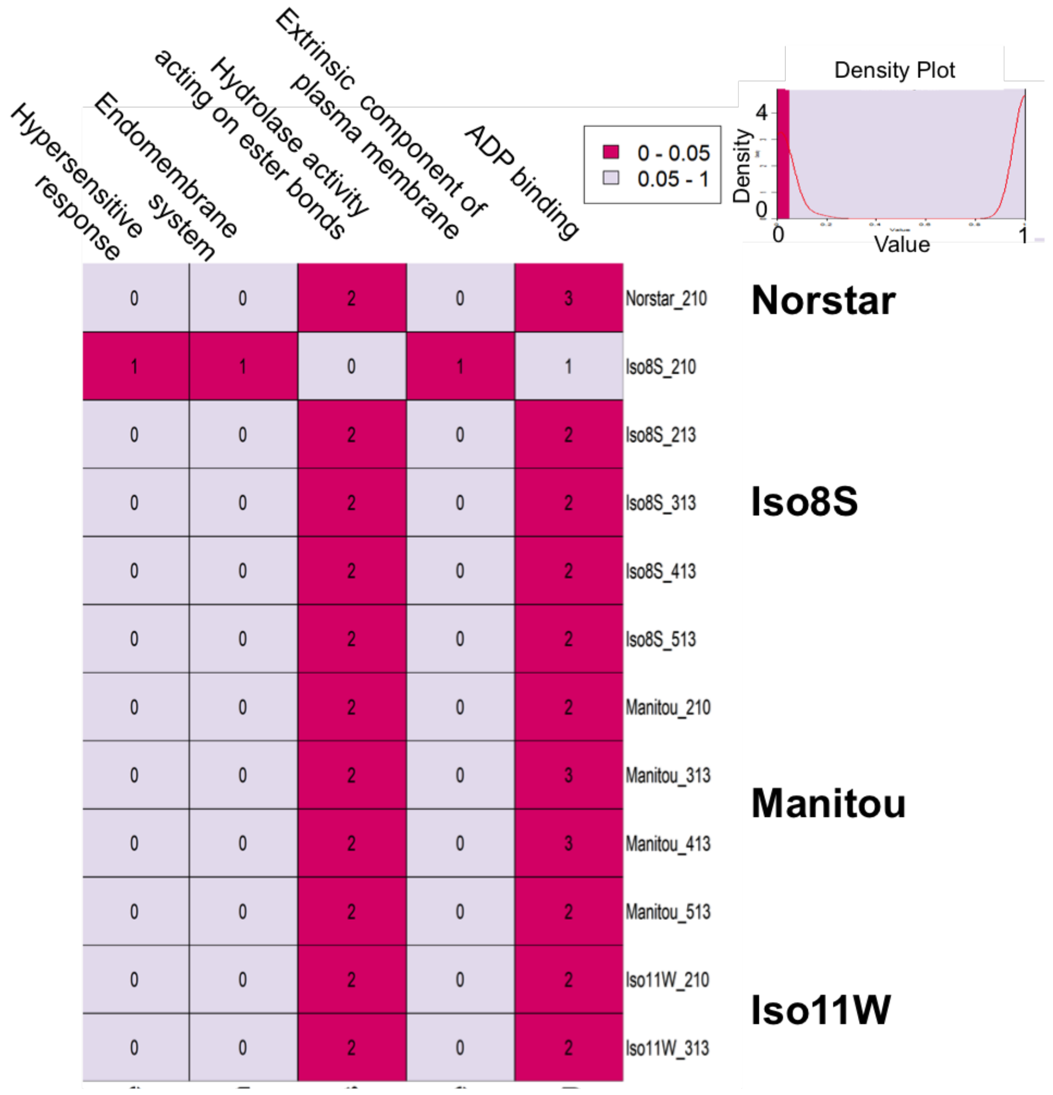
New161	CCAAAGTCGGCAATTTTGGGC	2AL,2BL,2DL,4AL,6AL,6BL,6DL,7AS,7DS	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	No
New162	TACCGAAAATAGCTGACTTTG	6AL,6BL,6DL,7AS	unknown	unknown	unknown	UP
New163	ATATAATCATGAAGACTCCCT	4BL,4DL,6AL,6BL,6DL,7AS	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	No
New164	GATATAATCATGAAGACTCCCT	4BL,4DL,6AL,6BL,6DL,7AS	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	No
New165	ATATACCAAAGATTGCTCTGCT	2AS,3B,3DL,6AL,6BL,6DL	unknown	unknown	unknown	No
New166	TATACCAAAGATTGCTCTGCTT	2AS,3B,3DL,6AL,6BL,6DL,7BL,7DL	unknown	unknown	unknown	No
New167	ATACCAAAGATTGCTCTGCTT	2AS,3B,3DL,6AL,6BL,6DL	unknown	unknown	unknown	No
New168	AAGACTGCTCTGCTTTGAGTA	2AS,2BL,2DL,3B,4AL,5AL,5BL,5DS,6AL,6BL,6BS,6DL,7AL,7BL	LEAF RUST 10 DISEASE-RESISTANCE LOCUS RECEPTOR-LIKE PROTEIN KINASE-like 2.4	unknown	unknown	No
New169	TATACCAAAGACTGCTCTGCTT	2AS,6AL,6BL,6DL,7BL	unknown	unknown	unknown	No
New170	GCACGAAAGTACTTTGAGAGAC	6AL,6BL,6DL	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	UP
New171	CGGCACGAAAGTACTTTGAGAG	6AL,6BL,6DL	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	No
New172	TTCGAGTGAGAGGCTGAATGTT	6DL	Cysteine-rich receptor-like protein kinase 25	unknown	unknown	No
New173	CATCTCGTGGACTTGAATGGGG	6AL,6AS,6BL,6DL,6DS,7AS	unknown	XLOC_011227 (1.5)	Putative laccase-19	No
New174	TCACCGATGCATCTCGTGGACT	na	unknown	unknown	unknown	No
New175	CATCGGGTTATTTGGCACCAG	na	Cysteine-rich receptor-like protein kinase 25 ;LEAF RUST 10 DISEASE-RESISTANCE LOCUS RECEPTOR-LIKE PROTEIN KINASE-like 2.4	unknown	unknown	No
New176	AGCTTTGAGAGACTTAATCGGC	5DL,6AL,6BL,6DL	unknown	XLOC_117461 (0.0)	LEAF RUST 10 DISEASE-RESISTANCE LOCUS RECEPTOR-LIKE PROTEIN KINASE-like 2.4	No
New177	ACCATTGATGAGCTTTGAGAGA	6BL,6DL	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	No
New178	TTTGTGAGGGCTTATATTACC	6AL,6BL,6DL,7AS	unknown	unknown	unknown	No
New179	AATAATTAGATACCTCTTGCCC	6AL,6DL	unknown	XLOC_125168 (0.0)	Cysteine-rich receptor-like protein kinase 25	UP
New180	GAAGAGCGCGTGGCGAGGGAA	2BS,6AL,6BL	unknown	unknown	unknown	UP/down
New181	CCCTCGCACCGCGCTCTTCTT	6AL,6DL	unknown	unknown	unknown	UP/down
New182	TCTTTCTTCGCCGTCCACCTC	6AL,6BL,6DL	unknown	XLOC_074566 (1.5)	Myb-related protein P	No
New183	TCTACTCACCCCGTCTATCCC	6AL,6BL,6DL	unknown	unknown	unknown	No
New184	CGAGCACCACCTCTCAGGCTG	6DL	unknown	unknown	unknown	UP
New185	TGTAGACGAGCTCAGGGAGGA	6AL,6BL,6BS,6DL	unknown	unknown	unknown	No

New186	GATAGGGAAACTGACATGCCT	1BS,2BS,2DL,3B,4BS,5DL,6AS,6BS,6DS,7BL	Disease resistance protein RPM1	XLOC_129568 (0.0)	Putative disease resistance RPP13-like protein 3	No
New187	TGCTCCATATCATGTACTGTA	2BS,4BS,6AS,6BS,6DS,7BL	Disease resistance protein RPM1	XLOC_128777 (0.0)	Putative disease resistance protein At1g50180	No
New188	TCCCCGAGCTCAAATGACCCT	6DS	Disease resistance protein RPM1	unknown	unknown	DOWN
New189	ATGGCACAAGTTTGCGAGGAC	6DS	Disease resistance protein RPM1	XLOC_129568 (0.0)	Putative disease resistance RPP13-like protein 3	No
New190	AGACGAGAAAGGAGTGGAACAC	1BS,6AS,6BS,6DS,7BL	Disease resistance protein RPM1	XLOC_128038 (0.0)	Disease resistance protein RPM1	UP
New191	CCCGGCATCCCTCTTGAGCCC	1DL	Subtilisin-like protease SBT1.4	unknown	unknown	No
New192	ACCGATGATGACTCCCTCGCC	1DL,2DL	Subtilisin-like protease SBT1.4	unknown	unknown	UP
New193	CTGTACAGCCTCAACAGGGGC	1DL	Subtilisin-like protease SBT1.4	XLOC_016652 (0.0)	Subtilisin-like protease SBT1.4	UP
New194	ATAGATGCCGGTGTCAACTAC	1AL,1BL,1DL	Subtilisin-like protease SBT1.4	XLOC_014001 (1.0)	Subtilisin-like protease SBT1.2	No

3. false RNA

ID	RNA sequence	Pre-miRNA genomic loci	Precursor functions	Target gene (Expectation)	Target gene functions	Regulation
New195	AAGAACCACTCTCACGCACCGAAA	7BS	unknown	unknown	unknown	DOWN
New196	TCCGTAGAAGAGACAAAACCCT	1DL,3B,5BL,6AL,6BL,6DL,7BS,7DL	unknown	unknown	unknown	No
New197	AGACGTATTGTTAGCCTCGCC	2AS,7BS	unknown	unknown	unknown	No
New198	TCAAGAATCATGTCATGGACA	1BS,3AL,3DL,3DS,4AL,4AS,4DL,5AS,5BL,5BS,5DS,6AL,6DL,6DS	Disease resistance protein RPM1	XLOC_042430 (0.5)	Disease resistance protein RPM1	No
New199	CGAATTGTTTGTCAAGCTCCGCCT	3AS,3B	unknown	unknown	unknown	No
New200	AGCACTGTGGCCTCCGACTCGAAT	3AS,3B	unknown	XLOC_127478 (1.5)	B-box zinc finger protein 22	No
New201	TTATAGGAAAATTCTGCGAGCACC	5DL	unknown	unknown	unknown	No
New202	GCAGAACTCTTGGCCTTGGTACTC	1AL,1DL	unknown	unknown	unknown	UP
New203	CAACAATACGTCCGATGGTGT	2DS,3DL,6DL,7BS	unknown	unknown	unknown	No
New204	GTATCAACTCTCATCATGGCGGCC	1BL,1DL	Calmodulin	unknown	unknown	UP
New205	AATTGAGATGTAGAACAACTA	5BL	Leucine-rich repeat receptor-like kinase protein THICK TASSEL DWARF1	XLOC_104717 (1.0)	Leucine-rich repeat receptor-like protein kinase PXL1	UP

New206	CCAACGATCGGTGACAGAGAA	2DL	unknown	unknown	unknown	UP
New207	ACGACTGGGTTGTTTGCTCGT	2AS	unknown	unknown	unknown	No
New208	GTAGCCGGGATAGACGAAGCT	1AL,4AL	unknown	unknown	unknown	UP
New209	CTCTGGCCAGTTGGATCAAAAGGC	6DL	unknown	unknown	unknown	No
New210	ATTCCGGATGACGAAAGCACAATT	1BL,3B,3DL	unknown	unknown	unknown	DOWN
New211	AAGAGAATCGGGTATTGAGCCACT	5AL	unknown	XLOC_098600 (0.5)	LRR receptor-like serine/threonine-protein kinase GSO1	DOWN
New212	AAGCGAATCGGATAATGAGCCACT	5DL	unknown	unknown	unknown	up/DOWN
New213	AGTACCGGAGACGTTAGATCAATC	2BL,3AL,3DS,4AL,4DL,4DS,6AL,6DL,6DS,7DL	unknown	unknown	unknown	No
New214	TTTAGTTTGATTGATCTAACGTCT	1AS,1BL,1BS,2BL,2DS,3AL,3B,3DS,4AL,4DL,4DS,6AL,6AS,6DL,6DS,7DL	unknown	unknown	unknown	No
New215	TTTCCTTGACTCGTAGCTGCAGA	6AL,6AS,6BL,6BS,6DL	unknown	unknown	unknown	UP
New216	TGCCCAAGATGTAGACACTCGT	1AL,1AS,1BL,2AL,2AS,2BL,2BS,2DL,2DS,3AL,3AS,3B,3DL,3DS,4AL,4AS,4BL,4BS,4DL,5AL,5AS,5BL,5BS,5DL,5DS,6AL,6AS,6BL,6BS,6DS,7AL,7AS,7BL,7BS,7DL	unknown	XLOC_065233 (1.5)	Serine/threonine-protein phosphatase 7 long form homolog	No
New217	CGAGGGCTTGTGATCATGGACACT	3AS,3B	unknown	unknown	unknown	No
New218	AGAGGATCTTCGCCTGATGGA	4AL,7AS,7DS	unknown	unknown	unknown	DOWN
New219	GCTCGGCTACATCTCTGAAGTGGA	3AS	unknown	unknown	unknown	No
New220	TTCGGAACCAATGGACACAAACAC	2BS	unknown	XLOC_078292 (1.5)	Purple acid phosphatase 3	UP



	Hypersensitive response	Endomembrane system	Hydrolase activity acting on ester bonds	Extrinsic component of plasma membrane	ADP binding
Norstar_210	0	0	2	0	3
Iso8S_210	1	1	0	1	1
Iso8S_213	0	0	2	0	2
Iso8S_313	0	0	2	0	2
Iso8S_413	0	0	2	0	2
Iso8S_513	0	0	2	0	2
Manitou_210	0	0	2	0	2
Manitou_313	0	0	2	0	3
Manitou_413	0	0	2	0	3
Manitou_513	0	0	2	0	2
Iso11W_210	0	0	2	0	2
Iso11W_313	0	0	2	0	2

Figure 4.5 Enriched GO terms by precursor XLOCs

Numbers shown in the table indicates the counts of occurrence; dark pink color indicates p-value $<$ 0.05 in the statistic test. The small plot indicates the distribution of p-values between 0 and 1. Same plotting legends for the next enrichment figures.

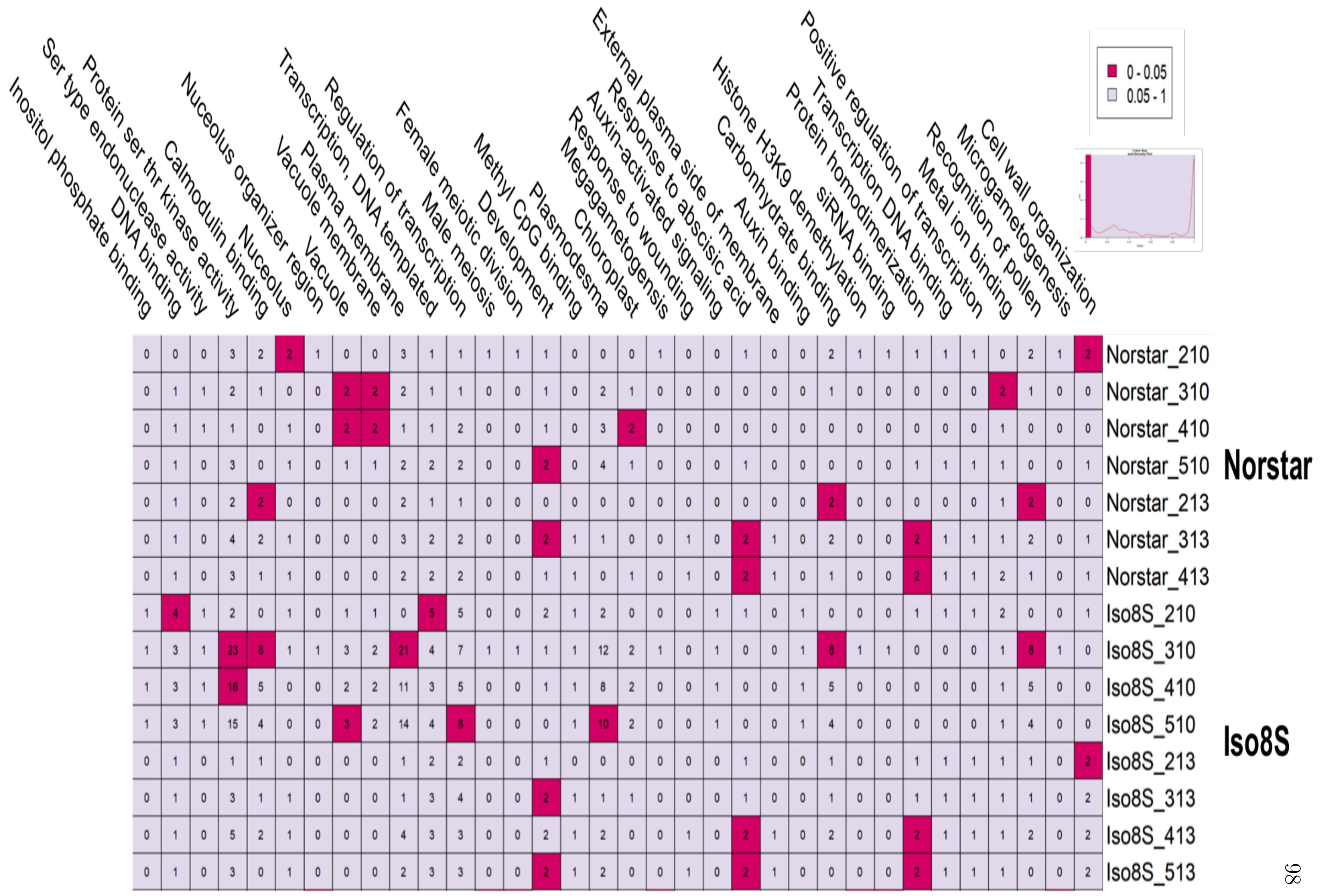
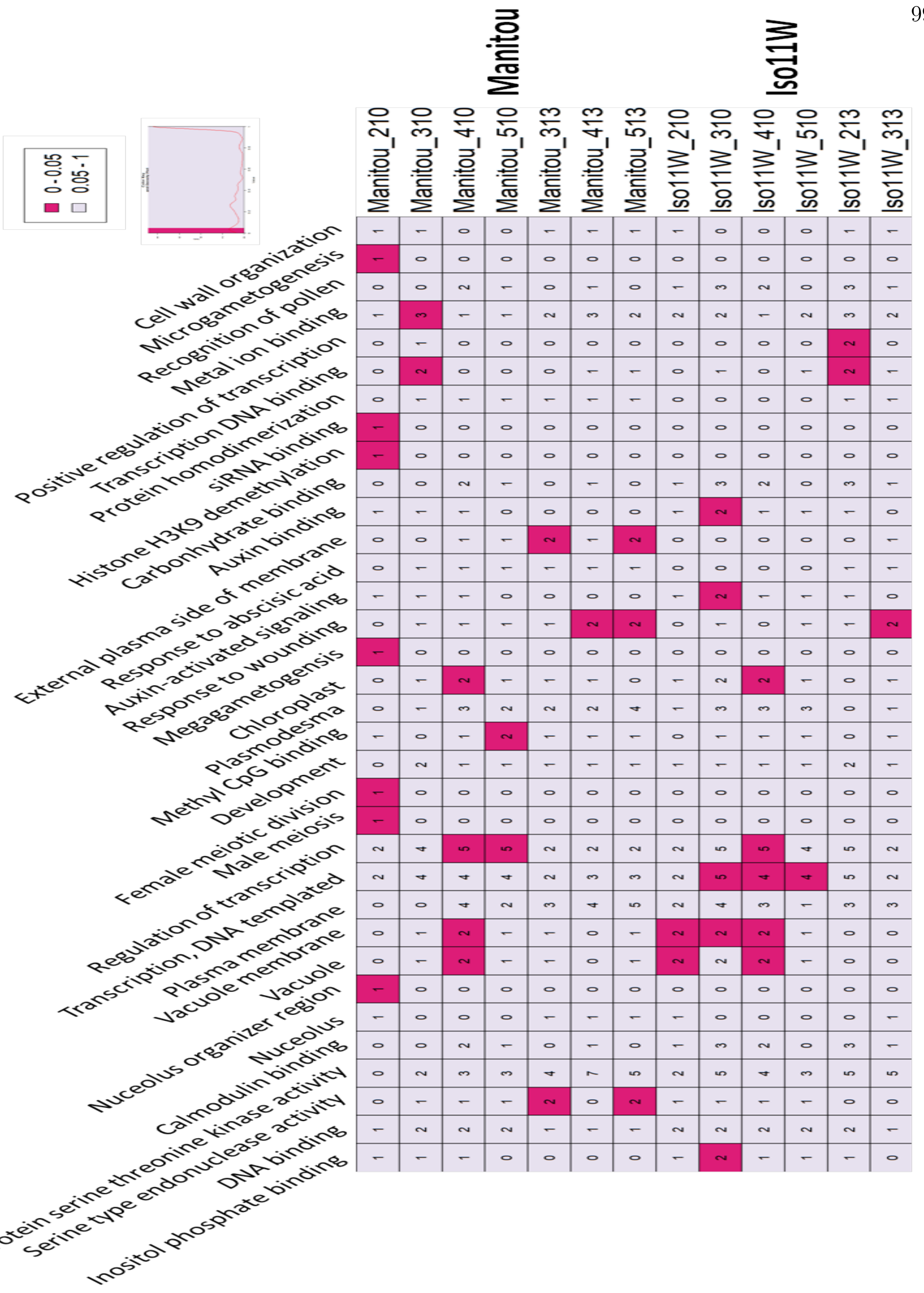


Figure 4.6 Enriched GO terms by top 1 target gene



	Manitou_210	Manitou_310	Manitou_410	Manitou_510	Manitou_313	Manitou_413	Manitou_513	Iso11W_210	Iso11W_310	Iso11W_410	Iso11W_510	Iso11W_213	Iso11W_313
Cell wall organization	1	1	0	0	1	1	1	1	0	0	0	1	1
Microgametogenesis	**1**	0	0	0	0	0	0	0	0	0	0	0	0
Recognition of pollen	0	0	2	1	0	1	0	1	3	2	0	3	1
Metal ion binding	1	**3**	1	1	2	3	2	2	2	1	2	3	2
Positive regulation of transcription	0	1	0	0	0	0	0	0	0	0	0	**2**	0
Transcription DNA binding	0	**2**	0	0	0	1	1	0	1	0	1	**2**	1
Protein homodimerization	0	1	1	1	1	1	1	0	0	0	0	1	1
siRNA binding	**1**	0	0	0	0	0	0	0	0	0	0	0	0
Histone H3K9 demethylation	**1**	0	0	0	0	0	0	0	0	0	0	0	0
Carbonhydrate binding	0	0	2	1	0	1	0	1	3	2	0	3	1
Auxin binding	1	1	1	0	0	0	0	1	**2**	1	1	1	0
External plasma side of membrane	0	0	1	1	**2**	1	**2**	0	0	0	0	0	1
Response to abscisic acid	0	1	1	1	1	1	1	0	0	0	0	1	1
Auxin-activated signaling	1	1	1	0	0	0	0	1	**2**	1	1	1	0
Response to wounding	0	1	1	1	1	**2**	**2**	0	1	0	1	1	**2**
Megagametogensis	**1**	0	0	0	0	0	0	0	0	0	0	0	0
Chloroplast	0	1	**2**	1	1	1	0	1	2	**2**	1	0	1
Plasmodesma	0	1	3	2	2	2	4	1	3	3	3	0	1
Methyl CpG binding	1	0	1	**2**	1	1	1	0	1	1	1	0	1
Development	0	2	1	1	1	1	1	1	1	1	1	2	1
Female meiotic division	**1**	0	0	0	0	0	0	0	0	0	0	0	0
Male meiosis	**1**	0	0	0	0	0	0	0	0	0	0	0	0
Regulation of transcription	2	4	**5**	**5**	2	2	2	2	5	**5**	4	5	2
Transcription, DNA templated	2	4	4	4	2	3	3	2	**5**	**4**	**4**	5	2
Plasma membrane	0	0	4	2	3	4	5	2	4	3	1	3	3
Vacuole membrane	0	1	**2**	1	1	0	1	**2**	**2**	**2**	1	0	0
Vacuole	0	1	**2**	1	1	0	1	**2**	2	**2**	1	0	0
Nuceolus organizer region	**1**	0	0	0	0	0	0	0	0	0	0	0	0
Nuceolus	1	0	0	0	1	1	1	1	0	0	0	0	1
Calmodulin binding	0	0	2	1	0	1	0	1	3	2	0	3	1
Protein serine threonine kinase activity	0	2	3	3	4	7	5	2	5	4	3	5	5
Serine type endonuclease activity	0	1	1	1	**2**	0	**2**	1	1	1	1	0	0
DNA binding	1	2	2	2	1	1	1	2	2	2	2	2	1
Inositol phosphate binding	1	1	1	0	0	0	0	1	**2**	1	1	1	0

