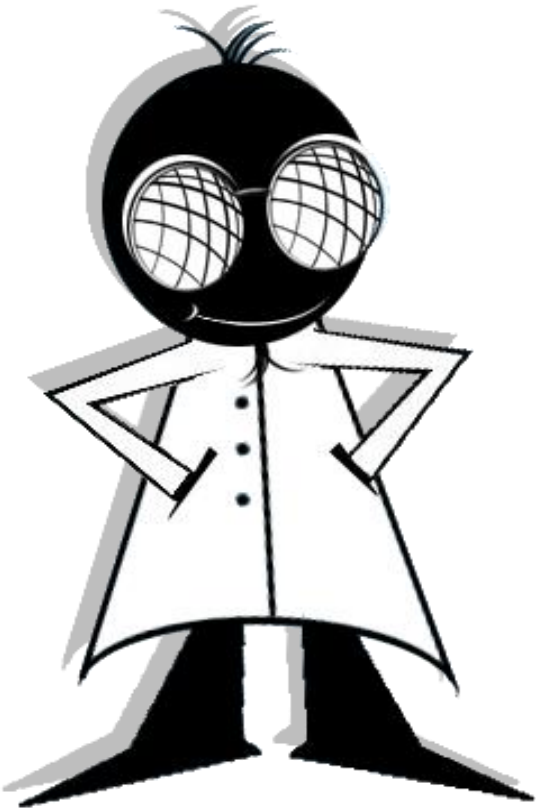
MiGA

Microsatellite Genome Analysis



Machine Learning and Knowledge Discovery Group

Computer Science Department

Aristotle University of Thessaloniki

In collaboration with the laboratory of

Population Genetics of Animal Organisms

Department of Genetics, Development and Molecular Biology

School of Biology

Aristotle University of Thessaloniki

Version 1.01

1. User 's Guide

In this section we are going to describe the MiGA application.

1.1 Getting Started

To start MiGA application, double click on the MiGA.jar icon in the installation folder. MiGA's mainframe has 2 tabs. The first tab named "Ensembl or local" implements SSR search for organisms included at ensembl database. The second tab named "File from User" implements SSR search for FASTA file chosen by the user.

"Ensembl or local"

In that case, choose the organisms (one or more) you want to analyse and then click on the button "Check last updated".

