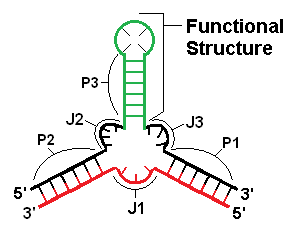
SxRNA\_Producer

A Java Based Genetic Algorithm Application

**A Summary User Manual**

****

**Introduction**

The SxRNA\_Producer application allows the generation of candidate sxRNA switch sequences for a specified trigger miRNA while also optionally avoiding or encouraging suppressive or reinforcing interactions with other miRNA based on input parameters. The program has been implemented in Java and requires a suitable JRE be installed on the system it is to be executed on. Additionally, the program requires that the RNAfold and RNAcofold programs (Vienna package for RNA) be installed on the system and that the command line for invoking these be specified in a properties file if different from the defaults ("/usr/local/bin/RNAfold -p -d2 --noLP --noPS --noDP" & "/usr/local/bin/RNAcofold -d2 --noLP --noPS"). The application has been written and tested using versions 2.4.14 of these programs. If a different version is installed, it may require code changes to SxRNA\_Producer if the input or output format has been changed. Though written in Java, it is assumed the code will be executed on Linux.

The source code for SxRNA\_Producer and its underlying libraries is on Github and can be found at <https://github.com/fjdoyle2002/genetic_algorithm>

**Background**

Genetic algorithms employ rules derived from natural selection to evolve data. Descriptions of their implementation therefore borrow terms from biology but may use these terms in ways that might seem odd to biologists. Individuals in the population undergoing evolution are represented by an array of “genes” that is termed a “chromosome”. Possible values for each of the genes are termed “alleles”. In this particular implementation, each gene actually represents a single RNA nucleotide in a sequence (represented by the Java RibonucleotideGene class), which may be particularly confusing to some biologists without clarification.

For background on sxRNA concepts, please see the Frontiers in Genetics and Scientific Reports articles from this lab.

**Running the Program**

The program is invoked as follows(note, there should be no line breaks in actual invocation, also… file paths are examples and should not be assumed to be valid as shown):

java -Xmx15360m -cp /opt/sxrna/bin

io.github.fjdoyle2002.ga.sxrna.SxRNA\_Producer

nc=NNNNNNNNNNNNNNNNNNNNAAMGGYYYUUUUHARRRCYMMNNNNNNNNNNNNNNNNNNNNN

mc=yyyyyyyyyyyyyyyyyyyynnnnnnnnnnnnnnnnnnnnnyyyyyyyyyyyyyyyyyyyyy

"ms=.......................((((((....))))))......................."

tt=UGGAGUGUGACAAUGGUGUUUG ncrf=/opt/sequence/hsa.fa

ntf=/home/*someuser*/non\_targets.txt

sptf=/home/*someuser*/secondary\_positive\_targets.txt

netf=/home/*someuser*/negative\_targets.txt

pf=/home/*someuser*/enviro.properties

**NOTE:** If running from a terminal, it is advised to use “nohup” to avoid process termination due to disconnect upon screensaver, etc…

**Argument Details**

**nc –** nucleotide constraints. The value of this parameter determines the number of “genes” in the genome of each individual that will be produced by the program. Additionally, it specifies a portion of the available alleles for each of these genes with regard to the group of specific RNA bases that can satisfy the FASTA code at that position. The other impact on available alleles comes from the **mc** argument and will be discussed in its section. Some internal portion of the specified nucleotide constraint argument must include the consensus sequence for the functional structure to be switched. In this example, it is a version of the Histone Stem Loop consensus (**AAMGGYYYUUUUHARRRCYMM**). The ‘N’ characters flanking this consensus represent any possible RNA base (i.e., A,C,G, or U) and will constitute the region for hybridizing to the target RNA to form the 3WJ as well as potential “destabilizing sequence” to promote alternative structural conformation without the trigger. Centering the motif portion within the input constraint sequence tends to result in candidates that have the structural motif centered with portions of both flanks acting as destabilizers without the trigger. Given the ability for blank genes, it is not impossible that a candidate could have the structure biased to one side. If a user wished to force a bias (i.e., toward the 3’ end, as the manual designs had done) this could be accomplished via the **nc**, **mc** and **ms** arguments (see figure 1 at end of doc).

