Co-evolution of RNA Virus and Vaccine

LAVI IONAS, Queen's University, Canada
PRIYANK THAKKAR, Queen's University, Canada
CYRIL ZHANG, Queen's University, Canada

This model aimed to take the biological structure of an RNA virus and represent it as a growing population that is capable of mutating and reproducing. Concurrently, a population of vaccines try to mimic the virus in order to neutralize it. This cat and mouse chase between a co-evolving problem and solution space is made in hopes to better understand viral patterns and behaviour, their biological structures, and the necessary models that would aid in the production of a vaccine. Realistically, this model is nowhere near the biological representations of a real virus or vaccine, as that is beyond the scope of our capabilities and resources. This model does however substitute the genetic data of a virus with binary values, in order to replicate some of its biology. The node arrays are the protein spikes, the lethal points dictate its effectiveness, its mutation represents its RNA nature. We simulated to the best of our ability a virus that is natural, cloning and mutating without interference. At the same time, we sought to artificially manufacture a vaccine that would be able to deal with this virus. If this model did not contribute to the production of vaccines, it has in the least given us a practical understanding of virology and its challenges when modelling.

Additional Key Words and Phrases: Virus, Vaccine, Co evolution, Model, Evolutionary Algorithm, Genetic Algorithm

ACM Reference Format:
Lavi Ionas, Priyank Thakkar, and Cyril Zhang. 2018. Co-evolution of RNA Virus and Vaccine. 1, 1 (May 2018), 7 pages. https://doi.org/XXXXXXX.XXXXXXX

Our GitHub repo can be accessed at Virus Vaccine Co-Evolution

1 PROBLEM DESCRIPTION

It is not necessary to justify the importance of the scientific value in better understanding RNA virus' given the last 3 years and the impact that SARS-19 has made on humanity. Often things must get much worse before we bring our full attention to the issue at hand. Covid has made all of us extremely aware of our biological vulnerabilities to virus' and the time sensitive efforts necessary to building a vaccine. Although we are not virologist, we are computer scientists, and we were inspired to aid in the better understanding of virus' through various modelling techniques. In this project, we aimed to make as complex of a virus model as we could, given our capabilities and resources, in order to contribute to the general library of virus models that already exist. Our hopes are that the next people that decide to model a virus can be inspired by us in the same way we were inspired by other people and their work.

Authors' addresses: Lavi Ionas, Queen's University, Kingston, Canada; Priyank Thakkar, Queen's University, Kingston, Canada; Cyril Zhang, Queen's University, Kingston, Canada.

2 LITERARY REVIEW

Given that our backgrounds do not intercept with biology at any point in our educational careers, tackling on a deeply intricate biological system was no easy task. Fortunately, there are thousands of scholarly works on the nature and structures of a virus , some of which we read in order to get a general understanding. We stumbled upon a numbered amount of papers that worked on modelling a virus. All were extremely inspiring, focusing on different properties of virus'. Some papers simulated a virus within a host cell, where the bodies immune system combated an evolving virus. The paper we generated most of our inspiration from was *$ModellingSARS - COV - 2$ co-evolution with genetic algorithms* by Aymeric Vié [2]. This paper details the stochastic nature of the virus and the deterministic nature of social policies. It co-evolves the virus and policies where the virus aims to survive, and the social policies aim to minimizing infection rates. We were inspired by this model, and decided to make a similar, yet structurally different representation between the virus and a potential vaccine. We were also very keen in learning the basic details of RNA viruses and their special properties, like their high mutation rates. The paper *Why are RNA virus mutation rates so damn high?* by S. Duffy [1]gave us great insights into the evolutionary capability of these viruses.

3 EA DESIGN

3.1 Representation

Virus and Vaccine Representation

Virus:

The representation we used for the virus was a 64 bit string with each bit representing a protein spike. Every virus in the population was also assigned properties which determined how the virus would do in later generations; the properties were encoded to each virus and consisted of the lethal point index array, reproduction rate, name of the virus and the mutation of each virus.