Figure 4.6 Enriched GO terms by top 1 target gene, continued

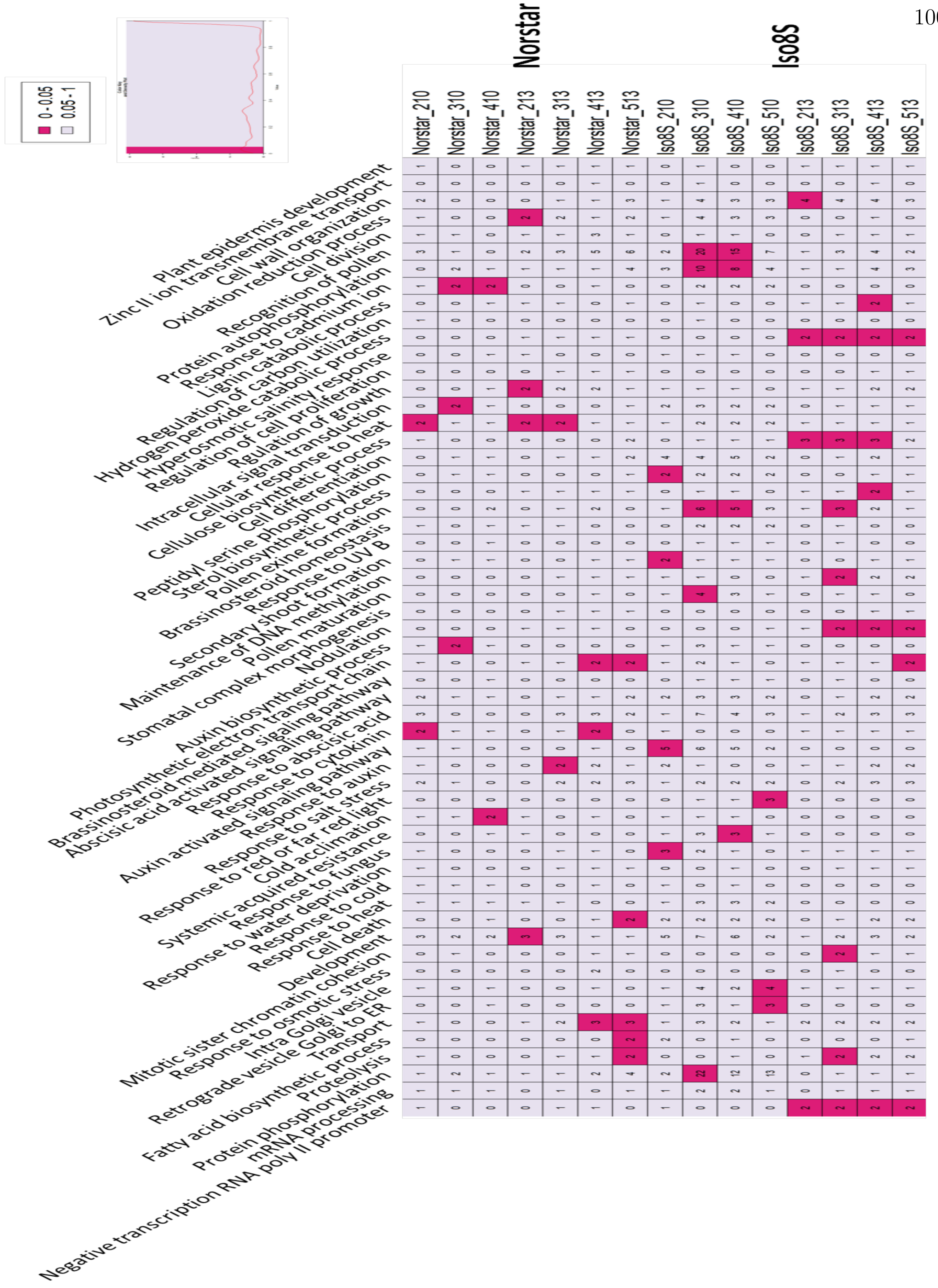
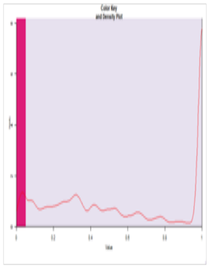


Figure 4.7 Enriched GO Biological Process terms by top 5 scored target genes



🟥 0 - 0.05
▢ 0.05 - 1

	Manitou						Iso11W						
	Manitou_310	Manitou_510	Manitou_213	Manitou_313	Manitou_413	Manitou_513	Iso11W_210	Iso11W_310	Iso11W_510	Iso11W_213	Iso11W_313	Iso11W_413	Iso11W_513
Plant epidermis development	1	0	1	0	1	0	0	1	**2**	2	1	2	2
Zinc II ion transmembrane transport	0	0	0	**2**	1	0	0	0	0	0	0	2	2
Cell wall organization	1	2	4	3	3	**5**	1	1	0	3	4	4	5
Oxidation reduction process	0	1	0	1	0	0	0	2	1	0	1	1	2
Cell division	0	0	**4**	3	2	1	0	0	0	1	**4**	**5**	3
Recognition of pollen	0	1	4	4	6	2	2	3	1	5	2	8	8
Protein autophosphorylation	2	3	3	4	3	5	1	2	3	1	2	3	3
Response to cadmium ion	0	0	0	1	0	1	1	1	1	1	1	1	1
Lignin catabolic process	0	1	0	**2**	1	1	0	0	1	1	1	2	2
Regulation of carbon utilization	0	0	**2**	1	1	0	0	0	0	0	1	2	2
Hydrogen peroxide catabolic process	1	**2**	1	1	1	1	0	0	0	1	1	1	1
Hyperosmotic salinity response	1	0	1	0	1	1	0	**2**	1	1	**2**	1	1
Regulation of cell proliferation	0	1	1	**2**	1	**2**	0	0	0	0	1	1	0
Regulation of growth	1	2	1	1	1	1	0	1	0	0	2	1	0
Intracellular signal transduction	2	1	1	**3**	**3**	**3**	1	2	1	2	2	**4**	**4**
Cellular response to heat	1	1	0	1	0	1	1	**2**	1	1	0	0	2
Cellulose biosynthetic process	0	1	2	2	2	2	1	1	0	1	**3**	3	3
Cell differentiation	1	3	2	2	2	2	2	3	**4**	3	2	2	1
Peptidyl serine phosphorylation	1	1	1	**2**	1	**2**	1	**2**	1	1	1	1	1
Sterol biosynthetic process	1	**2**	1	1	1	1	0	0	1	0	0	0	0
Pollen exine formation	1	2	0	1	1	0	0	2	1	1	1	2	2
Brassinosteroid homeostasis	1	1	0	0	0	0	0	**2**	1	0	1	1	2
Response to UV B	2	1	1	0	1	1	0	2	**2**	1	2	2	2
Secondary shoot formation	1	1	1	1	1	**2**	1	1	1	0	0	0	0
Maintenance of DNA methylation	0	1	2	2	2	1	0	0	0	1	1	2	2
Pollen maturation	0	0	0	1	1	0	0	0	0	1	0	0	0
Stomatal complex morphogenesis	0	1	1	**2**	1	**2**	0	0	0	0	1	1	0
Nodulation	1	**2**	1	1	1	1	0	1	0	1	0	1	1
Auxin biosynthetic process	1	1	1	2	0	**3**	1	1	1	0	1	1	1
Photosynthetic electron transport chain	0	0	0	0	0	0	0	0	1	1	0	1	2
Brassinosteroid mediated signaling pathway	**2**	1	0	0	0	0	0	1	1	1	1	1	1
Abscisic acid activated signaling pathway	1	1	2	**4**	**4**	2	2	2	1	1	3	4	4
Response to abscisic acid	2	1	**6**	3	3	1	2	1	2	4	3	6	6
Response to cytokinin	1	0	0	0	0	0	0	0	1	1	0	1	2
Auxin activated signaling pathway	2	1	2	2	1	2	2	4	2	3	3	4	2
Response to auxin	1	1	2	1	1	1	0	0	0	1	1	1	1
Response to salt stress	2	1	1	**4**	**4**	3	1	1	1	2	**4**	**5**	4
Response to red or far red light	0	0	0	0	0	0	0	0	1	1	0	1	1
Cold acclimation	1	0	1	0	1	1	0	2	**2**	1	2	2	2
Systemic acquired resistance	0	0	0	0	0	0	0	1	0	0	1	0	0
Response to fungus	1	0	1	0	0	0	0	0	0	1	0	0	1
Response to water deprivation	1	0	0	1	1	0	**2**	1	1	1	1	1	1
Response to cold	1	0	2	0	1	2	1	2	**2**	1	1	2	2
Response to heat	1	0	1	0	1	1	1	**3**	1	2	1	1	2
Cell death	1	0	0	2	2	2	1	1	0	2	2	2	2
Development	3	2	2	2	4	2	1	4	2	5	3	3	3
Mitotic sister chromatin cohesion	1	0	1	**2**	1	**2**	0	0	0	1	1	2	2
Response to osmotic stress	2	0	0	2	2	1	0	0	0	**3**	0	3	2
Intra Golgi vesicle	0	1	2	1	1	1	1	2	1	1	1	2	2
Retrograde vesicle Golgi to ER	0	1	1	0	0	0	0	1	1	0	0	1	1
Transport	0	0	0	1	0	1	0	1	1	2	**3**	3	**4**
Fatty acid biosynthetic process	0	0	1	0	0	2	1	1	0	0	0	0	1
Proteolysis	0	1	1	2	2	1	1	0	0	1	2	2	2
Protein phosphorylation	1	3	9	2	3	5	1	2	2	6	1	8	10
mRNA processing	2	1	2	1	2	1	0	1	**2**	1	1	2	0
Negative transcription RNA poly II promoter	1	0	1	1	1	0	0	0	0	2	0	1	2

Figure 4.7 Enriched GO Biological Process terms by top 5 scored target genes, continued

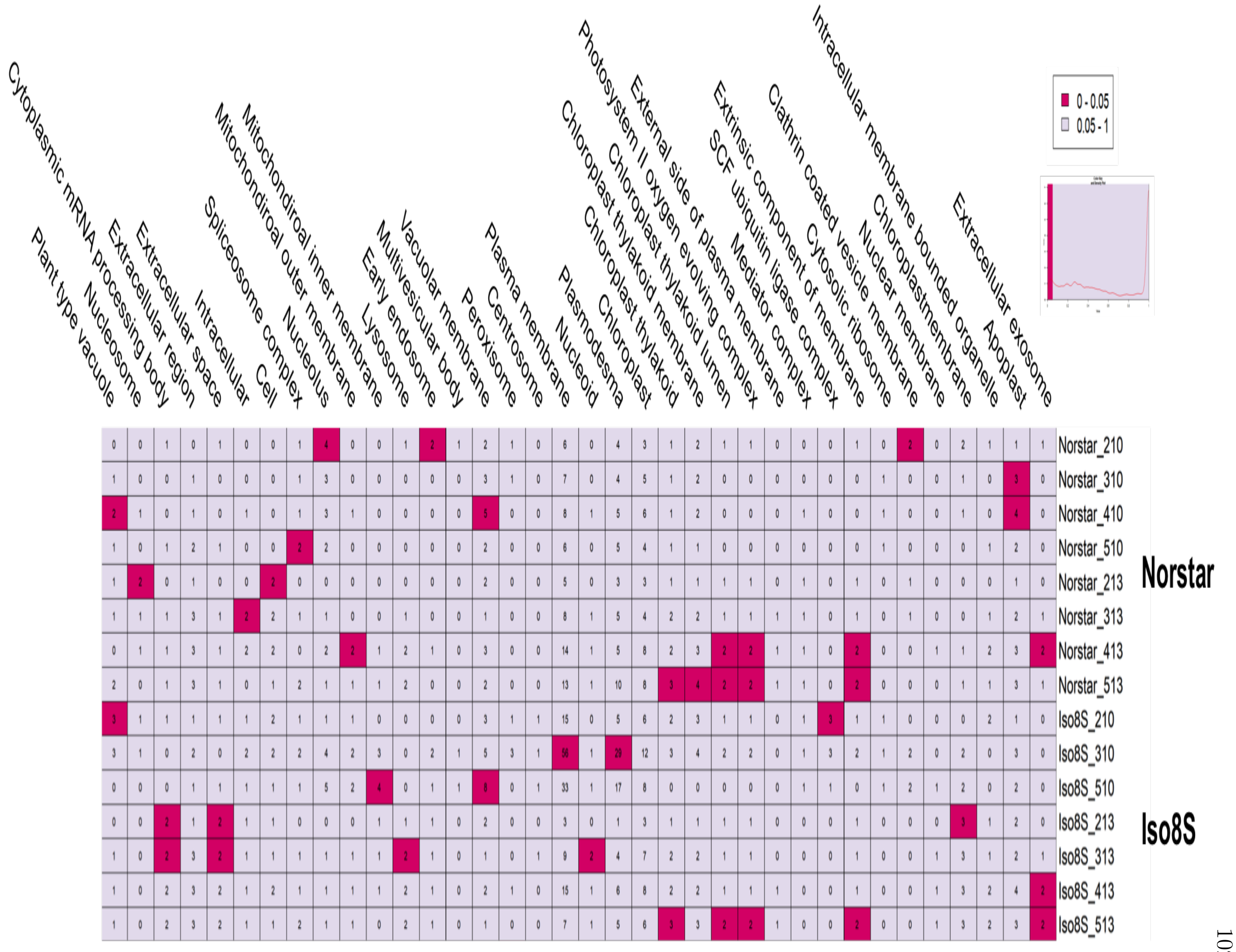
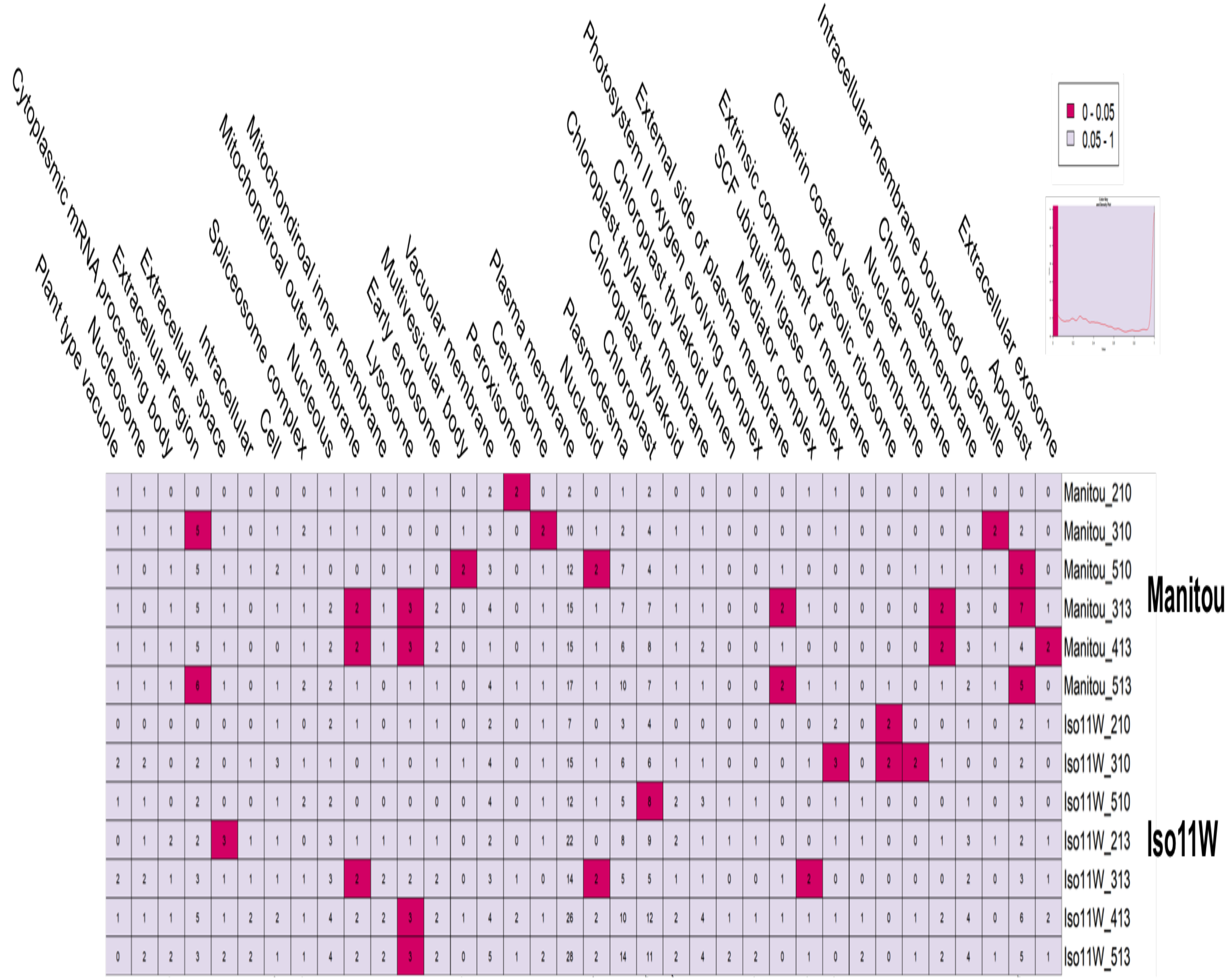


Figure 4.8 Enriched GO Cellular Component terms by top 5 scored target genes



	Manitou_210	Manitou_310	Manitou_510	Manitou_313	Manitou_413	Manitou_513	Iso11W_210	Iso11W_310	Iso11W_510	Iso11W_213	Iso11W_313	Iso11W_413	Iso11W_513
Extracellular exosome	0	0	0	1	**2**	0	1	0	0	1	1	2	1
Apoplast	0	2	**5**	**7**	4	**5**	2	2	3	2	3	6	5
Intracellular membrane bounded organelle	0	**2**	1	0	1	1	0	0	0	1	0	0	1
Chloroplastmembrane	1	0	1	3	3	2	1	0	1	3	2	4	4
Nuclear membrane	0	0	1	**2**	**2**	1	0	1	0	1	0	2	2
Clathrin coated vesicle membrane	0	0	1	0	0	0	0	**2**	0	0	0	1	1
Cytosolic ribosome	0	0	0	0	0	1	**2**	**2**	0	0	0	0	0
Extrinsic component of membrane	0	0	0	0	0	0	0	0	1	1	0	1	2
SCF ubiquitin ligase complex	1	1	0	0	0	1	2	**3**	1	1	0	1	0
Mediator complex	1	0	0	1	0	1	0	1	0	0	**2**	1	1
External side of plasma membrane	0	0	1	**2**	1	**2**	0	0	0	0	1	1	0
Photosystem II oxygen evolving complex	0	0	0	0	0	0	0	0	1	1	0	1	2
Chloroplast thylakoid lumen	0	0	0	0	0	0	0	0	1	1	0	1	2
Chloroplast thylakoid membrane	0	1	1	1	2	1	0	1	3	1	1	4	4
Chloroplast thylakoid	0	1	1	1	1	1	0	1	2	2	1	2	2
Chloroplast	2	4	4	7	8	7	4	6	**8**	9	5	12	11
Plasmodesma	1	2	7	7	6	10	3	6	5	8	5	10	14
Nucleoid	0	1	**2**	1	1	1	0	1	1	0	**2**	2	2
Plasma membrane	2	10	12	15	15	17	7	15	12	22	14	26	28
Centrosome	0	**2**	1	1	1	1	1	1	1	1	0	1	2
Peroxisome	**2**	0	0	0	0	1	0	0	0	0	1	2	1
Vacuolar membrane	2	3	3	4	1	4	2	4	4	2	3	4	5
Multivesicular body	0	1	**2**	0	0	0	0	1	0	0	0	1	0
Early endosome	1	0	0	2	2	1	1	1	0	1	2	2	2
Lysosome	0	0	1	**3**	**3**	1	1	0	0	1	2	**3**	**3**
Mitochondiroal inner membrane	0	0	0	1	1	0	0	1	0	1	2	2	2
Mitochondiroal outer membrane	1	1	0	**2**	**2**	1	1	0	0	1	**2**	2	2
Nucleolus	1	1	0	2	2	2	2	1	2	3	3	4	4
Spliceosome complex	0	2	1	1	1	2	0	1	2	0	1	1	1
Cell	0	1	2	1	0	1	1	3	1	1	1	2	1
Intracellular	0	0	1	0	0	0	0	1	0	1	1	2	2
Extracellular space	0	1	1	1	1	1	0	0	0	**3**	1	1	2
Extracellular region	0	**5**	5	5	5	**6**	0	2	2	2	3	5	3
Cytoplasmic mRNA processing body	0	1	1	1	1	1	0	0	0	2	1	1	2
Nucleosome	1	1	0	0	1	1	0	2	1	1	2	1	2
Plant type vacuole	1	1	1	1	1	1	0	2	1	0	2	1	0

Figure 4.8 Enriched GO Cellular Component terms by top 5 scored target genes, continued

