# Utilizing Genetic Algorithms to Optimize Spider Web Design

Nigel Woodhouse

1412716

Faculty of Computer Engineering

University of Alberta

Dec 23, 2019

ECE 720

Petr Musilek

## Contents

## Abstract

The essence of evolutionary programming stems from observations in the biological world. In this paper, an evolutionary programming technique is applied back to the biological world through the construction of a spider’s web. A spiderweb is constructed using basic, well-established rules. Its function, too, is basic. This evolutionary adaptation took millions of years to implement and many more millions of years to perfect into the typical spiral shapes one can observe in nature. It is possible to recreate very similar attributes using an evolutionary computer program following the same ruleset. Applying a random walk approach to a simple initialization, the performance of a web can increase efficiency up to 32% by balancing the fitness of the web’s function (number of insects caught) and cost (amount of thread used). After 100 generations, the digital web can resemble the geometry of a regular polygon, validating the results qualitatively.

## Introduction

Charles Darwin promoted two main premises in is work, On The Origin of Species [1], published in 1859. The first and most famous conclusion is the survival of the fittest. That is, individuals of a population that can satisfy their niche to a higher order than other members are more likely to survive to the point of sexual maturity and pass along their genetic information to the next generation. The attributes that made this individual successful proliferate through future generations. Further ideas proposed, such as those by evolutionary zoologist, Professor Richard Dawkins, has coined this as an evolutionary "arms race" [2] as different species compete against one another. The other keystone inferred by Darwin is that the origins of life had simple beginnings. Looking closely at a biological entity displays enormous complexity. Rewinding the clock, however, reveals that these complexities occurred gradually over millions of generations, gradually improving on the previous design through slight mutations. Following this line of reasoning, it is easy to fathom simple origins to all biological life.

A spiderweb, from initial inspection, can appear complicated. Watching a spider construct its web reveals a simple ruleset that creates the beautiful evolutionary adaptation. When analyzing an orb-weaver spider, one can observe that it creates radial threads first, followed by frame threads, and lastly, spiral threads. The elegant dance of the spider creates a biological masterpiece with a well-defined function; to catch insects. Following Darwin's ideas described above, one can assume that spider webs did not always appear like this and must have originated from a basic and unsophisticated design. Professor Richard Dawkins elaborates on this premise in his Christmas lecture in 1991, Growing up in the Universe [3] [4] and his book, Climbing Mount Improbable [5]. The focal point of Dawkins's discussion is supplemented with an evolutionary computer program designed by Peter Fuchs and Thiemo Krink [5], designed precisely to accomplish Darwin's ideas.

Biological evolution is slow. Evolutionary programming accelerates the process. The goal of this line of research is to stitch Darwin's premise of simple beginnings with the ruleset governing a spiderwebs design in a computer program. The algorithm considers the completeness of the function (catching insects) with the cost of doing so (amount of thread used). From initial design, the web should evolve through several generations of a random walk to optimize both components of the fitness function to resemble a spiderweb design one may find in nature.

## Materials and Methods

Some basic terms regarding the anatomy of a spider web must be defined. The three main components focused on are radial threads, frame threads, and spiral threads. Radial threads are those that begin at the origin and move radially outward. Frame threads connect the ends of the radial threads. Spiral threads occur anywhere along the length of one spiral thread and connect to an adjacent radial thread, providing the web with its stereotypical spiral shape. Other parts of a spider's web exist in nature but are not analyzed in this research.

The algorithm described is programmed in Python 3.7 in the Visual Studio Code environment. The algorithm borrows heavily from the TkInter and PIL libraries. The TkInter library displays the results of the algorithm to the user. The PIL library uses the same data; however, the PIL library saves the results to memory for image processing.

Every generation, eight new child spiderwebs are generated through random mutations of the parent web through parameters described below. The parent web is compared against the 8 child webs. Ergo, 9 webs are compared per generation, which are organized in a 3x3 grid and presented to the user. Figure 1 in the Appendix displays an outline of the algorithm.

A web is constructed in a TkInter canvas. The canvas is 300x300 pixels. The center of the canvas operates as the origin of the web. Created around the origin are 8 points drawn in polar coordinates that act as the ends of the radial thread. A straight line connects these points. The ends of the radial lines are connected to the adjacent radial line by a straight line, acting as frame threads. The initial parent web contains no spiral threads.