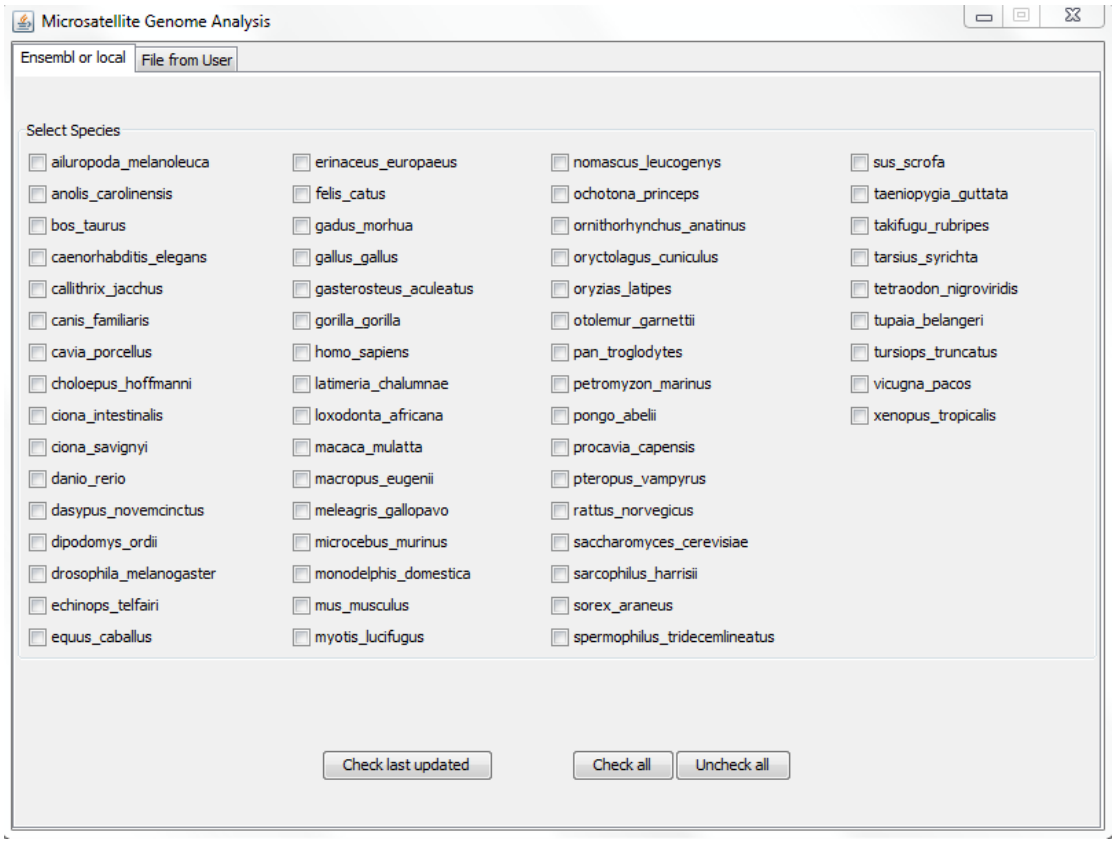


Figure 1: List of Organisms

In the next window (figure 1), there is a list with the organisms chosen before. For each one of them, there is either a timestamp with the date and time from the last update in the local database or a message indicates that the organism have to be updated before proceed. The next figure (figure 2) shows the message that the application shows for update or not.

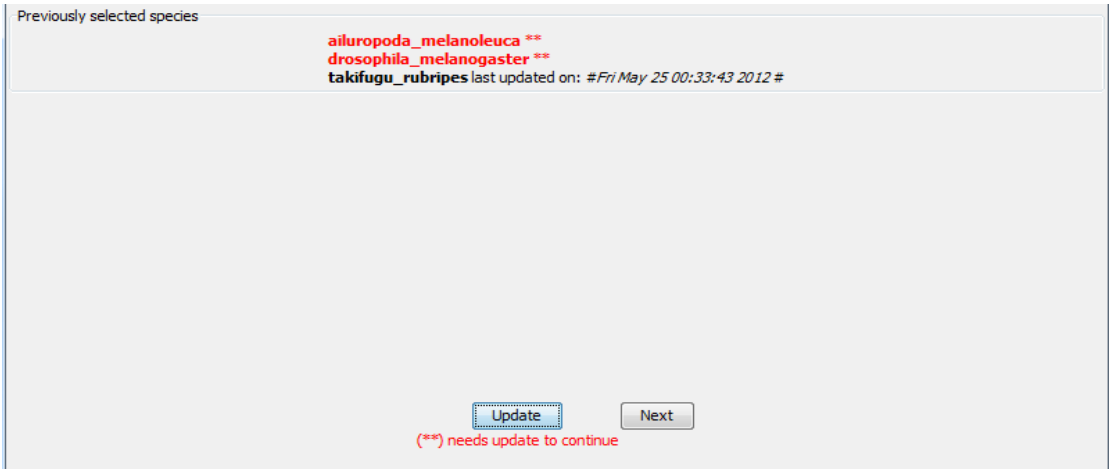


Figure 2: Organisms do need to be updated and others do not

CAUTION! Internet connection is necessary for this step. If you choose Update, program asks to choose those organisms you want to update. You can choose to update every organism you want even if it is already updated, but it is wise to do that only if the timestamp is really old because the update operation requires several hours. Choose the checkbox for each organism you want to update, click on the button "Update" and let the program update your local database.

CAUTION!! You can minimize the window but under no circumstances should you close the window, the internet connection or the computer. After minimizing the window – during update – the window will appear black until the end of the update. There is no reason to worry about, the program continues execution normally and will proceed to the next frame as long as the update is completed. Updating speed depends on the internet connection's speed and the computer's abilities. Update might need several hours to complete. As soon as the update is completed, the previous window will show up and you can now choose the "Next" button.

Once every chosen organism has been updated, you can proceed analysing their genome in order to find SSR's.

"File from User"

In this tab, you can choose the FASTA file you want to analyse. Click on the "Browse" button and choose the file. Then, give a project name and click on "Start scanning" button. Once file scanning is completed the next window will show up.

