Silverfish Simulation

M I R A N A L I
a n d V L A D A N N I K O L I C

KTH Computer Science
and Communication

Bachelor of Science Thesis
Stockholm, Sweden 2012

Silverfish Simulation

M I R A N A L I
a n d V L A D A N N I K O L I C

DD143X, Bachelor's Thesis in Computer Science (15 ECTS credits)
Degree Progr. in Computer Science and Engineering 300 credits
Royal Institute of Technology year 2012
Supervisor at CSC was Henrik Eriksson
Examiner was Mårten Björkman

URL: www.csc.kth.se/utbildning/kandidatexjobb/datateknik/2012/
ali_miran_OCH_nikolic_vladan_K12002.pdf

Kungliga tekniska högskolan
Skolan för datavetenskap och kommunikation

KTH CSC
100 44 Stockholm

URL: www.kth.se/csc

Abstract

This essay is part of the course DD143X, Degree Project in Computer Science, First Level. The authors chose to simulate evolution in a silverfish population. The problem statement sought to answer if one could find the fittest silverfish after several generations with the created simulation. The simulation was implemented in Java and it was stated that silverfishes' Cowardliness Level (CL) controlled their behaviour. They have two distinct behaviours - feign death and flee to the hideout. Three different scenarios were run from which a uniform result was obtained and the simulation succeeded in providing a clear answer to the problem statement.

Referat

Silverfisksimulering

Denna rapport är ett moment i kursen DD143X, Examensarbete inom datalogi, grundnivå. Författarna valde att simulera evolution i en silverfiskbefolkning. Problemspecifikationen syftar på att, med hjälp av simulationen som skapades, ge ett lämpligt svar på frågan om vilken silverfisk som klarar sig bäst. Simulationen implementerades i Java och det skrevs att silverfiskarnas Cowardliness Level (CL) kontrollerar deras beteende. Silverfiskarna har två distinkta beteenden - att spela död och att fly till gömstället. Tre olika scenarion kördes och man lyckades få fram ett entydigt resultat från dessa körningar.

Contents

Chapter 1

Statement of Collaboration

This project is a product of the collaboration of Vladan Nikolic and Miran Ali and constitutes of a planning phase, the implementation of code and the presenting of the code and its results in this essay. The authors met with the supervisor at four different occasions to discuss the essay's progress and brainstorm various possibilities to improve the work.

Both authors have a full understanding of what has been done and the contents of this essay. The structure of the code was decided after a few sessions of brainstorming between Mr Ali and Mr Nikolic. The ideas that surfaced during these sessions were written down to be used later on.

The bulk of the code was also composed together, with one of the authors performing the actual writing while the other followed and made various suggestions and vice versa. After the final meeting with professor Eriksson the code was functioning properly but there were not any clear results to present. The code had to be improved and this was done individually, but both parts contributed with roughly the same amount of work.

During this, the essay was continuously being improved by both authors. The improvement consisted of writing new material and improving existing paragraphs.

Chapter 2

Introduction

A silverfish is a wingless insect in the order Thysanura [1]. The common name is derived from its fish-like appearance and wiggling movement [1]. The silverfish is a nocturnal creature that prefers to live in moist areas and its diet consists of starchy foods [1].

In this essay, different assumptions will be made on the silverfishes' behaviour, birth rate and mortality rates. Because of this, many statements regarding the simulation and the premises surrounding it may or may not be based on actual facts.

It is important for the reader to know that the simulation of evolution is the important aspect of this essay, not the silverfish. The silverfish is merely a container of properties and functions.

2.1 Problem Statement

When the lights are on, silverfishes demonstrate different behaviours. One type of silverfish will feign death to avoid detection, while others will try to flee to their hideout before getting stomped to death. The question that is to be answered is:

After several generations of silverfishes have had time to grow in a bathroom, what kind of silverfish will prevail? The bold one that feigns death or the cowardly one that flees to the hideout?