The radial threads are the critical elements to the evolution of the web, as all other features rely upon its parameters. Since these threads are constructed using polar coordinates, two parameters influence its properties: length and angle. The initial length of all radial threads is set randomly between 50 to 100 pixels. The length of any one radial thread can vary by ± 10 pixels per generation, with the minimum length being 5 pixels. The maximum length a radial thread can be is half of the canvas width or 150 pixels. The initial angle of the radial threads is between 0 to 2π radians. The angle of any single radial thread can vary by ± 0.2 radians per generation.

The creation of the spiral threads is based upon the properties of the radial threads, as described above. Radial threads are interpreted as a unit vector, and thus, the spiral threads exist some length along that unit vector and connect to an adjacent unit vector. Two properties influence the manifestation of spiral threads. The first property is how far along the spiral thread exists along the unit vector of the corresponding radial thread. The position of a spiral thread along one radial thread and does not influence the location of the same spiral thread on the adjacent radial thread. The other condition is that the spiral thread can randomly be turned on and off to add more variability to the solution space. A spiral thread has a 50% chance of being turned on if it is off and vice versa. The position of a spiral thread can only be modified if it is on. 20 spiral threads can exist on a radial thread at a single time.

An artificial selection method is first implemented in the program. This allows the user to select which of the 9 webs the individual deems best based upon their own set of standards. The web selected becomes the parent for the next generation.

A spiderweb has a clearly defined function. Its purpose is to serve as an evolutionary adaptation for the host spider to catch nearby flying insects as a method to obtain food. It is self-evident that the more insects a spider web catches, the more food the spider has access to, and the more likely to survive and pass on its genes to the next generation. Red pixels in the canvas space represent nearby flying insects. Their positions are randomly scattered across the canvas and modified every generation.

However, a spider contains only so much silk to construct its web, so the more web used serves as a cost. The number of insects caught versus the amount of thread used serves the basis for a fitness function. The governing equation to describe the fitness function is described as:

(1)
$$Fitness = \frac{[(Thread\ Fitness)^n + (Insect\ Fitness)^m]}{2}$$

Where thread fitness is the consideration of how much thread explores the space and insect fitness calculates how many insects the web captures. The implementation of exponents n and m adjust the weight of the fitness parameters and increase flexibility to obtain desirable results. Implementing Equation 1 moves the algorithm from one that requires human input (artificial selection), to one that is automatic, mimicking natural selection.

The PIL library evaluates the insect fitness of each web per generation to satisfy Equation 1. To calculate the number of insects caught, the number of red pixels per canvas is evaluated. The fewer the number of red pixels per canvas, the better the web, as the black outline of the web's threads overlap that of the red insect pixels. The pixel length of the radial threads and spiral threads determines the web cost.

The thread fitness and insect fitness are normalized between [0,1]. Of the 9 webs, the canvas that contains the fewest red pixels and uses the least amount of thread scores the highest fitness and serves as the parent for the next generation.

The number of radial threads, spiral threads, canvas size, and insects, as well as the mutations per generation, can be modified easily. The values provided are one example to explore the solution space that produces desirable quantitative and qualitative results.

## Results

Equation 1 describes the fitness of all 9 webs per generation. While the fitness per generation is always normalized between [0,1] for each parameter, looking at raw values provides a gradient of improvement across numerous generations.

In the first generation, the initial thread length used can vary quite wildly based upon the random variables from the initialization process. Through multiple trials, the initial thread length is approximately 1500 ± 150 pixels. This only contains radial threads. Due to the implementation of spiral threads, the thread length used by the 100th generation can increase to 15000 pixels, taking up 16% of the canvas. The length of thread used per web is normalized by subtracting and dividing each web by the maximum thread length that exists amongst any canvas. This finds a relative difference per generation.