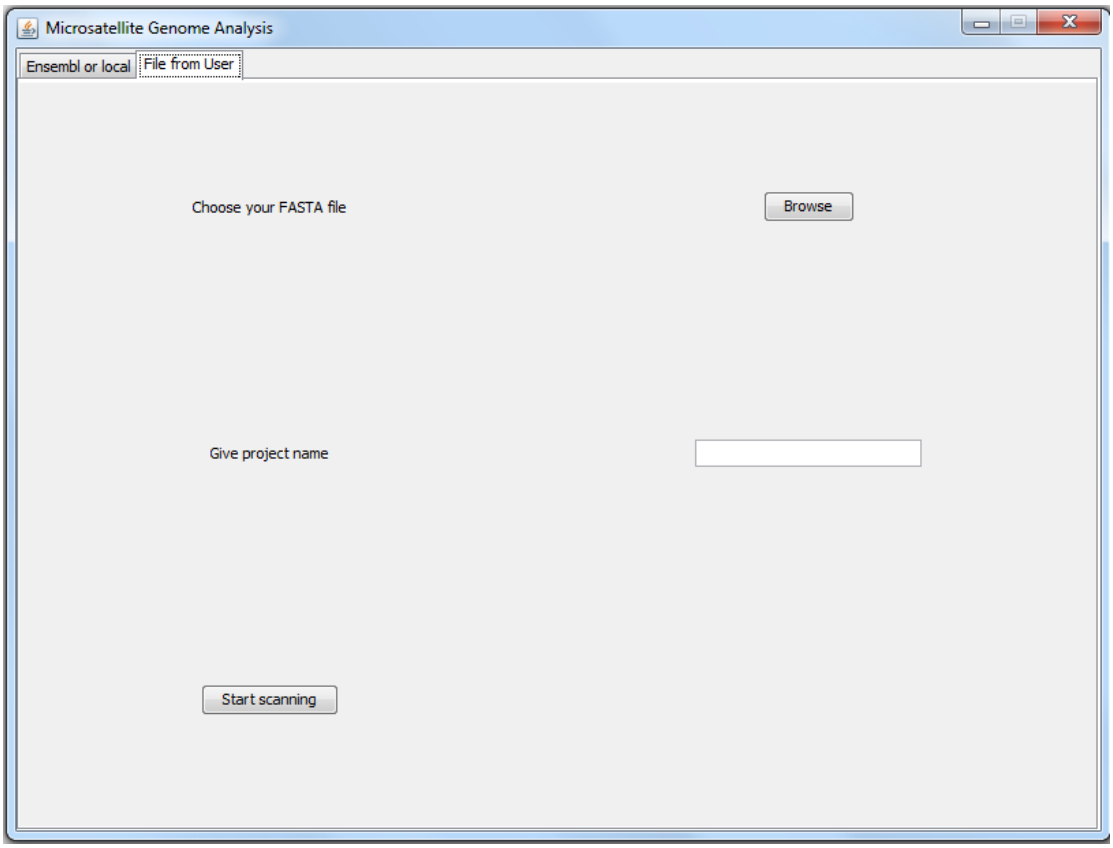


Figure 3: File from User

1.2 Final Window

The final window of the program has 2 tabs. This window will appear either you have initially chosen “Ensembl or local” tab or “File from User” tab. In the first tab named “Statistics” the user have to insert the parameters according to which the program will search the genome for SSRs. The second tab, named “Sequence Retrieval” returns the SSR’s sequence along with its flanking regions.

Statistics

The main choices on this tab are:

- ✓ Perfect
- ✓ Imperfect
- ✓ Compound
- ✓ Perfect Compound
- ✓ Imperfect Compound

