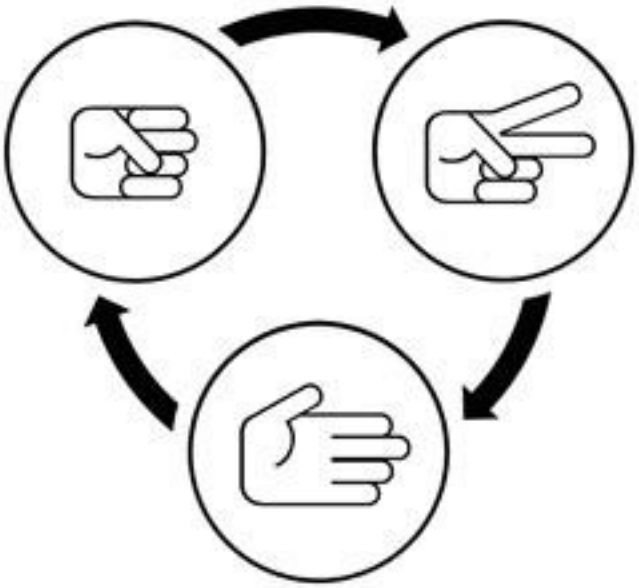
Rock Paper Scissors



Bachelorscriptie CKI

7.5 ECTS

Student: Koen Klinkers

3213129

Supervisor: Jan Broersen

April 2011

Universiteit Utrecht

Table of Contents

1 Introduction

1.1 Inspiration

The inspiration for the thesis came from online multi-player video games. I noticed that the strategies used in these games evolved over time. Certain strategies would become popular, but over time these strategies would be abandoned in favour of other strategies. The strategies were getting more advanced and better, but there was also another factor determining which strategies would become popular. Namely, strategies would become popular because they were really effective against other strategies that were used a lot. As a consequence the strategy that was countered would get less popular and another strategy that was effective against the new popular strategy would emerge. This pattern keeps repeating itself. A strategy becomes popular, as a consequence its counter-strategy becomes popular. Next, the counter of the counter becomes popular etc.

1.2 Rock-Paper-Scissors

The above described dynamics are similar to the children's game Rock-Paper-Scissors (RPS). The game is used as a decision method like flipping a coin. The game is played by two players. The players simultaneously choose one of the three strategies: Rock, Paper or Scissors. This is done by 'throwing a hand'. A closed fist represents Rock, a hand with all fingers extended against each other represents Paper, and a fist with only the index finger and middle finger extended represents Scissors. The rules of the games are: Rock blunts Scissors, Scissors cuts Paper and Paper wraps Rock. The interesting part of this game is that all strategies are equal. All strategies beat one strategy and lose from another strategy. Therefore, none of the three strategies should become more popular then the other strategies. But the odd thing is that in certain models which use the RPS dynamics, some strategies do become more popular than others. Similar to the pattern in video games. The above described video games use RPS dynamics in their game mechanics. This is done because it ensures balanced game play. Although the games are completely balanced, the dominant strategies change over time.

1.3 Goal of the Thesis

The goal of this thesis to develop some theory that describes the patterns that emerge in RPS-like models. I will design a model in NetLogo[1] in which the changing strategy dynamics and other patterns are reproduced. First I will use RPS as base for my model. Later I will try different variations of the RPS game and implement them in the model and try to explain their behaviour.

1.4 Relevance with Artificial Intelligence(AI)

The thesis is about game theory. The relevance between game theory and AI is clear. Game theory can help understand how AI should behave and reason faced with certain problems faced in real life. In this thesis I will also try to give some biological examples. This shows that the study of artificial intelligence is not only relevant for understanding our brains, but can also help understanding complex system found all around us, for example ecosystems. This part of Artificial Intelligence will be relevant for this thesis.