Insect fitness is evaluated based upon the number of red pixels that are covered by the black web. The colour of the web and the insects must be different colours for this evaluation to take place. The number of insects produced is based upon the size of the canvas. With the canvas being 300x300 pixels, the number of insects produced is equal to one dimension (300, in this case). Each insect has a diameter equal to the number of insects divided by 100. In this case, each insect has a diameter of 3 pixels that fills an area of approximately 12 pixels. With the size of the canvas (300x300) and the area of insects (300x12), it predicts that insects represent a maximum limit of 4% (or 3600 pixels) of the canvas. The position of insects is created randomly throughout the canvas. This upper limit prediction of 4% does not account for any overlap that may occur. The arrangements of insects are identical in each canvas and vary every generation.

On the first iteration, approximately 2590 ± 30 pixels are expected to be red. The best web is the canvas that contains the fewest red pixels. The difference between the best web and every other web is found, to keep the results relative per generation. The differences are normalized. On the 100th  generation, the number of red pixels in a canvas decreases to 2180 ± 100, showing a 16% increase in performance.

In the equation, both the thread fitness and insect fitness are raised to the exponents n and m, respectively, allowing the weight of the parameters to be adjusted. These exponents influence what the algorithm deems is more critical, searching the domain space or capturing known insects, providing a balance between exploration and exploitation. Through experimentation, $n = 0.3$ and $m = 1.3$ provides a balance between exploration and exploitation.

Since the radial threads, maximum length can only be one half of the canvas width; this effectively inscribes a circle within the square canvas with the same diameter/side length. Ergo, the maximum area the web could cover is 78.5% of the canvas, if there are enough spiral threads to cover the entire inside of the inscribed circle. Therefore, a maximum of 78.5% of insects.

What is essential, however, is the general shape that the web takes place. Regardless of the initial starting configuration, it is observed that, given enough generations, the web tends towards some regular polygon with a number of sides less than or equal to the number of radial lines. One can also observe that the spiral threads tend towards even spacing along the radial line's length and little crossing of the spiral threads occur with each other. These qualities are not implemented in the programmed yet are desirable emergent properties.

## Discussion

The following discussion borrows from the parameters described in the methods section. Regardless of the parameter values, qualitative results remain constant.

The purpose of this investigation is to apply evolutionary programming back to the biological world. Applying this concept onto spiderwebs is beneficial for two ways, as described by evolutionary zoologist, Professor Richard Dawkins, during his 1991 Christmas Lectures, Growing up in the Universe [3][4]. In his presentation, he explored the same concept through a program written by Peter Fuchs and Thiemo Krink [5]. The great thing about spiderwebs, as Richard Dawkins describes, is that "the webs are two-dimensional structures and we know what they're for, they're for catching insects. So, the benefit is simply going to be the number of insects caught and the cost we can calculate because the cost is the amount of silk used." [3].

Through simple rules, it is possible to recreate the configuration of the web in mere minutes, as it would take millions of years and generations to yield similar results through biological evolution.

The main objective is the geometry of the web produced and that it should mimic webs observed in nature. After 100 generations, the resultant figure does resemble that of a spider web. An example of a webs progression is displayed in Figure 2 in the Appendix. There are points to address after observing multiple iterations of the web's development.

The first point to examine is that both the thread fitness and insect fitness both yield a 16% increase in their respective functionality, yielding a web that is 32% more efficient than its starting design when evaluating the objective fitness function. This increase in efficiency shows that the methods used to evaluate said properties are proportional and are dependant upon one another. Continuous incremental improvement is the desired outcome of this random walk.

The ideal web should display frame threads that are regular polygon whose number of sides is equal to the number of radial threads, and that the spiral threads should follow the same shape. This geometry maximizes the area covered with the smallest perimeter. As the number of radial threads increases, the resultant shape should approach that of a circle, for a circle has the greatest area to perimeter ratio of any shape. Regarding the problem at hand, this translates to exploring the largest search space with the least cost.

In the same line of thinking, optimally spiral threads should not cross each other. Having spiral threads cross means that there are resources wasted covering the same location. Observations of later generations show that crossing does decrease yet will always be present. Spiral threads crossing will always exist, for the position of a spiral web along the length of its radial thread can only be modified when it is on. As established, a thread that is on has a 50% likelihood of being turned off. It suggests that the number of spiral threads that crossover should not exceed 50% if the claims mentioned above are true, it can be observed from Figure 2d in the Appendix that less than 50% of the threads cross one another. This property, while not implemented into code, is an expected emergent property through optimization of the goal.