What the problem statement really asks, is if it is possible to simulate evolution in such a way that an answer is clearly given. More specifically, is there an optimal solution for the silverfish spieces?

It is important to capture the two fundamental parts of evolution. That is, natural selection and random variation. What Charles Darwin called natural selection is simply the preservation of characteristics useful to the species through generations [2]. Random variations are changes that occur randomly at birth, both good and bad. The idea is that the good changes will be preserved and the bad changes will not be preserved.

The main approach to this problem will be to make the creation of the simulation

as simple as possible while it still should give presentable results. If one starts incorporating complicating factors, which might not even lead to remarkable results, the entire model will probably be much harder to implement within the limited time span.

2.2 Programming Environment and Simulation Technique

Originally, it was thought that MATLAB would prove to be a convenient choice for programming the simulation. However, during the course of the project, several difficulties were encountered and it was decided that implementing the model in Java would be more convenient.

The supervisor for this project, Henrik Eriksson, spoke about the differences in using event-based or time-based simulation. That is, if the model will be based on a chronological sequence of events where operations are made once an event is initiated or if the model will be based on events spread across a time-interval, where an event is initiated depending on current position in the time-interval [3]. Before modelling can start, a choice has to be made between these two different simulation techniques.

Chapter 3

Method

This part of the essay covers how most factors of the simulation were implemented. These factors involve the implementation of a silverfish and its most important functions, storage of several silverfishes and the events that compose the simulation.

3.1 Event-based Simulation

A choice was made for the use of event-based simulation. The reasons for this decision are that the authors felt that it would be simple to understand the process of the simulation and that it would be simple to write the code.

Introducing event-based simulation means that certain operations are performed once a certain event is triggered [3]. Instead of implementing time, the simulation depends on iterations and one iteration equates to one generation of silverfishes.

3.2 A Silverfish

This subsection covers how a silverfish functions in the simulation and how it has been implemented in Java.

3.2.1 Cowardliness Levels and Functions Incorporating it

As mentioned in the introduction, silverfishes will demonstrate different behaviours once the lights are turned on. Some will flee to the hide-out and others will feign death, hoping that they will not be noticed. The property of a silverfish that controls this type of behaviour and what this essay will reflect upon is the cowardliness level (CL) of a silverfish. Each silverfish has a CL - a decimal number between 0.0 and 1.0 that determines the likelihood of said silverfish to feign death when the lights are turned on. A silverfish with CL = 0.0 will never flee and one with CL = 1.0 will always flee. For instance, a silverfish with CL = 0.5 will only flee if the distance to the hideout is less than (or equal to) half of the maximum distance from the

hide-out. The CL of a silverfish also correlates with its distance to the hide-out when it steps outside. The braver the silverfish, the further away it is on average.

Silverfishes demonstrate these behaviours with the help of simple algorithms. These algorithms decide if the silverfish will flee or feign death and the probability of death after making its choice. The code for these algorithms can be found in the next subsection.

3.2.2 Silverfish Algorithms

The algorithm for **putPosition** is very straightforward. It simply randomizes a point for any silverfish moving out of the hideout. It also adjusts the position depending on the CL of the silverfish. Braver silverfishes are driven further away from the hide-out.

```
1     public void putPosition(int possiblePositions) {
2
3         //Invert CL, used to make sf with low CL
4         //go further away from hide-out
5         double invert = 1.0 - this.getCl();
6
7         //Adjust the length they go depending on invert
8         //adjustment factor goes from 1 to 101
9         int adjust = brando.nextInt( (int) (invert * 100
              + 1)) ;
10        this.setPosition(brando.nextInt(possiblePositions)
              + 1 + adjust);
11
12        //Set to 100 if out of bounds (> 100)
13        if(this.getPosition() > 100){
14            this.setPosition(100);
15        }
16    }
```