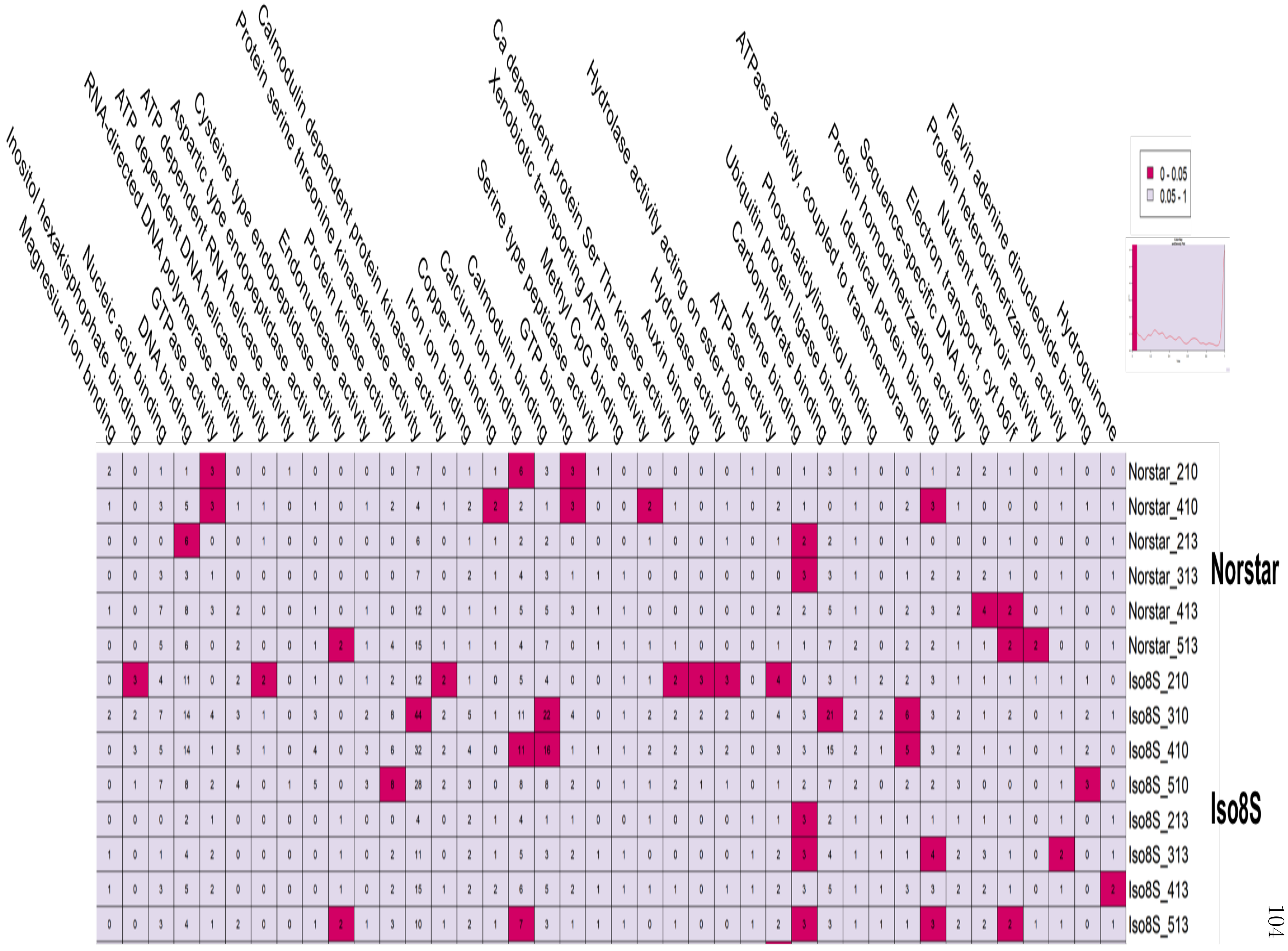


Figure 4.9 Enriched GO Molecular Function terms by top 5 scored target genes

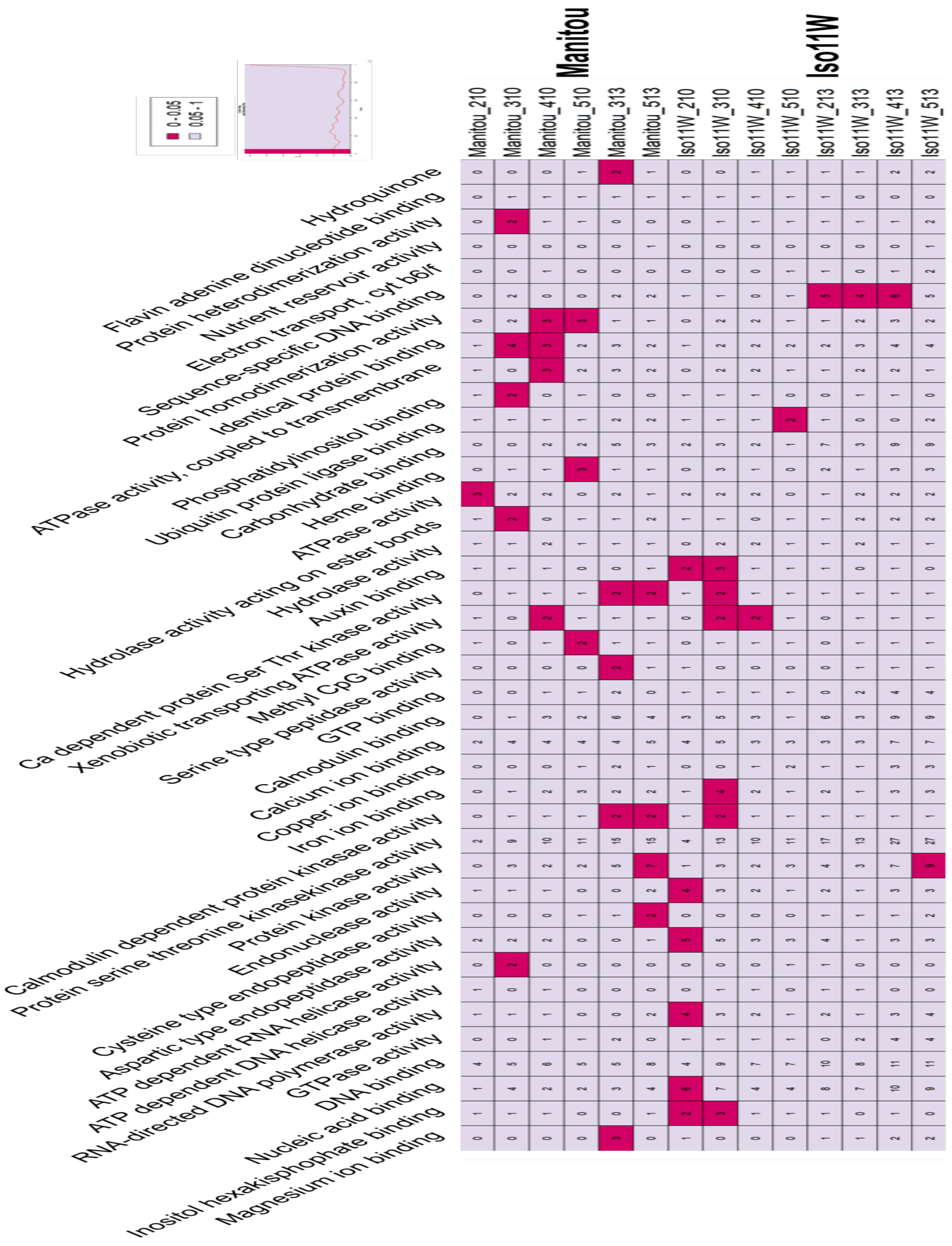
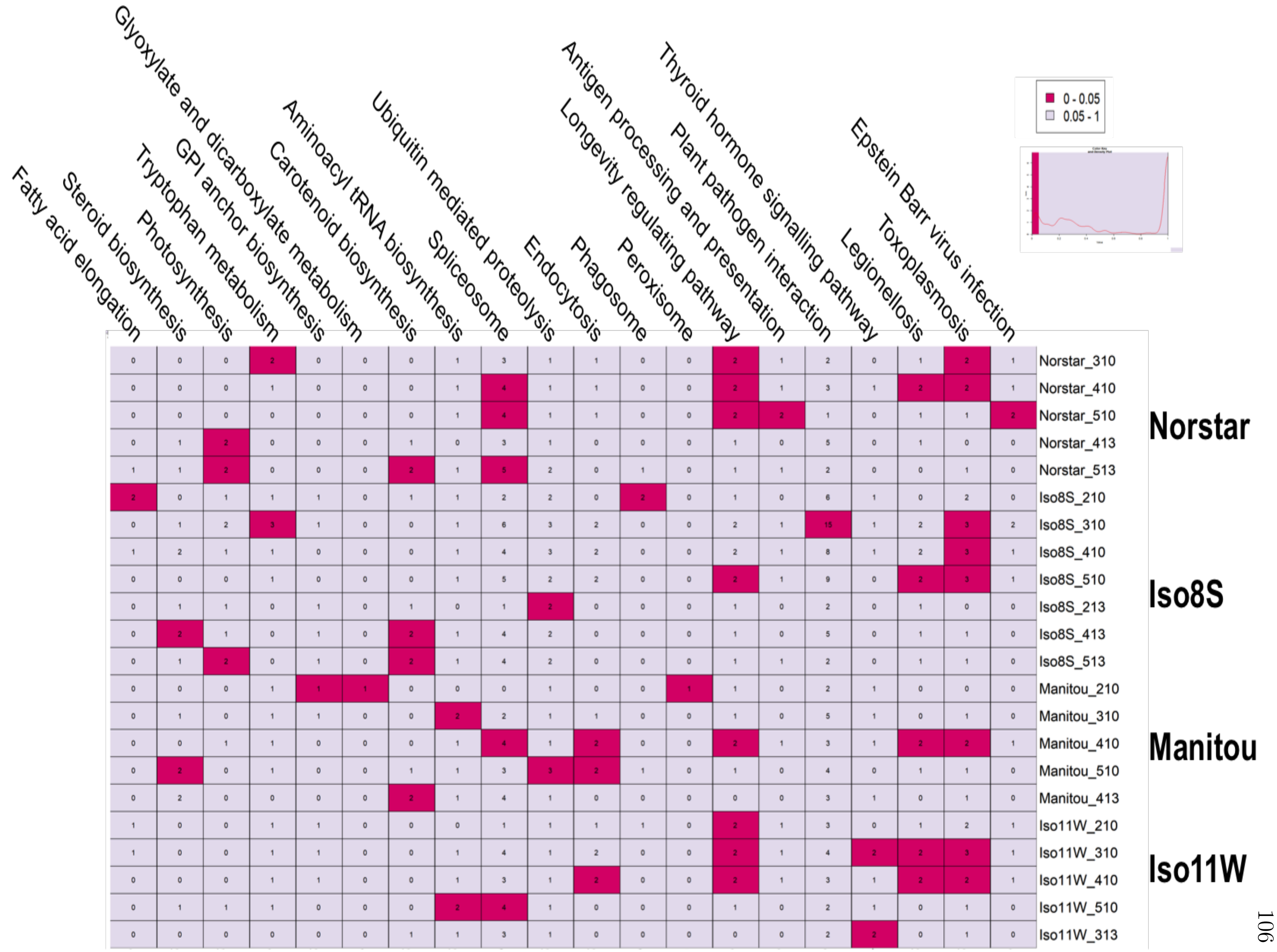


Figure 4.9 Enriched GO Molecular Function terms by top 5 scored target genes, continued



	Fatty acid elongation	Steroid biosynthesis	Photosynthesis	Tryptophan metabolism	GPI anchor biosynthesis	Glyoxylate and dicarboxylate metabolism	Carotenoid biosynthesis	Aminoacyl tRNA biosynthesis	Spliceosome	Ubiquitin mediated proteolysis	Endocytosis	Phagosome	Peroxisome	Longevity regulating pathway	Antigen processing and presentation	Plant pathogen interaction	Thyroid hormone signalling pathway	Legionellosis	Toxoplasmosis	Epstein Barr virus infection
Norstar_310	0	0	0	2	0	0	0	1	3	1	1	0	0	2	1	2	0	1	2	1
Norstar_410	0	0	0	1	0	0	0	1	4	1	1	0	0	2	1	3	1	2	2	1
Norstar_510	0	0	0	0	0	0	0	1	4	1	1	0	0	2	2	1	0	1	1	2
Norstar_413	0	1	2	0	0	0	1	0	3	1	0	0	0	1	0	5	0	1	0	0
Norstar_513	1	1	2	0	0	0	2	1	5	2	0	1	0	1	1	2	0	0	1	0
Iso8S_210	2	0	1	1	1	0	1	1	2	2	0	2	0	1	0	6	1	0	2	0
Iso8S_310	0	1	2	3	1	0	0	1	6	3	2	0	0	2	1	15	1	2	3	2
Iso8S_410	1	2	1	1	0	0	0	1	4	3	2	0	0	2	1	8	1	2	3	1
Iso8S_510	0	0	0	1	0	0	0	1	5	2	2	0	0	2	1	9	0	2	3	1
Iso8S_213	0	1	1	0	1	0	1	0	1	2	0	0	0	1	0	2	0	1	0	0
Iso8S_413	0	2	1	0	1	0	2	1	4	2	0	0	0	1	0	5	0	1	1	0
Iso8S_513	0	1	2	0	1	0	2	1	4	2	0	0	0	1	1	2	0	1	1	0
Manitou_210	0	0	0	1	1	1	0	0	0	1	0	0	1	1	0	2	1	0	0	0
Manitou_310	0	1	0	1	1	0	0	2	2	1	1	0	0	1	0	5	1	0	1	0
Manitou_410	0	0	1	1	0	0	0	1	4	1	2	0	0	2	1	3	1	2	2	1
Manitou_510	0	2	0	1	0	0	1	1	3	3	2	1	0	1	0	4	0	1	1	0
Manitou_413	0	2	0	0	0	0	2	1	4	1	0	0	0	0	0	3	1	0	1	0
Iso11W_210	1	0	0	1	1	0	0	0	1	1	1	1	0	2	1	3	0	1	2	1
Iso11W_310	1	0	0	1	1	0	0	1	4	1	2	0	0	2	1	4	2	2	3	1
Iso11W_410	0	0	0	1	1	0	0	1	3	1	2	0	0	2	1	3	1	2	2	1
Iso11W_510	0	1	1	1	0	0	0	2	4	1	0	0	0	1	0	2	1	0	1	0
Iso11W_313	0	0	0	0	0	0	1	1	3	1	0	0	0	0	0	2	2	0	1	0

Figure 4.10 Enriched KEGG pathways by top 5 scored target genes

CHAPTER V

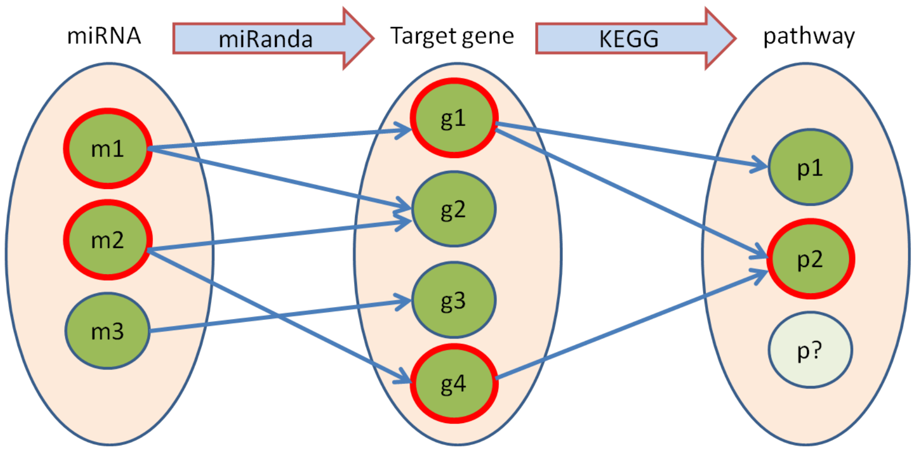
DISCUSSIONS AND CONCLUSIONS

This work has achieved the acceleration of miRNAs prediction through the implementation of mirLibSpark project. The target gene prediction module using miRanda has also been incorporated. With the benchmarking by *Arabidopsis* libraries, the project is at least 568 times faster than an automated version of than miRDeep-P. Several components of mirLibSpark pipeline have been adapted to apply in the analysis of real-world data predictions for field wheat miRNA. A total of 220 sRNAs are discovered, classified into miRNA, siRNA and false (or undecided) classes. Among them, 79 sRNAs are cold-responsive having the trend of up-, down- or mixed- (up/down) regulation throughout the time course of cold acclimation. The significantly enriched functions of these sRNAs were predicted through their precursor loci and target gene functions in GO terms and KEGG pathways. However, there are still certain limitations and important future directions to be considered.

5.1 Network enrichment graph

In the field of gene expression analysis, one of the major obstacles is to identify the true causal effect relationships from the differential expressed genes to the phenotype of interest [116, 117], in this case miRNA to cold response. It has been hypothesized that network analysis using numerous gene expression profiles enhances the discovery of true causal-effect pairs. How the functional enrichment graph help to reduce the false-positive predictions is illustrated in Figure 5.1. Say three miRNAs were differentially expressed in the 80 libraries, denoted as m_1, m_2 and m_3. By miRanda or other target gene predictors, 4 best hits of target candidates were identified, denoted as g_1, g_2 g_3 and g_4. The arrows indicate the targeting. Then through data integration, some of the target genes were annotated with KEGG pathway, denoted as p_1, p_2 and p_3. The arrows indicate the data integration. By enrichment analysis, only p_2 is enriched in certain cold environment and genetic background. Then the graph path could be traced back to the important genes involved in this pathway, g_1 and g_4, and the functional regulators of these genes, m_1 and m_2. With this selection process, m_3 could be discarded. In this study, functional networks KEGG and GO terms were used to analyze 80 trasncriptomic profiles. KEGG pathway enrichment, for example, could help to focus our attention on 66 sRNAs from all differentially expressed cold-responsive sRNAs. Due to the time constraints of this study, automation and adjustment for this retrograde enrichment analysis could not be fully carried out. One of the planned adjustment will be to separate the up- and down-regulated miRNA lists to infer to down- and up-regulated enriched functions. Therefore, future directions should include building the network enrichment graph for functional miRNA validation. In addition to KEGG and GO term, there are still other databases. For example, an application of genetic interaction network for functional miRNA validation by Vafaee et al [44] has been described in Chapter I, Section 1.4.2. Protein-protein interaction network and the precursor promoter analysis are two other potentially useful approaches for miRNA functional analysis.

Figure 5.1 Enrichment graph for true positive prediction enhancement



5.2 Statistical methods

How to select for the enriched functions and genes is a statistical quest throughout the functional miRNA analysis and has been standardized in our previous work [38] and applied in this study. The numbers of sequence reads are normalized to enable inter-library comparison. Significance of fold changes for differential gene expression is calculated based on z-score and FDR. Upper cumulative hypergeometrics analysis selects the significantly enriched functions by the target genes of the significantly differential expressed miRNA sets.

These statistical methods are essentially coupled with specific indices. For example, z-score and FDR for fold change, and the permutation test is a popular alternative method for enrichment. Not only that network enrichment graph may introduce a new index to the analysis framework, but also there are constant improvement for the in-use indices. Therefore, another future direction to enhance functional miRNA prediction could be a thorough review for the construction of indices and the usage of associated statistical methods.

5.3 Parallelization scheme

In the implementation logic of mirLibSpark, each element (RNA sequence) is distributed and processed individually. Such parallelization scheme works very well for small-genome *Arabidopsis* but not for mega-genome wheat. When each element is being processed, the genome and all supporting annotations and reference data need to be loaded to the executor node since the pipeline requires the access to the genome index for alignment and precursor extraction. Limited optimization could still be made to improve the current scheme of memory use but this approach is far from sufficient for wheat. In order to solve this problem, a new way of genome index hashing and distribution mechanism should be incorporated in the pipeline. In other words, not only the input sequences but also the genome references are partitioned and distributed.

5.4 User friendliness

Task parallelization and distribution is relatively easy to achieve by Apache Spark programming architecture as compared to python multi-threading and subprocesses at the cost of Spark dependency. Moreover, the choice to use Spark is acknowledged by its speeding-up performance [118]. Built in user-friendly Unix-like environment, current version of mirLibSpark requires users to have intermediate level of computer skills because they need to install Spark, a few bioinformatics tools and the mirLibSpark package from Git. To launch jobs in a cloud server not supported by default, users also needs to locate and load the dependencies by themselves.

Since the bioinformatics tools incorporated in mirLibSpark are all under permessive licences (MIT, BSD, Apache, LGPL) and could be distributed for academic purpose, mirLibSpark could be upgraded by including the bioinformatics tools within the package directly. Users will no longer need to install or load any third-party default dependency tool except Spark. Docker can also be considered to wrap up the package but it does not work well for the encapsulation of distributed Spark environment.

Another level of mirLibSpark upgrade is the agility of tool swapping. In the script *mirLibRules.py* each category of processing can be re-written in python inheritance abstraction. For instance, the alignment step is a class. An oriented object inherited from alignment class that executes BLAST or HISAT2 can replace the current class that executes Bowtie.

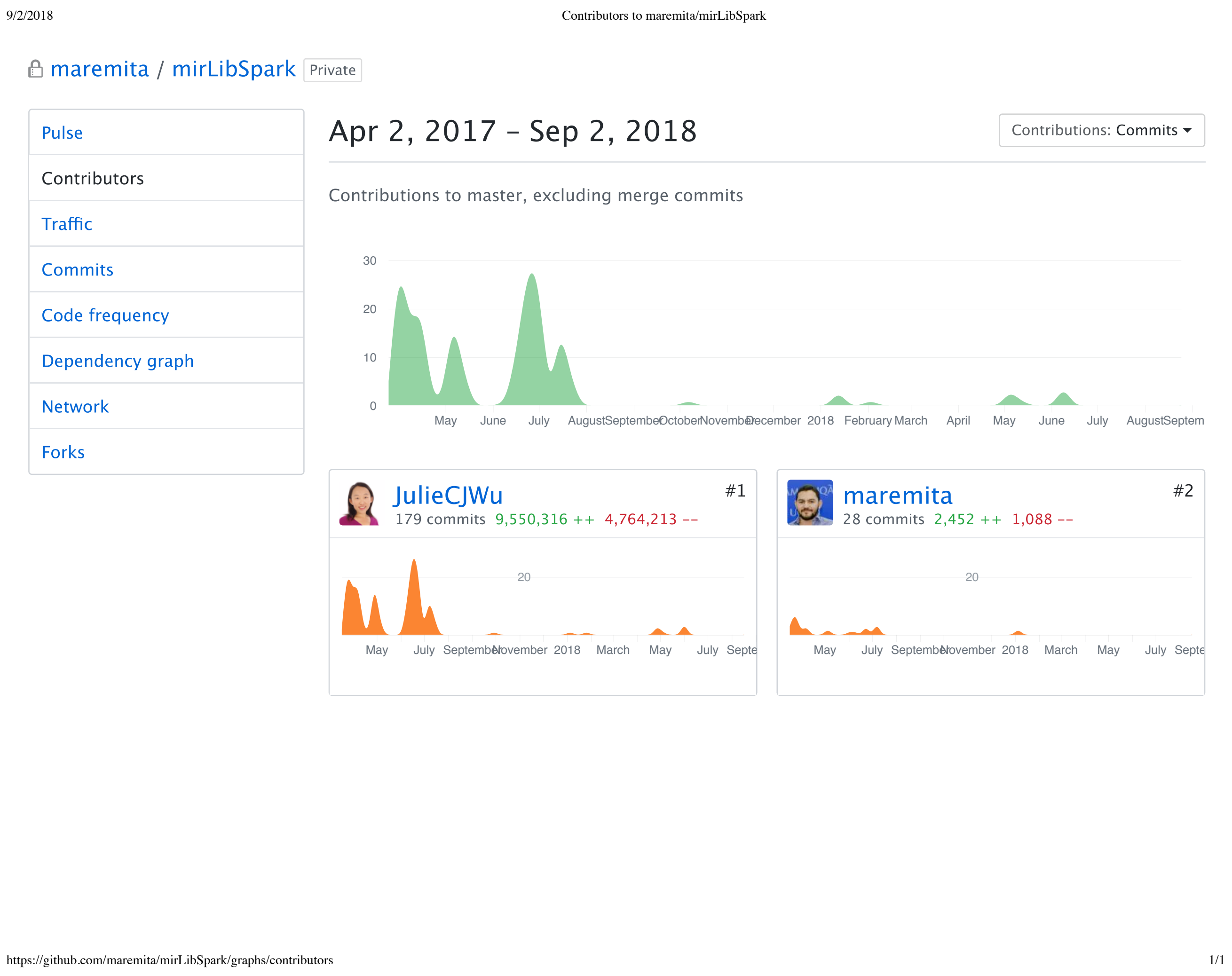
It is a standard but time-consuming procedure to predict target genes once the miRNAs are predicted. PsRNATarget is a HTML-interphase task submission platform connecting to a computing server with more than a thousand of nodes and the number of nodes is still increasing [99]. As long as the annotated reference genome of interest is supported by the platform, users simply fill out the web submission form to launch the task. If we have our own privileged server, it can be considered to deliver mirLibSpark as HTML submissions in our web server.

Finally, the mirLibSpark project is available in Dropbox and GitHub. The detail annotations of all predicted wheat small RNAs, including the predicted target genes of each miRNA, precursor sequences and loci, GO terms, KEGG pathways, field transcriptome expression reads and XLOC annotations, can also be found in Dropbox. Web addresses are listed in Appendix for availability.

APPENDIX A

SOURCE CODES

A.1 Contributors to mirLibSpark



9/2/2018 Contributors to maremita/mirLibSpark

maremita / mirLibSpark Private

Apr 2, 2017 – Sep 2, 2018

Contributions: Commits

Contributions to master, excluding merge commits

JulieCJWu #1
179 commits 9,550,316 ++ 4,764,213 --

maremita #2
28 commits 2,452 ++ 1,088 --

https://github.com/maremita/mirLibSpark/graphs/contributors 1/1

A.2 Structure of the mirLibSpark project

Figure A.1 Structure of the mirLibSpark project in Linux tree view

```
mirLibSpark
├── .gitattributes
├── .gitignore
├── LICENSE(.txt)
├── README.md
├── paramfile.txt
├── dbs
│   ├── ATH
│   │   └── Genome
│   │       ├── TAIR10_chr1.fas
│   │       ├── ***
│   │       └── TAIR10_chrM.fas
│   ├── TAIR10_cdna_20101214_updated.fasta
│   ├── TAIR10_ncRNA_CDS.gff
│   ├── ath_miRBase21.gff3
│   └── bowtie_index
│       ├── a_thaliana_t10.1.ebwt
│       ├── ***
│       └── a_thaliana_t10.rev.2.ebwt
├── input_samples
│   ├── 100.txt
│   ├── high_conf_mature_ath_uniq_raw.txt
│   ├── mirbase_all_349_real_ath_uniq.txt
│   └── neg_ath1000.txt
├── lib
│   ├── miRcheck.pm
│   ├── miRdup_1.4
│   │   ├── lib
│   │   │   └── weka.jar
│   │   ├── miRdup.jar
│   │   ├── model
│   │   │   ├── Viridiplantae.model
│   │   │   └── thaliana.model
│   │   └── src
│   │       ├── miRdup
│   └── miranda
├── mirdeep_p
│   ├── miRDP1.3
│   │   ├── alignedselected.pl
│   │   ├── compare_annotated_retrieved.pl
│   │   ├── convert_SAM_to_blast.pl
│   │   ├── convert_bowtie_to_blast.pl
│   │   ├── excise_candidate.pl
│   │   ├── fetch_extended_precursors.pl
│   │   ├── filter_alignments.pl
│   │   ├── miRDP.pl
│   │   ├── overlap.pl
│   │   └── rm_redundant_meet_plant.pl
│   ├── run_mirdeep_P.py
│   ├── tair10
│   │   ├── TAIR10_GFF3_genes.gff
│   │   ├── annotated_miRNA_v21_extended.fa
│   │   ├── ath.gff3
│   │   ├── ath.gff3.edited
│   │   ├── chromosome_length
│   │   └── ncRNA_CDS.gff
│   └── tair9
│       └── ***
├── src
│   ├── eval_mircheck.pl
│   ├── mirLibPipeline.py
│   ├── mirLibRules.py
│   ├── neg.py
│   ├── run_miranda.py
│   ├── sequential.py
│   └── utils.py
└── workdir
    └── submit_j.pbs
```

*NOTES: *** = not shown to save page space*

A.3 paramfile.txt

```
#=the char leading message is essential. #message is for users to print a short message in the report to identify this experiment. Support \n new line.
#message=write your message here line1\nline2
#==NOTE that project_path and genome_path have to be absolute paths while the input_path, output_path and b_index_path can be relative paths to the project_path.
project_path=/path/to/mirLibHadoop/
genome_path=/path/to/ATH/Genome/
input_path=/path/to/mirLibHadoop/input/
output_path=/path/to/mirLibHadoop/output/
b_index_path=/path/to/mirLibHadoop/dbs/bowtie_index/a_thaliana_t10
#== input_types: a = raw (seq\tfreq); b = reads (each line may be a redundant sequence); c = fasta (>name\nseq); d = fastq
input_type=a
adapter=none
limit_s_freq=10
limit_m_freq=100
limit_len=18
limit_nbLoc=15
pri_l_flank=500
pri_r_flank=200
pre_flank=30
mirdup_model=thaliana.model
mirdup_limit=0.98
mcheck_param=def
Max_Score_cutoff=170
Max_Energy_cutoff=-15
Gap_Penalty=-15
target_file=TAIR10_cdna_20101214_updated.fasta
#=================== Spark configuration
sc_appname=mirLibHadoop
sc_partition=12
#=================== local mode
sc_master=local[*]
sc_mstrmemory=20g
sc_execmemory=20g
#=================== cluster mode
sc_execcores=24
```

