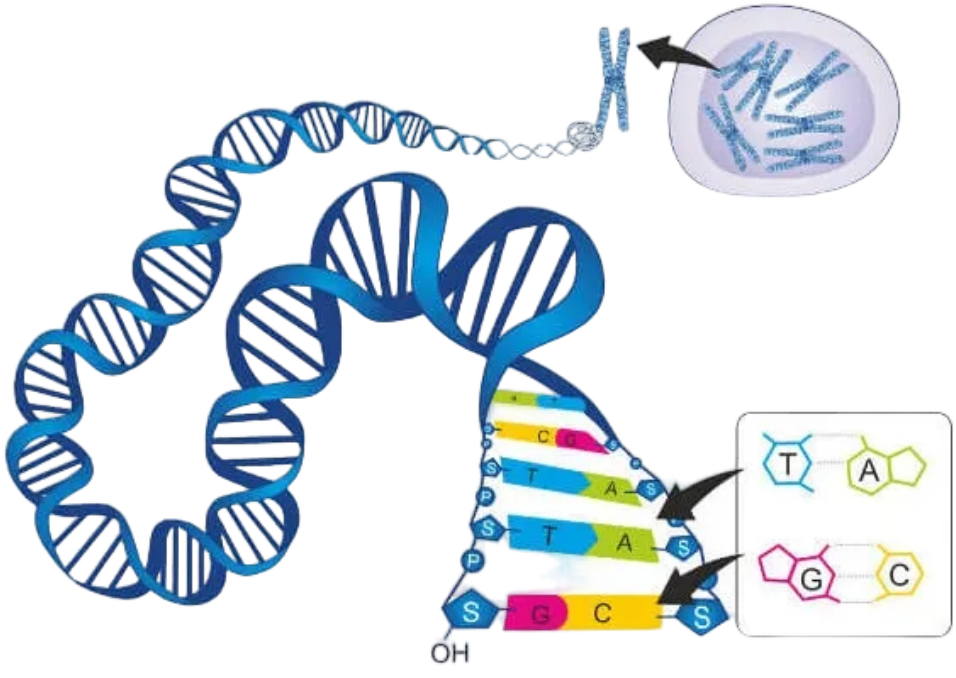
Software Engineering Department
Ort Braude College

Capstone Project Phase B-23-1-R-11

Cracking the DNA protein binding code

Supervisor: Dr. Zakharia Frenkel
Majd Msallam - Email: msalm46@gmail.com
Mohamed Abo Saleh - Email: abosalehmohamed800@gmail.com



Contents

Abstract:

This research delves into the mechanisms behind the process of DNA-binding proteins (DBPs) locating their target sites within the genome. Despite a significant amount of research in this area, there is still debate about how DBPs are able to quickly locate their target sites. This study proposes a new hypothesis that certain nucleotide sequences in DNA act as "pointers" that direct DBPs to the appropriate binding sites.

In our project we created a tool for finding the best candidates for such “pointers” by different approaches. We also tested our algorithm for this goal. This algorithm iterates through the DNA sequence using a sliding window of size k, moving the window one position at a time. saving all unique combinations obtained during this process, and analyzing them based on the properties of the pointers. The properties include different occurrences in downstream and upstream regions, in the main and complementary DNA strand, the asymmetry of the pointer pattern and other.

In the project implementation we met a difficulty with accessibility of the scientific data concerning the in vivo occupied protein binding sites. It caused a significant delay and only a part of the planned methods were applied. However, a clear signal in biological sequence was detected, significantly stronger than in corresponding shuffled data. The biological meaning of the discovered signal required additional investigation.

You can access our project on GitHub using this link:
https://github.com/majdmsallam/Cracking-the-DNA-pointers-code.git

1.Project Review

1.1 Introduction

There have been many studies on DNA-binding proteins, particularly those with high specificity, and there is still debate about how these proteins are able to quickly locate their target sites within the genome. This extended research offers a new hypothesis that may help to explain the speed at which these proteins locate their targets and could potentially have a significant impact on our understanding of genetics. There are several models for the mechanisms of the sequence search process, but none have yet provided an explanation for the speed at which DNA-binding proteins are able to find their target sites.

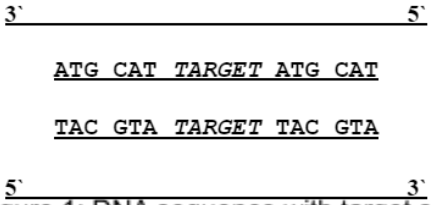
In our model it is postulated that certain nucleotide sequences in DNA act as "pointers" that direct proteins to the appropriate binding sites. These sequences are made up of base pairs, which are thought to serve as the pointers. The effectiveness of these pointers is believed to be determined by their frequency; if a particular sequence points to a specific location, it is likely to occur frequently. Further research into this hypothesis may provide useful insights into this particular issue.

1.2 Algorithms

1.2.1 The DNA pointers at a target site should have the following properties[1]:

- The occurrence of the pointer pattern downstream of the target site should be significantly higher than the occurrence upstream of the target site.
- The occurrence of the pointer pattern in the complementary DNA strand upstream of the target site in the primary strand should be significantly higher than the occurrence downstream of the target site.
- The pointer pattern should not be asymmetrical and can include wildcard elements.

As an example, if AAA and ATG are considered pointers, their occurrence will be high downstream and low upstream, while the occurrence of TTT and CAT will be low downstream and high upstream.



[Figure 1: DNA sequence with target site Source: Previous research]

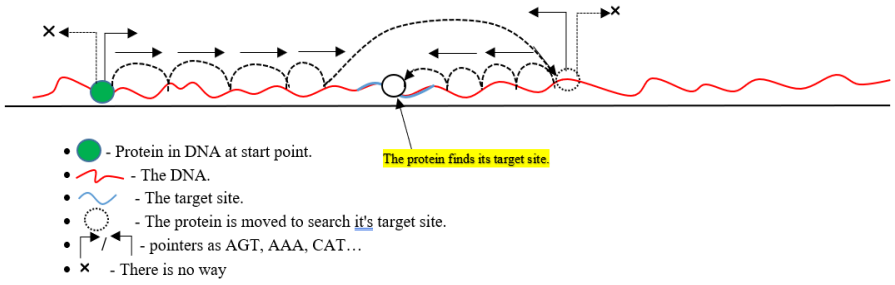
1.2.2 To detect the corresponding pointer patterns according to the postulated properties[1]:

- Calculate the content of all words in the downstream and upstream regions of the target site.
- Select the words with the greatest difference in occurrence between the word and its palindromic pair (such as ATG and CAT) in both regions.
- Select the pattern that best meets the mentioned above conditions.

1.2.3 The cases of interrupted and inexact patterns[1]:

There are four cases of patterns that a protein can bind to DNA (with the target site being AGCGGGCT):

- Exact pattern: the protein finds his target and is strongly caught (sequence of letters).
    - Example: **AGCGGGCT (Figure 7).**
- Inexact pattern: The protein finds his target site but is caught weakly because of substitutions.
    - Example: **AGCGGGCT** instead of G there is A, C, T OR instead of C there is A, G, T.
- Exact pattern but **interrupted**. It can be due to sometimes the protein interacting only with one side of the DNA duplex, as explained **(in Figure. 8)**. An expected distance between neighboring parts of the interrupted patterns corresponds to the period of the DNA helix (about 10.4 nucleotides). So, it can vary between 10 and 11, and even more.
    - Example: AGCG ????? GGCT
- Inexact pattern and **not sequentially** exactly what we explain above.
    - Example: AGCG ????? GGCT instead of G there is A, C, T OR instead of C there is A, G, T.