The algorithm below decides if the silverfish will flee or feign death by incorporating CL and distance to hide-out.

```
1     public boolean willRun(int possiblePositions) {
2         return (position / possiblePositions <= cl);
3     }
```

In the function **run**, it is decided if the silverfish will be stomped to death before he makes it to the hide-out. The ratio between current position and all of the possible positions has been reused. In this case, the further away a silverfish is from the hideout, the larger is the probability of death by foot.

```
1      public void run(int possiblePositions) {
2          if(brando.nextDouble() <= (position /
               possiblePositions)){
3              this.setDead(true);
4          }
5      }
```

In the function **feign**, it is decided if the silverfish will succeed in feigning death. If so, the silverfish will become fertile from the heat of battle and fertility guarantees it twin babies - a very large benefit for those who survive when feigning death. The probability of actually succeeding is 50 %.

```
1      public void feign(int possiblePositions) {
2          double rand = brando.nextDouble();
3          if(0.5 <= rand){
4              this.setDead(true);
5              return;
6          }
7          //Makes silverfish fertile
8          this.fertile = true;
9      }
```

Apart from these algorithms, the silverfish also carries a hand-full of get- and set-functions used to fetch and change certain properties of the silverfish, respectively.

3.2.3 The Circle of Life

Below is a chart depicting the process a silverfish goes through when it is time for the silverfishes to come out of the hide-out.

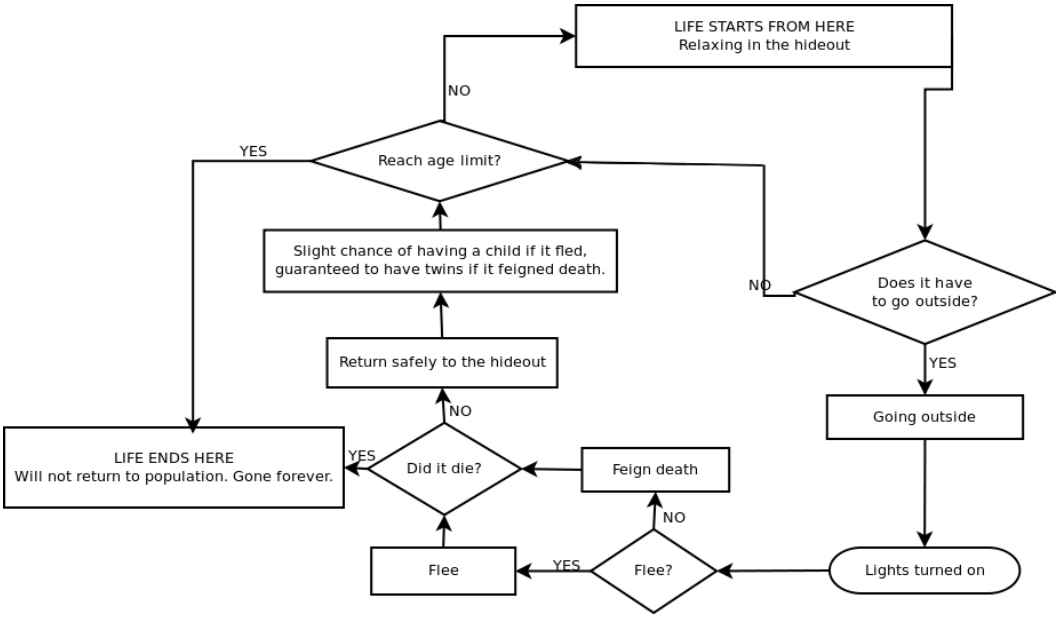

Figure 3.1: A chart that illustrates the circle of life and its processes for a silverfish.

The chart starts from the event labeled as “LIFE STARTS FROM HERE” and the silverfish goes through this process until it is either murdered by a foot or old age, ending the chart in the event labeled as “LIFE ENDS HERE”.