A.4 submit_j.pbs

```
#!/bin/bash
#PBS -N testMirlibHadoop
#PBS -l nodes=3:ppn=8
#PBS -l walltime=00:10:00
#PBS -A hvg-164-aa
#PBS -o jobsOut/$MOAB_JOBID.out
#PBS -e jobsOut/$MOAB_JOBID.err
#PBS -E

cd "${PBS_O_WORKDIR}"

module load apps/spark/2.0.0

module load compilers/gcc/4.8.5
module load compilers/java/1.8
# This line is required only if we use cutadapt (this pipeline works with 2.7, tested)
module load apps/python/2.7.5
#
module load apps/bowtie/1.1.0
module load apps/mugqic_pipeline/2.1.1
module load mugqic/blast/2.2.31+
module load mugqic/ViennaRNA/2.1.8
#mugqic/ViennaRNA/2.3.5

# Launch Spark cluster
start-all.sh

# Submit SparkPi.jar application to the Spark cluster
spark-submit --master spark://$HOSTNAME:7077 ../src/mirLibPipeline.py ../paramfile.txt
# Stop Spark cluster
stop-all.sh

```

A.5 mirLibPipeline.py

```
'''
program: mirLibPipeline.py
author: M.A.Remita
author: Chao-Jung Wu
date: 2017-03-28
version: 1.01.00

Le programme implemente le pipeline d'analyse des sRAN et prediction des miRNAs. Les principales etapes de predictions sont :
  - 1 Filtrage
  - 2 Alignement
  - 3 Extraction du precurseur
  - 4 Repliement du precurseur
  - 5 prediction/validation du miRNA
  - 6 validaiton/filtrage avec l'expression
'''

from __future__ import print_function
import sys
import os.path
import time
from os import listdir
#
import utils as ut
import mirLibRules as mru


if __name__ == '__main__' :

  if not len(sys.argv) == 2:
    sys.stderr.write('Three arguments required\nUsage: spark-submit mirLibPipeline.py <path to paramfile> 2>/dev/null\n')
    sys.exit()

  paramfile = sys.argv[1]
  paramDict = ut.readParam (paramfile)

  #message = paramDict['message'].split()
  #for i in message: print(i)
  #= Parameters and cutoffs
  input_type = paramDict['input_type']
  adapter = ut.tr_U_T (paramDict['adapter'])
  project_path = paramDict['project_path'][:-1]
  rep_input = paramDict['input_path']
  rep_output = paramDict['output_path']
  rep_msub_jobsOut = project_path + '/workdir/jobsOut'
  #my_sep = paramDict['my_sep']                    # Separator
  rep_tmp = project_path + '/tmp/'                 # tmp file folder

  #= spark configuration
  appMaster = paramDict['sc_master']              #"local"
  appName = paramDict['sc_appname']               #"mirLibHadoop"
  mstrMemory = paramDict['sc_mstrmemory']         #"4g"
  execMemory = paramDict['sc_execmemory']         #"4g"
  execCores = paramDict['sc_execcores']           #2
  partition = int(paramDict['sc_partition'])

  #= genome
  genome_path = paramDict['genome_path']

  #= cutoffs
  limit_srna_freq = int(paramDict['limit_s_freq'])  #10     # exclude sRNA freq < limit_srna_freq
  limit_mrna_freq = int(paramDict['limit_m_freq'])  #200    # exclude miRNA freq < limit_mrna_freq
  limit_len = int(paramDict['limit_len'])           #18     # exclude RNA length < limit_len
  limit_nbLoc = int(paramDict['limit_nbLoc'])      #3       # exculde nbLoc mapped
  with bowtie  > limit_nbLoc

  #= bowtie
  b_index_path = paramDict['b_index_path']

  #= file and list of known non miRNA
  #known_non = '../dbs/TAIR10_ncRNA_CDS.gff'
  known_non = project_path + '/dbs/TAIR10_ncRNA_CDS.gff'###########################
  d_ncRNA_CDS = ut.get_nonMirna_coors (known_non) #= nb = 198736

  #= pri-mirna
  pri_l_flank = int(paramDict['pri_l_flank'])      #120
  pri_r_flank = int(paramDict['pri_r_flank'])      #60
  pre_flank = int(paramDict['pre_flank'])          #30

  #= mircheck parameter
  mcheck_param = paramDict['mcheck_param']         #'def'    # def : default
  parameters / mey : meyers parameters

  #= miRdup parameter
  path_RNAfold = ut.find_RNAfold_path ()
  mirdup_model = project_path + '/lib/miRdup_1.4/model/' + paramDict['mirdup_model']
  mirdup_jar = project_path + '/lib/miRdup_1.4/miRdup.jar'
  mirdup_limit =  float(paramDict['mirdup_limit'])

  #= miRanda parameter
  target_file = project_path + '/dbs/' + paramDict['target_file']
  miranda_exe = project_path + '/lib/miranda'
  Max_Score_cutoff = paramDict['Max_Score_cutoff'] #= need string or buffer
  Max_Energy_cutoff = paramDict['Max_Energy_cutoff'] #= NOT WORKING YET
  Gap_Penalty = paramDict['Gap_Penalty']

  ## EXMAMIN OPTIONS ###################################
  ut.validate_options(paramDict)
  ####################################################

  #= make required folders if not exist
  reps = [rep_output, rep_tmp, rep_msub_jobsOut]
  ut.makedirs_reps (reps)

  #= Spark context
  sc = ut.pyspark_configuration(appMaster, appName, mstrMemory, execMemory, execCores)
  #
  sc.addFile(known_non)################################
  sc.addPyFile(project_path + '/src/utils.py')
  sc.addPyFile(project_path + '/src/mirLibRules.py')
  sc.addFile(project_path + '/src/eval_mircheck.pl')
  sc.addFile(project_path + '/lib/miRcheck.pm')
  sc.addFile(project_path + '/lib/miRdup_1.4/lib/weka.jar')
  sc.addFile(mirdup_jar)
  sc.addFile(mirdup_model)
  sc.addFile(target_file)
  sc.addFile(miranda_exe)

  #= Spark application ID
  appId = str(sc.applicationId)

  #= Objects for rule functions
  dmask_obj = mru.prog_dustmasker()
  dmask_cmd, dmask_env = dmask_obj.dmask_pipe_cmd()
  bowtie_obj = mru.prog_bowtie(b_index_path)
  kn_obj = mru.prog_knownNonMiRNA(d_ncRNA_CDS)
  bowtie_cmd, bowtie_env = bowtie_obj.Bowtie_pipe_cmd()
  prec_obj = mru.extract_precurosrs(genome_path, pri_l_flank, pri_r_flank, pre_flank)
  rnafold_obj = mru.prog_RNAfold()
  mircheck_obj = mru.prog_mirCheck(mcheck_param)
  mirdup_obj = mru.prog_miRdup (rep_tmp, mirdup_model, mirdup_jar, path_RNAfold)
  profile_obj = mru.prog_dominant_profile()
  #
  miranda_obj = mru.prog_miRanda(Max_Score_cutoff, Max_Energy_cutoff, target_file, rep_tmp, miranda_exe, Gap_Penalty)

  #= Fetch library files in rep_input
  infiles = [f for f in listdir(rep_input) if os.path.isfile(os.path.join(rep_input, f))]
  
  #= Time processing of libraries
  timeDict = {}
  
  
  for infile in infiles :
    if infile[-1:] == '~': continue
    print ("--Processing of the library: ", infile)

    inBasename = os.path.splitext(infile)[0]
    infile = rep_input+infile
    inKvfile = rep_tmp + inBasename + '.kv.txt'

    # hdfsFile = inBasename + '.hkv.txt'

    if input_type == 'd': #= fastq
      ut.convert_fastq_file_to_KeyValue(infile, inKvfile)
      infile = inKvfile
      
    #
    print ("  Start of the processing...", end="\n")
    startLib = time.time()
    
    #= Convert the text file to RDD object
    ## in : file
    ## out: (a) u'seq\tfreq', 
    ##      (b) u'seq1', u'seq2', u'seq1', 
    ##      (c) u'>name1\nseq1', u'>name2\nseq2', u'>name3\nseq1',
    ##      (d) u'seq\tquality'
    distFile_rdd = sc.textFile("file:///" + infile, partition) #= NumPartitions = 4 (default for 100.txt was 2)
    #print('NB distFile_rdd: ', len(distFile_rdd.collect()))#

    #= Unify different input formats to "seq freq" elements

    if input_type == 'a': #= raw
    ## in : u'seq\tfreq'
    ## out: ('seq', freq)
      ## note that type_a does not need to collapse nor trim.
      collapse_rdd = distFile_rdd.map(lambda line: mru.rearrange_rule(line, '\t'))#\
                             #.distinct() ##= .distinct() might not be necessary
    else:
      if input_type == 'b': #= reads
      ## in : u'seq1', u'seq2', u'seq1'
      ## out: u'seq1', u'seq2', u'seq1'
        input_rdd = distFile_rdd

      elif input_type == 'c': #= fasta
      ## in : u'>name1\nseq1', u'>name2\nseq2', u'>name3\nseq1'
      ## out: u'seq1', u'seq2', u'seq1'
        input_rdd = distFile_rdd.filter(lambda line: not line[0] == '>')

      elif input_type == 'd': #= processed fastq
      ## in : u'seq\tquality'
      ## out: u'seq1', u'seq2', u'seq1'
        input_rdd = distFile_rdd.map(lambda word: word.split('\t')[0])

    #= trim adapters
      if not adapter == 'none':
        trim_adapter_rdd = input_rdd.map(lambda e: trim_adapter (e, adapter))
      else: trim_adapter_rdd = input_rdd
      
    #= colapse seq and calculate frequency
      ## in : u'seq1', u'seq2', u'seq1'
      ## out: ('seq', freq)
      collapse_rdd = trim_adapter_rdd.map(lambda word: (word, 1)).reduceByKey(lambda a, b: a+b)
  
    #'''
   
    #= Filtering sRNA low frequency
    ## in : ('seq', freq)
    ## out: ('seq', freq)
    sr_low_rdd = collapse_rdd.filter(lambda e: int(e[1]) > limit_srna_freq) #.persist()#########################
    #print('NB sr_low_rdd: ', len(sr_low_rdd.collect()))##################################################
    
    #= Filtering short length
    ## in : ('seq', freq)
    ## out: ('seq', freq)
    sr_short_rdd = sr_low_rdd.filter(lambda e: len(e[0]) > limit_len).persist()  # TO KEEP IT
    #print('NB sr_short_rdd: ', len(sr_short_rdd.collect()))###################################################
    
    #= Filtering with DustMasker
    ## in : ('seq', freq)
    ## out: 'seq'
    dmask_rdd = sr_short_rdd.map(lambda e: '>s\n'+e[0])\
                            .pipe(dmask_cmd, dmask_env)\
                            .filter(lambda e: e.isupper() and not e.startswith('>'))\
                            .map(lambda e: str(e.rstrip()))\
                            .persist()
    #print('NB dmask_rdd: ', len(dmask_rdd.collect()))############################################
    
    #= Mapping with Bowtie
    ## in : 'seq'
    ## out: ('seq', [nbLoc, [['strd','chr',posChr],..]])
    bowtie_rdd = dmask_rdd.map(lambda e: " "+e)\
                          .pipe(bowtie_cmd, bowtie_env)\
                          .map(bowtie_obj.bowtie_rearrange_map)\
                          .groupByKey()\
                          .map(lambda e: (e[0], [len(list(e[1])), list(e[1])]))\
                          .persist()
    #print('NB bowtie_rdd: ', len(bowtie_rdd.collect()))################################################

    #= Getting the expression value for each reads
    ## in : ('seq', [nbLoc, [['strd','chr',posChr],..]])
    ## out: ('seq', [freq, nbLoc, [['strd','chr',posChr],..]])
    bowFrq_rdd = bowtie_rdd.join(sr_short_rdd)\
                           .map(bowtie_obj.bowtie_freq_rearrange_rule)\
                           .persist()
    #print('NB bowFrq_rdd: ', len(bowFrq_rdd.collect()))################################################
    
    #= Filtering miRNA low frequency
    ## in : ('seq', [freq, nbLoc, [['strd','chr',posChr],..]])
    ## out: ('seq', [freq, nbLoc, [['strd','chr',posChr],..]])
    mr_low_rdd = bowFrq_rdd.filter(lambda e: e[1][0] > limit_mrna_freq) #.persist()########################
    #print('NB mr_low_rdd: ', len(mr_low_rdd.collect()))################################################
    
    #= Filtering high nbLocations and zero location
    ## in : ('seq', [freq, nbLoc, [['strd','chr',posChr],..]])
    ## out: ('seq', [freq, nbLoc, [['strd','chr',posChr],..]])
    nbLoc_rdd = mr_low_rdd.filter(lambda e: e[1][1] > 0 and e[1][1] < limit_nbLoc)
    #.persist()############
    #print('NB nbLoc_rdd: ', 
    len(nbLoc_rdd.collect()))##################################################
    
    #= Flatmap the RDD
    ## in : ('seq', [freq, nbLoc, [['strd','chr',posChr],..]])
    ## out: ('seq', [freq, nbLoc, ['strd','chr',posChr])
    flat_rdd = nbLoc_rdd.flatMap(mru.flatmap_mappings)#.persist() ###
    #print('NB flat_rdd distinct (this step flats elements): ', 
    len(flat_rdd.groupByKey().collect()))#################
    #print('NB flat_rdd not distinct: ', len(flat_rdd.collect()))#################

    ###############################
    ## Filtering known non-miRNA ##
    ###############################
    ## in : ('seq', [freq, nbLoc, ['strd','chr',posChr])
    ## out: ('seq', [freq, nbLoc, ['strd','chr',posChr])
    ##excluKnownNon_rdd = flat_rdd.filter(kn_obj.knFilterBySeq) #= defunct
    #excluKnownNon_rdd = 
    flat_rdd.repartition(100).filter(kn_obj.knFilterByCoor)#.persist()#######
    excluKnownNon_rdd = flat_rdd.filter(kn_obj.knFilterByCoor)#.persist()#######
    #print('excluKnownNon_rdd distinct: ', 
    len(excluKnownNon_rdd.groupByKey().collect()))########
    
    #= Extraction of the pri-miRNA
    ## in : ('seq', [freq, nbLoc, ['strd','chr',posChr])
    ## out: ('seq', [freq, nbLoc, ['strd','chr',posChr], ['priSeq',posMirPri]])
    primir_rdd = excluKnownNon_rdd.flatMap(prec_obj.extract_prim_rule)

    #= pri-miRNA folding
    ## in : ('seq', [freq, nbLoc, ['strd','chr',posChr], ['priSeq',posMirPri]])
    ## out: ('seq', [freq, nbLoc, ['strd','chr',posChr], ['priSeq',posMirPri,'priFold']])
    pri_fold_rdd = primir_rdd.map(lambda e: rnafold_obj.RNAfold_map_rule(e, 3))
    
    #= Validating pri-mirna with mircheck
    ## in : ('seq', [freq, nbLoc, ['strd','chr',posChr], ['priSeq',posMirPri,'priFold']])
    ## out: ('seq', [freq, nbLoc, ['strd','chr',posChr], 
    ['priSeq',posMirPri,'priFold','mkPred','mkStart','mkStop']])
    pri_vld_rdd = pri_fold_rdd.map(lambda e: mircheck_obj.mirCheck_map_rule(e, 3))\
                              .filter(lambda e: any(e[1][3]))#.persist()##
    #print('NB pri_vld_rdd distinct (mircheck): ', 
    len(pri_vld_rdd.groupByKey().collect()))##

    #= Filtering structure with branched loop
    ## in : ('seq', [freq, nbLoc, ['strd','chr',posChr], 
    ['priSeq',posMirPri,'priFold','mkPred','mkStart','mkStop']])
    ## out: ('seq', [freq, nbLoc, ['strd','chr',posChr], 
    ['priSeq',posMirPri,'priFold','mkPred','mkStart','mkStop']])
    one_loop_rdd = pri_vld_rdd.filter(lambda e: ut.containsOnlyOneLoop(e[1][3][2][int(e[
    1][3][4]) : int(e[1][3][5])+1]))#.persist()##
    #print('NB one_loop_rdd distinct : ', len(one_loop_rdd.groupByKey().collect()))##
    
    #= Extraction of the pre-miRNA
    ## in : ('seq', [freq, nbLoc, ['strd','chr',posChr], 
    ['priSeq',posMirPri,'priFold','mkPred','mkStart','mkStop']])
    ## out: ('seq', [freq, nbLoc, ['strd','chr',posChr], 
    ['priSeq',posMirPri,'priFold','mkPred','mkStart','mkStop'], ['preSeq',posMirPre]])
    premir_rdd = one_loop_rdd.map(lambda e: prec_obj.extract_prem_rule(e, 3)) ## use 
    one-loop rule
    #premir_rdd = pri_vld_rdd.map(lambda e: prec_obj.extract_prem_rule(e, 3)) ## ignore 
    one-loop rule
    
    #= pre-miRNA folding
    ## in : ('seq', [freq, nbLoc, ['strd','chr',posChr], ['priSeq',posMirPri,'priFold', 
    'mkPred','mkStart','mkStop'], ['preSeq',posMirPre]])
    ## out: ('seq', [freq, nbLoc, ['strd','chr',posChr], ['priSeq',posMirPri,'priFold', 
    'mkPred','mkStart','mkStop'], ['preSeq',posMirPre,'preFold']])
    pre_fold_rdd = premir_rdd.map(lambda e: rnafold_obj.RNAfold_map_rule(e, 4))

    #= Validating pre-mirna with mircheck II -- replaced by mirdup
    ## in : ('seq', [freq, nbLoc, ['strd','chr',posChr], ['priSeq',posMirPri,'priFold', 
    'mkPred','mkStart','mkStop'], ['preSeq',posMirPre,'preFold']])
    ## out: ('seq', [freq, nbLoc, ['strd','chr',posChr], ['priSeq',posMirPri,'priFold', 
    'mkPred','mkStart','mkStop'], 
    ['preSeq',posMirPre,'preFold','mkPred','mkStart','mkStop']])
    #pre_vld_rdd0 = pre_fold_rdd.map(lambda e: mircheck_obj.mirCheck_map_rule(e, 4))\
                            #.filter(lambda e: any(e[1][4])).persist()
    #print('NB pre_vld_rdd0 distinct (mircheck II): (', 
    len(pre_vld_rdd0.groupByKey().collect()), ')')



    #= Validating pre-mirna with miRdup zipWithUniqueId
    ## in : ('seq', [freq, nbLoc, ['strd','chr',posChr], ['priSeq',posMirPri,'priFold', 
    'mkPred','mkStart','mkStop'], ['preSeq',posMirPre,'preFold']])
    ## out: ('seq', [freq, nbLoc, ['strd','chr',posChr], ['priSeq',posMirPri,'priFold', 
    'mkPred','mkStart','mkStop'], ['preSeq',posMirPre,'preFold','mpPred','mpScore']])
    pre_vld_rdd = pre_fold_rdd.zipWithIndex()\
                            .map(mirdup_obj.run_miRdup)\
                            .filter(lambda e: e[1][4][3] == "true")#.persist()##
    #print('NB pre_vld_rdd distinct (mirdup): ', 
    len(pre_vld_rdd.groupByKey().collect()))##

    #= Create dict, chromo_strand as key to search bowtie blocs in the following dict
    dict_bowtie_chromo_strand = profile_obj.get_bowtie_strandchromo_dict(bowFrq_rdd.
    collect())
    broadcastVar = sc.broadcast(dict_bowtie_chromo_strand) #= get the value by 
    broadcastVar.value
    
    #= Filtering by expression profile (< 20%)
    ## in : ('seq', [freq, nbLoc, ['strd','chr',posChr], ['priSeq',posMirPri,'priFold', 
    'mkPred','mkStart','mkStop'], ['preSeq',posMirPre,'preFold','mpPred','mpScore']])
    ## out: ('seq', [freq, nbLoc, ['strd','chr',posChr], ['priSeq',posMirPri,'priFold', 
    'mkPred','mkStart','mkStop'], ['preSeq',posMirPre,'preFold','mpPred','mpScore'], 
    totalfrq])
    profile_rdd = pre_vld_rdd.map(lambda e: profile_obj.computeProfileFrq(e, 
    broadcastVar.value))\
                        .filter(lambda e: e[1][0] / float(e[1][5]) > 0.2).persist()##

    print('NB profile_rdd distinct: ', len(profile_rdd.groupByKey().collect()))##
    results = profile_rdd.collect()


    #= target prediction
    miranda_rdd = profile_rdd.map(miranda_obj.computeTargetbyMiranda).persist()####
    print('NB miranda_rdd distinct : ', len(miranda_rdd.groupByKey().collect()))####
    results = miranda_rdd.collect()
    
    endLib = time.time() 
    print ("  End of the processing     ", end="\n")
    
    
    #= write results to a file
    outFile = rep_output  +  appId + '_miRNAprediction_' + inBasename + '.txt'
    ut.writeToFile (results, outFile)
    
    timeDict[inBasename] = endLib - startLib


  sc.stop() #= allow to run multiple SparkContexts
```