[Figure 2:(exact pattern) this fig shows the pathways for transferring a protein from one site to another along a long DNA molecule are 'sliding', 'hopping' and 'intersegmental transfer'. (Top) A protein might 'slide' along the double helix, transferring from one base pair position to the adjacent one without dissociating from the DNA.
Source: Previous research]

[Figure 3: Exact pattern but interrupted
Source: Previous research]

The four cases of patterns that a protein can bind to DNA - exact pattern, inexact pattern, exact pattern interrupted, and inexact pattern not sequentially exact - are the reality of how proteins interact with DNA. Our project will primarily concentrate on the precise pattern of DNA sequences. We will employ various algorithms and approaches to analyze the DNA sequence, such as utilizing PWM (Position Weight Matrix),

examining both ascending and descending sequences, determining the frequency of occurrences, and dividing the DNA sequence into multiple segments.

1.2.4 Algorithms Based on Mentioned Case:

We have proposed different algorithms to identify the anticipated DNA pointers, considering that the specific characteristics and structure of these pointers are uncertain. All of these algorithms utilize a DNA sequence and a word size (k) as the fundamental input, and additional parameters such as Position Frequency Matrix (PFM) may be included. They generate a table of potential patterns that could potentially function as pointers.
The algorithms discussed in this research have been examined and explored by previous students. Nevertheless, certain algorithms have not been effectively implemented or widely embraced thus far [1].

Algorithm 1 : Segmented DNA Analysis:

- Take a DNA sequence where the binding site is positioned in the middle of the sequence.
- Divide the DNA sequence into segments:
    - Divide the DNA sequence into segments before and after the binding site 1/2, 1/4, 1/8, and so on, up to 1/32.
- Calculate the occurrences of each word in each segment:
    - Iterate through each segment.
    - Count the occurrences of each word within the segment.
    - Store the occurrence counts for each word in each segment.
- Focus specifically on the segment before the binding site (1/32 segment):
    - Use a sliding window approach to iterate through the segment.
    - Move the window one position at a time.
- Consider a word as a potential pointer if:
    - It appears more frequently in the segments before the binding site.
    - The reverse complement of the word appears less frequently in the segments before the binding site.

Algorithm 2: Word Occurrences DNA Analysis:

- Take a DNA sequence where the binding site is positioned in the middle of the sequence.
- Extract all words from the downstream and upstream regions of the target site, along with their occurrences, using a sliding window size k and save the unique words
- Select the words that meet the following conditions:
    - The occurrence of the word (i.e. pointing in the forward direction) in the downstream region is more frequent than in the upstream region.
    - The occurrence of the corresponding palindromic word (i.e. pointing in the opposite direction) in the downstream region is less frequent than in the upstream region.
- Return all the words that satisfy the conditions.

Algorithm 3: Segmented Word Distribution Analysis:

- Take a DNA sequence where the binding site is positioned in the middle of the sequence.
- Divide the DNA sequence into a given number of segments.
- Iterate through the DNA sequence using a sliding window approach with window size k, moving the window one position at a time.
- For each new unique sequence encountered, add it to the set of unique sequences.
- Check if the number of occurrences in each segment forms an ascending order series before the binding site, and a descending order series after the binding site.
- Check if the complement word occurs in descending order before the binding site and ascending order after the binding site.
- If both conditions are satisfied, consider it a potential pointer.
- Return all potential pointers.

To access the PFM (Position Frequency Matrix) for this algorithm, you can download it from the Jaspar website [2].

Algorithm 4:Potential PWM Pointers :

- Given a DNA sequence with the binding site located in the middle:
    - Upload PFM for the DNA sequence.

    - Construct a Position Probability Matrix (PPM) from the PFM.
    - Build a Position Weight Matrix (PWM) from the PPM.
- Calculate the maximum score in the PWM.
- Iterate through the DNA sequence using a sliding window approach with a window size of "k", moving the window one position at a time.
- For each window:
    - Calculate the sum of the weights of the bases within the window.
    - If the score of the word is 0.9 times the maximum score and the number of occurrences is greater downstream (before the binding site) than upstream (after the binding site), consider the word as a potential pointer.
- Return the list of potential pointers identified in the DNA sequence.

1.3 Research process

In order to construct an optimal tool and achieve the desired outcomes, we undertook a thorough exploration of the field of biology, which was initially unfamiliar to us. This involved dedicating significant time and effort to gain a comprehensive understanding of the necessary biological concepts. This foundational knowledge acquisition formed the core focus of Phase A. Subsequently, we proceeded to search for a suitable database and evaluate different programming languages, including C++, Python, and Java, to identify the most suitable choice for our tool's development. After careful consideration, we determined that Java was the optimal language due to its superior performance with our sequences and its compatibility with JavaFX, allowing us to effectively build a graphical user interface (GUI).
In our project, we conducted basic tests to validate the efficiency of our tool in reading files, handling DNA sequences, and calculating the occurrences of words within those sequences.

1.4 Results

As mentioned earlier, our tool comprises four algorithms, each with its own unique approach. These algorithms aim to identify the most suitable subsequences that could potentially serve as pointers. We conducted experiments using different sizes of pointers and varied the number of segments on the same chromosome.
To differentiate between sequences associated with other binding sites, we examined a relative sequence of 3 million nucleic acid bases surrounding each binding site position.

After carefully examining the results, and in order to obtain more accurate findings, we focused on subsequences of size 6. Specifically, we selected the binding site (chr1 AGGGCCTCCGGCTC 114510535 114510548 +) from chromosome 1. To ensure the reliability of our results and to rule out the possibility of random discoveries, we employed the widely accepted shuffle method as a test tool in the field of DNA analysis. The following images display the obtained results:

1.4.1 Results of Segmented DNA Analysis algorithm

The output of this algorithm is a text file containing all potential pointers, along with the frequency of occurrences of the word at various regions (1/2, 1/4, 1/8, 1/16, 1/32) before and after the binding site. To provide a clearer understanding of the structure, we have also included an illustrative example:

[Figure 4: Example of segmentation.]

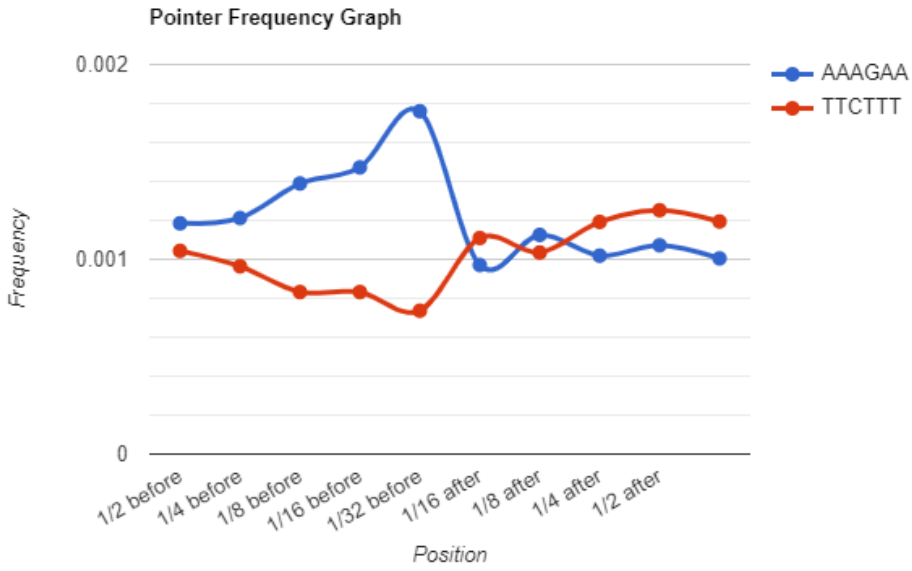
These are the obtained results:

```
AAGGAA:
Occurrences in first and second halves: [1095, 989].
Occurrences in second and 3rd quarters: [555, 478].
Occurrences in 3rd and fifth eighths: [299, 210].
Occurrences in 7/16 and 8/16 1/16: [141, 119].
Occurrences in 15/32 and 16/32 1/32: [73, 46].
TTCCTT:
Occurrences in first and second halves: [869, 1093].
Occurrences in second and 3rd quarters: [443, 556].
Occurrences in 3rd and fifth eighths: [201, 268].
Occurrences in 7/16 and 8/16 1/16: [103, 127].
Occurrences in 15/32 and 16/32 1/32: [35, 68].
```