3.3 Storing Silverfishes

The Linked List that can be found in Java’s API was extensively used to store silverfishes. To keep tabs on the population a Linked List was used to store every living silverfish. When a silverfish has gone outside it has left the hide-out and also the population, which is why the silverfish is moved from a population list to an outside list.

3.4 The Simulation

This section covers the implementation of events in the simulation and the variables used in the simulation.

3.4.1 Variables

The actual simulation is comprised of several variables and algorithms that combine these variables together with different probability functions. A table of the most important variables and what data they store can be found below.

Name	Description	Type
possiblePos	Used to decide the amount of possible positions a silverfish can be present on.	Integer
startPop	The size of the starting population	Integer
iter	A variable that decides how many times the process of a silverfish is to be carried out. In layman terms, one could say that iter equates to the amount of generations born.	Integer
ageLimit	A universal age limit for the silverfishes. Any silverfish reaching this ageLimit will perish. The age of a silverfish increases in the same rate as aforementioned iter.	Integer
popLimit	An integer that works as an upper limit for the total amount of silverfishes. Used to avoid lists with such sizes that makes simulation computationally infeasible.	Integer
outsideLimit	An integer that sets a limit to how many silverfishes that can come outside of the hide-out.	Integer
population	A linked list that contains all of the silverfishes currently in the hide-out.	Linked List
outside	A linked list that contains all of the silverfishes currently outside.	Linked List
brando	A generator that is used to generate random integers and decimal numbers.	Random

Table 3.1: A table showing the variables used in the simulation, their description and what they are.

In order for the simulation to give just results, the aforementioned variables must be chosen and used in such a way that the silverfishes with a low CL have similar chances of survival as the silverfishes with a high CL.

3.4.2 Birth Limits, Probabilities and Variations

There is a limit of one thousand (1 000) silverfishes in the population. When this limit is reached all silverfishes become sterile until enough silverfishes are either

stomped to death or dead because of old age. When the population becomes less than the population limit, silverfishes are again able to give birth to infants.

A silverfish returning to its hide-out after successfully fleeing will be partially shunned by its community. Silverfish females have a desire to carry the babies of the silverfishes who managed to come back to the hide-out after successfully feigning death. Because of this, the silverfishes who fled will only have a 30 % chance to meet with a woman and give birth to a child. The child will have the CL of the silverfish who fled to the hide-out added with a variation that ranges from -0.05 to 0.05.

As stated previously, female silverfishes have a desire to carry the child of a brave silverfish. Because of this, the silverfishes who return to the hide-out after feigning death will be guaranteed twin silverfishes. These silverfishes will have the CL of the silverfish who feigned death added with a variation that ranges from -0.06 to 0.04.

The random variation in this section is the implementation of the random variation mentioned in the problem statement.

3.4.3 Scenarios

The results that will be presented and discussed are the outcomes from three different simulations where the difference lies in the CL-distribution of the starting population. These scenarios are:

- A starting population where every inhabitant starts with a random CL in the interval [0.0 1.0].
- A starting population where every inhabitant starts with CL = 1.0, which can be interpreted as a population comprised of cowards.
- A starting population where every inhabitant starts with CL = 0.0, the population comprised of brave silverfishes who are willing to sacrifice themselves for the good of the hive and stare death in the eye, hopefully living to die another day.

For the coming simulations, aforementioned variables have been given certain values. The values are illustrated in table 3.2.

Variable	Value
possiblePos	100
startPop	100
iter	300
ageLimit	2
popLimit	1000
outsideLimit	45 % of population

Table 3.2: A table showing the values used for the simulations.

Chapter 4

Results

This section reveals the results of the aforementioned scenarios and explains how to interpret the diagrams.

It was chosen to present the data in histograms because it was felt that histograms would make it easier for the reader to find an answer for the problem statement; what kind of silverfish will prevail?

4.1 Scenario 1 - Population Starting With Random CL

The first histogram is depicted on the next page. This histogram is a result of silverfishes breeding for 300 generations when the inhabitants of the starting population are given random CLs.