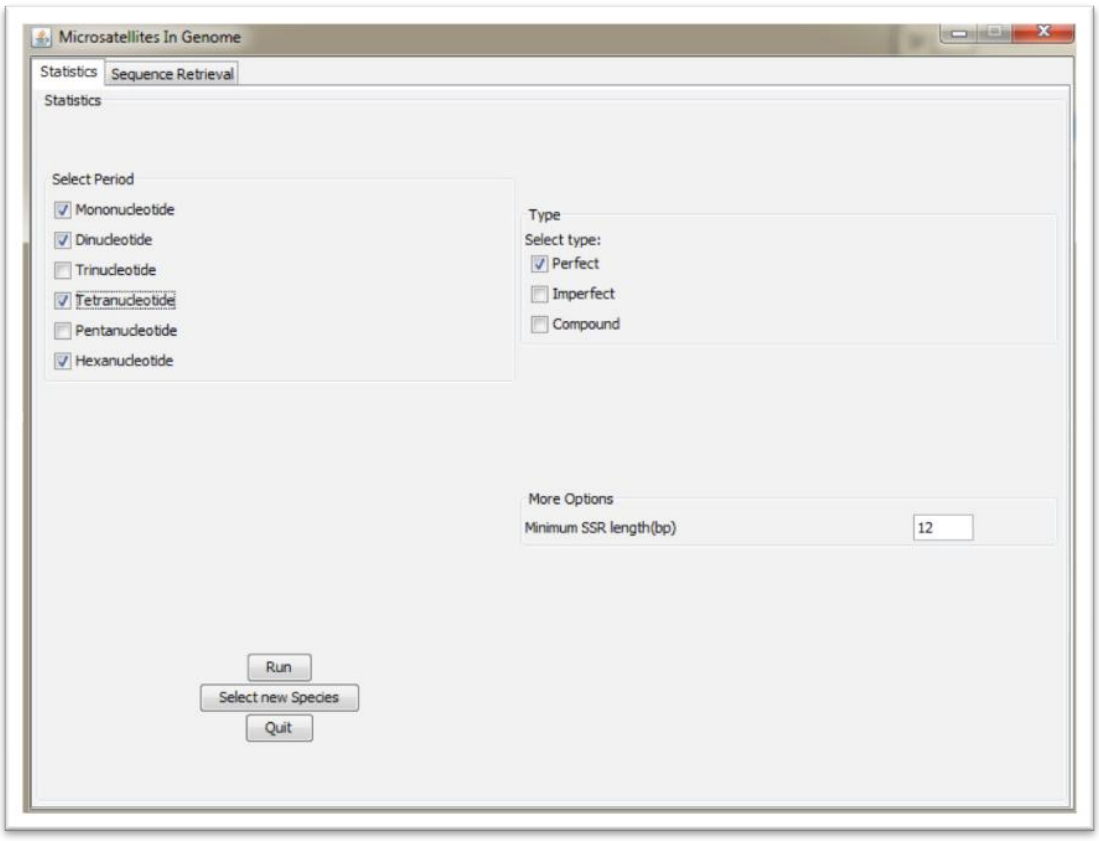
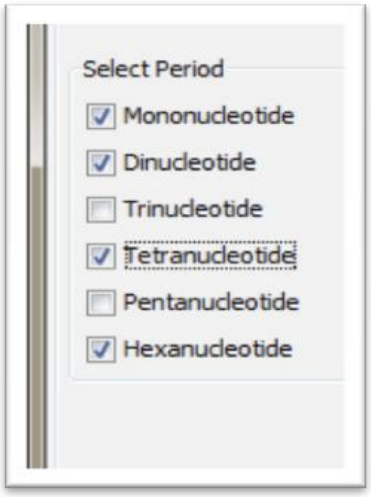


Figure 4: Statistics Window

If choices 1. *Perfect* or 2. *Imperfect* have been selected it is necessary to choose the Motif/Period (Mono-Di-Tri-Tetra-Penta-Hexa nucleotide) of the microsatellites SSRs.

More specifically for each SSR type chosen:



Perfect

In case perfect SSRs are chosen it is necessary to fill in the minumum SSR length (bp – base pair).

Imperfect

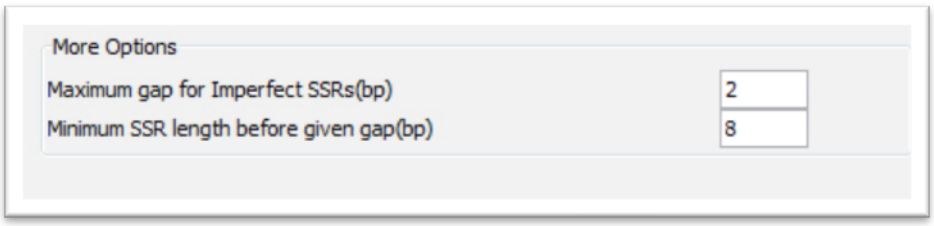
In case imperfect SSRs are chosen it is necessary to fill in the maximum gap for imperfect SSRs (bp – base pair) and the minimum SSR length before the given gap (bp – base pair). For Instance:

- ✓ Maximum gap for imperfect SSRs: 2 bp
- ✓ Minimum SSR length before given gap: 8 bp

(Maximum gap)