A.6 mirLibRules.py

```
'''
program: mirLibRules.py
author: M.A.Remita
author: Chao-Jung Wu
date: 2017-03-28
version: 1.00.02

Le programme

'''

import subprocess as sbp
import os
import utils as ut
from operator import itemgetter


def rearrange_rule(kv_arg, kv_sep):
  tab = kv_arg.split(kv_sep)
  return (str(tab[0]), int(tab[1]))

def rearrange_sam_rule(line):
  data = line.rstrip('\n').split('\t')
  chromo = data[2]
  posChr = int(data[3]) - 1 #= because Dr Diallo's file is not zero based
  strand = data[4]
  if strand == '1': strand = '+'
  else: strand = '-'
  seq = data[9]
  return (str(seq), [500, 1, [strand, chromo, posChr]])

def flatmap_mappings(elem) :
  '''
  ## in : ('seq', [freq, nbLoc, [['strd','chr',posChr],..]])
  ## out: ('seq', [freq, nbLoc, ['strd','chr',posChr])
  '''
  newElems = []
  
  for mapping in elem[1][2]:
    newElem = (elem[0], [elem[1][0], elem[1][1], mapping])
    newElems.append(newElem)
  
  return newElems

class prog_dustmasker ():

  def __init__(self):
    #= The object has to be initialized in the driver program
    #= to permit the capture of its env variables and pass them
    #= to the subprocess in the worker nodes
    self.env = os.environ

  def dmask_filter_rule(self, elem):
    sRNAseq = str(elem[0])
    line1 = ['echo', '>seqMir\n' + sRNAseq]
    line2 = ['dustmasker']
    
    p1 = sbp.Popen(line1, stdout=sbp.PIPE, env=self.env)
    p2 = sbp.Popen(line2, stdin=p1.stdout, stdout=sbp.PIPE, env=self.env)
    p1.stdout.close()  #= Allow p1 to receive a SIGPIPE if p2 exits.
    
    output = p2.communicate()[0].rstrip('\n')
    nblines = len(output.split('\n'))
    if nblines == 1:
        return True
    return False  ##= false data will be automatically excluded in the new RDD

  def dmask_pipe_cmd(self):
    return 'dustmasker -outfmt fasta', self.env

class prog_bowtie ():

  def __init__(self, b_index):
    self.bowtie_index = b_index
    
    #= The object has to be initialized in the driver program
    #= to permit the capture of its env variables and pass them
    #= to the subprocess in the worker nodes
    self.env = os.environ

  def run_bowtie(self, seq):
    mappings = []
    FNULL = open(os.devnull, 'w')
    
    # cmd = 'bowtie --mm -a -v 0 --suppress 1,5,6,7,8 -c ' + self.bowtie_index + ' '+
    seq  # shell=True
    cmd = ['bowtie', '--mm', '-a', '-v', '0', '--suppress', '1,5,6,7,8', '-c', self.
    bowtie_index, seq] # shell=False
    
    sproc = sbp.Popen(cmd, stdout=sbp.PIPE, stderr=FNULL, shell=False, env=self.env)
    bsout = sproc.communicate()[0]
    bwout = bsout.rstrip('\n')
    
    FNULL.close()
    
    if bwout :
      bwList = bwout.split('\n')
      for line in bwList:
        map_value = line.rstrip('\n').split('\t')
        map_value[0] = str(map_value[0])
        map_value[1] = str(map_value[1])
        map_value[2] = int(map_value[2])
        mappings += [map_value]
      
    return mappings

  def Bowtie_map_rule(self, elem):
    sRNAseq = str(elem[1][0])
    append_value = self.run_bowtie(sRNAseq)
    elem[1].append(len(append_value))
    elem[1].append(append_value)
    return elem
  
  def Bowtie_pipe_cmd (self):
    cmd = "bowtie --mm -a -v 0 --suppress 1,6,7,8 -r " + self.bowtie_index + " - "
    # cmd = "bowtie --mm -a -v 0 --suppress 1,6,7,8 " + self.bowtie_index + " - "
    
    return cmd, self.env
  
  def bowtie_rearrange_map (self, elem):
    #= u'+\tChr3\t7922370\tAAATGTAAACATCTGATCGTTTGA'
    elemTab = map(str, elem.split("\t"))
    
    #= if negative strand, get the reverse complement
    if elemTab[0] == "-" :
      elemTab[3] = ut.getRevComp(elemTab[3])
      
    return (elemTab[3], [elemTab[0], elemTab[1], int(elemTab[2])])
    
  def bowtie_freq_rearrange_rule(self, elem):
    '''
    in : ('seq',([nbLoc, [['strd','chr',posChr],..]], freq))
    out: ('seq', [freq, nbLoc, [['strd','chr',posChr],..]])
    '''
    return (elem[0], [elem[1][1], elem[1][0][0], elem[1][0][1]])

class extract_precurosrs ():

  def __init__(self, genome_path, ext_left, ext_right, pre_flank):
    self.genome_path = genome_path
    self.ext_left = ext_left
    self.ext_right = ext_right
    self.pre_flank = pre_flank
    #
    self.genome = ut.getGenome(genome_path, ".fas")

  def extract_precursors (self, contig, strand, start_srna, len_srna):
    prims = []
    ext_left = self.ext_left
    ext_right = self.ext_right
    
    #= all positions are zero-based
    s1 = start_srna - ext_left
    pos_5p = ext_left
    if s1 < 0:
      s1 = 0
      pos_5p = start_srna
      ext_left = start_srna
    
    t_len1 = len_srna + ext_left + ext_right
    
    s2 = start_srna - ext_right
    pos_3p = ext_right
    if s2 < 0:
      s2 = 0
      pos_3p = start_srna
      ext_right = start_srna
      
    t_len2 = len_srna + ext_left + ext_right
    
    seq_5p = contig[s1:(s1+t_len1)]
    seq_3p = contig[s2:(s2+t_len2)]
    
    if strand == "-":
      seq_5p = ut.getRevComp(seq_5p);
      pos_5p = len(seq_5p) - pos_5p - len_srna
      
      seq_3p = ut.getRevComp(seq_3p);
      pos_3p = len(seq_3p) - pos_3p - len_srna
    
    if seq_5p == seq_3p :
      prims = [[seq_5p, pos_5p]]
    else :
      prims = [[seq_5p, pos_5p], [seq_3p, pos_3p]]
      
    return prims

  def extract_prim_rule(self, elem):
    '''
    ## in : ('seq', [freq, nbLoc, ['strd','chr',posChr])
    ## out: ('seq', [freq, nbLoc, ['strd','chr',posChr], ['priSeq',posMirPri]])
    '''
    newElems = []
    len_srna = len(elem[0])
    
    mapping = elem[1][2]
    contig = self.genome[mapping[1]]
    prims = self.extract_precursors(contig, mapping[0], mapping[2], len_srna)
      
    for prim in prims :
      newElem = (elem[0], [elem[1][0], elem[1][1], mapping, prim])
```

```
            newElems.append(newElem)
        
        return newElems
    
    def extract_sub_seq(self, contig, posMir, fback_start, fback_stop):
        fold_len = fback_stop - fback_start + 1
        pos = posMir - fback_start + self.pre_flank          #= 0-based
        deb = fback_start - self.pre_flank; 
        
        if deb < 0 :
            deb = 0
            pos = posMir
        
        fin = fback_stop + self.pre_flank + 1
        newSeq = contig[deb:fin]
        
        return [newSeq, pos]
    
    def extract_prem_rule(self, elem, field):
        '''
        olde : elem = (id, [seq, frq, nbloc, [bowtie], [pri_miRNA, posMirPrim, Struct, 
        mircheck, fbstart, fbstop]])
        new : elem = (seq, [frq, nbloc, [bowtie], [pri_miRNA, posMirPrim, Struct, mircheck, 
        fbstart, fbstop]])
        '''
        priSeq = elem[1][field][0]
        posMir = int(elem[1][field][1])
        fback_start = int(elem[1][field][4])
        fback_stop = int(elem[1][field][5])
        
        elem[1].append(self.extract_sub_seq(priSeq, posMir, fback_start, fback_stop))
        return elem


class prog_RNAfold ():

    def __init__(self):
        self.env = os.environ

    def run_RNAfold(self, seq):
        '''
        example line = echo GUGGAGCUCCUAUCAUUCC| RNAfold
        task: echo and pipe the sequence to RNAfold
        this requires two subprocesses
        '''
        line1 = ['echo', seq]
        #line2 = ['RNAfold','--noPS', '--noLP', '--temp=25.0']
        line2 = ['RNAfold','--noPS', '--noLP']
        
        p1 = sbp.Popen(line1, stdout=sbp.PIPE, env=self.env)
        p2 = sbp.Popen(line2, stdin=p1.stdout, stdout=sbp.PIPE, env=self.env)
        p1.stdout.close()  #= Allow p1 to receive a SIGPIPE if p2 exits.
        
        output = p2.communicate()[0].rstrip('\n').split('\n')
        
        RNAfold_results = output[1].split(' (')
        folding = RNAfold_results[0]               #= ...(((....)))......
        # MFE = float(RNAfold_results[1][:-1])      #= minimun free energy
        
        return folding

    def RNAfold_map_rule(self, elem, field):
        '''
        old : elem = (id, [seq, frq, nbloc, [bowtie], [pri_miRNA]])
        new : elem = (seq, [frq, nbloc, [bowtie], [pri_miRNA]])
        '''
        elem[1][field].append(self.run_RNAfold(elem[1][field][0]))
        return elem

class prog_mirCheck ():

    def __init__(self, param):
        self.param = param
        self.env = os.environ

    def run_mirCheck(self, folding, miRNA_start, miRNA_stop):
        '''
        example line = perl eval_mircheck.pl "((((((.((((((....).))))).))))))........." 46
        64 def
        '''
        cmd = ['perl', 'eval_mircheck.pl', folding, str(miRNA_start), str(miRNA_stop), self.
        param]
        FNULL = open(os.devnull, 'w')
        sproc = sbp.Popen(cmd, stdout=sbp.PIPE, shell=False, stderr=FNULL, env=self.env)
        mirCheck_results = sproc.communicate()[0].rstrip('\n').split('\t') #= ['3prime',
        '1', '173']
        FNULL.close()
        return mirCheck_results

    def mirCheck_map_rule(self, elem, field):
        '''
        elem = (id, [seq, frq, nbloc, [bowtie], [pri_miRNA, posMirPrim, Struct]])
        elem[1][field][0]
        elem = (seq, [frq, nbloc, [bowtie], [pri_miRNA, posMirPrim, Struct]])
        '''
        len_miRNAseq = len(elem[0])

        pos_miRNA_start = elem[1][field][1]
        pos_miRNA_stop  = pos_miRNA_start + len_miRNAseq - 1
        folding = elem[1][field][2]

        mirCheck_results = self.run_mirCheck(folding, pos_miRNA_start, pos_miRNA_stop)

        if 'prime' in mirCheck_results[0]:
            elem[1][field].extend(mirCheck_results)
        else :
            del elem[1][field][:]

        return elem

class prog_dominant_profile :

    def __init__(self):
        self.env = os.environ

    def get_bowtie_strandchromo_dict (self, bowtie_rdd_collect):
        '''elem : (seq, [frq, nbloc, [bowties]])
        '''
        dict_bowtie_chromo_strand = {}

        for elem in bowtie_rdd_collect :
            bowties = elem[1][2]

            for bowtie in bowties :
                #= concatenate chromosome (bowtie[1]) and strand (bowtie[0])
                chromo_strand = bowtie[1] + bowtie[0]

                if chromo_strand not in dict_bowtie_chromo_strand.keys():
                    dict_bowtie_chromo_strand[chromo_strand] = []

                dict_bowtie_chromo_strand[chromo_strand].append(elem)

        return dict_bowtie_chromo_strand

```

```python
  def calculateTotalfrq (self, bowbloc, x, y):
    ''' old elem in bowbloc = (id, [seq, frq, nbloc, [bowties]])
        new elem in bowbloc = (seq, [frq, nbloc, [bowties]])
    '''
    totalfrq = 0
    for elem in bowbloc :
      bowties = elem[1][2]
      frq = elem[1][0]
      for bowtie in bowties :
        posgen = bowtie[2]
        if (x < posgen < y) :
          totalfrq += frq
    return totalfrq

  def profile_range (self, elem):
    ''' define x, y with pre_vld_rdd
        old : elem = (id, [seq, frq, nbloc, [bowtie], [pri_miRNA], [pre_miRNA]])
        new : elem = (seq, [frq, nbloc, [bowtie], [pri_miRNA], [pre_miRNA]])
    '''
    posgen = elem[1][2][2]
    mirseq = elem[0]
    mirpos_on_pre = elem[1][4][1]
    preseq = elem[1][4][0]
    strand = elem[1][2][0]

    if strand == '+':
      x = posgen - mirpos_on_pre    #= inclusive
      y = x + len(preseq) - 1       #= inclusive
    else:
      y = posgen + len(mirseq) + mirpos_on_pre -1
      x = y-len(preseq) + 1
    return x-1, y+1                 #= exclusive  x < a < y

  def exp_profile_filter (self, elem, dict_bowtie_chromo_strand):
    ''' old : elem = (id, [seq, frq, nbloc, [bowtie], [pri_miRNA], [pre_miRNA]])
        new : elem = (seq, [frq, nbloc, [bowtie], [pri_miRNA], [pre_miRNA]])
    '''
    x, y = self.profile_range (elem)
    bowtie_bloc_key = elem[1][2][1] + elem[1][2][0]  #chrom+strand
    bowbloc = dict_bowtie_chromo_strand[bowtie_bloc_key]
    totalfrq = self.calculateTotalfrq (bowbloc, x, y)
    miRNAfrq = elem[1][0]
    ratio = miRNAfrq / float(totalfrq)

    if ratio > 0.2 :
        return True
    return False

  def computeProfileFrq(self, elem, dict_bowtie_chromo_strand):
    x, y = self.profile_range (elem)
    bowtie_bloc_key = elem[1][2][1] + elem[1][2][0]  #=chrom+strand
    bowbloc = dict_bowtie_chromo_strand[bowtie_bloc_key]
    totalfrq = self.calculateTotalfrq (bowbloc, x, y)

    elem[1].append(totalfrq)
    return elem


class prog_miRanda ():
  def __init__ (self, Max_Score_cutoff, Max_Energy_cutoff, target_file, rep_tmp, 
  miranda_exe, Gap_Penalty):
    self.env = os.environ

    #== variables ==
    self.Max_Score_cutoff = Max_Score_cutoff
    self.Max_Energy_cutoff = Max_Energy_cutoff
    self.Gap_Penalty = Gap_Penalty
```

A.7 sequential.py

```
'''
program: sequential.py
author: Chao-Jung Wu
date: 2017-10-11
version: 0.00.03
'''



from __future__ import print_function
import sys
import os.path
import time
#

import utils as ut

import mirLibRules as mru

#tmp_rep = '/home/cloudera/Desktop/mirLibHadoop/tmp/'
tmp_rep = '/home/cjwu/gitproject/mirLibHadoop/tmp/'


#known_non = '../dbs/TAIR10_ncRNA_CDS.gff'
known_non = '/home/cjwu/gitproject/mirLibHadoop/dbs/TAIR10_ncRNA_CDS.gff'
d_ncRNA_CDS = ut.get_nonMirna_coors (known_non)
kn_obj = mru.prog_knownNonMiRNA(d_ncRNA_CDS)
profile_obj = mru.prog_dominant_profile()


limit_srna_freq = 10

limit_len = 18 
limit_mrna_freq = 100
limit_nbloc = 15

#genome_path = '/home/cloudera/workspace/mirLibHadoop/dbs/ATH/Genome/'
genome_path = '/scratch/hvg-164-aa/mirlibhadoop/ATH/Genome/'
pri_l_flank = 500 #20
pri_r_flank = 200 #160
pre_flank = 30
extr_obj = mru.extract_precurosrs (genome_path, pri_l_flank, pri_r_flank, pre_flank)




def readRaw (infile):
  ''' seq\tfreq'''
  dict_mirna = {}
  with open (infile, 'r') as fh:
    for line in fh:
      data = line.rstrip('\n').split('\t')
      seq = data[0]
      freq = int(data[1])
      dict_mirna[seq] = [freq]
  return dict_mirna


def filterFreq (limit_srna_freq, dict_mirna):
  dict_mirna2 = dict_mirna.copy()
  for k, v in dict_mirna2.items():
    if int(v[0]) < limit_srna_freq:
      del dict_mirna[k]


def filterShort (limit_len, dict_mirna):
  dict_mirna2 = dict_mirna.copy()
  for k in dict_mirna2.keys():
    if len(k) < limit_len:
      del dict_mirna[k]



def filterdust (dict_mirna):
  #= echo $'>seq1\nCGTGGCTATGATAGCGATATTCGTTTTTTT' | dustmasker
  dict_mirna2 = dict_mirna.copy()
  for k in dict_mirna2.keys():
      cmd = 'echo $\'>seq1\n' + k + '\' | dustmasker > ' + tmp_rep + 'dustmasker.tmp2'
      os.system(cmd)
      with open (tmp_rep + 'dustmasker.tmp2', 'r') as fh:
        data = fh.readlines()
        if len(data) == 2: 
          del dict_mirna[k]
  
def bowtiemap (dict_mirna):
  #bowtie_index = '/home/cloudera/workspace/mirLibHadoop/dbs/bowtie_index/a_thaliana_t10'
  bowtie_index = '/home/cjwu/gitproject/mirLibHadoop/dbs/bowtie_index/a_thaliana_t10'
  for k in dict_mirna.keys():
    cmd = 'bowtie --mm -a -v 0 --suppress 1,5,6,7,8 -c ' + bowtie_index + ' '+ k + ' > '
     + tmp_rep + 'bowtiemap.tmp2 2>/dev/null'
    os.system(cmd)
    locs = []
    with open (tmp_rep + 'bowtiemap.tmp2', 'r') as fh:
      for line in fh:
        data = line.rstrip('\n').split('\t') #= +	Chr2	1040947
        strand = data[0]
        chromo = data[1]
        posSeq = int(data[2])
        data = [strand, chromo, posSeq]
        locs.append(data)
    dict_mirna[k].append(len(locs))
    dict_mirna[k].append(locs) #= v =[freq, 2, [[strand, chromo, pos],[strand, chromo, 
    pos]]]

def filterMirnaFreq (limit_mrna_freq, dict_mirna):
  dict_mirna2 = dict_mirna.copy()
  for k, v in dict_mirna2.items():

    if int(v[0]) < limit_mrna_freq:

      del dict_mirna[k]

def filterNbLoc (limit_nbloc, dict_mirna):
  dict_mirna2 = dict_mirna.copy()
  for k, v in dict_mirna2.items():
    if v[1] == 0 or v[1] > limit_nbloc:
      del dict_mirna[k]

def filterKnowNon (dict_mirna):
  dict_mirna2 = dict_mirna.copy()
  for k, v in dict_mirna.items():
    if not kn_obj.knFilterByCoor ([k, v]):
      del dict_mirna2[k]

def extractPri (dict_mirna):
  for k, v in dict_mirna.items():
    for i in range(len(v[2])):
      bow = v[2][i]
      strand = bow[0]
      chromo = bow[1]
      start_srna = int(bow[2])
      len_srna = len(k)
      if chromo not in extr_obj.genome.keys(): 
        prims = [['-', -2]]
      else:
        contig = extr_obj.genome[chromo]
        prims = extr_obj.extract_precursors (contig, strand, start_srna, len_srna)
      dict_mirna[k][2][i].append(prims)

def fold1 (dict_mirna):
  for k, v in dict_mirna.items():
    for i in range(len(v[2])):
      prims = v[2][i][3]
      for j in range(len(prims)):
        cmd = 'echo ' + prims[j][0] + '| RNAfold  > ' + tmp_rep + 'rnafold.tmp2'
        os.system(cmd)
        with open (tmp_rep + 'rnafold.tmp2', 'r') as fh:
          fh.readline()
          for line in fh:
            fold = line.split(' (')[0]
            dict_mirna[k][2][i][3][j].append(fold)

def check (dict_mirna):
  for k, v in dict_mirna.items():
    for i in range(len(v[2])):
      prims = v[2][i][3]
      for j in range(len(prims)):
        miRNA_start = int(prims[j][1])
        miRNA_stop = miRNA_start + len(k) -1
        folding = prims[j][2]
        cmd = 'perl eval_mircheck.pl \"' + folding + '\" ' + str(miRNA_start) + ' ' +
        str(miRNA_stop) + ' def > ' + tmp_rep + 'testcheck_tmp.txt'
        os.system(cmd)
        with open (tmp_rep + 'testcheck_tmp.txt', 'r') as fh:
          for line in fh:
            check = line.rstrip('\n').split('\t')
            if 'prime' in check[0]:
              dict_mirna[k][2][i][3][j].append(check[0])
              dict_mirna[k][2][i][3][j].append(int(check[1]))
              dict_mirna[k][2][i][3][j].append(int(check[2]))
            else:
              del dict_mirna[k][2][i][3][j][:]

def filterOneLoop (dict_mirna):
  for k, v in dict_mirna.items():
    for i in range(len(v[2])):
      prims = v[2][i][3]
      for j in range(len(prims)):
        if len(prims[j]) == 0: continue
        folding = prims[j][2]
        s = int(prims[j][4])
        e = int(prims[j][5])+1
        if not ut.containsOnlyOneLoop (folding[s:e]):
          del dict_mirna[k][2][i][3][j][:]

def extractPre_fromPri (dict_mirna):
  for k, v in dict_mirna.items():
    for i in range(len(v[2])):
      prims = v[2][i][3]
      for j in range(len(prims)):
        if len(prims[j]) == 0: continue
        prim = prims[j]
        priSeq = prim[0]
        posMir = int(prim[1])
        fback_start = int(prim[4])
        fback_stop = int(prim[5])
        data = extr_obj.extract_sub_seq (priSeq, posMir, fback_start, fback_stop)
        dict_mirna[k][2][i][3][j].append(data[0])
        dict_mirna[k][2][i][3][j].append(data[1])

def fold2 (dict_mirna):
```

```python
  for k, v in dict_mirna.items():
    for i in range(len(v[2])):
      pres = v[2][i][3]
      for j in range(len(pres)):
        if len(pres[j]) == 0: continue
        preSeq = pres[j][6]
        cmd = 'echo ' + preSeq + '| RNAfold  > ' + tmp_rep + 'rnafold2.tmp2'
        os.system(cmd)
        with open (tmp_rep + 'rnafold2.tmp2', 'r') as fh:
          fh.readline()
          for line in fh:
            fold = line.split(' (')[0]
            dict_mirna[k][2][i][3][j].append(fold)

def mirdupcheck (dict_mirna):
  #= cmd = 'java -jar ../lib/miRdup_1.4/miRdup.jar -v mirdupcheck.tmp2 -c ../lib/miRdup_1.4//model/thaliana.model -r /usr/local/bin/'
  pred_file = tmp_rep + 'mirdupcheck.tmp2.thaliana.model.miRdup.txt'

  for k, v in dict_mirna.items():
    for i in range(len(v[2])):
      pres = v[2][i][3]
      for j in range(len(pres)):
        if len(pres[j]) == 0: continue
        preSeq = pres[j][6]
        miseq = k
        fold = pres[j][8]
        #print(preSeq, miseq, fold)

        with open (tmp_rep + 'mirdupcheck.tmp2', 'w') as fhtmp:
          print('seqx\t' + miseq + '\t' + preSeq + '\t' + fold, file=fhtmp)

        cmd = 'java -jar ../lib/miRdup_1.4/miRdup.jar -v ' + tmp_rep + 'mirdupcheck.tmp2 -c ../lib/miRdup_1.4//model/thaliana.model -r /usr/local/bin/ 1>/dev/null'
        os.system(cmd)

        with open (pred_file, 'r') as fh_tmp :
          for line in fh_tmp :
            if line.startswith("#PR") :
              mirdup_pred = line.rstrip("\n").split("\t")[2]
            elif line.startswith("#SC") :
              mirdup_score = '%.2f' % round(float(line.rstrip("\n").split("\t")[2]), 2)
        dict_mirna[k][2][i][3][j].append(mirdup_pred)
        dict_mirna[k][2][i][3][j].append(mirdup_score)

def dict_mirna_for_profile (dict_mirna):
  ''' 'elem : (seq, [frq, nbloc, [bowties]])  '''
  list_profile = []
  for k, v in dict_mirna.items():
    i = [k, v]
    list_profile.append(i)
  return list_profile

def filterProfile (dict_mirna):
  dict_mirna2 = dict_mirna.copy()
  for k, v in dict_mirna2.items():
    frq = v[0]
    nbLoc = v[1]
    locs = v[2]
    for l in range(len(locs)):
      loc = locs[l]
      bowtie = loc[0:3]
      items = loc[3]
      for i in range(len(items)):
        item = items[i]
        if len(item) == 0: continue
        prim = item[0:6]
        prem = item[6:11]
        elem = [k, [frq, nbLoc, bowtie, prim, prem]]
        elem = profile_obj.computeProfileFrq(elem, dict_bowtie_chromo_strand)
        totalFrq = elem[1][5]
        ratio = frq / float (totalFrq)
        if ratio > 0.2:
          dict_mirna[k][2][l][3][i].append(ratio)
        else:
          del dict_mirna[k][2][l][3][i][:]

def createBowFrqDict (dict_mirna):
  ''' note that it must use copy module to make a copy for this operation, otherwise, the return dict_bowtie_chromo_strand will be mutabel with dict_mirna
      By using the copied d2, it dissociates the mutation link. This might be a particular inconvenience in python 2.7
  '''
  import copy
  d2 = copy.deepcopy(dict_mirna)
  bowtie_collect = dict_mirna_for_profile (d2)
  dict_bowtie_chromo_strand = profile_obj.get_bowtie_strandchromo_dict (bowtie_collect)
  return dict_bowtie_chromo_strand

def keepTrue(dict_mirna):
  dict_mirna2 = dict_mirna.copy()
  count = 0
  for k, v in dict_mirna2.items():
    TRUE = 0
    frq = v[0]
    nbLoc = v[1]
    locs = v[2]
    for l in range(len(locs)):
      loc = locs[l]
      bowtie = loc[0:3]
      items = loc[3]
      for i in range(len(items)):
        item = items[i]
        if not len(item) == 0:
          TRUE = 1
          count += 1
    if TRUE == 0:
      del dict_mirna[k]
  return count

#infile = 'test.txt'
#infile = '/home/cloudera/Desktop/mirLibHadoop/input/100.txt'
#infile = '/home/cloudera/Desktop/mirLibHadoop/input/high_conf_mature_ath_uniq_raw.txt'
infile = '/home/cjwu/gitproject/mirLibHadoop/input/high_conf_mature_ath_uniq100.txt'

dict_mirna = readRaw (infile)

filterFreq (limit_srna_freq, dict_mirna)

filterShort (limit_len, dict_mirna)
filterdust (dict_mirna)
bowtiemap (dict_mirna)

dict_bowtie_chromo_strand = createBowFrqDict (dict_mirna)

filterMirnaFreq (limit_mrna_freq, dict_mirna)
filterNbLoc (limit_nbloc, dict_mirna)
extractPri (dict_mirna)

fold1 (dict_mirna)
check (dict_mirna)
filterOneLoop (dict_mirna)
```

```
extractPre_fromPri (dict_mirna)
fold2 (dict_mirna)
mirdupcheck (dict_mirna)
filterProfile (dict_mirna)
nbTrueDistinct = keepTrue(dict_mirna)

#for k, v in dict_mirna.items():
#  print(k, v)

print('nbTrueDistinct: ', nbTrueDistinct)
print('nbNonDistinct: ', len(dict_mirna))