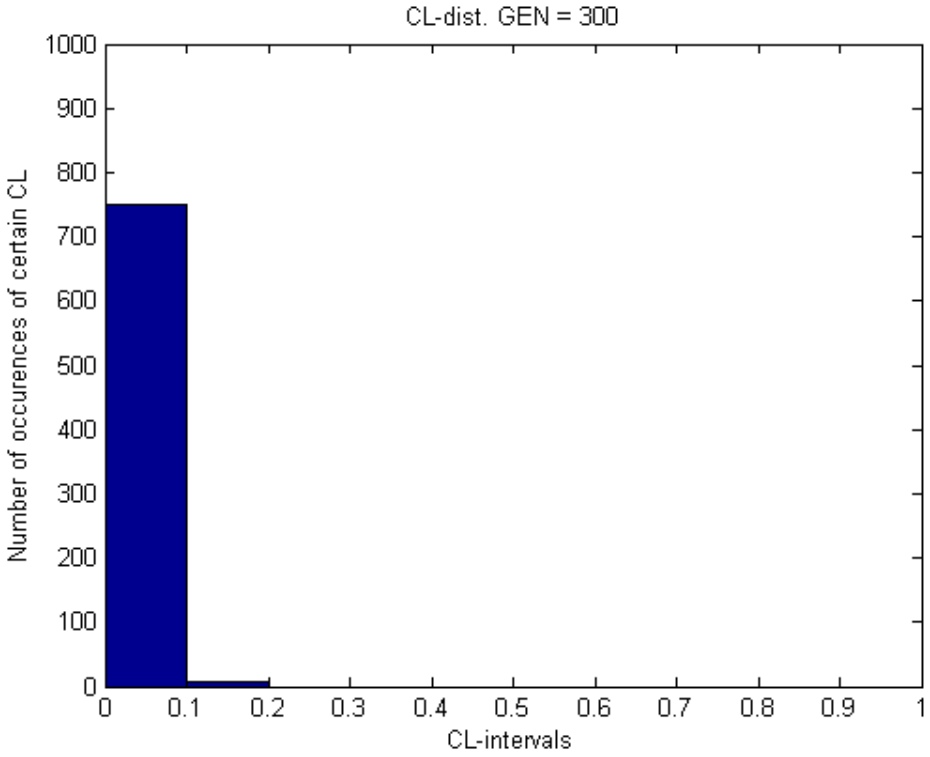

Figure 4.1: A histogram depicting CL-distribution after 300 generations.

The large bar between 0 and 0.1 are the number of occurrences where a silverfish with a CL in said interval exists in the population. The histogram above depicts a population where silverfishes with a CL larger than 0.1 have been almost completely extinct.

Figure 4.1 does answer the question to some extent, but it might also be relevant to know how the starting population looked like in terms of CL. The starting population's CL-distribution and how the population's CL-distribution evolves is shown in figure 4.2.

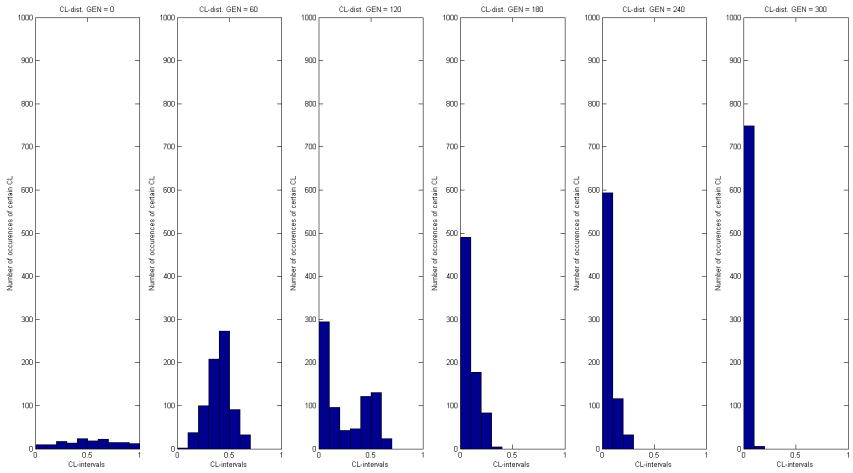

Figure 4.2: The evolution of CL-distribution every 60th generation.