[Figure 5: results of Segmented DNA Analysis algorithm]

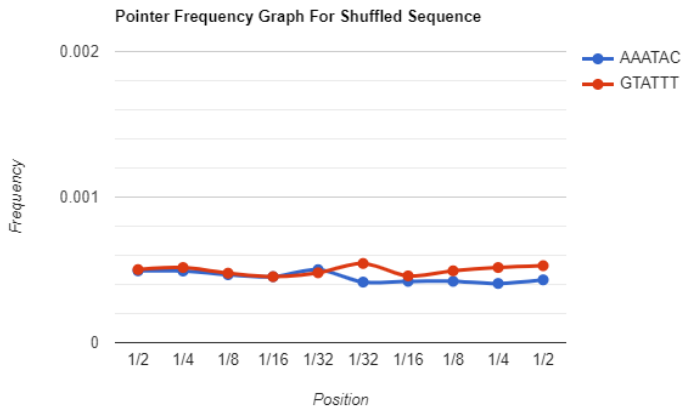
Based on the observed results, we can conclude that the frequency of the pointer "AAGGAA" increases as we approach the binding site, while it decreases as we move further away from the binding site. Conversely, the reverse complement word "TTCCTT"

exhibits the opposite trend, with its frequency decreasing as we approach the binding site and increasing as we move away from it. These findings support the hypotheses proposed in our research.



[Figure 6: Graphical visualization of the results.]

As observed in the frequency graph, the frequency of the word "AAAGAA" increases as we approach the binding site, while the reverse complement word "TTCTTT" exhibits the opposite pattern. This suggests that the proposed theory may hold true. Additionally, we note that the frequency of "AAAGAA" increases initially after the binding site, but then stabilizes, Similar observations can be made for the reverse complement sequence. We are proposing that this does not invalidate our claims. To verify this, we took the identical DNA sequence and rearranged it by segments, resulting in the displayed graph:



[Figure 7: Graphical visualization of the results.]

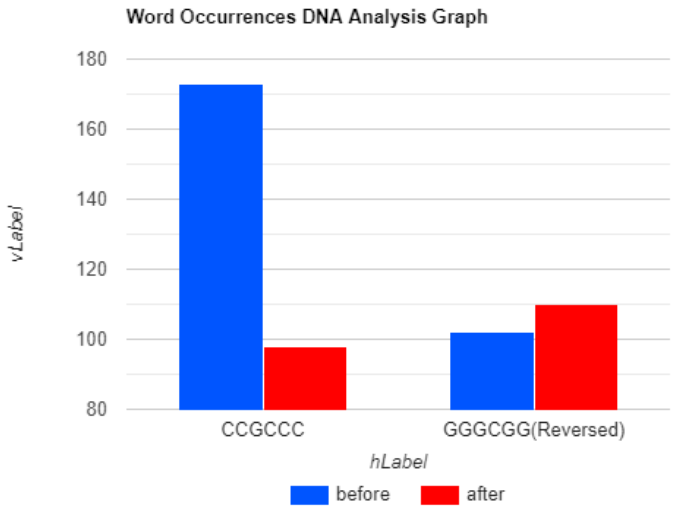
The in vivo signal is observed to be twice as strong as the in vitro signal (generated from a shuffled sequence). This indicates that these noises are not significantly meaningful.

1.4.2 Results of Word Occurrences DNA Analysis algorithm

```
CCCGTT [53, 35] AACGGG [37, 41]
CCGCCC [173, 102] GGGCGG [98, 110]
GCCGCC [111, 59] GGCGGC [77, 80]
CGGCCA [92, 61] TGGCCG [63, 68]
```

[Figure 8: results of Word Occurrences DNA Analysis algorithm]

This algorithm aims to validate our hypothesis that the frequency of a particular pointer in the downstream region is greater than its frequency in the upstream region, while the reverse complement of the pointer exhibits the opposite pattern, with higher occurrences in the upstream region compared to the downstream.
Based on the observed results, these pointers meet the required conditions, which allows us to consider them as potential pointers.



[Figure 9: Graphical visualization of the results.]

1.4.3 Results of Segmented Word Distribution Analysis algorithm

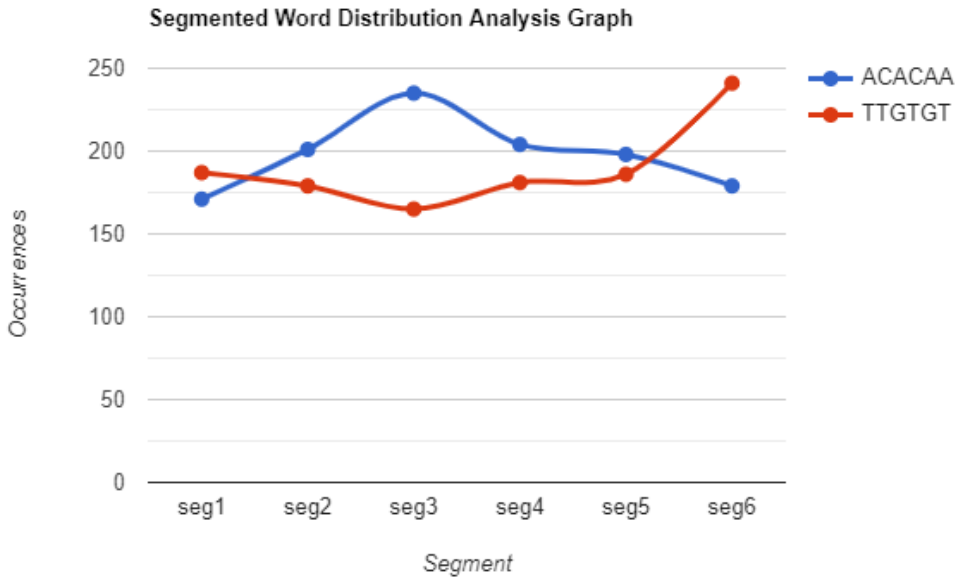
The algorithm yields a list of potential pointers along with their occurrences in the segments. The results are organized in ascending order before the binding site and

descending order after the binding site for the original word, while the reverse complement word follows the opposite pattern.
These are the obtained results:

```
Results for word size = 6, and number of segments = 6:

chr1 AGGGCCTCCGGCTC 114510535 114510548 +
ACACAA [171, 201, 235, 204, 198, 179] | TTGTGT [187, 179, 165, 181, 186, 241]
ACACGG [23, 26, 29, 26, 22, 20] | CCGTGT [26, 26, 20, 21, 22, 32]
ACCAAT [116, 121, 128, 107, 103, 85] | ATTGGT [120, 119, 101, 111, 113, 135]
CAATCA [121, 121, 167, 136, 125, 107] | TGATTG [124, 116, 115, 124, 127, 148]
```

[Figure 10: results of Segmented Word Distribution Analysis algorithm]



[Figure 11: Graphical visualization of the results.]