```

A.8 run_mirdeep_p.py

```
'''
This program wraps to run mirdeep-p. Run under the folder of mirdeep_p.

It takes input file in raw format (seq\tfreq)from the input folder in this mirLibHadoop
project. MirDeep-P does not allow any adjustment of the parameters.
I can choose to use genome tair9 or tair10 by changing a program parameter in the
calling of the function: init_mirdeep_p (). Default is tair10.

author: Chao-Jung Wu
date: 2017-08-
version: 1.00.02
'''
import os
import time
from os import listdir

options = ['tair9', 'tair10']

def convert_raw_to_fasta500 (infile, rep_input):
  inBasename = os.path.splitext(infile)[0]
  outfile = '../tmp/' + inBasename + '.fa'
  fh_out = open (outfile, 'w')
  nb = 0
  with open (rep_input + infile, 'r') as fh:
    for i in fh:
      seq = i.split('\t')[0]
      freq = i.rstrip('\n').split('\t')[1]
      header = '>seq_' + str(nb) + '_x' + freq
      print >> fh_out, header, '\n', seq
      nb += 1
  fh_out.close()
  return outfile

def init_mirdeep_p (op):
  os.system('cp miRDP1.3/* .')
  os.system('mkdir bowtie-index 2>/dev/null')
  if op == options[0]:
    genome = 'TAIR9_genome.fa'
    rep = 'tair9/'
    f_annotated = 'annotated_miRNA_extended.fa'
    #os.system('bowtie-build -f ' + genome + ' bowtie-index/' + genome[:-3] + '
    >/dev/null')
    os.system('cp ' + rep + 'ath.gff .')#= miRBase v15
    #os.system('cp ' + rep + 'annotated_miRNA_extended.fa .')
    os.system('cp genome/' + genome + ' .')
    os.system('cp ' + rep + 'ncRNA_CDS.gff .')#
    os.system('cp ' + rep + 'chromosome_length .')#
    os.system('perl fetch_extended_precursors.pl ' + genome + ' ath.gff3.edited >' +
    f_annotated)
    os.system('bowtie-build -f annotated_miRNA_extended.fa
    bowtie-index/annotated_miRNA_extended.fa >/dev/null')
  elif op == options[1]:
    genome = 'a_thaliana_t10.fa'
    rep = 'tair10/'
    f_annotated = 'annotated_miRNA_v21_extended.fa'
    #os.system('bowtie-build -f ' + genome + ' bowtie-index/' + genome[:-3] + '
    >/dev/null')
    os.system('cp ../dbs/bowtie_index/*.ebwt bowtie-index/')
    os.system('cp ' + rep + 'ath.gff3.edited .')#= miRBase v21
    os.system('cp ' + rep + 'annotated_miRNA_v21_extended.fa .')
    os.system('cp genome/' + genome + ' .')
    #os.system('cat ../dbs/ATH/Genome/TAIR10_*.fas > ' + genome)#########
    os.system('cp ' + rep + 'ncRNA_CDS.gff .')#
    os.system('cp ' + rep + 'chromosome_length .')#
    #os.system('perl fetch_extended_precursors.pl ' + genome + ' ath.gff3.edited >' +
    f_annotated)
    os.system('bowtie-build -f annotated_miRNA_v21_extended.fa
    bowtie-index/annotated_miRNA_v21_extended.fa >/dev/null')
  return genome, f_annotated

def run_mirdp_new (infile):
  os.system('bowtie -a -v 0 bowtie-index/' + genome[:-3] + ' -f ' + infile + '>indata.aln 2>/dev/null')
  os.system('perl convert_bowtie_to_blast.pl indata.aln ' + infile + ' ' + genome + ' >indata.bst 2>/dev/null')
  os.system('perl filter_alignments.pl indata.bst -c 15 >indata_filter15.bst 2>/dev/null')
  os.system('perl overlap.pl indata_filter15.bst ncRNA_CDS.gff -b >indata_id_overlap_ncRNA_CDS 2>/dev/null')
  os.system('perl alignedselected.pl indata_filter15.bst -g indata_id_overlap_ncRNA_CDS >indata_filter15_ncRNA_CDS.bst 2>/dev/null')
  os.system('perl filter_alignments.pl indata_filter15_ncRNA_CDS.bst -b ' + infile + ' > indata_filtered.fa 2>/dev/null')
  #################
  os.system('perl excise_candidate.pl ' + genome + ' indata_filter15_ncRNA_CDS.bst 700 >indata_precursors.fa 2>/dev/null')
  #os.system('perl excise_candidate.pl ' + genome + ' indata_filter15_ncRNA_CDS.bst 250 >indata_precursors.fa 2>/dev/null')
  #################
  os.system('cat indata_precursors.fa | RNAfold --noPS > indata_structures')
  os.system('bowtie-build -f indata_precursors.fa bowtie-index/indata_precursors >/dev/null 2>/dev/null')
  os.system('bowtie -a -v 0 bowtie-index/indata_precursors -f indata_filtered.fa > indata_precursors.aln 2>/dev/null')
  os.system('perl convert_bowtie_to_blast.pl indata_precursors.aln indata_filtered.fa indata_precursors.fa >indata_precursors.bst 2>/dev/null')
  os.system('sort +3 -25 indata_precursors.bst >indata_signatures')
  os.system('perl miRDP.pl indata_signatures indata_structures > result_new_predictions')
  #os.system('perl rm_redundant_meet_plant.pl chromosome_length indata_precursors.fa indata_predictions indata_nr_prediction indata_filter_P_prediction')

def run_mirdp_known (infile, f_annotated):
  os.system('bowtie -a -v 0 bowtie-index/' + f_annotated + ' -f ' + infile + ' >indata.aln 2>/dev/null')
  os.system('perl convert_bowtie_to_blast.pl indata.aln ' + infile + ' ' + f_annotated + ' > indata_extended.bst 2>/dev/null')
  ###############
  #os.system('perl excise_candidate.pl ' + f_annotated + ' indata_extended.bst 250 >precursors_250.fa 2>/dev/null')
  os.system('perl excise_candidate.pl ' + f_annotated + ' indata_extended.bst 700 >precursors_250.fa 2>/dev/null')
  ###############
  os.system('bowtie-build -f precursors_250.fa bowtie-index/precursors_250 >/dev/null 2>/dev/null')
  os.system('cat precursors_250.fa|RNAfold --noPS >precursors_250_structure')
  os.system('bowtie -a -v 0 bowtie-index/precursors_250 -f ' + infile + ' >indata_250.aln 2>/dev/null')
  os.system('perl convert_bowtie_to_blast.pl indata_250.aln ' + infile + ' precursors_250.fa >indata_250.bst 2>/dev/null')
  os.system('sort +3 -25 indata_250.bst >indata_250_signature')
  os.system('perl miRDP.pl indata_250_signature precursors_250_structure > result_known_250_predictions')

def run_miRDP ():
  #rep_input = '/home/cloudera/Desktop/mirLibHadoop/input/'
  #rep_input = '/home/cjwu/demo/srna2/'
  rep_input = '../input/'

  infiles = [f for f in listdir(rep_input) if os.path.isfile(os.path.join(rep_input, f))]
  totaltime = 0
  for infile in infiles:
    if infile[-1:] == '~': continue
    print 'start processing', infile
    infile = convert_raw_to_fasta500 (infile, rep_input)
    start = time.time()

    run_mirdp_new (infile)
    run_mirdp_known (infile, f_annotated)

    end = time.time()

    duration = end - start
    totaltime += duration
    duration = format(duration, '.0f')
    print 'miRDP duration: ', infile, duration, 'sec'

    os.system('cat result_* > result_combinded.txt')
    os.system('cut -f1 result_combinded.txt | grep \'seq_\' | sort | uniq | wc -l')
    os.system('rm -f indata* *.gff* *.pl precursors_250_structure *.fa
    chromosome_length result_combinded.txt bowtie-index/*precursors*')
  
  totaltime = format(totaltime, '.0f')
  print 'miRDP duration: ', totaltime, 'sec'

genome, f_annotated = init_mirdeep_p ('tair10')
run_miRDP ()
```

APPENDIX B

MISCELLANY

B.1 MirLibSpark and wheat miRNA availability

-Project name: mirLibSpark;
-Project home page: `http://github.com/maremita/mirLibHadoop`
-Operating system(s): Oracle VM VirtualBox, Compute Canada Server;
-Supplementary materials and data: Dropbox `https://goo.gl/EAJmQu`
-License: BSD 3-Clause License

Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Manuscript Category

OXFORD

Subject Section

mirLibSpark: a scalable NGS microRNA prediction pipeline with data aggregation

Chao-Jung Wu [1,2], **Amine M. Remita** [1,2] **and Abdoulaye Baniré Diallo** [1,*]

[1] Department of Computer Science, Université du Québec à Montréal, Montréal, H3C 3P8 Canada
[2] These authors contributed equally to this work

[*] To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: The emergence of the Next Generation Sequencing increases drastically the volume of transcriptome data. Although many algorithms and workflows for novel microRNA (miRNA) prediction have been proposed, they are mostly semi-automated and few are designed for processing large volume of sequence data. This work aims to develop an efficient pipeline that analyzes short RNAs libraries faster while increasing the classification accuracy.

Results: We propose an improved pipeline for a high volume data facility, by implementing the mirLibSpark prediction pipeline based on the Apache Spark framework. It is firstly oriented towards the annotation of miRNAs in plants such as *Arabidopsis thaliana*. It can process data 100 times faster than a program that executes the same steps without parallelization, and at least 568 times faster than miRDeep-P after we made a script to run all their manual steps automatically. Its performance measures are better than those of miRDeep-P, a predictor of plant miRNAs. We have tested it on computer clusters as well as local environments. We have also applied our pipeline on more than one hundred libraries of *A. thaliana* small RNAs and demonstrated that analyzing the expression patterns of the predicted miRNAs among the libraries enables the classification of functional miRNAs. We deliver here the first fully automated and distributed miRNA predictor. It is an efficient and accurate miRNA predictor with functional insight. Eventually this pipeline will be extended to analyze small RNAs of other organisms and conditions such as human diseases. It will benefit the miRNA research community for its applications.

Availability: mirLibSpark program will be available as a docker image in the Wheat MicroRNA Portal wheat.bioinfo.uqam.ca. Currently it is available in the Dropbox goo.gl/wm8Y3X

Contact: diallo.abdoulaye@uqam.ca

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

The emergence of the Next Generation Sequencing (NGS) allows researchers to quickly and cheaply collect billions of small RNAs (sRNAs) sequences expressed according to a given stress condition so that the volume of small-RNAome data increases dramatically. However, the annotation of sRNAs has never been rapidly and systematically achieved. The needs to quickly classify sRNAs and analyze their related data will soon become a burden (Marx, 2013). It has become a trend for researchers to request service hours on a cloud cluster rather than invest in the hardware and maintenance to compute big data of high volume, variety and velocity (Dai *et al.*, 2012). Distributed computing frameworks such as Hadoop MapReduce and Apache Spark have recently benefited bioinformatics researchers to realize efficient, scalable and reliable computing performance on both private clusters and cloud computing services (Zou *et al.*, 2013). Scalable methods are revolutionizing traditional bioinformatics tools in RNA research (Nellore *et al.*, 2016; de Castro *et al.*, 2017).

Apache Spark has been recently proven powerful to scale up and accelerate one of the widely-used bioinformatics tools, BLAST (de Castro *et al.*, 2017). Spark exposes an abstraction of a distributed memory called *Resilient distributed dataset* (RDD) and a set of operations including

B.3 More resources for mirLibSpark

Cloudera deployment

On the host master:

- Download and install the Cloudera Manager Installer
  ```
  $ wget http://archive.cloudera.com/cm5/installer/
    latest/cloudera-manager-installer.bin
  $ chmod u+x cloudera-manager-installer.bin
  $ sudo ./cloudera-manager-installer.bin
  ```
- Open the Cloudera Manager Console in a web browser (`http://localhost:7180/`)
- Accept the Cloudera Manager End User License Terms and Conditions
- Add the hosts by specifying hostnames and IP address ranges
- Install and add Hadoop services (HDFS, YARN, Spark, etc.)
- Assign Hadoop service roles to each host

  More details of Cloudera deployment can be found here `https://www.cloudera.com/documentation/enterprise/5-10-x/topics/cm_ig_install_path_a.html`

Some useful resources

Cloudera

- `https://www.cloudera.com/downloads/cdh/5-10-0.html`
- `https://blog.cloudera.com/blog/2014/01/how-to-create-a-simple-hadoop-cluster-with-virtualbox/`
- `http://www.cse.scu.edu/~mwang2/projects/CDH_installConfig1_13m.pdf`
- `https://www.cloudera.com/documentation/enterprise/5-8-x/topics/cm_ig_host_allocations.html`
- `http://blog.cloudera.com/blog/2013/01/a-guide-to-python-frameworks-for-hadoop/`
- `http://crazyadmins.com/install-multinode-cloudera-hadoop-cluster-cdh5-4-0-manually/`

Spark

- `http://spark.apache.org/docs/latest/programming-guide.html`
- `http://blog.cloudera.com/blog/2015/03/how-to-tune-your-apache-spark-jobs-part-1/`
- `https://www.cloudera.com/documentation/enterprise/5-6-x/topics/cdh_ig_running_spark_on_yarn.html`
- `http://blog.cloudera.com/blog/2016/02/making-python-on-apache-hadoop-easier-with-anaconda-and-cdh/`

Bowtie index

Users can build their own index or download pre-built genome index for many other organisms from the ftp site

- `ftp://ftp.ccb.jhu.edu/pub/data/bowtie_indexes/`

B.4 Description of wheat 365 KEGG pathways

Wheat KEGG pathway

Pathway	Name	Class	Covrage
ko00010	Glycolysis / Gluconeogenesis	Metabolism; Carbohydrate metabolism	31.31%
ko00020	Citrate cycle (TCA cycle)	Metabolism; Carbohydrate metabolism	33.33%
ko00030	Pentose phosphate pathway	Metabolism; Carbohydrate metabolism	25.64%
ko00040	Pentose and glucuronate interconversions	Metabolism; Carbohydrate metabolism	13.64%
ko00051	Fructose and mannose metabolism	Metabolism; Carbohydrate metabolism	17.92%
ko00052	Galactose metabolism	Metabolism; Carbohydrate metabolism	21.33%
ko00053	Ascorbate and aldarate metabolism	Metabolism; Carbohydrate metabolism	33.33%
ko00061	Fatty acid biosynthesis	Metabolism; Lipid metabolism	33.33%
ko00062	Fatty acid elongation	Metabolism; Lipid metabolism	30.43%
ko00071	Fatty acid degradation	Metabolism; Lipid metabolism	20.75%
ko00072	Synthesis and degradation of ketone bodies	Metabolism; Lipid metabolism	37.50%
ko00073	Cutin, suberine and wax biosynthesis	Metabolism; Lipid metabolism	83.33%
ko00100	Steroid biosynthesis	Metabolism; Lipid metabolism	56.67%
ko00120	Primary bile acid biosynthesis	Metabolism; Lipid metabolism	5.56%
ko00130	Ubiquinone and other terpenoid-quinone biosynthesis	Metabolism; Metabolism of cofactors and vitamins	35.71%
ko00140	Steroid hormone biosynthesis	Metabolism; Lipid metabolism	10.64%
ko00190	Oxidative phosphorylation	Metabolism; Energy metabolism	30.70%
ko00195	Photosynthesis	Metabolism; Energy metabolism	47.62%
ko00196	Photosynthesis - antenna proteins	Metabolism; Energy metabolism	23.81%
ko00220	Arginine biosynthesis	Metabolism; Amino acid metabolism	34.55%
ko00230	Purine metabolism	Metabolism; Nucleotide metabolism	31.39%
ko00232	Caffeine metabolism	Metabolism; Biosynthesis of other secondary metabolites	22.22%
ko00240	Pyrimidine metabolism	Metabolism; Nucleotide metabolism	39.43%
ko00250	Alanine, aspartate and glutamate metabolism	Metabolism; Amino acid metabolism	43.28%
ko00254	Aflatoxin biosynthesis	Metabolism; Biosynthesis of other secondary metabolites	7.69%
ko00260	Glycine, serine and threonine metabolism	Metabolism; Amino acid metabolism	35.11%
ko00261	Monobactam biosynthesis	Metabolism; Biosynthesis of other secondary metabolites	17.86%
ko00270	Cysteine and methionine metabolism	Metabolism; Amino acid metabolism	37.11%
ko00280	Valine, leucine and isoleucine degradation	Metabolism; Amino acid metabolism	30.43%
ko00281	Geraniol degradation	Metabolism; Metabolism of terpenoids and polyketides	6.25%
ko00290	Valine, leucine and isoleucine biosynthesis	Metabolism; Amino acid metabolism	58.82%
ko00300	Lysine biosynthesis	Metabolism; Amino acid metabolism	21.74%
ko00310	Lysine degradation	Metabolism; Amino acid metabolism	21.05%
ko00330	Arginine and proline metabolism	Metabolism; Amino acid metabolism	25.25%
ko00340	Histidine metabolism	Metabolism; Amino acid metabolism	26.09%
ko00350	Tyrosine metabolism	Metabolism; Amino acid metabolism	23.19%
ko00360	Phenylalanine metabolism	Metabolism; Amino acid metabolism	22.67%
ko00361	Chlorocyclohexane and chlorobenzene degradation	Metabolism; Xenobiotics biodegradation and metabolism	2.63%
ko00362	Benzoate degradation	Metabolism; Xenobiotics biodegradation and metabolism	2.11%
ko00363	Bisphenol degradation	Metabolism; Xenobiotics biodegradation and metabolism	25.00%
ko00380	Tryptophan metabolism	Metabolism; Amino acid metabolism	22.39%
ko00400	Phenylalanine, tyrosine and tryptophan biosynthesis	Metabolism; Amino acid metabolism	33.33%
ko00401	Novobiocin biosynthesis	Metabolism; Biosynthesis of other secondary metabolites	10.00%
ko00402	Benzoxazinoid biosynthesis	Metabolism; Biosynthesis of other secondary metabolites	37.50%
ko00410	beta-Alanine metabolism	Metabolism; Metabolism of other amino acids	37.21%
ko00430	Taurine and hypotaurine metabolism	Metabolism; Metabolism of other amino acids	12.50%
ko00440	Phosphonate and phosphinate metabolism	Metabolism; Metabolism of other amino acids	11.43%
ko00450	Selenocompound metabolism	Metabolism; Metabolism of other amino acids	35.71%
ko00460	Cyanoamino acid metabolism	Metabolism; Metabolism of other amino acids	36.36%
ko00471	D-Glutamine and D-glutamate metabolism	Metabolism; Metabolism of other amino acids	16.67%
ko00473	D-Alanine metabolism	Metabolism; Metabolism of other amino acids	20.00%
ko00480	Glutathione metabolism	Metabolism; Metabolism of other amino acids	34.15%
ko00500	Starch and sucrose metabolism	Metabolism; Carbohydrate metabolism	28.44%
ko00510	N-Glycan biosynthesis	Metabolism; Glycan biosynthesis and metabolism	69.77%
ko00511	Other glycan degradation	Metabolism; Glycan biosynthesis and metabolism	42.11%
ko00513	Various types of N-glycan biosynthesis	Metabolism; Glycan biosynthesis and metabolism	45.10%
ko00514	Other types of O-glycan biosynthesis	Metabolism; Glycan biosynthesis and metabolism	26.92%
ko00515	Mannose type O-glycan biosynthesis	Metabolism; Glycan biosynthesis and metabolism	4.76%
ko00520	Amino sugar and nucleotide sugar metabolism	Metabolism; Carbohydrate metabolism	25.33%
ko00521	Streptomycin biosynthesis	Metabolism; Biosynthesis of other secondary metabolites	16.67%
ko00523	Polyketide sugar unit biosynthesis	Metabolism; Metabolism of terpenoids and polyketides	2.22%
ko00524	Neomycin, kanamycin and gentamicin biosynthesis	Metabolism; Biosynthesis of other secondary metabolites	1.75%
ko00531	Glycosaminoglycan degradation	Metabolism; Glycan biosynthesis and metabolism	26.67%
ko00532	Glycosaminoglycan biosynthesis - chondroitin sulfate / dermatan sulfate	Metabolism; Glycan biosynthesis and metabolism	5.56%
ko00534	Glycosaminoglycan biosynthesis - heparan sulfate / heparin	Metabolism; Glycan biosynthesis and metabolism	4.55%
ko00540	Lipopolysaccharide biosynthesis	Metabolism; Glycan biosynthesis and metabolism	18.42%
ko00550	Peptidoglycan biosynthesis	Metabolism; Glycan biosynthesis and metabolism	7.50%
ko00561	Glycerolipid metabolism	Metabolism; Lipid metabolism	32.47%

ko00562	Inositol phosphate metabolism	Metabolism; Carbohydrate metabolism	37.50%
ko00563	Glycosylphosphatidylinositol (GPI)-anchor biosynthesis	Metabolism; Glycan biosynthesis and metabolism	76.92%
ko00564	Glycerophospholipid metabolism	Metabolism; Lipid metabolism	33.03%
ko00565	Ether lipid metabolism	Metabolism; Lipid metabolism	31.03%
ko00590	Arachidonic acid metabolism	Metabolism; Lipid metabolism	22.22%
ko00591	Linoleic acid metabolism	Metabolism; Lipid metabolism	16.67%
ko00592	alpha-Linolenic acid metabolism	Metabolism; Lipid metabolism	60.87%
ko00600	Sphingolipid metabolism	Metabolism; Lipid metabolism	34.15%
ko00603	Glycosphingolipid biosynthesis - globo and isoglobo series	Metabolism; Glycan biosynthesis and metabolism	13.33%
ko00604	Glycosphingolipid biosynthesis - ganglio series	Metabolism; Glycan biosynthesis and metabolism	14.29%
ko00620	Pyruvate metabolism	Metabolism; Carbohydrate metabolism	26.32%
ko00624	Polycyclic aromatic hydrocarbon degradation	Metabolism; Xenobiotics biodegradation and metabolism	2.38%
ko00625	Chloroalkane and chloroalkene degradation	Metabolism; Xenobiotics biodegradation and metabolism	16.67%
ko00626	Naphthalene degradation	Metabolism; Xenobiotics biodegradation and metabolism	6.90%
ko00627	Aminobenzoate degradation	Metabolism; Xenobiotics biodegradation and metabolism	2.74%
ko00630	Glyoxylate and dicarboxylate metabolism	Metabolism; Carbohydrate metabolism	31.58%
ko00640	Propanoate metabolism	Metabolism; Carbohydrate metabolism	11.88%
ko00643	Styrene degradation	Metabolism; Xenobiotics biodegradation and metabolism	12.00%
ko00650	Butanoate metabolism	Metabolism; Carbohydrate metabolism	12.79%
ko00660	C5-Branched dibasic acid metabolism	Metabolism; Carbohydrate metabolism	20.00%
ko00670	One carbon pool by folate	Metabolism; Metabolism of cofactors and vitamins	30.30%
ko00680	Methane metabolism	Metabolism; Energy metabolism	11.98%
ko00710	Carbon fixation in photosynthetic organisms	Metabolism; Energy metabolism	69.44%
ko00720	Carbon fixation pathways in prokaryotes	Metabolism; Energy metabolism	10.38%
ko00730	Thiamine metabolism	Metabolism; Metabolism of cofactors and vitamins	38.71%
ko00740	Riboflavin metabolism	Metabolism; Metabolism of cofactors and vitamins	24.24%
ko00750	Vitamin B6 metabolism	Metabolism; Metabolism of cofactors and vitamins	34.78%
ko00760	Nicotinate and nicotinamide metabolism	Metabolism; Metabolism of cofactors and vitamins	15.73%
ko00770	Pantothenate and CoA biosynthesis	Metabolism; Metabolism of cofactors and vitamins	43.24%
ko00780	Biotin metabolism	Metabolism; Metabolism of cofactors and vitamins	30.43%
ko00785	Lipoic acid metabolism	Metabolism; Metabolism of cofactors and vitamins	75.00%
ko00790	Folate biosynthesis	Metabolism; Metabolism of cofactors and vitamins	20.00%
ko00830	Retinol metabolism	Metabolism; Metabolism of cofactors and vitamins	12.50%
ko00860	Porphyrin and chlorophyll metabolism	Metabolism; Metabolism of cofactors and vitamins	27.10%
ko00900	Terpenoid backbone biosynthesis	Metabolism; Metabolism of terpenoids and polyketides	58.49%
ko00901	Indole alkaloid biosynthesis	Metabolism; Biosynthesis of other secondary metabolites	36.36%
ko00902	Monoterpenoid biosynthesis	Metabolism; Metabolism of terpenoids and polyketides	32.00%
ko00903	Limonene and pinene degradation	Metabolism; Metabolism of terpenoids and polyketides	15.38%
ko00904	Diterpenoid biosynthesis	Metabolism; Metabolism of terpenoids and polyketides	27.45%
ko00905	Brassinosteroid biosynthesis	Metabolism; Metabolism of terpenoids and polyketides	60.00%
ko00906	Carotenoid biosynthesis	Metabolism; Metabolism of terpenoids and polyketides	34.78%
ko00908	Zeatin biosynthesis	Metabolism; Metabolism of terpenoids and polyketides	62.50%
ko00909	Sesquiterpenoid and triterpenoid biosynthesis	Metabolism; Metabolism of terpenoids and polyketides	23.53%
ko00910	Nitrogen metabolism	Metabolism; Energy metabolism	18.33%
ko00920	Sulfur metabolism	Metabolism; Energy metabolism	13.68%
ko00940	Phenylpropanoid biosynthesis	Metabolism; Biosynthesis of other secondary metabolites	57.58%
ko00941	Flavonoid biosynthesis	Metabolism; Biosynthesis of other secondary metabolites	63.16%
ko00942	Anthocyanin biosynthesis	Metabolism; Biosynthesis of other secondary metabolites	57.14%
ko00943	Isoflavonoid biosynthesis	Metabolism; Biosynthesis of other secondary metabolites	46.15%
ko00944	Flavone and flavonol biosynthesis	Metabolism; Biosynthesis of other secondary metabolites	41.67%
ko00945	Stilbenoid, diarylheptanoid and gingerol biosynthesis	Metabolism; Biosynthesis of other secondary metabolites	46.15%
ko00950	Isoquinoline alkaloid biosynthesis	Metabolism; Biosynthesis of other secondary metabolites	33.33%
ko00960	Tropane, piperidine and pyridine alkaloid biosynthesis	Metabolism; Biosynthesis of other secondary metabolites	42.31%
ko00965	Betalain biosynthesis	Metabolism; Biosynthesis of other secondary metabolites	14.29%
ko00966	Glucosinolate biosynthesis	Metabolism; Biosynthesis of other secondary metabolites	20.00%
ko00970	Aminoacyl-tRNA biosynthesis	Genetic Information Processing; Translation	41.27%
ko00980	Metabolism of xenobiotics by cytochrome P450	Metabolism; Xenobiotics biodegradation and metabolism	18.18%
ko00981	Insect hormone biosynthesis	Metabolism; Metabolism of terpenoids and polyketides	5.56%
ko00982	Drug metabolism - cytochrome P450	Metabolism; Xenobiotics biodegradation and metabolism	16.67%
ko00983	Drug metabolism - other enzymes	Metabolism; Xenobiotics biodegradation and metabolism	45.45%
ko00984	Steroid degradation	Metabolism; Xenobiotics biodegradation and metabolism	13.33%
ko01040	Biosynthesis of unsaturated fatty acids	Metabolism; Lipid metabolism	29.63%
ko01051	Biosynthesis of ansamycins	Metabolism; Metabolism of terpenoids and polyketides	3.12%
ko01053	Biosynthesis of siderophore group nonribosomal peptides	Metabolism; Metabolism of terpenoids and polyketides	3.33%
ko01200	Carbon metabolism	Metabolism; Overview	na
ko01210	2-Oxocarboxylic acid metabolism	Metabolism; Overview	na
ko01212	Fatty acid metabolism	Metabolism; Overview	na
ko01220	Degradation of aromatic compounds	Metabolism; Overview	na
ko01230	Biosynthesis of amino acids	Metabolism; Overview	na
ko01502	Vancomycin resistance	Human Diseases; Drug resistance	na