An unusual property that has arisen, however, is that the web may approach the geometry of a regular polygon that has a number of sides less than the number of radial threads. The algorithm can get locked inside of a local minimum, which is a known flaw with random walks. Up to 25% of radial threads may continually reduce their lengths to a minimum of 5 pixels. Neighbouring radial threads converge towards the angles of the short radial threads. Conclusions derived from analysis of the raw data infer that the manifestation of this behaviour is that the existence of radial threads is expensive. A balance between the number of radial threads and exploration is found. In doing so, most spiral threads connected to the short radial threads turn off, reducing cost further and yielding a local minimum of the objective function. Adjusting the weighting of the thread fitness parameters in Equation 1 (n and m) can improve the algorithm's desire to explore and avoid these local minimums. Examples of a web finding these local minima are shown in Figure 3 in the Appendix.

While this behaviour may be undesirable, it illustrates a valuable lesson of evolution; flaws can pass through the organism's lineage. Evolution has no foresight and cannot go back to resolve said shortcomings. In the concept of the research, a local minimum is difficult to escape as the algorithm has no previous knowledge of what could be better, and the algorithm cannot select a series of less adequate individuals in hopes of a better solution. As Professor Richard Dawkins puts it (in a metaphorical sense) when describing evolution as Mount Improbable that the organism "is now unable to escape because escaping would mean going downhill… and the one thing you cannot do on Mount

Improbable is ever go downhill" [6]. This property was not anticipated when setting out in this line of research. These traits align well with established knowledge of genetic algorithms, particularly random walks.

There is no optimum design present, as the position of insects changes every generation; the solution space is continually changing. Hence why there is no convergence criterium shown in the algorithm in Figure 1 in the Appendix. The No Free Lunch Theorem suggests that there is never one genetic algorithm that will always perform better in every problem domain. By extension from this theorem, there will never be one best web that can satisfy every solution space. Observations do show that one web can remain dominant over several sequential generations before being overthrown. Although, when this occurs, the solution may deviate for a few generations and return to a geometry that had expressed persistent dominance previously.

The algorithm designed is based upon previous work by Peter Fuchs and Thiemo Krink, as shown in Professor Richard Dawkins's presentation and book [3] [4] [5]. While not having access to their program, it is assumed that it, too, is based upon random walks. From the videos and book, however, it can be suggested that other rules are implemented into Fuchs and Krink's algorithm that this one does not consider. Such rules can include, but not limited to, an equal spacing of spiral threads, the prevention of spiral threads from crossing over, and a different approach to spiral threads being turned on and off. A more elegant initial construction is also present in Fuchs and Krink's algorithm. These rules that the other researchers implemented are rules observed in nature and aid in the avoidance of a local minimum and increase the likelihood of producing a geometry of a polygon whose degree is equivalent to that of the number of radial lines. While the rules suggested are not followed in this algorithm, similar results can still be obtained, yet is less robust. Regardless, the premise between the two algorithms is the same, and similar conclusions can be derived from both.

## Conclusion

Darwin's ideas are revolutionary and redefined all of biology; it is the lens used to observe the natural world. Areas of programming and computation have adapted these concepts from biology. The goal of this line of research is to take these ideas and reapply them back to biology. The method undertaken is the replication of a spiderweb through a random walk. A spiderweb is a simple 2D figure with simple rules and a known function, making it an ideal candidate. From a random initial case, the rules implemented provided a 32% increase in the web's overall efficiency through 100 generations. The web took upon features of a regular polygon whose number of sides is equal to the number of radial lines. These quantitative and qualitative characteristics are the expected outcomes based on biological observation and mathematical optimization. This simple realization continues to validate Darwin's claims 160 years later through means unthinkable in his day.

## Future Research

The application of optimizing a spiderweb does not hold much value in of itself in the engineering world. However, knowing the function of the web can lead one to investigate similar problems. The function of the web is to behave as a net to catch nearby insects. A similar engineering application that could take benefit is, for example, capturing plastics adrift in the ocean. This concept could be extremely beneficial in capturing microplastics, as the size of the "insects" in the described algorithm is adjustable. The chaotic nature of a random walk may yield designs for nets previously unthought-of by humans.