The graph above provides supporting evidence for our hypotheses. By dividing the DNA sequence into six segments, each with a size of 500,000, we can observe that the number of occurrences of the word increases as we move closer to the binding site, and conversely, decreases as we move away from it. This pattern holds true for the word itself, as well as its reverse complement (moving closer to the binding site the number of occurrences decreases, and moving after the binding site the number of occurrences increase) .

1.4.4 Results of Potential PWM Pointers algorithm

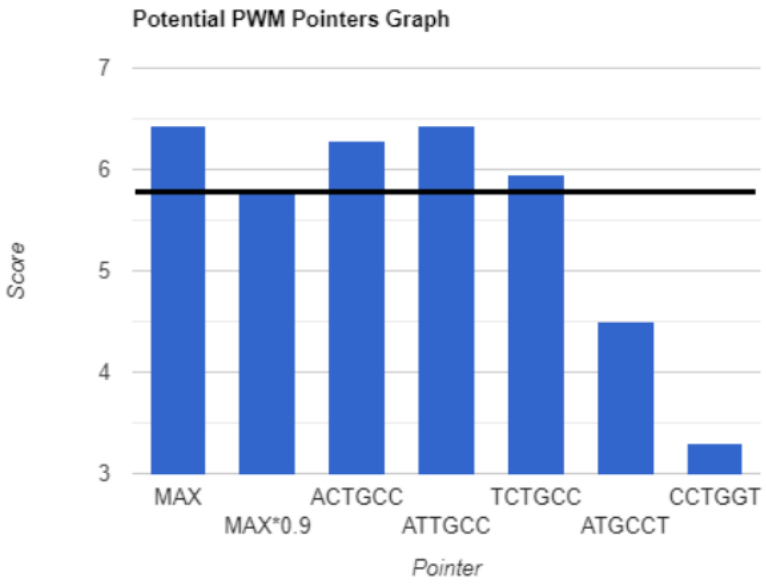
The results obtained from this algorithm consist of a list of potential pointers. These pointers are assigned a score equal to 0.9 times the maximum score. Additionally, the number of occurrences of these pointers before the binding site is higher than the number of occurrences after the binding site.
The scores in the PWM quantify the probability or affinity of a specific nucleotide or amino acid being found at a given position. Higher scores indicate a higher likelihood of that particular nucleotide or amino acid being preferred at that position, while lower scores indicate a lower likelihood.

These are the obtained results:

```
Max Score (word size = 6): 6.453288209048953

chr1 AGGGCCTCCGGCTC 114510535 114510548 +
ACTGCC=> Score: 6.279620805920825, downstream: 342, upstream: 322
ATTGCC=> Score: 6.453288209048953, downstream: 303, upstream: 296
TCTGCC=> Score: 5.958108653098009, downstream: 695, upstream: 650
GTTGCC=> Score: 6.3629937770154505, downstream: 378, upstream: 319
CCTGCC=> Score: 6.196394957556956, downstream: 719, upstream: 639
```

[Figure 12: Results of Potential PWM Pointers algorithm.]



[Figure 13: Graphical visualization of the results.]

As shown in the results, we specifically focused on words that exhibited a significantly high score and had a greater number of occurrences before the binding site compared to after it.

1.5 Conclusion

To summarize, our research aimed to identify potential pointers within DNA sequences that could point towards the presence of protein binding sites. However, the project faced challenges in accessing necessary scientific data on in vivo protein binding sites, resulting in delays and limitations in using planned methods. Despite this, we were able to detect a distinct and strong signal in the biological sequence, surpassing that of shuffled data. Further investigation is needed to determine the significance of this signal.
We applied various algorithms based on the hypothesis that subsequences occurring at higher frequencies before the binding site and decreasing in frequency further away may serve as potential pointers. Our analysis yielded multiple sequences that met these conditions, consistently exhibiting a pattern of higher occurrence near the binding site and lower occurrence as we moved away. These findings suggest that these identified subsequences could serve as informative pointers for locating binding sites.
In addition, we took the initiative to create a comprehensive and reliable database, extracting relevant data for future researchers. This database will save significant time and effort for subsequent work in this area. However, further investigation is still required to delve deeper into this problem.

2.User documentation

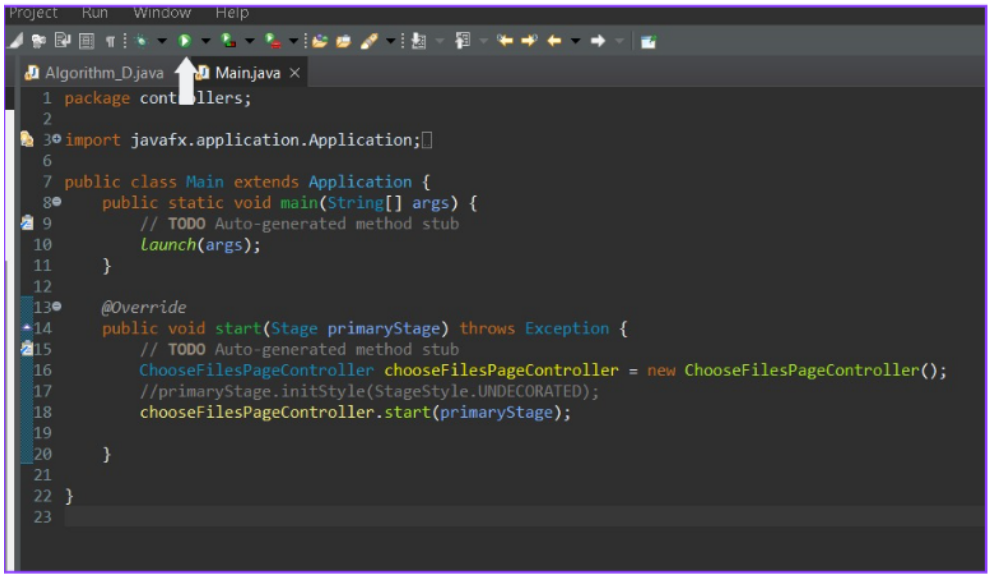
2.1 User Guide

The purpose of the "Cracking the DNA Protein Binding Code" tool is to identify potential pointers of a binding site within the DNA sequence.
The utilization of JAVA within the Eclipse IDE for enterprise java developers - specifically for the 2021-09 version and java version 1.8.0_341 - has been employed to implement all the algorithms. The graphical user interface (GUI) has been constructed using SceneBuilder and JAVAFX version (javafx-sdk-17.0.0.1). In our tool, we have solely relied on fundamental java collections and libraries, abstaining from incorporating any external libraries.