ko01503	Cationic antimicrobial peptide (CAMP) resistance	Human Diseases; Drug resistance	na
ko01521	EGFR tyrosine kinase inhibitor resistance	Human Diseases; Drug resistance	na
ko01522	Endocrine resistance	Human Diseases; Drug resistance	na
ko01523	Antifolate resistance	Human Diseases; Drug resistance	na
ko01524	Platinum drug resistance	Human Diseases; Drug resistance	na
ko02010	ABC transporters	Environmental Information Processing; Membrane transport	1.26%
ko02020	Two-component system	Environmental Information Processing; Signal transduction	1.46%
ko02024	Quorum sensing	Cellular Processes; Cellular community - prokaryotes	4.61%
ko02025	Biofilm formation - Pseudomonas aeruginosa	Cellular Processes; Cellular community - prokaryotes	2.27%
ko02026	Biofilm formation - Escherichia coli	Cellular Processes; Cellular community - prokaryotes	3.28%
ko03008	Ribosome biogenesis in eukaryotes	Genetic Information Processing; Translation	69.51%
ko03010	Ribosome	Genetic Information Processing; Translation	82.52%
ko03013	RNA transport	Genetic Information Processing; Translation	65.22%
ko03015	mRNA surveillance pathway	Genetic Information Processing; Translation	81.36%
ko03018	RNA degradation	Genetic Information Processing; Folding, sorting and degradation	65.38%
ko03020	RNA polymerase	Genetic Information Processing; Transcription	54.90%
ko03022	Basal transcription factors	Genetic Information Processing; Transcription	82.86%
ko03030	DNA replication	Genetic Information Processing; Replication and repair	45.76%
ko03040	Spliceosome	Genetic Information Processing; Transcription	82.79%
ko03050	Proteasome	Genetic Information Processing; Folding, sorting and degradation	66.67%
ko03060	Protein export	Genetic Information Processing; Folding, sorting and degradation	66.67%
ko03070	Bacterial secretion system	Environmental Information Processing; Membrane transport	8.11%
ko03320	PPAR signaling pathway	Organismal Systems; Endocrine system	11.48%
ko03410	Base excision repair	Genetic Information Processing; Replication and repair	58.54%
ko03420	Nucleotide excision repair	Genetic Information Processing; Replication and repair	74.47%
ko03430	Mismatch repair	Genetic Information Processing; Replication and repair	43.18%
ko03440	Homologous recombination	Genetic Information Processing; Replication and repair	41.67%
ko03450	Non-homologous end-joining	Genetic Information Processing; Replication and repair	42.11%
ko03460	Fanconi anemia pathway	Genetic Information Processing; Replication and repair	57.41%
ko04010	MAPK signaling pathway	Environmental Information Processing; Signal transduction	7.33%
ko04011	MAPK signaling pathway - yeast	Environmental Information Processing; Signal transduction	17.78%
ko04012	ErbB signaling pathway	Environmental Information Processing; Signal transduction	10.29%
ko04013	MAPK signaling pathway - fly	Environmental Information Processing; Signal transduction	15.48%
ko04014	Ras signaling pathway	Environmental Information Processing; Signal transduction	7.19%
ko04015	Rap1 signaling pathway	Environmental Information Processing; Signal transduction	3.85%
ko04016	MAPK signaling pathway - plant	Environmental Information Processing; Signal transduction	85.19%
ko04020	Calcium signaling pathway	Environmental Information Processing; Signal transduction	5.63%
ko04022	cGMP-PKG signaling pathway	Environmental Information Processing; Signal transduction	7.69%
ko04024	cAMP signaling pathway	Environmental Information Processing; Signal transduction	6.16%
ko04062	Chemokine signaling pathway	Organismal Systems; Immune system	5.30%
ko04064	NF-kappa B signaling pathway	Environmental Information Processing; Signal transduction	11.24%
ko04066	HIF-1 signaling pathway	Environmental Information Processing; Signal transduction	22.54%
ko04068	FoxO signaling pathway	Environmental Information Processing; Signal transduction	17.00%
ko04070	Phosphatidylinositol signaling system	Environmental Information Processing; Signal transduction	39.62%
ko04071	Sphingolipid signaling pathway	Environmental Information Processing; Signal transduction	23.08%
ko04072	Phospholipase D signaling pathway	Environmental Information Processing; Signal transduction	10.10%
ko04075	Plant hormone signal transduction	Environmental Information Processing; Signal transduction	91.11%
ko04110	Cell cycle	Cellular Processes; Cell growth and death	50.47%
ko04111	Cell cycle - yeast	Cellular Processes; Cell growth and death	44.26%
ko04112	Cell cycle - Caulobacter	Cellular Processes; Cell growth and death	12.90%
ko04113	Meiosis - yeast	Cellular Processes; Cell growth and death	37.37%
ko04114	Oocyte meiosis	Cellular Processes; Cell growth and death	40.79%
ko04115	p53 signaling pathway	Cellular Processes; Cell growth and death	19.67%
ko04120	Ubiquitin mediated proteolysis	Genetic Information Processing; Folding, sorting and degradation	51.20%
ko04122	Sulfur relay system	Genetic Information Processing; Folding, sorting and degradation	29.63%
ko04130	SNARE interactions in vesicular transport	Genetic Information Processing; Folding, sorting and degradation	47.06%
ko04139	Mitophagy - yeast	Cellular Processes; Transport and catabolism	41.18%
ko04140	Autophagy	Cellular Processes; Transport and catabolism	56.00%
ko04141	Protein processing in endoplasmic reticulum	Genetic Information Processing; Folding, sorting and degradation	55.00%
ko04142	Lysosome	Cellular Processes; Transport and catabolism	28.43%
ko04144	Endocytosis	Cellular Processes; Transport and catabolism	32.20%
ko04145	Phagosome	Cellular Processes; Transport and catabolism	27.37%
ko04146	Peroxisome	Cellular Processes; Transport and catabolism	52.78%
ko04150	mTOR signaling pathway	Environmental Information Processing; Signal transduction	24.78%
ko04151	PI3K-Akt signaling pathway	Environmental Information Processing; Signal transduction	10.42%
ko04152	AMPK signaling pathway	Environmental Information Processing; Signal transduction	29.27%
ko04210	Apoptosis	Cellular Processes; Cell growth and death	11.93%
ko04211	Longevity regulating pathway	Organismal Systems; Aging	20.59%
ko04212	Longevity regulating pathway - worm	Organismal Systems; Aging	32.73%
ko04213	Longevity regulating pathway - multiple species	Organismal Systems; Aging	20.69%

ko04214	Apoptosis - fly	Cellular Processes; Cell growth and death	21.43%
ko04215	Apoptosis - multiple species	Cellular Processes; Cell growth and death	10.26%
ko04260	Cardiac muscle contraction	Organismal Systems; Circulatory system	15.00%
ko04261	Adrenergic signaling in cardiomyocytes	Organismal Systems; Circulatory system	8.70%
ko04270	Vascular smooth muscle contraction	Organismal Systems; Circulatory system	8.14%
ko04310	Wnt signaling pathway	Environmental Information Processing; Signal transduction	16.50%
ko04320	Dorso-ventral axis formation	Organismal Systems; Development	7.41%
ko04330	Notch signaling pathway	Environmental Information Processing; Signal transduction	29.03%
ko04340	Hedgehog signaling pathway	Environmental Information Processing; Signal transduction	13.89%
ko04341	Hedgehog signaling pathway - fly	Environmental Information Processing; Signal transduction	37.50%
ko04350	TGF-beta signaling pathway	Environmental Information Processing; Signal transduction	13.04%
ko04360	Axon guidance	Organismal Systems; Development	5.74%
ko04370	VEGF signaling pathway	Environmental Information Processing; Signal transduction	15.38%
ko04380	Osteoclast differentiation	Organismal Systems; Development	7.07%
ko04390	Hippo signaling pathway	Environmental Information Processing; Signal transduction	6.36%
ko04391	Hippo signaling pathway - fly	Environmental Information Processing; Signal transduction	13.46%
ko04392	Hippo signaling pathway -multiple species	Environmental Information Processing; Signal transduction	16.00%
ko04510	Focal adhesion	Cellular Processes; Cellular community - eukaryotes	6.08%
ko04520	Adherens junction	Cellular Processes; Cellular community - eukaryotes	10.00%
ko04530	Tight junction	Cellular Processes; Cellular community - eukaryotes	14.85%
ko04540	Gap junction	Cellular Processes; Cellular community - eukaryotes	11.86%
ko04550	Signaling pathways regulating pluripotency of stem cells	Cellular Processes; Cellular community - eukaryotes	4.63%
ko04611	Platelet activation	Organismal Systems; Immune system	3.45%
ko04612	Antigen processing and presentation	Organismal Systems; Immune system	24.39%
ko04614	Renin-angiotensin system	Organismal Systems; Endocrine system	9.09%
ko04620	Toll-like receptor signaling pathway	Organismal Systems; Immune system	6.58%
ko04621	NOD-like receptor signaling pathway	Organismal Systems; Immune system	13.24%
ko04622	RIG-I-like receptor signaling pathway	Organismal Systems; Immune system	13.21%
ko04623	Cytosolic DNA-sensing pathway	Organismal Systems; Immune system	27.45%
ko04624	Toll and Imd signaling pathway	Organismal Systems; Immune system	10.64%
ko04626	Plant-pathogen interaction	Organismal Systems; Environmental adaptation	41.03%
ko04630	Jak-STAT signaling pathway	Environmental Information Processing; Signal transduction	3.10%
ko04650	Natural killer cell mediated cytotoxicity	Organismal Systems; Immune system	7.41%
ko04657	IL-17 signaling pathway	Organismal Systems; Immune system	11.11%
ko04658	Th1 and Th2 cell differentiation	Organismal Systems; Immune system	3.03%
ko04659	Th17 cell differentiation	Organismal Systems; Immune system	4.71%
ko04660	T cell receptor signaling pathway	Organismal Systems; Immune system	7.06%
ko04662	B cell receptor signaling pathway	Organismal Systems; Immune system	10.53%
ko04664	Fc epsilon RI signaling pathway	Organismal Systems; Immune system	8.51%
ko04666	Fc gamma R-mediated phagocytosis	Organismal Systems; Immune system	27.59%
ko04668	TNF signaling pathway	Environmental Information Processing; Signal transduction	4.60%
ko04670	Leukocyte transendothelial migration	Organismal Systems; Immune system	2.67%
ko04710	Circadian rhythm	Organismal Systems; Environmental adaptation	27.27%
ko04711	Circadian rhythm - fly	Organismal Systems; Environmental adaptation	12.50%
ko04712	Circadian rhythm - plant	Organismal Systems; Environmental adaptation	77.78%
ko04713	Circadian entrainment	Organismal Systems; Environmental adaptation	6.58%
ko04720	Long-term potentiation	Organismal Systems; Nervous system	16.28%
ko04721	Synaptic vesicle cycle	Organismal Systems; Nervous system	47.50%
ko04722	Neurotrophin signaling pathway	Organismal Systems; Nervous system	14.44%
ko04723	Retrograde endocannabinoid signaling	Organismal Systems; Nervous system	9.46%
ko04724	Glutamatergic synapse	Organismal Systems; Nervous system	9.41%
ko04725	Cholinergic synapse	Organismal Systems; Nervous system	4.55%
ko04726	Serotonergic synapse	Organismal Systems; Nervous system	4.76%
ko04727	GABAergic synapse	Organismal Systems; Nervous system	15.38%
ko04728	Dopaminergic synapse	Organismal Systems; Nervous system	12.05%
ko04730	Long-term depression	Organismal Systems; Nervous system	13.33%
ko04740	Olfactory transduction	Organismal Systems; Sensory system	12.00%
ko04742	Taste transduction	Organismal Systems; Sensory system	2.50%
ko04744	Phototransduction	Organismal Systems; Sensory system	11.11%
ko04745	Phototransduction - fly	Organismal Systems; Sensory system	11.11%
ko04750	Inflammatory mediator regulation of TRP channels	Organismal Systems; Sensory system	5.97%
ko04810	Regulation of actin cytoskeleton	Cellular Processes; Cell motility	11.92%
ko04910	Insulin signaling pathway	Organismal Systems; Endocrine system	20.93%
ko04911	Insulin secretion	Organismal Systems; Endocrine system	2.99%
ko04912	GnRH signaling pathway	Organismal Systems; Endocrine system	9.52%
ko04913	Ovarian steroidogenesis	Organismal Systems; Endocrine system	2.56%
ko04914	Progesterone-mediated oocyte maturation	Organismal Systems; Endocrine system	37.29%
ko04915	Estrogen signaling pathway	Organismal Systems; Endocrine system	10.77%
ko04916	Melanogenesis	Organismal Systems; Endocrine system	8.96%
ko04917	Prolactin signaling pathway	Organismal Systems; Endocrine system	7.02%

ko04918	Thyroid hormone synthesis	Organismal Systems; Endocrine system	11.54%
ko04919	Thyroid hormone signaling pathway	Organismal Systems; Endocrine system	19.54%
ko04920	Adipocytokine signaling pathway	Organismal Systems; Endocrine system	11.32%
ko04921	Oxytocin signaling pathway	Organismal Systems; Endocrine system	9.91%
ko04922	Glucagon signaling pathway	Organismal Systems; Endocrine system	23.64%
ko04923	Regulation of lipolysis in adipocytes	Organismal Systems; Endocrine system	6.67%
ko04924	Renin secretion	Organismal Systems; Endocrine system	6.52%
ko04925	Aldosterone synthesis and secretion	Organismal Systems; Endocrine system	5.17%
ko04930	Type II diabetes mellitus	Human Diseases; Endocrine and metabolic diseases	10.81%
ko04931	Insulin resistance	Human Diseases; Endocrine and metabolic diseases	17.81%
ko04932	Non-alcoholic fatty liver disease (NAFLD)	Human Diseases; Endocrine and metabolic diseases	33.87%
ko04933	AGE-RAGE signaling pathway in diabetic complications	Human Diseases; Endocrine and metabolic diseases	5.56%
ko04940	Type I diabetes mellitus	Human Diseases; Endocrine and metabolic diseases	8.00%
ko04960	Aldosterone-regulated sodium reabsorption	Organismal Systems; Excretory system	11.54%
ko04961	Endocrine and other factor-regulated calcium reabsorption	Organismal Systems; Excretory system	17.24%
ko04962	Vasopressin-regulated water reabsorption	Organismal Systems; Excretory system	15.15%
ko04964	Proximal tubule bicarbonate reclamation	Organismal Systems; Excretory system	21.43%
ko04966	Collecting duct acid secretion	Organismal Systems; Excretory system	58.82%
ko04970	Salivary secretion	Organismal Systems; Digestive system	4.76%
ko04971	Gastric acid secretion	Organismal Systems; Digestive system	4.08%
ko04972	Pancreatic secretion	Organismal Systems; Digestive system	7.69%
ko04973	Carbohydrate digestion and absorption	Organismal Systems; Digestive system	23.81%
ko04974	Protein digestion and absorption	Organismal Systems; Digestive system	5.36%
ko04975	Fat digestion and absorption	Organismal Systems; Digestive system	15.38%
ko04976	Bile secretion	Organismal Systems; Digestive system	1.69%
ko04977	Vitamin digestion and absorption	Organismal Systems; Digestive system	14.29%
ko04978	Mineral absorption	Organismal Systems; Digestive system	16.67%
ko05010	Alzheimer's disease	Human Diseases; Neurodegenerative diseases	36.36%
ko05012	Parkinson's disease	Human Diseases; Neurodegenerative diseases	41.80%
ko05014	Amyotrophic lateral sclerosis (ALS)	Human Diseases; Neurodegenerative diseases	23.26%
ko05016	Huntington's disease	Human Diseases; Neurodegenerative diseases	43.62%
ko05020	Prion diseases	Human Diseases; Neurodegenerative diseases	16.67%
ko05031	Amphetamine addiction	Human Diseases; Substance dependence	8.89%
ko05032	Morphine addiction	Human Diseases; Substance dependence	6.25%
ko05033	Nicotine addiction	Human Diseases; Substance dependence	3.70%
ko05034	Alcoholism	Human Diseases; Substance dependence	16.87%
ko05100	Bacterial invasion of epithelial cells	Human Diseases; Infectious diseases	13.70%
ko05110	Vibrio cholerae infection	Human Diseases; Infectious diseases	33.33%
ko05111	Biofilm formation - Vibrio cholerae	Cellular Processes; Cellular community - prokaryotes	0.93%
ko05120	Epithelial cell signaling in Helicobacter pylori infection	Human Diseases; Infectious diseases	14.44%
ko05130	Pathogenic Escherichia coli infection	Human Diseases; Infectious diseases	18.60%
ko05131	Shigellosis	Human Diseases; Infectious diseases	18.75%
ko05132	Salmonella infection	Human Diseases; Infectious diseases	12.20%
ko05133	Pertussis	Human Diseases; Infectious diseases	6.33%
ko05134	Legionellosis	Human Diseases; Infectious diseases	22.06%
ko05140	Leishmaniasis	Human Diseases; Infectious diseases	7.02%
ko05142	Chagas disease (American trypanosomiasis)	Human Diseases; Infectious diseases	9.76%
ko05143	African trypanosomiasis	Human Diseases; Infectious diseases	5.41%
ko05145	Toxoplasmosis	Human Diseases; Infectious diseases	8.51%
ko05146	Amoebiasis	Human Diseases; Infectious diseases	4.94%
ko05152	Tuberculosis	Human Diseases; Infectious diseases	14.50%
ko05160	Hepatitis C	Human Diseases; Infectious diseases	13.92%
ko05161	Hepatitis B	Human Diseases; Infectious diseases	12.39%
ko05162	Measles	Human Diseases; Infectious diseases	11.01%
ko05164	Influenza A	Human Diseases; Infectious diseases	16.13%
ko05166	HTLV-I infection	Human Diseases; Infectious diseases	21.36%
ko05168	Herpes simplex infection	Human Diseases; Infectious diseases	22.07%
ko05169	Epstein-Barr virus infection	Human Diseases; Infectious diseases	39.63%
ko05200	Pathways in cancer	Human Diseases; Cancers	10.44%
ko05202	Transcriptional misregulation in cancer	Human Diseases; Cancers	8.84%
ko05203	Viral carcinogenesis	Human Diseases; Cancers	30.60%
ko05204	Chemical carcinogenesis	Human Diseases; Cancers	13.89%
ko05205	Proteoglycans in cancer	Human Diseases; Cancers	8.64%
ko05206	MicroRNAs in cancer	Human Diseases; Cancers	7.79%
ko05210	Colorectal cancer	Human Diseases; Cancers	18.37%
ko05211	Renal cell carcinoma	Human Diseases; Cancers	15.69%
ko05212	Pancreatic cancer	Human Diseases; Cancers	11.32%
ko05213	Endometrial cancer	Human Diseases; Cancers	15.38%
ko05214	Glioma	Human Diseases; Cancers	14.89%
ko05215	Prostate cancer	Human Diseases; Cancers	15.71%

ko05216	Thyroid cancer	Human Diseases; Cancers	10.71%
ko05217	Basal cell carcinoma	Human Diseases; Cancers	4.76%
ko05218	Melanoma	Human Diseases; Cancers	9.52%
ko05219	Bladder cancer	Human Diseases; Cancers	10.53%
ko05220	Chronic myeloid leukemia	Human Diseases; Cancers	7.02%
ko05221	Acute myeloid leukemia	Human Diseases; Cancers	8.89%
ko05222	Small cell lung cancer	Human Diseases; Cancers	16.67%
ko05223	Non-small cell lung cancer	Human Diseases; Cancers	13.04%
ko05224	Breast cancer	Human Diseases; Cancers	10.42%
ko05230	Central carbon metabolism in cancer	Human Diseases; Cancers	23.91%
ko05231	Choline metabolism in cancer	Human Diseases; Cancers	22.95%
ko05322	Systemic lupus erythematosus	Human Diseases; Immune diseases	17.78%
ko05323	Rheumatoid arthritis	Human Diseases; Immune diseases	18.46%
ko05340	Primary immunodeficiency	Human Diseases; Immune diseases	2.86%
ko05410	Hypertrophic cardiomyopathy (HCM)	Human Diseases; Cardiovascular diseases	4.11%
ko05414	Dilated cardiomyopathy	Human Diseases; Cardiovascular diseases	1.22%
ko05416	Viral myocarditis	Human Diseases; Cardiovascular diseases	7.89%

REFERENCES

[1] Raja Ragupathy, Sridhar Ravichandran, Md Safiur Rahman Mahdi, Douglas Huang, Elsa Reimer, Michael Domaratzki, and Sylvie Cloutier. Deep sequencing of wheat srna transcriptome reveals distinct temporal expression pattern of mirnas in response to heat, light and uv. *Scientific reports*, 6:39373, 2016.

[2] Praveen Guleria, Monika Mahajan, Jyoti Bhardwaj, and Sudesh Kumar Yadav. Plant small rnas: biogenesis, mode of action and their roles in abiotic stresses. *Genomics, proteomics & bioinformatics*, 9(6):183–199, 2011.

[3] Matthew W Jones-Rhoades, David P Bartel, and Bonnie Bartel. Micrornas and their regulatory roles in plants. *Annu. Rev. Plant Biol.*, 57:19–53, 2006.

[4] Agata Stepien, Katarzyna Knop, Jakub Dolata, Michal Taube, Mateusz Bajczyk, Maria Barciszewska-Pacak, Andrzej Pacak, Artur Jarmolowski, and Zofia Szweykowska-Kulinska. Posttranscriptional coordination of splicing and mirna biogenesis in plants. *Wiley Interdisciplinary Reviews: RNA*, 8(3), 2017.

[5] What do the mirna names/identifiers mean? `http://www.mirbase.org/help/nomenclature.shtml`. Accessed: 2018-08-02.

[6] Victor Ambros, Bonnie Bartel, David P Bartel, Christopher B Burge, James C Carrington, Xuemei Chen, Gideon Dreyfuss, Sean R Eddy, SAM Griffiths-Jones, Mhairi Marshall, et al. A uniform system for microrna annotation. *Rna*, 9(3):277–279, 2003.

[7] Xin Zhao, Huiyong Zhang, and Lei Li. Identification and analysis of the proximal promoters of microrna genes in arabidopsis. *Genomics*, 101(3):187–194, 2013.

[8] Fan Jia and Christopher D Rock. Mir846 and mir842 comprise a cistronic mirna pair that is regulated by abscisic acid by alternative splicing in roots of arabidopsis. *Plant molecular biology*, 81(4-5):447–460, 2013.

[9] Dawid Bielewicz, Malgorzata Kalak, Maria Kalyna, David Windels, Andrea Barta, Franck Vazquez, Zofia Szweykowska-Kulinska, and Artur Jarmolowski. Introns of plant pri-mirnas enhance mirna biogenesis. *EMBO reports*, 14(7):622–628, 2013.

[10] Qingpo Liu, Ying Feng, and Zhujun Zhu. Dicer-like (dcl) proteins in plants. *Functional & integrative genomics*, 9(3):277–286, 2009.

[11] Katarzyna Knop, Agata Stepien, Maria Barciszewska-Pacak, Michal Taube, Dawid Bielewicz, Michal Michalak, Jan W Borst, Artur Jarmolowski, and Zofia Szweykowska-Kulinska. Active 5´ splice sites regulate the biogenesis efficiency of arabidopsis micrornas derived from intron-containing genes. *Nucleic acids research*, 45(5):2757–2775, 2016.