1.5 Structure of the Thesis

First I will explain my RPS model. I will describe the mating strategy of a particular lizard. The dynamics of the mating strategy were the basis of the RPS model I implemented. Than I will explain the model itself and its implementation in Netlogo. Than I will produce the results of the RPS model and give an explanation of the emerging dynamics. I will try to create a theoretical framework in which my model can be placed. This thesis belongs to the study of complex systems, a young research area which lacks any form of a grand overarching theory. My model can be seen as an extension of minority games. These kind of games were generalised in a framework called Crowd-Anticrowd Theory[2][3].I will show how my model can be an addition of the Crowd-Anticrowd Theory. Next I will describe some variations of RPS model. First the RPS-5 model which is an extension of the RPS model with five strategies instead of three. I will again explain the emerging dynamics using the results. The RPS model can be extended by any odd numbered amount of strategies. Therefore I will formalise the RPS model for *n* strategies. Then I will explain the RPSW model. This model is, unlike the previous models, unbalanced. I will again explain the emerging dynamics using the results. Finally I will try another model I created randomly. The goal of this variation will be to test the explaining ability of my findings in previous variations. In the conclusion I will reflect on my findings and discuss further research possibilities.

---

1 Netlogo is an environment for programming multi-agent models. http://ccl.northwestern.edu/netlogo/

2 Neil F johnsons and Micheal L. Hart,(2003) Crowd-Anticrowd Theory of Multi-Agent Minority Games, arXiv:cond-mat/0212088v2

3 Neil F.Johnsons and Pak Ming Hui (2003) Crow-Anticrowd Theory of Collective Dynamics in Competitive Multi-Agents Populations and Networks, arXiv:cond-mat/0306516v1

2 RPS-model

I based my model on the children's game Rock Paper Scissors. The reason for this is that RPS is a perfectly balanced game. In RPS there are three different strategies. All three strategies are equally effective. So if in the simulation one strategy would become more popular then the other strategies, there must be a different reason other than that strategy is better than the other strategies. This could explain the effect that one strategy is more popular, although it is not necessarily better.

2.1 RPS in Game Theory

RPS is game played with two players. Each player chooses one of the three strategies, Rock, Paper or Scissors. Rock beats Scissors, Scissor beats Paper and Paper beats Scissors. The goal of the game is to beat the opponents strategy. If both players choose the same strategy the game is tied. In game theory RPS is classified as a zero-sum game. A zero sum game means that the amount that one player wins is the same as the other player loses. This is the pay-off matrix of RPS. 1 means a win, -1 a loss and 0 a tie.

	Opponent Rock	Opponent Paper	Opponent Scissors
Rock	0	-1	1
Paper	1	0	-1
Scissors	-1	1	0

Table 1: Pay-off Matrix standard RPS

$$\text{Expected outcome} = \frac{1}{3}(-1) + \frac{1}{3}(0) + \frac{1}{3}(1) = 0$$

As you can see all three strategies have the same expected outcome. So if you would play this game repeatedly the best strategy is to play random, because this way your opponent can not predict your strategy.

You might wonder how in a game where the best strategy is to play random, one strategy may become dominant. In real life there are examples of RPS-like dynamics and they also produce patterns in which some strategies become more dominant than others.

2.2 Lizards

What do lizards an RPS have in common? In the inner Coast Range of California there lives the side-blotched lizard (*Uta stansbruriana).* The mating strategies of the males of these lizards follow the same rules as RPS[4]. There are three kinds of males, distinguished by the colour of their throats. There are orange, blue and yellow coloured males and each one represents a different kind of strategy. Orange males are large strong males who guard a large area with a lot of female lizards. The blue coloured male is weaker then the orange coloured male and guards a smaller area with fewer females. The yellow male doesn't guard an area ,but its appearance mimics females. It mates by sneaking into the region of another male and mating with his females without the other noticing him. In this 'game' orange beats blue, blue beats yellow and yellow beats orange. Orange is much stronger than blue and chases al competing blue males away. Blue males have a better bond with their females, because they have fewer females in their area. So they recognise the 'sneaker' yellow male and are able to chase him away. Finally the orange has such a large area to cover with a lot of females, the yellow male is able to sneak inside the area of the orange male and mate with his females without him being noticed.