VirusID	Representation
1	[[0,0,0,0,0,0,0,0]*8,[LethalPoints],[ReproductionRate],[MutationArray]]
2	[[0,0,0,0,0,0,0,0]*8,[LethalPoints],[ReproductionRate],[MutationArray]]
.	.
.	.
100	[[0,0,0,0,0,0,0,0]*8,[LethalPoints],[ReproductionRate],[MutationArray]]

Vaccine:

The representation we used for the vaccine was an 8 bit string with each string representing the protein spike. The motivation behind this representation was derived from biology, since vaccines are weak copies of viruses. Table 2 gives us a better insight for the representation of our vaccine.

VaccineID	Representation
1	[0,1,0,0,0,0,0,1]
2	[0,0,0,0,0,1,0,0]
.	.
.	.
25	[0,1,1,0,1,0,1,0]

3.2 Initialization

Virus and Vaccine Initialization

Virus:

The viruses were initialized as a 64 bit string consisting of all 0's as there are no mutations occurring at generation 0. The viruses were also assigned lethal points. These properties point to the protein spikes, nodes, of the virus that would actually be capable of penetrating a host cell and reproducing. The reproduction rate of the virus and the *virusID* are also initialized. A population size of 25 was chosen with each virus similar to the other with different lethal points assigned to it randomly.

Vaccine:

The vaccines were initialized as 8 bit strings which consisted of random binary values. An example vaccine [0,1,1,0,1,0,1,0] shows random mutation of the vaccine.

3.3 Parent Selection

As viruses biologically do not have parents, but clone themselves, we tried to retain that in our model. Our choice for selecting vaccines was implementing an FPS(Fitness Proportional) algorithm to select two parents, where the strongest two vaccines had the highest probability to be selected as parents. FPS allowed us to maintain the stochastic nature of selecting the parents which made sure that each individual had a non zero probability of being a parent regardless of the fitness. We also ensuring that the best two vaccines had the highest probability of being selected, in order to generate the best suiting vaccine as fast as possible.

3.4 Recombination and Survivor Selection

The recombination of the vaccines was achieved by using a crossover operator. The crossover operator generated a random point which would be used as the cutoff point to swap the portion of the two parents that are selected. This process gives two new copies of the vaccine. The two new copies of the vaccine also have a chance to mutate which is dictated by the vaccine mutation rate. The two new copies of vaccines were added to the vaccine population. No vaccines were removed from the population as they always had a chance to combat the next generation of viruses with better efficiency.

3.5 Vaccine Fitness

The fitness of the vaccine was calculated by its effectiveness in targeting virus lethal points. This was done by comparing the vaccine with the lethal point of the virus and calculating how similar it was to the lethal point, more points were awarded to the vaccine which matched the most lethal points in a virus population.

3.6 Virus Cloning

Biologically, viruses are known to clone rather than mate and reproduce, we tried to mimic this by assigning a reproduction rate to each virus in the population which is calculated by the sum of the mutation values in the mutation value array. If the reproduction rate was more than 0, the virus is allowed to produce n clones, where n ranged from 0 to the number of lethal points the virus had. This made it possible for viruses with more lethal point to produce more clones to infect the host, but as a drawback also made them more vulnerable to the vaccine.

3.7 Virus Mutation

The viruses are mutated based on the mutation rate parameter. A random location out of the 64 bits is chosen and the binary values are flipped to represent an activation or deactivation of the protein spike data. If the previous value at the location was 0, it is changed to 1 and if it was previously 1 it is changed to 0. An example showing the above process is shown below:

V1 - [0,1,0,0,1,0,0,0.....*56],

Location chosen - 4th and 5th

V1 after mutation - [0,1,0,0,0,1,0,0....*56].

3.8 Hyper-parameters

There were several hyper-parameters which contributed to the simulation of our evolutionary algorithm. These hyper-parameters are listed below:

- $VirusPopulationSize$
- $VaccinePopulationSize$
- $VirusLength$
- $VirusNodeLength$
- $VirusMutationRate$
- $VirusLethalPointMutRate$
- $VaccineMutationRate$

We tested the model with different hyper-parameters, each given different results of the model. We chose the best set of hyper-parameters by comparing the results of the model with each run. Slight changes in the mutation rate of the viruses gave us varying results and we encourage to experiment with the model with different values for the virus and vaccine mutation rate.