The diagrams are snapshots of the CL-distribution for every 60th generation. The starting generation's CL-distribution is seen in the first histogram and the final generation's CL-distribution is seen to the far right. The histograms clearly show that the largest group of silverfishes are those with a CL in the interval [0.0 0.1] at the end of the simulation.

4.2 Scenario 2 - Population Starting With CL = 1.0

The result from the second scenario can be seen in figure 4.3. In this case, the inhabitants started with CL = 1.0.

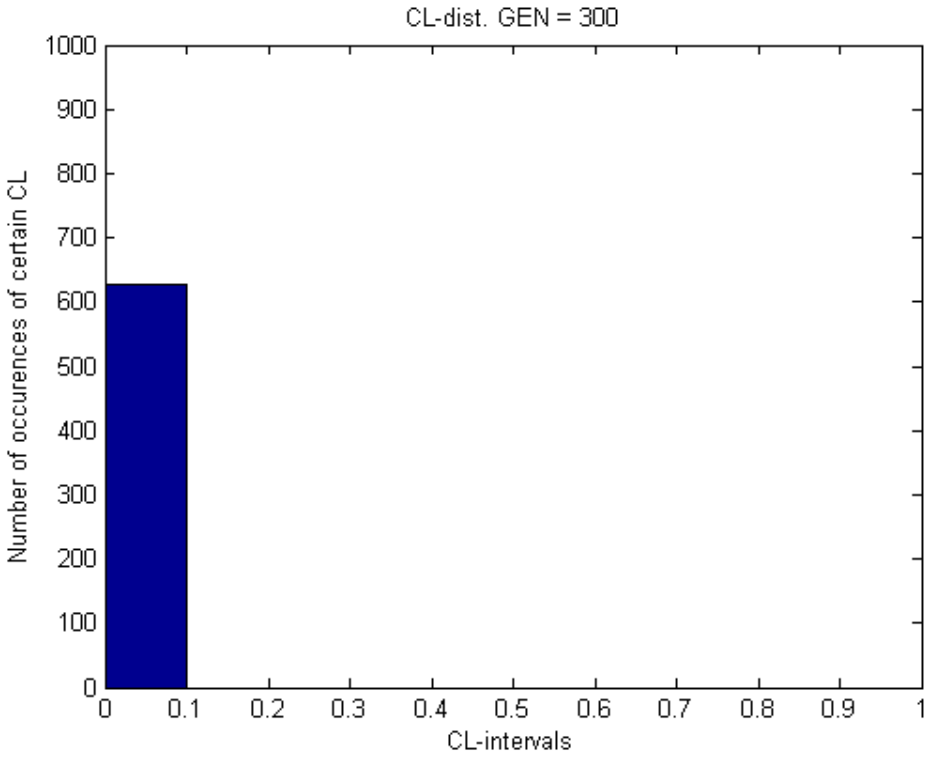

Figure 4.3: A histogram depicting CL-distribution after 300 generations.

Figure 4.3 reminds of the histogram from the first scenario. Even in this scenario it is relevant to show how CL-distribution evolves by showing histogram snapshots. This has been illustrated in figure 4.4.