However, this is just a starting point to bridge evolutionary algorithms with mechanical design and optimization. All that would need to change is the criterium for function and cost. Such examples could include the design of a drone body to be as aerodynamic as possible while reducing weight. Alternatively, the design of PCB boards with the most computational power while reducing resources used and waste heat generated. The possibilities are endless. With such manufacturing techniques like 3D printing and CNC routing, it would be possible to manufacture these more complex yet efficient designs. This thought process could revolutionize engineering design.

## Literature Cited

[1] – C. Darwin, On The Origin of Species by Means of Natural Selection, or Preservation of Favoured Races in the Struggle for Life. London: John Murray, 1859.

[2] - "Arms races between and within species," Proceedings of the Royal Society of London. Series B. Biological Sciences, vol. 205, no. 1161, pp. 489–511, 1979.

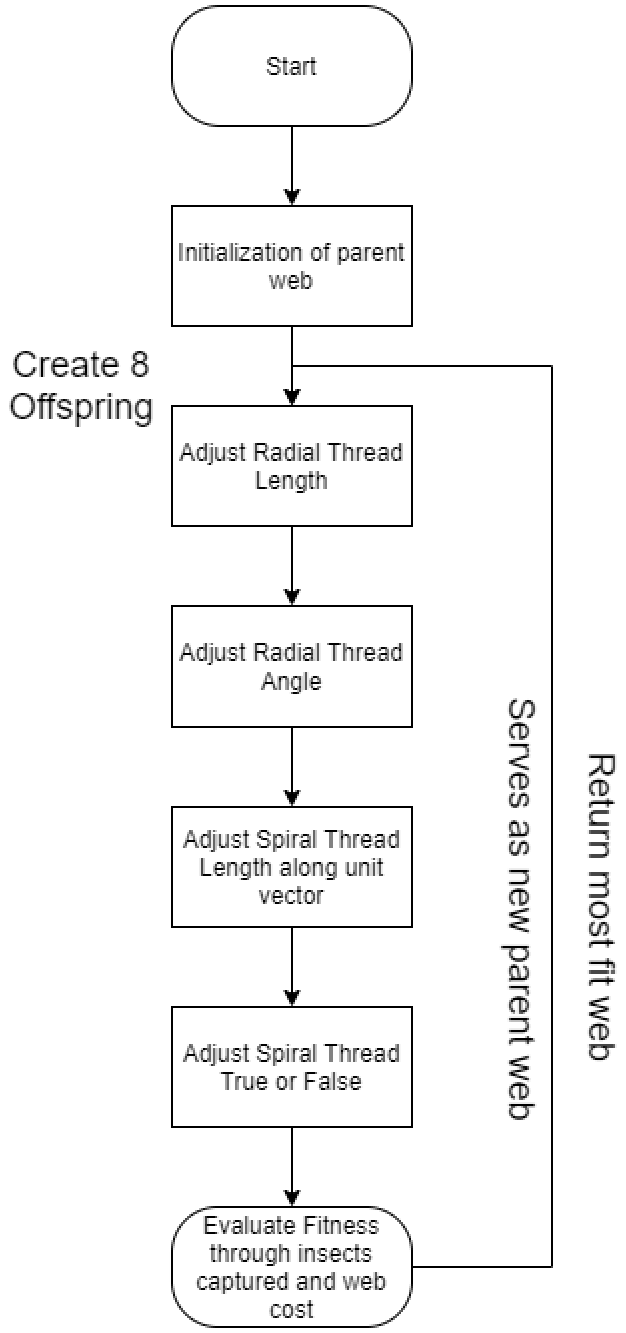
[3] - howlifebegan. (2011, August 14). Richard Dawkins -Spider web evolution [Video file]. Retrieved from https://www.youtube.com/watch?v=QdtYRJqNe9I

[4] – Richard Dawkins [Richard Dawkins Foundation for Reason & Science]. (2009, February 7). Ep2: Designed and Designoid Objects - Growing Up in the Universe - Richard Dawkins [Video File]. Retrieved from https://www.youtube.com/watch?v=xGyh1Qsw-Ak

[5] - R. Dawkins, Climbing Mount Improbable. Hoboken, New York: W. W. Norton and Company, Inc, 1996, pp. 58-62.