**mc** – meta constraints. Originally envisioned as an extensible code, it currently only represents “yes” (as ‘y’) or “no” (as ‘n’) to denote whether a corresponding gene (the **nc**, **mc** and **ms** arguments must all be the same length and characters of each **relate to each other by their position**) may be blank. This in addition to the values specified by the **nc** determines the full set of available allelesin each chromosome position.

**ms** – motif structure. This is a string that uses ‘dot bracket’ notation to denote the position and structure of the functional RNA motif that is to be switched. It must correspond positionally, and as appropriate, to the consensus sequence of the functional motif as positioned within the **nc** argument.This argument must be enclosed in quotation marks due to special character use.

**tt** – targeted trigger. This is the RNA sequence of the miRNA that we want to “turn on” the sxRNA via formation of a *trans* three-way junction.

**pf** – properties file. A list of the properties that can be set within the file specified by this argument are listed at the end of this document. The file is optional and there are default values for all properties. It is only needed if you wish to override any of these defaults.

**ncrf** – non-coding RNA file. This file should include all miRNA that may be referred to by name in the off-target files. The format of this file should be two-line FASTA entries consisting of first line header and second line RNA sequence. Example entries are presented at the end of this document. This can be retrieved from miRBase as an “all entries” file and subdivided via grep (Linux command line) for organism of interest.

**Off target files:** the next three arguments are used to determine fitness related to off target interactions. “non-targets” should be RNAs whose levels do not change markedly between contexts where the switch is expected to be off or on (i.e., RNAs with which any interaction, positive or negative, should generally be avoided). These contexts could be in different cell types or in the same cell type with some treatment, etc. The values might be derived from sequencing or may be mock data specified to drive the evolution of a sequence with specific intent, as the fitness score assigned for avoidance or engagement of interaction with these sequences is weighted (slightly differently based on type[non-target vs secondary positive vs negative]) via these values. Secondary positive targets are RNA that show a marked increase in the (i.e., fold enrichment) “on state context”. These are RNAs that are desirable to help promote the functional motif formation and which we do not want to induce cleavage of the sxRNA. Negative targets are RNAs that show a marked decrease from off state to on state. These are RNAs that are desirable to induce cleavage or disrupt the motif structure’s formation.

**ntf -** non-target file. The file specified by this argument should include a list of ncRNA names (that correspond with entries in the file specified by the **ncrf** argument) in the first column and numeric values in columns two and three that represent their presence in the off and on state contexts, respectively (for a single cell type undergoing treatment, these values may be identical for a given miRNA).

**sptf** – secondary positive target file. Same format as **ntf** file. Should contain RNAs that are increased fold in “on state” vs “off state”.

**netf** – negative target file. Same format as **ntf** file. Should contain RNAs that are reduced fold in “on state” context vs “off state”.

**Output:**

There are two output files produced by the program. Both include a timestamp as part of the name, so multiple runs can be executed in the same directory without overwriting previous results. These are large text files and should be viewed in a plain text viewer that is capable of having “word-wrap” disabled (e.g., TextPad on Windows, etc.). The two files are:

sxRNAProducer\_*timestamp*.log (e.g., “sxRNAProducer\_1647972122037.log”)

This file shows the top ten population members at each generation, along with their solo-folds and trigger co-folds. If program completes, the bottom of the file will have some descriptive stats for fitness across generations. Prior to those stats will be the top ten individuals at the final generation. You may choose amongst these for test candidates or run the program again(with or without altering input) to find more options. This is the general log file and also includes some amount of information that is not of high relevance for general use.