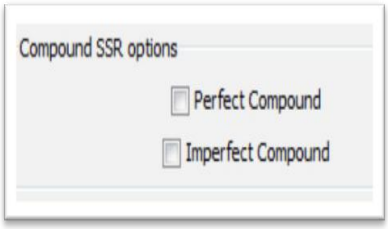
*AGAGAGAG**TC**AGAGAGAG*

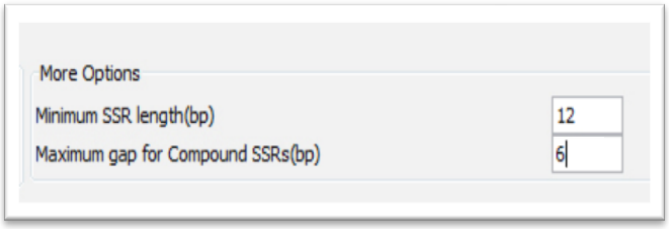
(Minimum SSR length)



Compound Perfect/Imperfect

In case compound SSRs are chosen you have to choose the type of the compound SSRs you are seeking for (Perfect/Imperfect Compound).





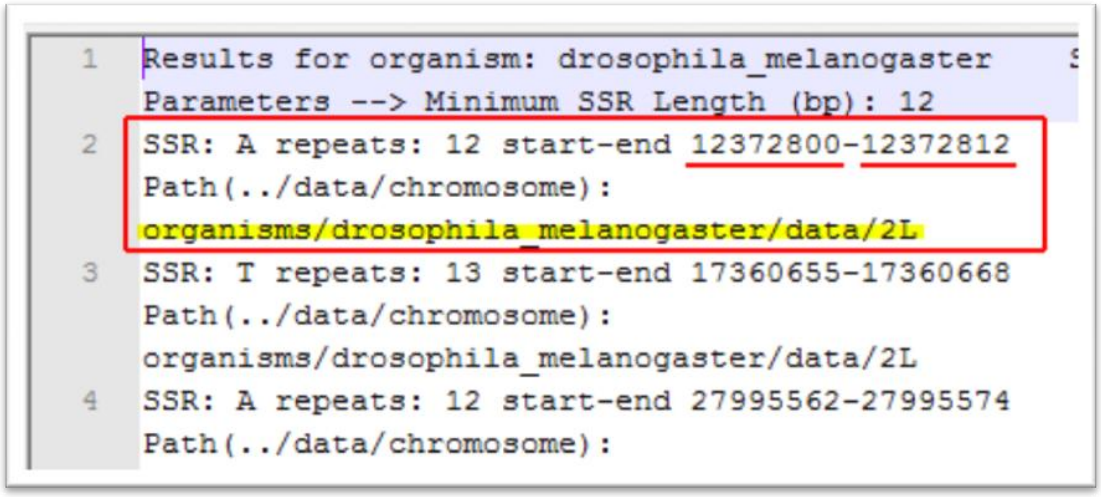
After that, it is necessary to fill in the minimum SSR length (bp – base pair) and the maximum gap for compound SSRs (bp) parameters.

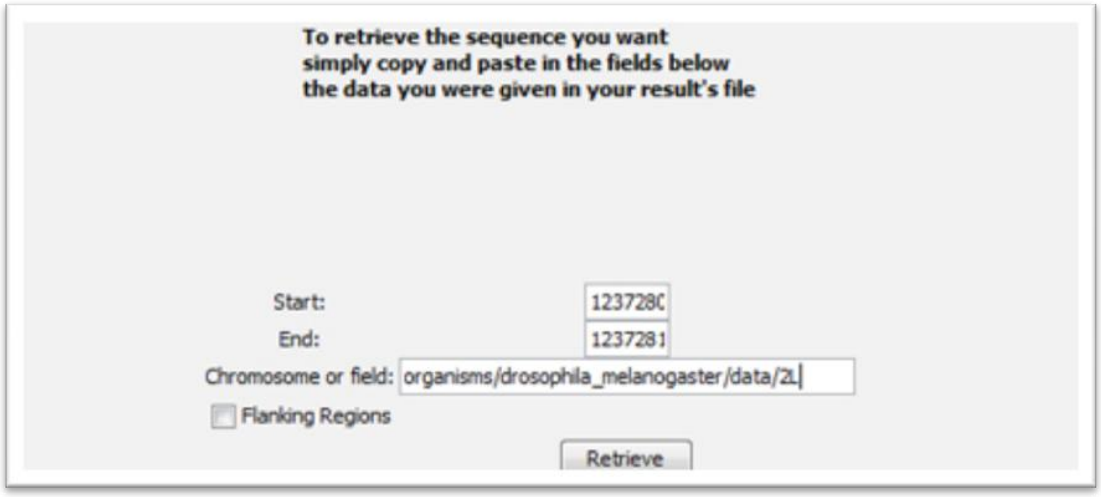
Once select search parameters click on the "Run" button and wait for a few minutes until results are shown. One .txt file opens in which you can see all the wanted SSRs found with information about their period (for perfect SSRs), their start and end base pair in the genome as well as the Path (see Sequence Retrieval). In addition, in the program's folder there is now a folder named **organisms** including folders for each one of the organisms you have selected. In every one of them there is a folder named **results** with the results file produced. Besides that, there is also one folder named **stats** with the statistical representation (summary statistics) for each one of the organisms chosen.