[6] - Richard Dawkins [Richard Dawkins Foundation for Reason & Science]. (2009, February 7). Ep3: Climbing Mount Improbable - Growing Up in the Universe - Richard Dawkins [Video File]. Retrieved from https://www.youtube.com/watch?v=YT1vXXMsYak

## Appendix


Start
Initialization of parent web
Create 8 Offspring
Adjust Radial Thread Length
Adjust Radial Thread Angle
Adjust Spiral Thread Length along unit vector
Adjust Spiral Thread True or False
Evaluate Fitness through insects captured and web cost
Serves as new parent web
Return most fit web

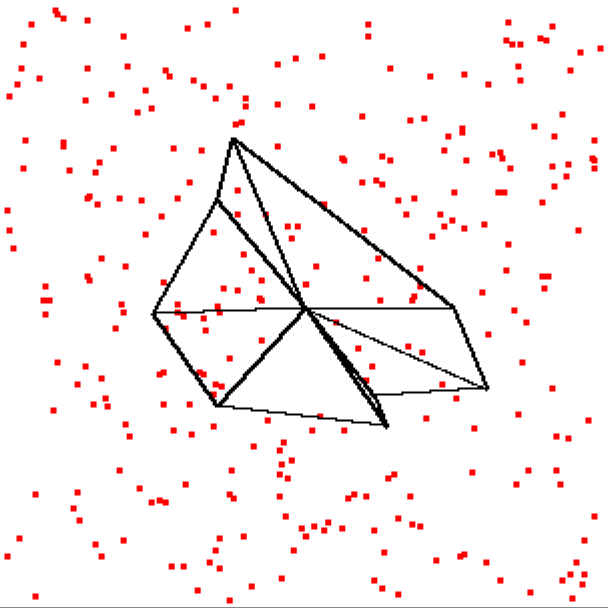


Figure 2a: Generation 0

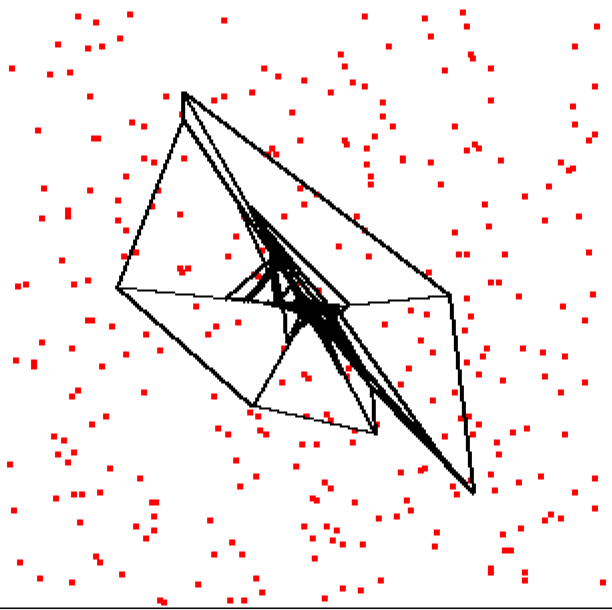


Figure 2b: Generation 10

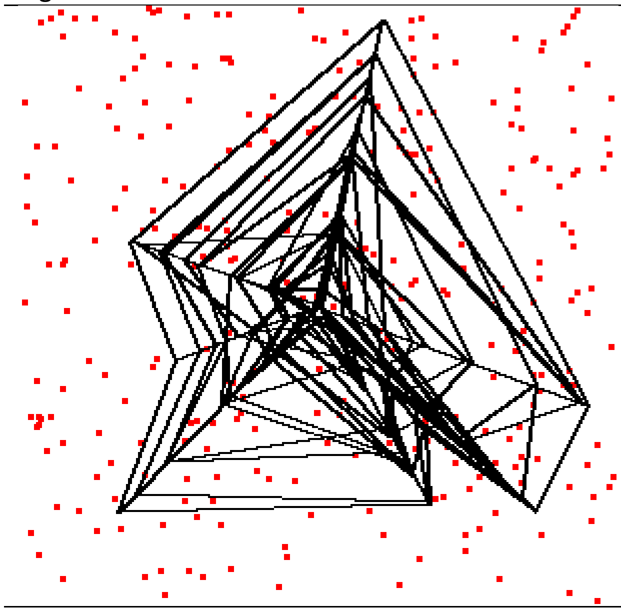


Figure 2c: Generation 50

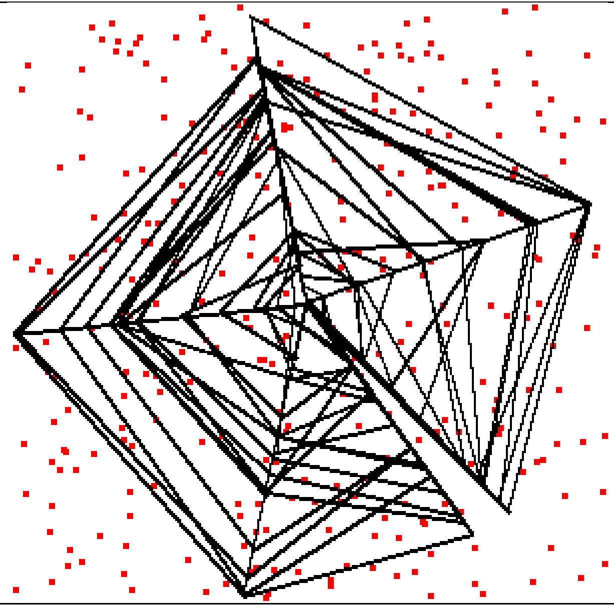


Figure 2d: Generation 100

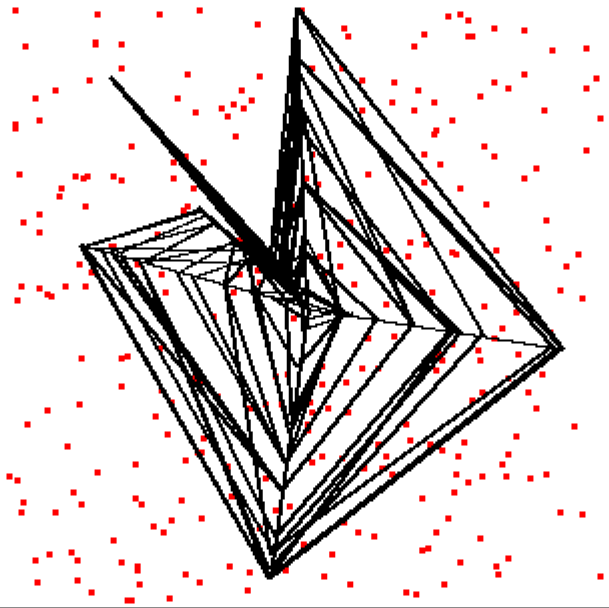


Figure 3b: Flaw found. n=1, m=0.5. Generation 90

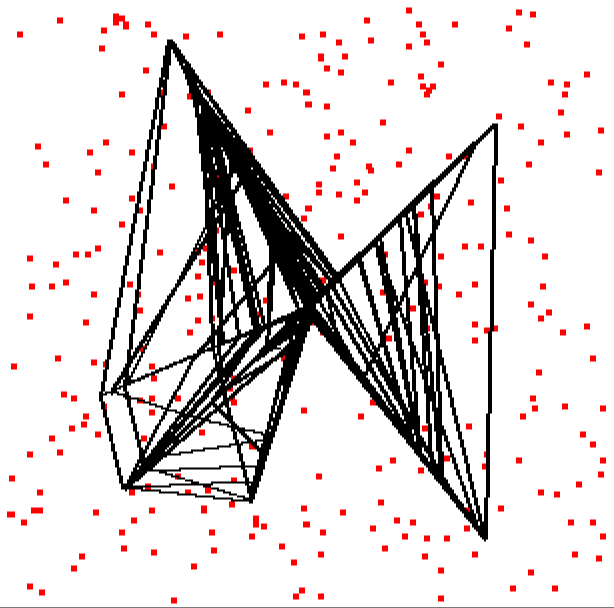


Figure 3b: Flaw found. n=1.5, m=1. Generation 70