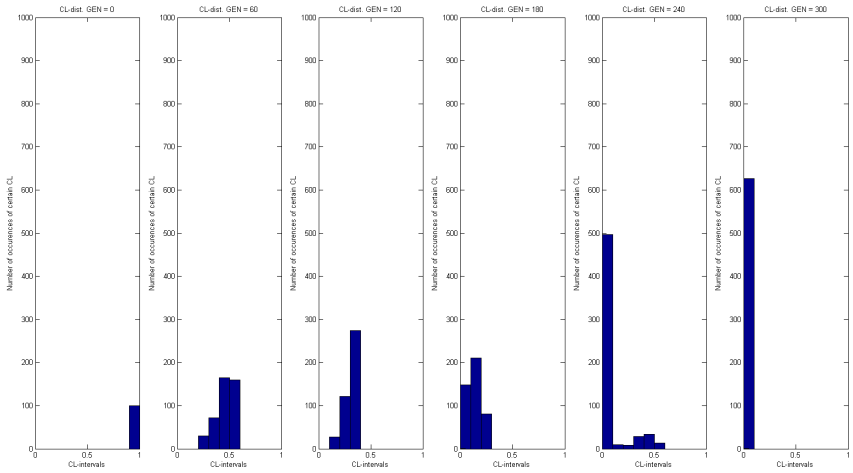

Figure 4.4: The evolution of CL-distribution every 60th generation.

It is possible to see that the population's CL-distribution converges to the interval [0.0 0.1], suggesting that the silverfishes who feign death the most are the fittest silverfishes in the hive.

4.3 Scenario 3 - Population Starting With CL = 0.0

On the next page is the histogram for the population starting with CL = 0.0. The only silverfishes alive are the ones with a CL in the interval [0.0 0.1].

The snapshots have been excluded from this scenario, because it is not as interesting when the starting population begins with CL = 0.0. This is due to the fact that the populations CL-distribution remains roughly unchanged for the entire course of the simulation, making the snapshots visually unpleasing.

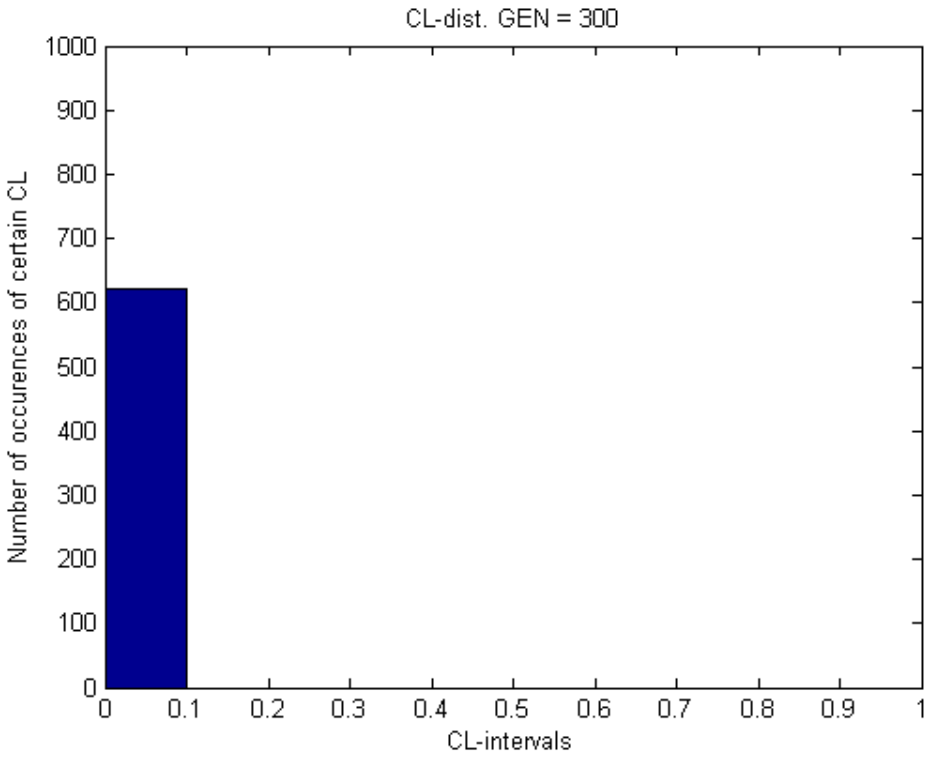

Figure 4.5: A histogram depicting CL-distribution after 300 generations.