3.9 Process

Our population of virus' begin their life with an inability to reproduce, but a complete ability to mutate. Given a variable chance to mutate (1-5 percent), each virus has the ability to either positively or negatively mutate. This ties in to its biological nature where RNA virus' do not have DNA proofing and can hence undergo unhelpful or potentially self stunting mutations []. These mutations are tracked, and a virus' overall ability to reproduce is calculated as the sum of their mutation values.

Each of the eight [0,0,0,0,0,0,0,0] nodes of a virus are binary values that represent its protein spikes. Each protein spike is capable of mutating, but not each protein spike is capable of penetrating a host cell in order to reproduce. We introduce this concept by structuring a virus to have X amount of lethal points, where X can be anywhere from 1-4. A lethal point is a protein spike, or a node array, that is capable of actually penetrating a host cell. This lethal node acts as a mark for what the vaccine should target. If the vaccine matches the lethal point of a virus, that virus is bound to be neutralized. Hence, there is risk involved in having more lethal points, but a virus with more lethal points can produce more clones of itself.

A 1 would indicate that a mutation has occurred. The mutation value array keeps track of these mutation values, and the sum of each value in the mutation array is the reproduction rate of that specific virus for that generation.

More specifically, a [1,0,0,0,0,0,0,0] node that has mutated could result in a mutation value array of [X,0,0,0…] where X can be any float between -1 and 1. A more complex set of mutations would produce a more complex mutation value array. For example, a node [1,0,1,0,1,1,0,0] could result in a mutation value array of [0.33,0,-0.24,0,0.11,-0.56,0,0…] where the sum of -0.36 would be the reproduction value of that virus.

A positive score would allow a virus to clone itself, a negative one would mean the virus needs to reverse its mutations by chance, or hope for a positive mutation in future generations. If a virus does reproduce, we simulated its life cycle. The virus that is capable of reproducing will have a chance to make as many clones as it can before it inevitably dies that generation. The number of clones the virus can make is a random integer from 0 to the number of lethal points of that virus. Hence, there is a probability that a virus will fail to reproduce, but at the same time, there is a possibility that it will quadruple itself. Once a virus reproduces, its job is complete, and whether the white blood cells track down the infected cells or the cell itself dies, the virus parent is no longer.

The clones of the virus then have a chance to mutate, as each clone has a chance of becoming better or worse than its parent. This ensures that the population of virus' are dynamic.

Unlike traditional EA's, our virus tries to simulate itself in nature, unbound by human interference. Since virus' clone themselves in nature, they cannot undergo parent selection, nor can they do the usual crossover. We implement these traditional strategies when we co-evolve the vaccine. The goal of the vaccine is to, by the end of the evolution, eradicated as many virus' as possible. The highest the neutralization rate, the higher the fitness of a vaccine. We select the best vaccines and cross them over, mutate them, and compare them to the lethal points of the virus. The more virus' the vaccine can target, the better its fitness, the more likely it is to be selected as a parent in the next generation. We judge the potency of a vaccine when the generational loop ends, where we simulate the vaccine being applied to the virus population and judge the percent of said population it would neutralize. The highest the percent, the more effective the vaccine.

4 EA RESULTS

We ran our model with different hyper-parameters, constantly changing the mutation rate of the virus and vaccine and also experimented with different initial populations of both the vaccine and the virus. Our model showed promising results in some of these runs as we were able to achieve 0.86 highest accuracy for the vaccine which resulted in 400 population of the virus being terminated after 15 generations. As we can see in the graphs , the vaccine in initial generations has a very low efficiency rate as it has been randomly generated and allowed to learn how the viruses are evolving and evolve according to the viruses to combat them. In the latter generations as the vaccine co-evolves with the viruses , it starts showing promising results.