Sequence Retrieval

This is the other tab on the final window. You can fill in Start, End, and Path (Chromosome or field) fields for one SSR that has been found and the program produces SSR's sequence for any use (e.g. primers design etc.). More specific, copy and paste the information from the results file into the fields and then click on the

"Retrieve" button. There is also an option with which you can retrieve flanking regions along with the SSR.





2. Results – Statistics

The program developed provides the user with two main functions. The first function has to do with genome retrieval and analysis used in microsatellite loci search and can assist in the development of primers. The second function has to do with the statistical presentation of these results which can be of great help understanding the evolutionary relationship among species.

Initially, the program accesses the local database (LoBid). Both retrieved data and the results obtained from the execution of the program on them will be stored in LoBid.

Then, the user is asked to choose the type of biological data that wants to analyze. There are two possible modes. The user can either chooses to update organisms via Ensembl (online biological database) and then analyze their genome, or to analyze already stored in LoBid data, as well as sequences saved in FASTA form (.fa or .fasta) from user input files. Once the data retrieval is finished, they are saved in LoBid database and txt files (slices), from where, eventually, will start the sequence analysis for microsatellite loci search.

2.1 Using Genomes from Ensembl

In this section we will describe MiGA's results when the user chooses to analyze genomes from organisms downloaded from ensembl.

Results

The results of the program are displayed to the user in text files. Each file contains as a header the organism name and all the search parameters used. Then, there are information about SSRs found during the process. These information are about:

- ✓ The motif of the repeated bases.

- ✓ The number of repeats for the motif in the SSR found.
- ✓ The start and the end of the SSR in the sequence.
- ✓ The path with which the user can retrieve the SSR sequence and its flanking regions.

Statistical analysis of SSRs (Tables Description)

Along with the results obtained in text files, the program provides a statistical representation of these. A separate text file is created which includes tables organized by the SSR type found during the search. At the bottom of the text file there is a table with the summary statistics of the search. Below there are some notes explaining the structure and content of every statistical table:

Every table has to do with one type of SSR (perfect, imperfect, perfect_compound, imperfect_compound).

Every table has the folder name (organism name) and the search parameters, given by the user, as a header. In this folder will be every result and file related with each organism analysis.

Both perfect (Figure 5) and imperfect (Figure 6) SSRs tables consist of 10 columns in each of which it is shown each of the following statistical information:

- ✓ The type of the repeated motif (mono-/di-/tri-/tetra-/penta-/hexa-nucleotide).
- ✓ The count of each type of SSR found in the analyzed genome. *e.g. 21059 mononucleotide repeats found in the genome – picture 1.*
- ✓ The total size that has each type of SSR found (count in base pair (bp)).
- ✓ The hundred percent (%) incidence of the nucleotide base of Adenine (A) on SSR found. *e.g. 45.61% of the nucleotide bases found in the perfect SSR of drosophila is adenine – picture 1.*
- ✓ The hundred percent (%) incidence of the nucleotide base of Thymine (T) on SSR found.
- ✓ The hundred percent (%) incidence of the nucleotide base of Cytosine (C) on SSR found.