Chapter 5

Discussion

This section aims to discuss the results presented in previous section.

5.1 CL-distribution

As the results show, the CL-distribution after several generations will be heavily focused in the interval [0.0 0.1]. The CL of the first generation of silverfishes will only moderately influence this outcome - as in the case of Scenario 1, where a tiny fraction of silverfishes remained in the CL-interval [0.1 0.2] at the end of the simulation.

5.2 CL-variations

It can be argued that the variation of CL displays a downward trend when silverfishes feign death, since the absolute value of the lower limit is larger than the absolute value of the upper limit. However, as stated before, silverfish females will prefer the braver silverfish males. As they grow up, these silverfishes will try to imitate the behaviour of their parents.

The variation for the silverfishes who flee is in the interval [-0.05 0.05]. This can be interpreted as fleeing silverfishes having a less dominant behaviour than the silverfishes who feign death. The children of fleeing silverfishes will not be particularly inspired to behave in a certain way as they grow up. This interpretation could serve as an explanation to why the silverfishes with particularly high CL perished.

While it is certain that a silverfish conceives two babies if it survives after feigning death, it is probably not ideal to have a CL of precisely 0.0. This is because there is always a 50 % chance of the silverfish surviving if it feigns death. The best CL is, most likely, slightly above 0.0 since the probability of survival is greater than 50 % if the silverfish flees when the distance to the hideout is less than half of the maximum distance.

Note that the different probabilities for conceiving babies depending if the silverfish flees or feigns death are there to make one type of behaviour highly favourable for the survival of the species. The authors of this essay are not suggesting that one should draw any conclusions of favourable behaviour for humans from this. The authors do not despise humans who act cowardly in certain situations.

5.3 Conclusion

The problem statement asked if it is possible to create a simulation that gives an optimal solution for the silverfish species. Given the results, which has been illustrated in the essay, the conclusion is that the silverfish with a CL in the interval [0.0 0.1] is the fittest silverfish after a long period of time. Therefore, the simulation was able to grant an optimal solution to the problem.

5.4 Further Development

Since this is a very limited project, there are several ways to improve this simulation. One idea is to use time-based simulation instead of event-based simulation. The advantages to this are that factors which are dependant on time are easily implemented. For example, the speed of the silverfishes can be implemented and altered, and the duration of an activated lamp can finally be implemnted in this simulation.

In this implementation, the bathroom is one-dimensional. A way to further develop this would be to increase the number of dimensions by adding a Y-axis as well. The distance from the hideout could then be calculated using Pythagora's theorem if it is assumed that the hideout is in a corner. If this is carried through, it might be interesting to present the positions of silverfishes graphically a few times during the span of the simulation.

The only "gene" that was spoken of in this essay was the Cowardliness Level (CL) of a silverfish. The CL is simply the authors' assumption to make the simulation simple. One could implement more complicated genes that control more than just the will to run or feign death.

Bibliography

[1] Eric Day, *Silverfish factsheet*, Webpage from the Internet, created August 1996 [obtained 10th April 2012], http://www.sites.ext.vt.edu/departments/entomology/factsheets/silverfi.html

[2] Charles Darwin, *On the Origin of Species*, [E-book]. Sydney, NSW: University of New South Wales, created 2009 [obtained 14th May 2012], http://embryology.med.unsw.edu.au/pdf/Origin_of_Species.pdf

[3] Bengt Sandblad, *Kort om simulering och simulatorer*, Lecture notes, Uppsala Universitet, created spring 2003 [obtained 11th April 2012], http://www.it.uu.se/edu/course/homepage/opgui/vt03/Simulatordokumentation