sxRNACensus\_*timestamp*.log (e.g., “sxRNACensus\_1647972122037.log”)

The census file produces a single line of output for every individual created in the run regardless of how fit it is deemed. Output is pipe delimited and shows:

**ID** – the unique ID (within this run) of the individual

**Fitness Score** – the fitness score calculated in this run for this individual

**Generation** – generation number this individual was produced

**Oldest Ancestor** – How far back in generations does this individual trace its lineage

**Youngest Ancestor** – How recent a generation was this individual’s youngest ancestor

**Ancestral Operations** – How many genetic operations contributed to the creation of this individual

**Genes** – the actual gene values (specific alleles) of this individual, including blanks. Remove blanks to produce phenotype

**nonTargetPosScore** – score for avoiding motif promoting interactions with non-targets

**nonTargetSuppressScore** – score for avoiding cleavage/motif suppressing interactions with non-targets

**negTargetPosScore** - score for avoiding motif promoting interactions with negative targets

**negTargetSuppressScore –** score for achieving cleavage/motif suppressing interactions with negative targets

**posTargetPosScore -** score for achieving motif promoting interactions with secondary positive targets

**posTargetSuppressScore -** score for avoiding cleavage/motif suppressing interactions with secondary positive targets

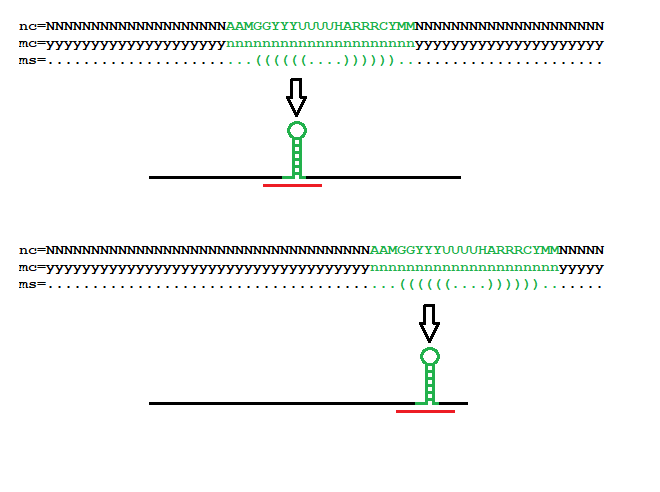


Figure Forcing a bias of motif position toward the 3’ end of the resulting candidate sxRNA sequences. Green represents the functional motif and red represents the trigger RNA that the sxRNAs are designed against. **NOTE:** something like the bottom candidate could be produced by the top arguments, given the ability for positions to be blank, but it is less likely.

Example **ncrf** file excerpt (first 10 lines, actual file has over 5,000):

>hsa-let-7a-5p MIMAT0000062 Homo sapiens let-7a-5p

UGAGGUAGUAGGUUGUAUAGUU

>hsa-let-7a-3p MIMAT0004481 Homo sapiens let-7a-3p

CUAUACAAUCUACUGUCUUUC

>hsa-let-7a-2-3p MIMAT0010195 Homo sapiens let-7a-2-3p

CUGUACAGCCUCCUAGCUUUCC

>hsa-let-7b-5p MIMAT0000063 Homo sapiens let-7b-5p

UGAGGUAGUAGGUUGUGUGGUU

>hsa-let-7b-3p MIMAT0004482 Homo sapiens let-7b-3p

CUAUACAACCUACUGCCUUCCC

Example **ntf** file excerpt (6 sample lines, actual size will vary):