- ✓ The hundred percent (%) incidence of the nucleotide base of Guanine (G) on SSR found.
- ✓ The relative frequency of each SSR type (count of SSR type (bp)/ total count of SSR in genome (bp)). *e.g. for the perfect SSR of drosophila is 21059/247191 = 0.085 – picture 1.*
- ✓ The relative abundance of each SSR type related to the genome length analyzed (total size of each type of SSR (bp)/ total genome length (bp)). *e.g. for the perfect SSR of drosophila is 313362/168736537 = 0.002 – picture 1.*
- ✓ The relative abundance of each SSR type related to the genome length analyzed (total size of each type of SSR (bp)/ total length of SSR found in the genome (bp)). *e.g. for the perfect SSR of drosophila is 313362/3615206 = 0.087 – picture 1.*

******* Perfect SSRs *******
Results for project: drosophila_melanogaster Search Parameters --> Minimum SSR Length (bp): 12

motif	count	bp	A%	T%	C%	G%	Relative Frequency	Relative Abundance 1	Relative Abundance 2
mono	21059	313362	45,61	43,558	5,711	5,121	0,085	0,002	0,087
di	19263	315994	32,321	33,221	17,274	17,184	0,078	0,002	0,087
tri	23273	342330	29,93	29,758	20,234	20,078	0,094	0,002	0,095
tetra	22824	333556	35,587	34,964	15,062	14,387	0,092	0,002	0,092
penta	12882	446760	38,834	28,839	18,45	13,876	0,052	0,003	0,124
hexa	147890	1863204	31,532	31,411	18,524	18,533	0,598	0,011	0,515
TOTAL	247191	3615206	33,946	32,476	17,137	16,441	1	0,021	1

Genome length (bp): 168736537
Relative Frequency: Count of each motif type / total SSR count
Relative Abundance 1: Mbp of each motif type / total sequence Mbp
Relative Abundance 2: Mbp of each motif type / total microsatellites Mbp

Figure 5. Statistical representation of perfect SSRs (example)

******* ImPerfect SSRs *******
Results for project: drosophila_melanogaster Search Parameters --> Maximum Gap for ImPerfect SSRs (bp): 2 minimum SSR length before given gap (bp): 8

motif	count	bp	A%	T%	C%	G%	Relative Frequency	Relative Abundance 1	Relative Abundance 2
mono	1213	28445	45,266	44,479	6,071	4,184	0,147	0	0,116
di	2724	2724	36,571	36,327	14,148	12,954	0,331	0	0,011
tri	1208	37792	37,743	36,764	12,347	13,146	0,147	0	0,155
tetra	1969	44416	36,248	33,877	15,049	14,826	0,239	0	0,182
penta	927	54707	39,712	31,618	15,704	12,967	0,112	0	0,224
hexa	201	10556	27,814	32,579	19,884	19,723	0,024	0	0,043
TOTAL	8242	244557	38,03	35,682	13,689	12,599	1	0,001	1

Genome length (bp): 168736537
Relative Frequency: Count of each motif type / total SSR count
Relative Abundance 1: Mbp of each motif type / total sequence Mbp
Relative Abundance 2: Mbp of each motif type / total microsatellites Mbp

Figure 6: Statistical representation of imperfect SSRs (example)

Compound SSRs tables (Figure 7 and 8) also consist of 10 columns. The only difference between compound SSRs tables and those of perfect and imperfect is the first from the left column where there is now the type of the SSRs found (Compound Perf. or Compound Imper.) instead of the type of the repeated motif. As a result of the lack of a specific type of one repeated motif, the table has one row – instead of six (perfect and imperfect) – which is the total statistics of the compound SSRs found.

******* Compound Perfect SSRs *******
Results for project: drosophila_melanogasterSearch Parameters --> Maximum Gap for Perfect Compound SSRs (bp) : 4minimum SSR length (bp): 12

	count	bp	A%	T%	C%	G%	Relative Frequency	Relative Abundance 1	Relative Abundance 2
Compound Perf.	5626	181930	31,351	28,43	20,389	19,831	1	0,001	1
TOTAL	5626	181930	31,351	28,43	20,389	19,831	1	0,001	1

Genome length (bp): 168736537
Relative Frequency: Count of each motif type / total SSR count
Relative Abundance 1: Mbp of each motif type / total sequence Mbp
Relative Abundance 2: Mbp of each motif type / total microsatellites Mbp

Figure 7: Statistical representation of compound perfect SSRs (example)

******* Compound Imperfect SSRs *******
Results for project: drosophila_melanogasterSearch Parameters --> Maximum Gap for Imperfect Compound SSRs(bp) : 4minimum SSR length(bp): 12

	count	bp	A%	T%	C%	G%	Relative Frequency	Relative Abundance 1	Relative Abundance 2
Compound Imper.	598	62973	35,272	29,967	18,1	16,661	1	0	1
TOTAL	598	62973	35,272	29,967	18,1	16,661	1	0	1

Genome length (bp): 168736537
Relative Frequency: Count of each motif type / total SSR count
Relative Abundance 1: Mbp of each motif type / total sequence Mbp
Relative Abundance 2: Mbp of each motif type / total microsatellites Mbp

Figure 8: Statistical representation of compound perfect SSRs (example)

The summary statistics table (Figure 9) appears at the bottom of the txt file and has similar format with the compound SSRs tables. In the first column of it there are the 4 types of SSRs that can be found and in the bottom of the table there is one row for the total statistics of the SSRs found.