[12] Stefan L Ameres and Phillip D Zamore. Diversifying microrna sequence and function. *Nature reviews Molecular cell biology*, 14(8):475, 2013.

[13] Julius Brennecke, Alexander Stark, Robert B Russell, and Stephen M Cohen. Principles of microrna–target recognition. *PLoS biology*, 3(3):e85, 2005.

[14] Josh T Cuperus, Noah Fahlgren, and James C Carrington. Evolution and functional diversification of mirna genes. *The Plant Cell*, pages tpc–110, 2011.

[15] Chenjiang You, Jie Cui, Hui Wang, Xinping Qi, Li-Yaung Kuo, Hong Ma, Lei Gao, Beixin Mo, and Xuemei Chen. Conservation and divergence of small rna pathways and micrornas in land plants. *Genome biology*, 18(1):158, 2017.

[16] Zhengrui Qin, Chunlian Li, Long Mao, and Liang Wu. Novel insights from non-conserved micrornas in plants. *Frontiers in plant science*, 5:586, 2014.

[17] Noah Fahlgren, Miya D Howell, Kristin D Kasschau, Elisabeth J Chapman, Christopher M Sullivan, Jason S Cumbie, Scott A Givan, Theresa F Law, Sarah R Grant, Jeffery L Dangl, et al. High-throughput sequencing of arabidopsis micrornas: evidence for frequent birth and death of mirna genes. *PloS one*, 2(2):e219, 2007.

[18] Arnaud T Djami-Tchatchou, Neeti Sanan-Mishra, Khayalethu Ntushelo, and Ian A Dubery. Functional roles of micrornas in agronomically important plants—potential as targets for crop improvement and protection. *Frontiers in plant science*, 8:378, 2017.

[19] J Han, ML Kong, H Xie, QP Sun, ZJ Nan, QZ Zhang, and JB Pan. Identification of mirnas and their targets in wheat (triticum aestivum l.) by est analysis. *Genet Mol Res*, 12:3793–805, 2013.

[20] Yu Zhang, Zaijie Wang, and Richard A Gemeinhart. Progress in microrna delivery. *Journal of controlled release*, 172(3):962–974, 2013.

[21] Jing Qu, Jian Ye, and Rongxiang Fang. Artificial microrna-mediated virus resistance in plants. *Journal of virology*, 81(12):6690–6699, 2007.

[22] Neha Sharma, Anita Tripathi, and Neeti Sanan-Mishra. Profiling the expression domains of a rice-specific microrna under stress. *Frontiers in plant science*, 6:333, 2015.

[23] Chiranjib Chakraborty, Ashish Ranjan Sharma, Garima Sharma, C George Priya Doss, and Sang-Soo Lee. Therapeutic mirna and sirna: moving from bench to clinic as next generation medicine. *Molecular Therapy-Nucleic Acids*, 8:132–143, 2017.

[24] Daniel H Chitwood and Marja CP Timmermans. Target mimics modulate mirnas. *Nature genetics*, 39(8):935, 2007.

[25] Jinping Zhao, Qingtao Liu, Pu Hu, Qi Jia, Na Liu, Kangquan Yin, Ye Cheng, Fei Yan, Jianping Chen, and Yule Liu. An efficient potato virus x-based microrna silencing in nicotiana benthamiana. *Scientific reports*, 6:20573, 2016.

[26] Ju Zhang, Deshui Yu, Yi Zhang, Kun Liu, Kedong Xu, Fuli Zhang, Jian Wang, Guangxuan Tan, Xianhui Nie, Qiaohua Ji, et al. Vacuum and co-cultivation agroinfiltration of (germinated) seeds results in tobacco rattle virus (trv) mediated whole-plant virus-induced gene silencing (vigs) in wheat and maize. *Frontiers in plant science*, 8:393, 2017.

[27] Lionel Navarro, Patrice Dunoyer, Florence Jay, Benedict Arnold, Nihal Dharmasiri, Mark Estelle, Olivier Voinnet, and Jonathan DG Jones. A plant mirna contributes to antibacterial resistance by repressing auxin signaling. *Science*, 312(5772):436–439, 2006.

[28] Carmen Simón-Mateo and Juan Antonio García. Microrna-guided processing impairs plum pox virus replication, but the virus readily evolves to escape this silencing mechanism. *Journal of virology*, 80(5):2429–2436, 2006.

[29] Tao Zhang, Yun-Long Zhao, Jian-Hua Zhao, Sheng Wang, Yun Jin, Zhong-Qi Chen, Yuan-Yuan Fang, Chen-Lei Hua, Shou-Wei Ding, and Hui-Shan Guo. Cotton plants export micrornas to inhibit virulence gene expression in a fungal pathogen. *Nature plants*, 2(10):16153, 2016.

[30] Om Prakash Gupta, Nand Lal Meena, Indu Sharma, and Pradeep Sharma. Differential regulation of micrornas in response to osmotic, salt and cold stresses in wheat. *Molecular biology reports*, 41(7):4623–4629, 2014.

[31] Zhanguo Xin and John Browse. Cold comfort farm: the acclimation of plants to freezing temperatures. *Plant, Cell & Environment*, 23(9):893–902, 2000.

[32] Francois Ouellet and Jean-Benoit Charron. Cold acclimation and freezing tolerance in plants. *eLS*, 2001.

[33] Sarvajeet Singh Gill and Narendra Tuteja. Reactive oxygen species and antioxidant machinery in abiotic stress tolerance in crop plants. *Plant physiology and biochemistry*, 48(12):909–930, 2010.

[34] Chao Jiang, Betty Iu, and Jas Singh. Requirement of a ccgac cis-acting element for cold induction of the bn115 gene from winter brassica napus. *Plant molecular biology*, 30(3):679–684, 1996.

[35] John E Koemel, Arron C Guenzi, Jeffrey A Anderson, and Edward L Smith. Cold hardiness of wheat near-isogenic lines differing in vernalization alleles. *Theoretical and Applied Genetics*, 109(4):839–846, 2004.

[36] Steve Babben, Edgar Schliephake, Philipp Janitza, Thomas Berner, Jens Keilwagen, Michael Koch, Fernando Alberto Arana-Ceballos, Sven Eduard Templer, Yuriy Chesnokov, Tatyana Pshenichnikova, et al. Association genetics studies on frost tolerance in wheat (triticum aestivum l.) reveal new highly conserved amino acid substitutions in cbf-a3, cbf-a15, vrn3 and ppd1 genes. *BMC genomics*, 19(1):409, 2018.

[37] Debbie Laudencia-Chingcuanco, Seedhabadee Ganeshan, Frank You, Brian Fowler, Ravindra Chibbar, and Olin Anderson. Genome-wide gene expression analysis supports a developmental model of low temperature tolerance gene regulation in wheat (triticum aestivum l.). *BMC genomics*, 12(1):299, 2011.

[38] Zahra Agharbaoui, Mickael Leclercq, Mohamed Amine Remita, Mohamed A Badawi, Etienne Lord, Mario Houde, Jean Danyluk, Abdoulaye Baniré Diallo, and Fathey Sarhan. An integrative approach to identify hexaploid wheat mirnaome associated with development and tolerance to abiotic stress. *BMC genomics*, 16(1):339, 2015.

[39] Bing Wang, Yan-fei Sun, Na Song, Jin-ping Wei, Xiao-jie Wang, Hao Feng, Zhi-yuan Yin, and Zhen-sheng Kang. Micrornas involving in cold, wounding and salt stresses in triticum aestivum l. *Plant physiology and biochemistry*, 80:90–96, 2014.

[40] Guoqi Song, Rongzhi Zhang, Shujuan Zhang, Yulian Li, Jie Gao, Xiaodong Han, Mingli Chen, Jiao Wang, Wei Li, and Genying Li. Response of micrornas to cold treatment in the young spikes of common wheat. *BMC genomics*, 18(1):212, 2017.

[41] Jarrod A Chapman, Martin Mascher, Aydın Buluç, Kerrie Barry, Evangelos Georganas, Adam Session, Veronika Strnadova, Jerry Jenkins, Sunish Sehgal, Leonid Oliker, et al. A whole-genome shotgun approach for assembling and anchoring the hexaploid bread wheat genome. *Genome biology*, 16(1):26, 2015.

[42] Blake C Meyers, Michael J Axtell, Bonnie Bartel, David P Bartel, David Baulcombe, John L Bowman, Xiaofeng Cao, James C Carrington, Xuemei Chen, Pamela J Green, et al. Criteria for annotation of plant micrornas. *The Plant Cell*, 20(12):3186–3190, 2008.

[43] Turgay Unver, Deana M Namuth-Covert, and Hikmet Budak. Review of current methodological approaches for characterizing micrornas in plants. *International journal of plant genomics*, 2009, 2009.

[44] Fatemeh Vafaee, Connie Diakos, Michaela B Kirschner, Glen Reid, Michael Z Michael, Lisa G Horvath, Hamid Alinejad-Rokny, Zhangkai Jason Cheng, Zdenka Kuncic, and Stephen Clarke. A data-driven, knowledge-based approach to biomarker discovery: application to circulating microrna markers of colorectal cancer prognosis. *NPJ systems biology and applications*, 4(1):20, 2018.

[45] Jerzy K Kulski. Next-generation sequencing—an overview of the history, tools, and "omic" applications. In *Next Generation Sequencing-Advances, Applications and Challenges*. Intech, 2016.

[46] Marcel Martin. Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnet. journal*, 17(1):pp–10, 2011.

[47] Heng Li and Nils Homer. A survey of sequence alignment algorithms for next-generation sequencing. *Briefings in bioinformatics*, 11(5):473–483, 2010.

[48] David J Lipman and William R Pearson. Rapid and sensitive protein similarity searches. *Science*, 227(4693):1435–1441, 1985.

[49] Stephen F Altschul, Warren Gish, Webb Miller, Eugene W Myers, and David J Lipman. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410, 1990.

[50] MS Waterman. Identification of common molecular subsequence. *Mol. Biol*, 147:195–197, 1981.

[51] W James Kent. Blat—the blast-like alignment tool. *Genome research*, 12(4):656–664, 2002.

[52] Heng Li, Jue Ruan, and Richard Durbin. Mapping short dna sequencing reads and calling variants using mapping quality scores. *Genome research*, pages gr–078212, 2008.

[53] Stephen M Rumble, Phil Lacroute, Adrian V Dalca, Marc Fiume, Arend Sidow, and Michael Brudno. Shrimp: accurate mapping of short color-space reads. *PLoS computational biology*, 5(5):e1000386, 2009.

[54] Heng Li and Richard Durbin. Fast and accurate short read alignment with burrows–wheeler transform. *bioinformatics*, 25(14):1754–1760, 2009.

[55] Ben Langmead, Cole Trapnell, Mihai Pop, and Steven L Salzberg. Ultrafast and memory-efficient alignment of short dna sequences to the human genome. *Genome biology*, 10(3):R25, 2009.

[56] Ruiqiang Li, Chang Yu, Yingrui Li, Tak-Wah Lam, Siu-Ming Yiu, Karsten Kristiansen, and Jun Wang. Soap2: an improved ultrafast tool for short read alignment. *Bioinformatics*, 25(15):1966–1967, 2009.

[57] Ben Langmead and Steven L Salzberg. Fast gapped-read alignment with bowtie 2. *Nature methods*, 9(4):357, 2012.

[58] Daehwan Kim, Ben Langmead, and Steven L Salzberg. Hisat: a fast spliced aligner with low memory requirements. *Nature methods*, 12(4):357, 2015.

[59] Mihaela Pertea, Daehwan Kim, Geo M Pertea, Jeffrey T Leek, and Steven L Salzberg. Transcript-level expression analysis of rna-seq experiments with hisat, stringtie and ballgown. *Nature protocols*, 11(9):1650, 2016.

[60] Daehwan Kim, Geo Pertea, Cole Trapnell, Harold Pimentel, Ryan Kelley, and Steven L Salzberg. Tophat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions. *Genome biology*, 14(4):R36, 2013.

[61] Mihaela Pertea, Geo M Pertea, Corina M Antonescu, Tsung-Cheng Chang, Joshua T Mendell, and Steven L Salzberg. Stringtie enables improved reconstruction of a transcriptome from rna-seq reads. *Nature biotechnology*, 33(3):290, 2015.

[62] Michael Zuker. Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic acids research*, 31(13):3406–3415, 2003.

[63] Zhi John Lu, Douglas H Turner, and David H Mathews. A set of nearest neighbor parameters for predicting the enthalpy change of rna secondary structure formation. *Nucleic acids research*, 34(17):4912–4924, 2006.

[64] Jessica S Reuter and David H Mathews. Rnastructure: software for rna secondary structure prediction and analysis. *BMC bioinformatics*, 11(1):129, 2010.

[65] Ivo L Hofacker. Vienna rna secondary structure server. *Nucleic acids research*, 31(13):3429–3431, 2003.

[66] Mickael Leclercq, Abdoulaye Banire Diallo, and Mathieu Blanchette. Computational prediction of the localization of micrornas within their pre-mirna. *Nucleic acids research*, 41(15):7200–7211, 2013.

[67] Anna Lukasik, Maciej Wójcikowski, and Piotr Zielenkiewicz. Tools4mirs–one place to gather all the tools for mirna analysis. *Bioinformatics*, 32(17):2722–2724, 2016.

[68] Matthew W Jones-Rhoades and David P Bartel. Computational identification of plant micrornas and their targets, including a stress-induced mirna. *Molecular cell*, 14(6):787–799, 2004.

[69] Fenglong Sun, Guanghui Guo, Jinkun Du, Weiwei Guo, Huiru Peng, Zhongfu Ni, Qixin Sun, and Yingyin Yao. Whole-genome discovery of mirnas and their targets in wheat (triticum aestivum l.). *BMC plant biology*, 14(1):142, 2014.

[70] Marcelo A German, Manoj Pillay, Dong-Hoon Jeong, Amit Hetawal, Shujun Luo, Prakash Janardhanan, Vimal Kannan, Linda A Rymarquis, Kan Nobuta, Rana German, et al. Global identification of microrna–target rna pairs by parallel analysis of rna ends. *Nature biotechnology*, 26(8):941, 2008.

[71] Xiaozeng Yang and Lei Li. mirdeep-p: a computational tool for analyzing the microrna transcriptome in plants. *Bioinformatics*, 27(18):2614–2615, 2011.

[72] Marc R Friedländer, Sebastian D Mackowiak, Na Li, Wei Chen, and Nikolaus Rajewsky. mirdeep2 accurately identifies known and hundreds of novel microrna genes in seven animal clades. *Nucleic acids research*, 40(1):37–52, 2011.

[73] Ramanjulu Sunkar and Guru Jagadeeswaran. In silico identification of conserved micrornas in large number of diverse plant species. *BMC plant biology*, 8(1):37, 2008.

[74] Yonggan Wu, Bo Wei, Haizhou Liu, Tianxian Li, and Simon Rayner. Mirpara: a svm-based software tool for prediction of most probable microrna coding regions in genome scale sequences. *BMC bioinformatics*, 12(1):107, 2011.

[75] Stuart J Lucas and Hikmet Budak. Sorting the wheat from the chaff: identifying mirnas in genomic survey sequences of triticum aestivum chromosome 1al. *PLoS One*, 7(7):e40859, 2012.

[76] Bernardo J Clavijo, Luca Venturini, Christian Schudoma, Gonzalo Garcia Accinelli, Gemy Kaithakottil, Jonathan Wright, Philippa Borrill, George Kettleborough, Darren Heavens, Helen Chapman, et al. An improved assembly and annotation of the allohexaploid wheat genome identifies complete families of agronomic genes and provides genomic evidence for chromosomal translocations. *Genome research*, 2017.

[77] Ana Kozomara and Sam Griffiths-Jones. mirbase: annotating high confidence micrornas using deep sequencing data. *Nucleic acids research*, 42(D1):D68–D73, 2013.

[78] Zhenhai Zhang, Jingyin Yu, Daofeng Li, Zuyong Zhang, Fengxia Liu, Xin Zhou, Tao Wang, Yi Ling, and Zhen Su. Pmrd: plant microrna database. *Nucleic acids research*, 38(suppl_1):D806–D813, 2009.

[79] Mohamed Amine Remita, Etienne Lord, Zahra Agharbaoui, Mickael Leclercq, Mohamed A Badawi, Fathey Sarhan, and Abdoulaye Baniré Diallo. A novel comprehensive wheat mirna database, including related bioinformatics software. *Current Plant Biology*, 7:31–33, 2016.

[80] Rudi Appels, Kellye Eversole, Catherine Feuillet, Beat Keller, Jane Rogers, Nils Stein, Curtis J Pozniak, Frédéric Choulet, Assaf Distelfeld, Jesse Poland, et al. Shifting the limits in wheat research and breeding using a fully annotated reference genome. *Science*, 361(6403):eaar7191, 2018.

[81] Mayumi Nakano, Kan Nobuta, Kalyan Vemaraju, Shivakundan Singh Tej, Jeremy W Skogen, and Blake C Meyers. Plant mpss databases: signature-based transcriptional resources for analyses of mrna and small rna. *Nucleic acids research*, 34(suppl_1):D731–D735, 2006.

[82] Gene Ontology Consortium et al. The gene ontology (go) database and informatics resource. *Nucleic acids research*, 32(suppl 1):D258–D261, 2004.

[83] Minoru Kanehisa, Miho Furumichi, Mao Tanabe, Yoko Sato, and Kanae Morishima. Kegg: new perspectives on genomes, pathways, diseases and drugs. *Nucleic Acids Research*, 45(D1):D353–D361, 2017.

[84] Mickael Leclercq, Abdoulaye Baniré Diallo, and Mathieu Blanchette. Prediction of human mirna target genes using computationally reconstructed ancestral mammalian sequences. *Nucleic acids research*, 45(2):556–566, 2016.

[85] Jianping Zhou, Yan Cheng, Meiqi Yin, Ennian Yang, Wenping Gong, Cheng Liu, Xuelian Zheng, Kejun Deng, Zhenglong Ren, and Yong Zhang. Identification of novel mirnas and mirna expression profiling in wheat hybrid necrosis. *PloS one*, 10(2):e0117507, 2015.

[86] Xingli Ma, Zeyu Xin, Zhiqiang Wang, Qinghua Yang, Shulei Guo, Xiaoyang Guo, Liru Cao, and Tongbao Lin. Identification and comparative analysis of differentially expressed mirnas in leaves of two wheat (triticum aestivum l.) genotypes during dehydration stress. *BMC Plant Biology*, 15(1):21, 2015.

[87] Habibullah Khan Achakzai, Muhammad Younas Khan Barozai, Muhammad Din, Iftekhar Ahmed Baloch, and Abdul Kabir Khan Achakzai. Identification and annotation of newly conserved micrornas and their targets in wheat (triticum aestivum l.). *PloS one*, 13(7):e0200033, 2018.

[88] Ren-Hao Pan, Lin-Yu Tseng, I-En Liao, Chien-Lung Chan, K Robert Lai, and Kai-Biao Lin. Design of an ngs microrna predictor using multilayer hierarchical mapreduce framework. In *Data Science and Advanced Analytics (DSAA), 2015. 36678 2015. IEEE International Conference on*, pages 1–8. IEEE, 2015.

[89] 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*, page 47. ACM, 2014.

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

[91] Lin Dai, Xin Gao, Yan Guo, Jingfa Xiao, and Zhang Zhang. Bioinformatics clouds for big data manipulation. *Biology direct*, 7(1):43, 2012.

[92] Aisling O'Driscoll, Jurate Daugelaite, and Roy D Sleator. 'big data', hadoop and cloud computing in genomics. *Journal of biomedical informatics*, 46(5):774–781, 2013.

[93] Christina Liu, Da-Yu Wu, and Jonathan D Pollock. Bioinformatic challenges of big data in non-coding rna research. *Frontiers in genetics*, 3:178, 2012.

[94] Vivien Marx. Biology: The big challenges of big data. *Nature*, 498(7453):255–260, 2013.

[95] Quan Zou, Xu-Bin Li, Wen-Rui Jiang, Zi-Yu Lin, Gui-Lin Li, and Ke Chen. Survey of mapreduce frame operation in bioinformatics. *Briefings in bioinformatics*, page bbs088, 2013.

[96] Aisling O'Driscoll, Vladislav Belogrudov, John Carroll, Kai Kropp, Paul Walsh, Peter Ghazal, and Roy D Sleator. Hblast: Parallelised sequence similarity–a hadoop mapreducable basic local alignment search tool. *Journal of biomedical informatics*, 54:58–64, 2015.

[97] Abhinav Nellore, Leonardo Collado-Torres, Andrew E Jaffe, José Alquicira-Hernández, Christopher Wilks, Jacob Pritt, James Morton, Jeffrey T Leek, and Ben Langmead. Rail-rna: Scalable analysis of rna-seq splicing and coverage. *Bioinformatics*, page btw575, 2016.

[98] Eric Bonnet, Ying He, Kenny Billiau, and Yves Van de Peer. Tapir, a web server for the prediction of plant microrna targets, including target mimics. *Bioinformatics*, 26(12):1566–1568, 2010.

[99] Xinbin Dai and Patrick Xuechun Zhao. psrnatarget: a plant small rna target analysis server. *Nucleic acids research*, 39(suppl 2):W155–W159, 2011.

[100] 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*, page gkv403, 2015.

[101] Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica. Spark: Cluster computing with working sets. *HotCloud*, 10(10-10):95, 2010.

[102] Todd K Moon. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60, 1996.

[103] Simon Andrews et al. Fastqc: a quality control tool for high throughput sequence data. 2010.

[104] Arnoud J Kal, Anton Jan van Zonneveld, Vladimir Benes, Marlene van den Berg, Marian Groot Koerkamp, Kaj Albermann, Normann Strack, Jan M Ruijter, Alexandra Richter, Bernard Dujon, et al. Dynamics of gene expression revealed by comparison of serial analysis of gene expression transcript profiles from yeast grown on two different carbon sources. *Molecular biology of the cell*, 10(6):1859–1872, 1999.

[105] Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the royal statistical society. Series B (Methodological)*, pages 289–300, 1995.

[106] Thomas Cokelaer, Dennis Pultz, Lea M Harder, Jordi Serra-Musach, and Julio Saez-Rodriguez. Bioservices: a common python package to access biological web services programmatically. *Bioinformatics*, 29(24):3241–3242, 2013.

[107] Xinbin Dai, Zhaohong Zhuang, and Patrick Xuechun Zhao. psrnatarget: a plant small rna target analysis server (2017 release). *Nucleic acids research*, 2018.

[108] Fengling Li, Xuemin Wu, Patricia Lam, David Bird, Huanquan Zheng, Lacey Samuels, Reinhard Jetter, and Ljerka Kunst. Identification of the wax ester synthase/acyl-coenzyme a: diacylglycerol acyltransferase wsd1 required for stem wax ester biosynthesis in arabidopsis. *Plant physiology*, 148(1):97–107, 2008.

[109] Karin Ljung. Auxin metabolism and homeostasis during plant development. *Development*, 140(5):943–950, 2013.

[110] Zheng-Jun Xu, Masatoshi Nakajima, Yoshihito Suzuki, and Isomaro Yamaguchi. Cloning and characterization of the abscisic acid-specific glucosyltransferase gene from adzuki bean seedlings. *Plant physiology*, 129(3):1285–1295, 2002.

[111] Masanori Okamoto, Ayuko Kuwahara, Mistunori Seo, Tetsuo Kushiro, Tadao Asami, Nobuhiro Hirai, Yuji Kamiya, Tomokazu Koshiba, and Eiji Nambara. Cyp707a1 and cyp707a2, which encode abscisic acid 8-hydroxylases, are indispensable for proper control of seed dormancy and germination in arabidopsis. *Plant Physiology*, 141(1):97–107, 2006.

[112] Celine Vanhee, Grzegorz Zapotoczny, Danièle Masquelier, Michel Ghislain, and Henri Batoko. The arabidopsis multistress regulator tspo is a heme binding membrane protein and a potential scavenger of porphyrins via an autophagy-dependent degradation mechanism. *The Plant Cell*, pages tpc–110, 2011.

[113] Vivek Verma, Pratibha Ravindran, and Prakash P Kumar. Plant hormone-mediated regulation of stress responses. *BMC plant biology*, 16(1):86, 2016.

[114] MHPJC Ashraf and PJC Harris. Photosynthesis under stressful environments: an overview. *Photosynthetica*, 51(2):163–190, 2013.

[115] Naoto Sano, Loïc Rajjou, Helen M North, Isabelle Debeaujon, Annie Marion-Poll, and Mitsunori Seo. Staying alive: molecular aspects of seed longevity. *Plant and Cell Physiology*, 57(4):660–674, 2015.

[116] David Amar, Tom Hait, Shai Izraeli, and Ron Shamir. Integrated analysis of numerous heterogeneous gene expression profiles for detecting robust disease-specific biomarkers and proposing drug targets. *Nucleic acids research*, 43(16):7779–7789, 2015.

[117] Yishai Shimoni. Association between expression of random gene sets and survival is evident in multiple cancer types and may be explained by sub-classification. *PLoS Computational Biology*, 14(2):e1006026, 2018.

[118] Sort benchmark. `http://sortbenchmark.org/`. Accessed: 2018-08-25.