hsa-miR-21-5p 7110.91 7110.91

hsa-let-7a-5p 6016.55 6016.55

hsa-miR-30a-5p 4797.86 4797.86

hsa-let-7i-5p 4019.6 4019.6

hsa-let-7f-5p 2195.62 2195.62

hsa-miR-181a-5p 1717.18 1717.18

The previous values were based on a single sequencing run on HeLa cells that were then to be used with either a synthetic analog trigger miRNA or transfected with a trigger miRNA producing plasmid. Only one sequencing run was performed and the values for miRNA present were used in the off and on context columns, as it was assumed they would not differ substantially based on the addition of the trigger. If multiple sequencing runs had been performed(on different batches of untreated cells) the average value from those runs would be used instead, but both columns would still hold identical numbers. The only reason this would have differing values would be if sequencing was done with and without the trigger analog/plasmid. If such runs had been performed and any of these changed by a fold increase/decrease, those RNA and their values should be put in the **sptf** or **netf** file as appropriate.

**sptf** and **netf** files will be same format but should contain fold increased values in the second column for **sptf** and fold decreased in the second column for **netf**. The actual values are normalized relative to each other within a file therefore units and range of values are not an issue as long as they represent a relative presence to each other in a single file. This is why mock values may be used to direct the evolution of a switch to avoid or engage certain RNAs. RNAs with larger average values in the **ntf** file will be weighted heavier in contribution to the fitness score.

Properties that may be set by **pf** file. Default values are shown in format expected in file.

#general GA environment variables

maximum\_mutation\_severity=.5

maximum\_generational\_fecundity=.5

maximum\_population\_size=1000

immigration\_rate=.01

maximum\_number\_of\_generations=100

allowMultiThreading=true

maximum\_time\_to\_wait\_for\_thread=25000

maximum\_threads=32

use\_fixed\_crossover=false

#sxRNA specific

cofoldCommand="/usr/local/bin/RNAcofold -d2 --noLP --noPS"

foldCommand="/usr/local/bin/RNAfold -p -d2 --noLP --noPS --noDP"

isTranslationalSwitch=true

**Property descriptions**

**maximum\_mutation\_severity –** (double) determines how many mutants will be produced and the severity of mutations (indirectly as number of genes mutated is based on random probability with a cutoff related to this number) they will experience. Number is chosen randomly per generation between zero and this value.

**maximum\_generational\_fecundity –** (double) determines how many offspring will be produced in a generation. Represents a fraction of current population size. Number is chosen randomly per generation between zero and this value.

**maximum\_population\_size –** (integer)How large of a population we want to maintain. This will affect how long it takes to process a generation, as the number of new offspring, mutants and new random members is based on the population size.

**immigration\_rate –** (double)determines how many completely random new population members will be introduced per generation. This is calculated as a fraction of the current population size.

**maximum\_number\_of\_generations** – (integer)currently, this is the only termination criteria. If you wish to run longer, you can set this to as large a size as desired, watch output file and terminate manually.

**allowMultiThreading –** (boolean)determines whether the system will use a single thread of execution to score each new population member iteratively or if multiple threads should be spawned to score in parallel. For sxRNA\_Producer it is suggested to generally use multithreading

**maximum\_time\_to\_wait\_for\_thread –** (long)number of milliseconds for thread pool to wait for a scoring thread to finish before assuming frozen and killing that thread.

**maximum\_threads** – (integer) determines how many threads the thread pool will maintain when multithreading option is enabled. It is suggested this be set at or below the maximum number of processing cores available.

**use\_fixed\_crossover –** (boolean)when true, this will create default fixed position crossover points based on motif structure position. It is preferable to use the false setting and allow for random position crossovers to occur.

**cofoldCommand, foldCommand -** These are the invocation commands used by the fitness algorithm to run secondary structure prediction on the candidate sxRNAs

**isTranslationalSwitch** – Boolean value. Set to false for a switch only expected to be used for in-vitro analysis with pure oligo trigger (e.g., for a diagnostic where an RNA purification step has occurred). Logic for translational switches takes rules pertaining to RISC cleavage into account that would not be applicable to a non-RISC incorporated trigger.