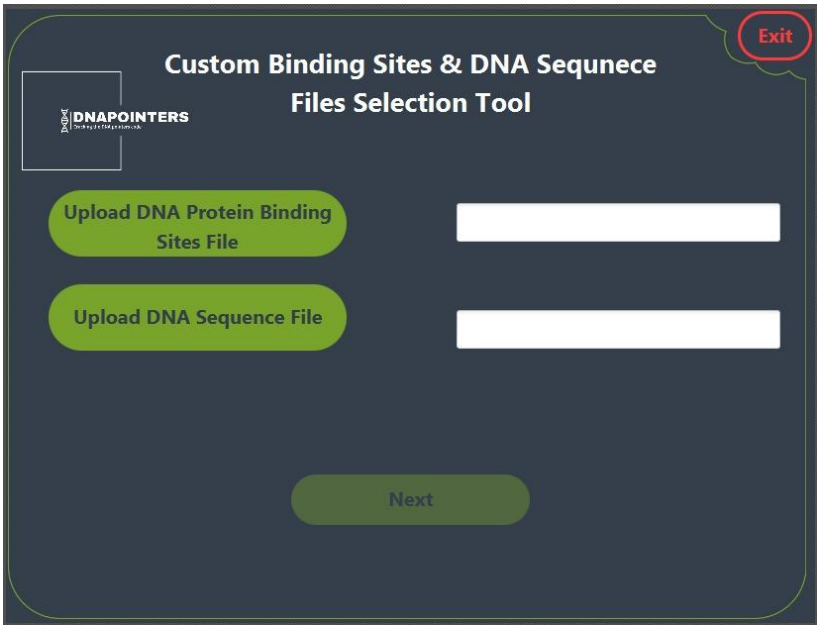
The tool's main objective is to identify and gather potential pointers of a binding site from a given DNA sequence using various methods. It then presents the collected findings to the user. Moreover, the tool offers the user the flexibility to select both the binding sites file and the DNA sequence file, allowing them to execute any desired algorithm on the chosen binding sites.

2.2 Operating Instructions

Running the tool is simple, all you need is an IDE to open the code with, and just press the “run” icon. See screenshot below:

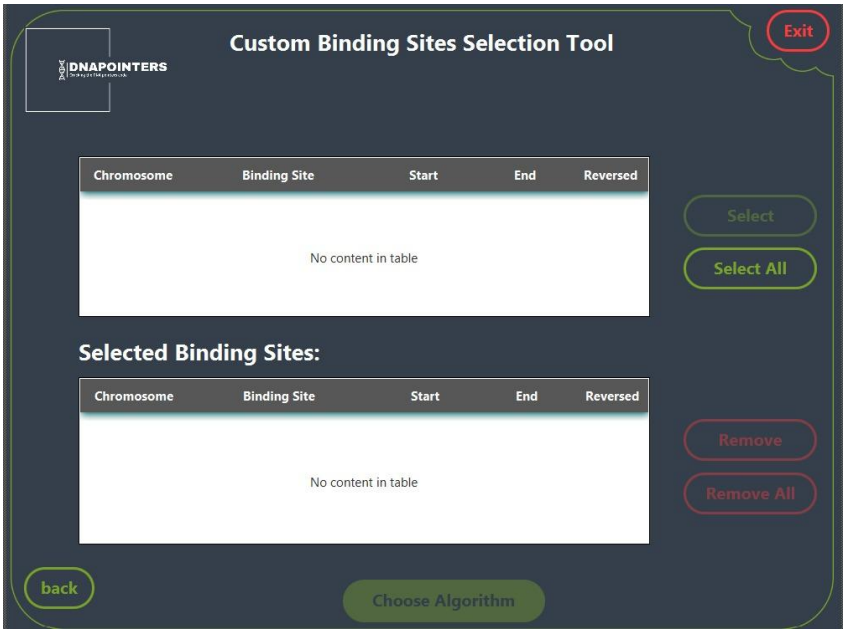


This will open the main page of the tool:

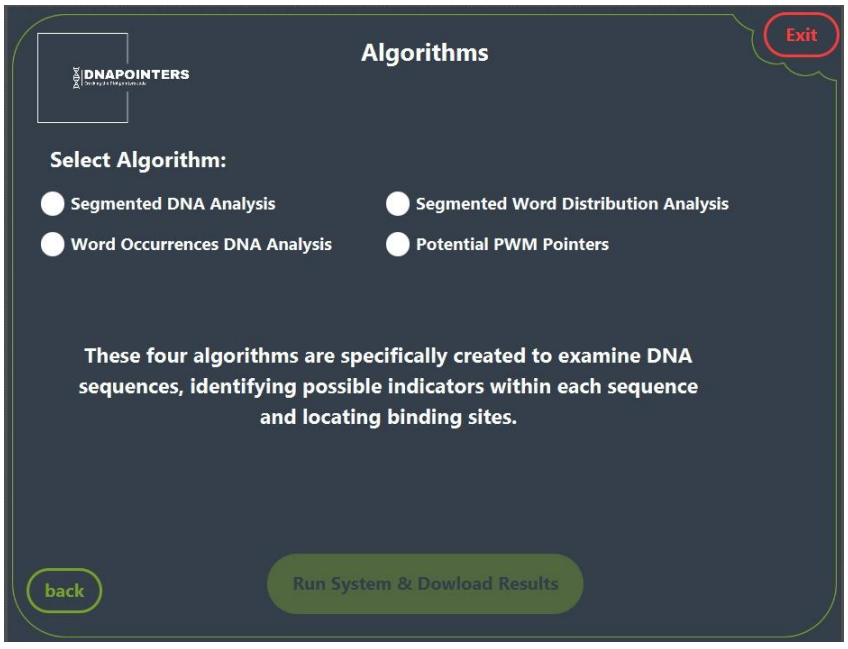


On this page the user will choose the DNA sequence and the binding sites, by clicking on one of these buttons a file chooser will open to select the files you want.

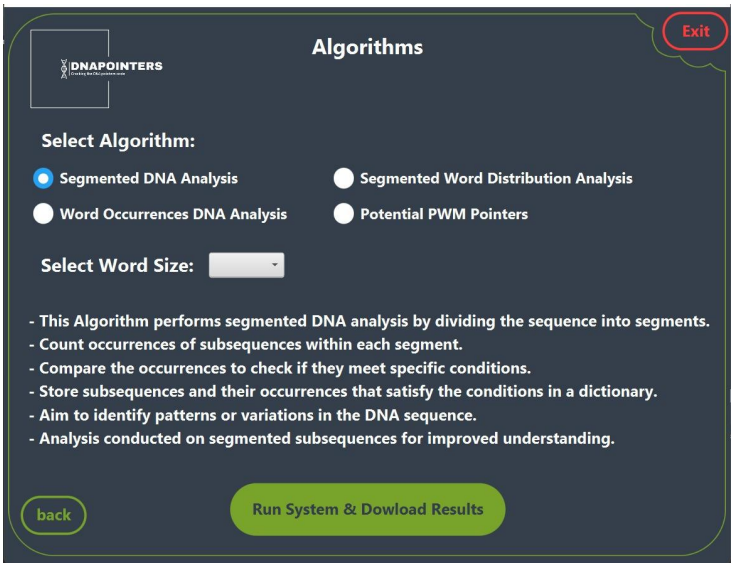
The user can select, select all or remove the binding sites he has selected through the page shown below:

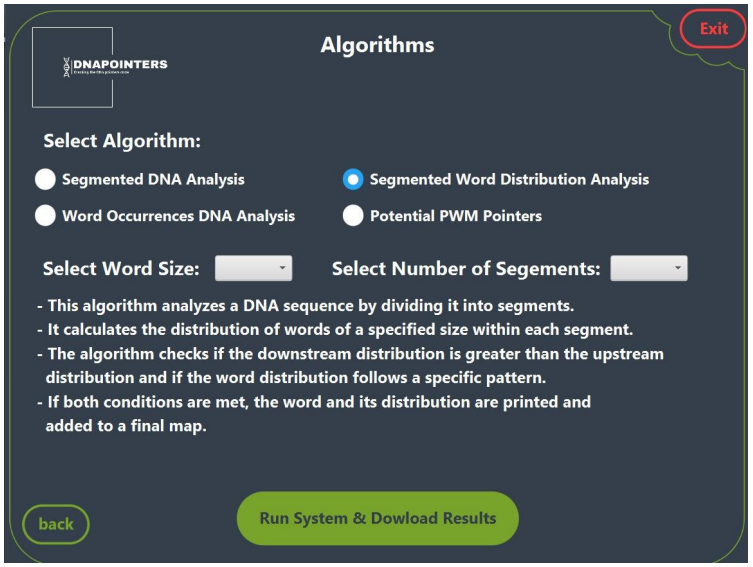


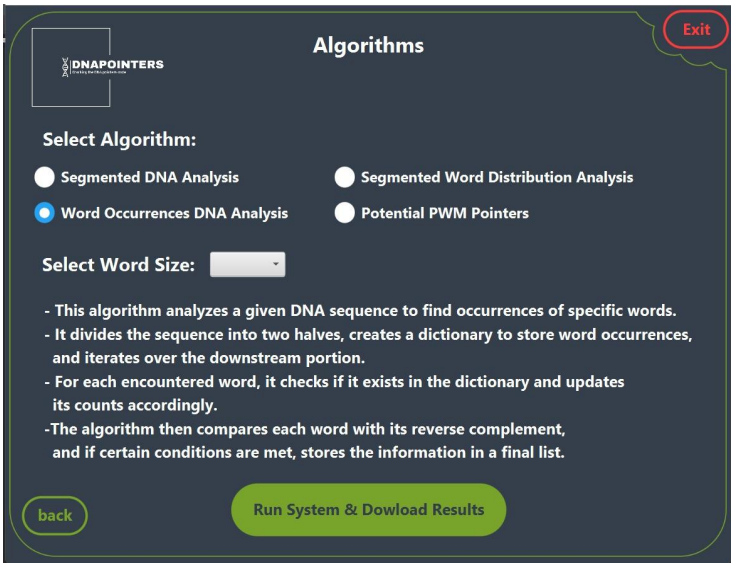
After selecting the binding sites of your choice the “choose Algorithm” button will be enabled and will take you to the page shown below

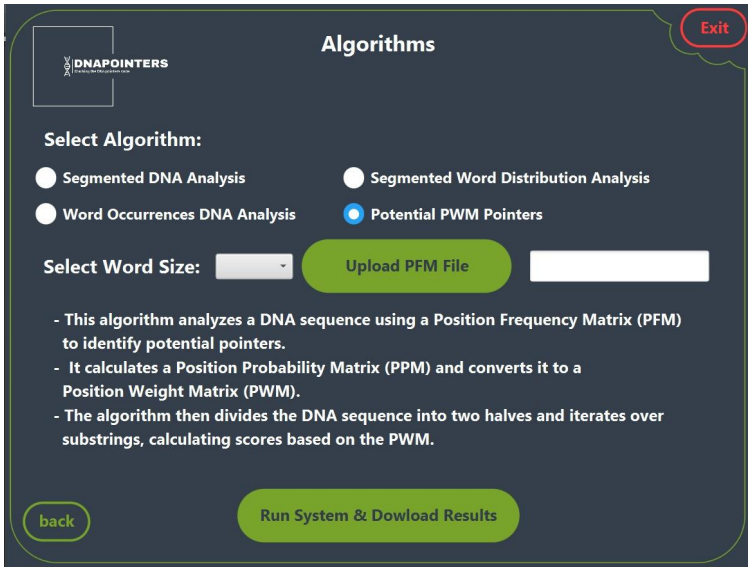


This page show the algorithms you can run on the data you selected on earlier pages, When choosing each algorithm a brief explanation about each algorithm is displayed as shown in the screenshots below:



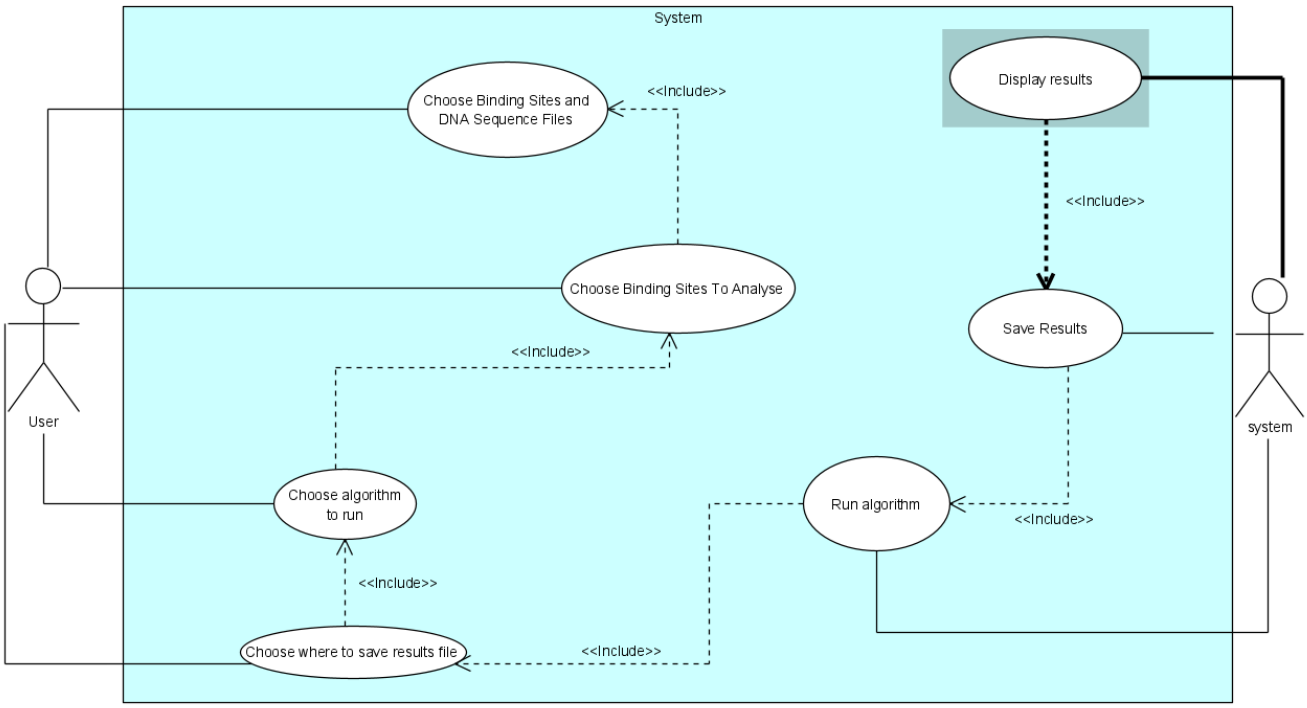




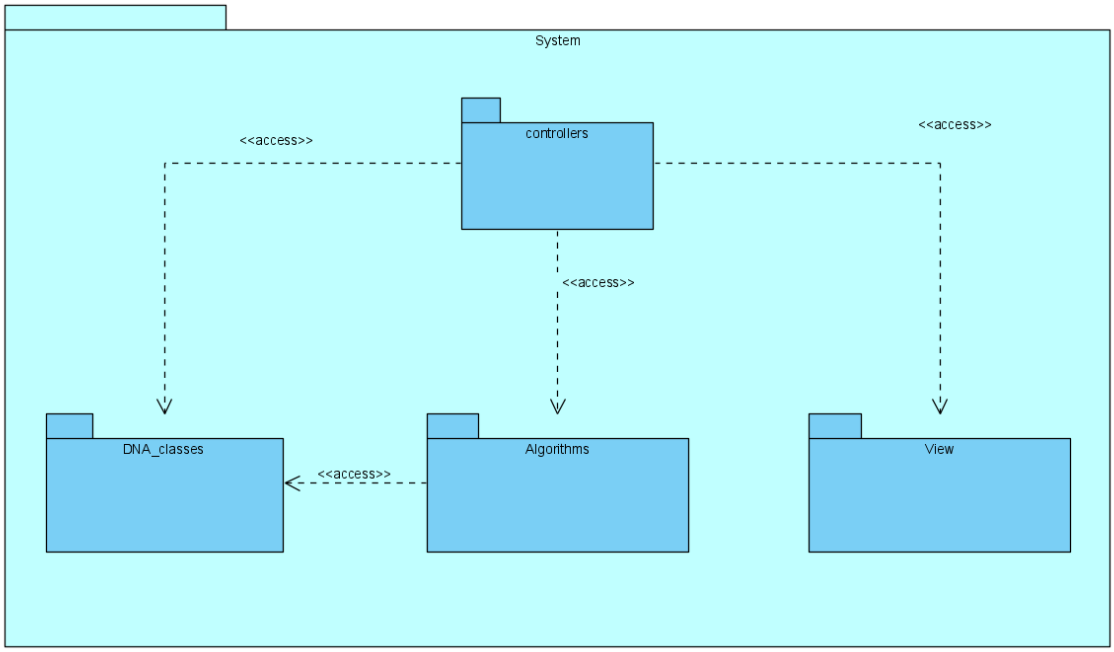


2.3 Maintenance Guide

UseCase Diagram:



Package Diagram:



2.4 DataBases construction

This research project involves the use of algorithms to analyze data. In order to effectively execute these algorithms, it is necessary to have access to a high-quality database containing relevant data.
Creating and compiling the database was a challenging task that consumed a significant portion of our project's time. We faced difficulties during the process, primarily due to the extensive search and selection required. We considered multiple databases, including paid options like TRANSFAC [3], but encountered obstacles such as inaccurate and irrelevant data in some of them. These issues necessitated careful consideration and thorough evaluation to ensure the reliability and relevance of the information included in our database.

After careful and thoughtful consideration We utilized two databases for our study, namely the JASPAR CORE database [2] and the NCBI [4] (National Center for Biotechnology Information). The JASPAR CORE database is a meticulously curated collection of profiles that accurately depict experimentally determined transcription factor binding sites for eukaryotes. It distinguishes itself from other comparable resources, such as TRANSFAC, by offering open data access, non-redundancy, and a high level of quality.
In addition, the NCBI (National Center for Biotechnology Information) is a renowned resource that provides access to various biological databases and tools. It offers a wide range of genomic, genetic, and protein-related information and serves as a valuable platform for researchers in the field.

The construction of our database involved these steps:

Initially, we obtained the file containing binding sites for a specific protein from the Jaspar website [5]. The downloaded file encompassed binding sites for all chromosomes present in the human genome version 38. After that, we obtained the GRCh38.p14 Primary Assembly dataset [6] from NCBI, which is a specific version of the human reference genome, which serves as a standardized representation of the DNA sequence and gene locations in the human genome. The dataset contained 23 files representing individual chromosomes, including the X chromosome.

[Figure 14: DNA chromosomes files]

```
CGCGCCGGCGCAGGCGCAGAGAGGCGCGCCGCGCCGGCGCAGGCGCAGAGAGGCGCGCCGCGCCGGCGCAGGCGCAGAGAGGC
CGCAGGCGCAGACACATGCTAGCGCGTCGGGGTGGAGGCGTGGCGCAGGCGCAGAGAGGCGCGCCGCGCCGGCGCAGGCGCAC
GCGCAGAGAGGCGCACCGCGCCGGCGCAGGCGCAGAGACACATGCTAGCGCGTCCAGGGGTGGAGGCGTGGCGCAGGCGCAGA
AGGAGCAAAGTCGCACGGCGCCGGGCTGGGGCGGGGGGAGGGTGGCGCCGTGCACGCGCAGAAACTCACGTCACGGTGGCGCC
ATCGACCGCCCCTTGCTTGCAGCCGGGCACTACAGGACCCGCTTGCTCACGGTGCTGTGCCAGGGCGCCCCCTGCTGGCGACT
GCCCCCTGCTGGCGCCGGGGCACTGCAGGGCCCTCTTGCTTACTGTATAGTGGTGGCACGCCGCCTGCTGGCAGCTAGGGACA
ACCTGCTGGCAGCTGGGGACACTGCCGGGCCCTCTTGCTCCAACAGTACTGGCGGATTATAGGGAAACACCCGGAGCATATGC
TTAAAAGTAAAAAATAAATATGTTTAATTTGTGAACTGATTACCATCAGAATTGTACTGTTCTGTATCCCACCAGCAATGTCT
TTGCCAGTCTAACAGGTGAAGCCCTGGAGATTCTTATTAGTGATTTGGGCTGGGGCCTGGCCATGTGTATTTTTTTAAATTTC
```

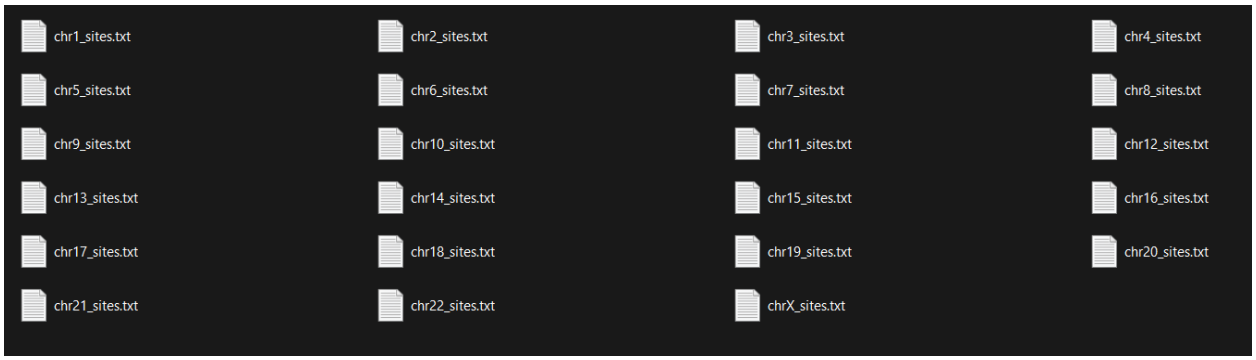
[Figure 15: DNA chromosome file structure.]

The structure of the binding sites file was as follows:

```
>hg38_chr1:246507317-246507330(-)
CGCCCCTCGGGCGT
>hg38_chr4:246583526-246583539(+)
TAGTCCTCAGGGCA
>hg38_chr4:246773587-246773600(+)
tctgcctgaggcat
>hg38_chr3:246773588-246773601(-)
aatgcctcaggcag
>hg38_chr3:246789291-246789304(-)
GTGGCCTCCGGCCG
>hg38_chr2:247291979-247291992(-)
TGAGCCTCAGGCCT
>hg38_chr2:248826675-248826688(-)
GCTCGCTCAGGGAC
```

[Figure 16: The binding sites file encompassed binding sites for all chromosomes.]

We partitioned the binding sites file into 23 distinct files, with each file containing the binding sites specific to a particular chromosome.