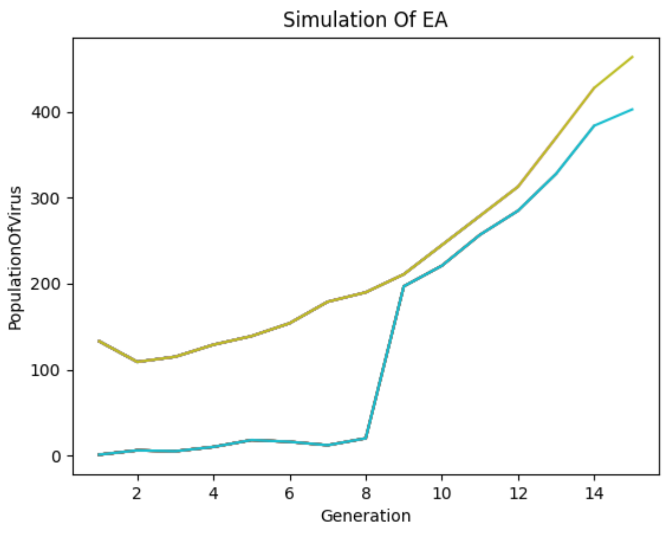

Fig. 1. Plot of Virus Population per generation

Generation	Virus Population	Vaccine Fitness(No of Virus Killed)	Efficiency Of Vaccine
1	133	1	0.007518796992
2	109	6	0.055045871559
3	115	5	0.043478260869
4	129	10	0.077519379844
5	139	18	0.129496402877
6	154	16	0.103896103896
7	179	12	0.067039106145
8	190	20	0.105263157894
9	211	197	0.933649289099
10	245	221	0.9020408163265
11	279	257	0.9211469534050
12	313	285	0.9105431309904
13	370	328	0.8864864864864
14	428	384	0.8971962616822
15	464	403	0.868534482758

5 COMPARISON

Considering the novel nature of viruses in EA designs, it was very challenging for us to find a library or algorithm that could perform in the same way to compare our methods without investing a considerable amount of our time. This EA has many variables that cannot be substituted and there is no clear, evident answer. Each generation requires a solution that is specific to the current population of viruses. For this reason, we failed to apply another algorithm to solve this issue.

6 DISCUSSION

Despite our initial lack of viral knowledge, we knew EA's were the right choice for this type of problem. Our model tackles on a multitude of variables and simulates the organic growth of a virus. In a problem space where the solution is not evident, where the virus is not engineered to be in a particular way, our EA model is very good at adapting the vaccine to fit the current state of the virus population. At each generation, the virus changes drastically, and with it so does the vaccine.

The limitations we faced in this project have been mostly technical. Our limited biological knowledge has held us back insurmountably in generating an accurate model for an RNA virus. The complexity of a virus cannot easily be translated to a code representation, and our research did not point us in any directions that would aid in that. Most of what we read orbited a binary array representation of viruses, shying away from genetic data. This is understandable, as genetic data is extremely complicated and beyond our scope of reasonable capability. Given our resources and knowledge, we were able to make the best of what we had, creating a model that, although it does not innately represent nature, tries to mimic it.

In the future, we seek to progress the model beyond this course. Our journey with virus' have just begun, and we have ideas on how to improve the model. Given more domain knowledge and better understanding of virology, we could make an even more complicated and targeted model that could better represent an RNA virus. Eventually, we seek to incorporate more basic genetic data and explore the interactions between artificial virus' and the hosts they infect. We are keen on exploring the chemistry behind virus host interactions, despite the feeling that this is beyond our capabilities. With each step, we will become more and more confident in tackling much more difficult representations and models.

7 REFERENCES

References Used in the Paper

[1] Duffy S. (2018). Why are RNA virus mutation rates so damn high?. PLoS biology, 16(8), e3000003. https://doi.org/10.1371/journal.pbio.3000003

[2] Vié, A. (2018, February 25). Modelling SARS-COV-2 coevolution with genetic algorithms. ArXiv. Retrieved February 11, 2023, from https://www.semanticscholar.org/paper/Modelling-SARS-CoV-2-coevolution-with-genetic-Vi

Received 25 April 2023