******* SUMMARY TABLE *******

type	count	bp	A%	T%	C%	G%	Relative Frequency	Relative Abundance 1	Relative Abundance 2
Perfect	247191	3615206	29,898	28,603	15,094	14,48	0,945	0,021	0,881
Imperfect	8242	244557	2,266	2,126	0,816	0,751	0,031	0,001	0,06
Compound Per	5626	181930	1,39	1,26	0,904	0,879	0,022	0,001	0,044
Compound Imp	598	62973	0,541	0,46	0,278	0,256	0,002	0	0,015
TOTAL	261657	4104666	34,095	32,449	17,091	16,366	1	0,024	1

Genome length (bp): 168736537
Relative Frequency: Count of each motif type / total SSR count
Relative Abundance 1: Mbp of each motif type / total sequence Mbp
Relative Abundance 2: Mbp of each motif type / total microsatellites Mbp

Figure 9: Statistical representation of all SSRs found (example)

2.2 Using Input Files from User

In this section we will describe MiGA's results when the user chooses to analyze an own FASTA file with biological sequences.

Results

The results of the program are displayed to the user in text files. Each file contains as a header the project name given by the user and all the search parameters used. Then, there are information about SSRs found during the process. These information are about:

- ✓ The microsatellite sequence as it is.
- ✓ The start and the end of the SSR in the sequence.
- ✓ The path with which the user can retrieve the SSR sequence and its flanking regions.

Statistical analysis of SSRs (Tables Description)

Along with the results obtained in text files, the program provides a statistical representation of these. A separate text file is created which includes tables organized by the SSR type found during the search. At the bottom of the text file there is a table with the summary statistics of the search. All tables have the same format with those created when analysing genomes downloaded from ensembl.

2.3 Sequence and Flanking Region Retrieval

One of the major goals of the research on genome analysis in bioinformatics, and more specifically the analysis of microsatellite sequences, is to clarify the composition of the flanking regions of each microsatellite loci. The importance of this goal is understandable if someone consider that flanking regions provide a great

guide to the efforts of biologists to gain “access” on genome parts necessary for each one of their researches.

The program provides the user the ability to retrieve microsatellite sequence found using the path that is stored in the results files as mentioned before. Both in the case of using genomes from ensembl and in the case of upload file from the user, next to the search parameters tab there is a tab for sequence retrieval.

The one thing the user has to do in order to retrieve one SSR sequence is to copy the path, the start and the end of the SSR as given in the results file. The program identifies the SSR and the slice from which it is retrieved in the local database (LoBid) and provides the user interface with the sequence as it is. In the same way, the program provides the user the additional opportunity of flanking regions retrieval for each SSR, which are the regions before and after the microsatellite sequence.

3. Pseudocode

Below you can find in pseudocode important sections of our application.

```
function findMonoNucleotideSSR(seq){
        for i=0 to seq.length-1{
                if seq_char[i] != 'N'{
                        if seq_char[i] == seq_char[i+1]{
                                CurrentSSR = seq_char[i];
                                Repeats = 2;
                                i++;
                                while seq_char[i] == seq_char[i+1]{
                                        Repeats++;
                                        i++;
                                }
                                Save -> SSR , Repeats , End
                        }
                }
        }
}
```

```
function findXNucleotideSSR(String seq, int motif_len){
        for i=0 to seq.length-1{
//to be sure that using recursion it will not give OutOfBoundsException
        If(seq.Substring(i,i+motif_len) =seq.Substring(i+motif_len,i+2*motif_len)){
                        CurrentSSR = seq_char[i];
                        Repeats = 2;
                        i=i+motif_len;
                        while seq_char[i] == seq_char[i+1]{
                                Repeats++;
                                i=i+motif_len;
                        }
                        Save -> SSR , Repeats , End
                        Recursively call findXNucleotideSSR(seq.Substring(i,seq.ending),motif_len);
                        i=seq.length; // to stop this loop
                }
        }
}
```