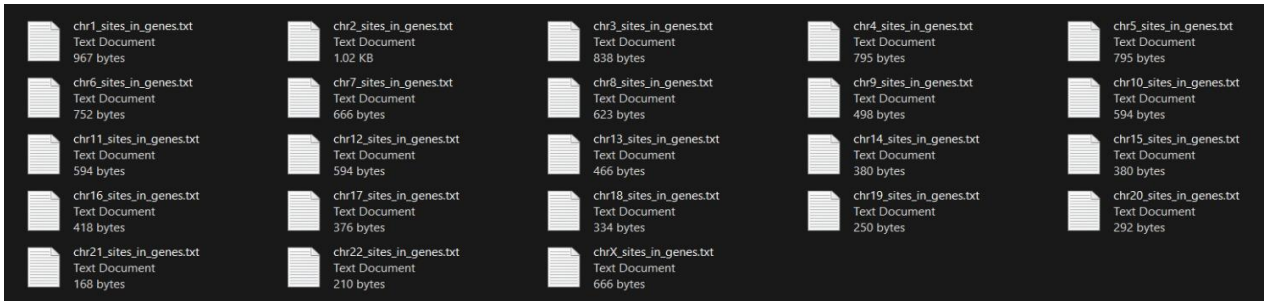
[Figure 17: There are 23 distinct binding sites files, with each one specifically assigned to a corresponding chromosome.]

The subsequent step involved obtaining the annotation file for the GRCh38.p14 Primary Assembly from NCBI. This file provides detailed information about each region within every chromosome. Our focus was on extracting the binding sites located within gene regions since these regions are recognized for their activity.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	# feature	class	assembly	assembly_	seq_type	chromoso	genomic_a	start	end	strand	product_a	non-redun	related_ac	name	symbol	GeneID	locus_tag	feature_in	product_le	attributes
2	misc_RNA		GCF_0000	Primary As	chromoso	1	NC_00000	11874	14409	+	NR_046018.2			DEAD/H-b	DDX11L1	1E+08		1652	1652	
3	gene	transcribe	GCF_0000	Primary As	chromoso	1	NC_00000	14362	29370	-				WASP fam	WASH7P	653635		15009		
4	misc_RNA		GCF_0000	Primary As	chromoso	1	NC_00000	14362	29370	-	NR_024540.1			WASP fam	WASH7P	653635		1769	1786	
5	gene	miRNA	GCF_0000	Primary As	chromoso	1	NC_00000	17369	17436	-				microRNA	MIR6859-1	1.02E+08		68		
6	precursor_	RNA	GCF_0000	Primary As	chromoso	1	NC_00000	17369	17436	-	NR_106918.1			microRNA	MIR6859-1	1.02E+08		68	68	
7	ncRNA	miRNA	GCF_0000	Primary As	chromoso	1	NC_00000	17369	17391	-				hsa-miR-6	MIR6859-1	1.02E+08		23		
8	ncRNA	miRNA	GCF_0000	Primary As	chromoso	1	NC_00000	17409	17431	-				hsa-miR-6	MIR6859-1	1.02E+08		23		
9	gene	lncRNA	GCF_0000	Primary As	chromoso	1	NC_00000	29774	35418	+				MIR1302-2	MIR1302-2	1.08E+08		5645		
10	ncRNA	lncRNA	GCF_0000	Primary As	chromoso	1	NC_00000	29774	35418	+	XR_007065314.1			MIR1302-2	MIR1302-2	1.08E+08		2263	2263	
11	gene	miRNA	GCF_0000	Primary As	chromoso	1	NC_00000	30366	30503	+				microRNA	MIR1302-2	1E+08		138		

[Figure 18: The annotation file.]

Following the filtration process, we obtained the resulting files:



[Figure 19: Filtered binding sites file in genes regions.]

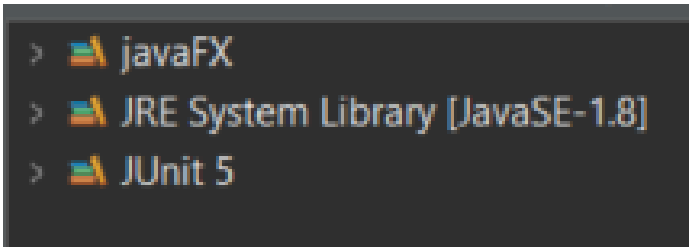
```
chr1 GCTGCCTGAGGCAG 3471641 3471654 -
chr1 CGCGCGTCACGCCG 15526971 15526984 -
chr1 TCTGCCTTAGGCAA 24600696 24600709 -
chr1 TTGCCCTCAGGCAT 34656961 34656974 -
chr1 ACTGCCTCTGGTCC 44329214 44329227 +
```

[Figure 20: example for binding site format (chromosome #, binding site sequence, start index, end index , reversed or not)]

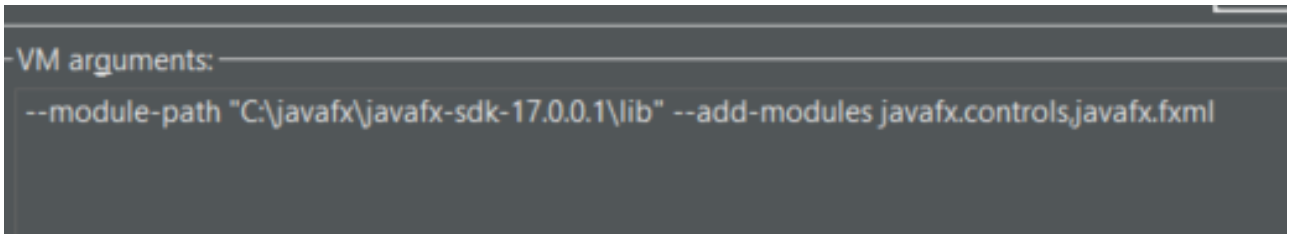
The final step entailed considering binding sites that were distantly located from one another.

2.5 System Requirements for Running Our Software

Ensure that you have files with a similar structure as previously demonstrated.
Initially, it is crucial to have an IDE that supports Java, such as Eclipse. Prior to installation, make sure to download and install the JDK from [7].
Furthermore, it is essential to install the JavaFX package, which can be acquired from [8].
Subsequently, import our project into the IDE and include the JavaFX library in the project libraries.



Include the JavaFX arguments in the run configurations, taking into account the location where you have stored the JavaFX files.



And you are ready to proceed.

3.References

[1] Adham Shahwan,Ammar Khutba, “Detection and characterization of the DNA-pointers”, January 2020.

[2] An open-access database of transcription factor binding profiles. JASPAR. (n.d.). https://jaspar.genereg.net/

[3] *TRANSFAC*. geneXplain. (2022, July 5). https://genexplain.com/transfac/

[4] National Center for Biotechnology Information, U.S. National Library of Medicine, https://www.ncbi.nlm.nih.gov/.

[5] *Matrix profile: TFAP2A - MA0003.4*. JASPAR. (n.d.-b). https://jaspar.genereg.net/matrix/MA0003.4/

[6] U.S. National Library of Medicine. (n.d.). *GRCH38.P14 - hg38 - genome - assembly - NCBI*. National Center for Biotechnology Information. https://www.ncbi.nlm.nih.gov/assembly/GCF_000001405.40#/def

[7] *Download the latest Java Lts Free*. Oracle Israel. (n.d.). https://www.oracle.com/il-en/java/technologies/downloads/

[8] *JavaFX*. Gluon. (2023, April 19). https://gluonhq.com/products/javafx/