These lizards also show the phenomenon of the changing strategy dynamic. The offspring of a male has the same strategy as their father. In a given season the orange males are quite successful and they produce a lot of offspring. So the next mating season there are a lot of orange males. Because their fathers strategy was successful it means that the strategy of their offspring won't be. This season the yellow males will be able to exploit the orange males and mate with lots of females. The following mating season there will be a lot of yellow males. In this season the strategy of the blue males will the most effective and will produce the most offspring. This cycle will keep repeating itself. In a particular season one strategy will be most prominent, but in the long run over many seasons all strategies are just as common.

I have described lizards cyclical pattern in a simplified way. In the real word it's a bit more complicated. One reason is that there are males who are hybrids between two colours. Another reason is that the physical world has an influence on the balance between the different strategies. For example the same region could accommodate much more blue males than orange males because blue male require a much smaller region. Sinervo & Lively have taken al these aspects into account and have shown that the dominant strategies fluctuates over time, but in the long run the different strategies are in equilibrium.

4 The rock-paper-scissors game and the evolution of alternative male strategies, B Sinervo & C.M. Lively, Nature vol 380, 21 March 1996

2.3 Description RPS-model

My model is inspired by the RPS dynamics from lizards described in the previous section. In my model a group of agents play the game of RPS against each other in an environment. After a round has been played, agents choose their new strategies accordingly to how the strategies performed in previous round. This way, the model mimics the changing strategy dynamics. Like the lizards, winning strategies produce a lot of offspring, therefore the strategy will be more abundant in the next season. Evenso in the video games, the tendency of people to adopt strategies that seem the most beneficial.

2.4 Implementation RPS-model

In this model a population of a 100 agents play RPS in an environment. The strategy each agent uses is decided before the game starts. So an agent will use the same strategy during the whole round. An agent wanders through the environment and when it meets another agent they play RPS. When an agent wins a game, it will update a global variable that contains the numbers of wins of the particular strategy for that round. After a certain amount of steps the round is over. I have chosen a hundred steps because, if a round doesn't have enough rounds the agents hadn't had a proper chance to fight and the results will be too random. Letting a round go on beyond a hundred steps is useless and just slows down the program.

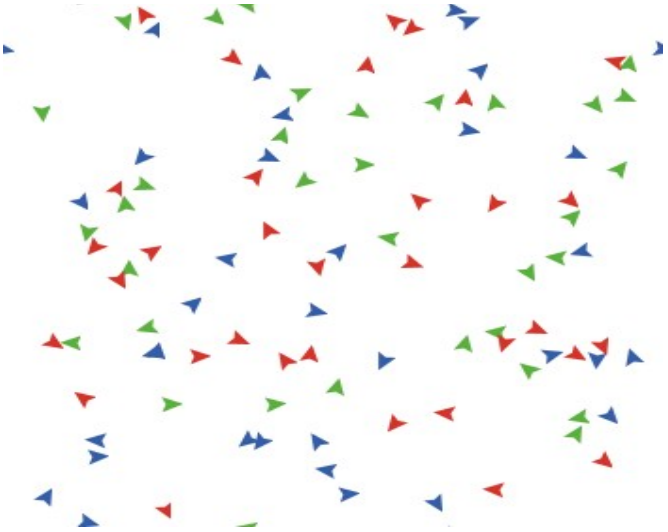


Illustration 1: Agents during a simulation

After a round is finished all old agents are deleted. Next, a new batch of hundred agents is created. Their strategies are based on the amount of wins of a particular strategy in the previous round using this formula.

$$(amount\ of\ agents\ using\ strategy\ x) = \frac{(amount\ of\ wins\ of\ stratiegy\ x\ during\ previous\ round) + 1}{(total\ amount\ of\ wins\ during\ previous\ round)} * 100$$

As you can see the percentage of a certain strategy in the new round is equal to the percentage of wins in the previous round. This also shows that all strategies will get one free win. This is done to prevent strategies to get extinct. Because of the dynamics of RPS, if one strategy dies out, all will follow and the simulations will end. In certain sense this is cheating, but I decided to do it this way. Because it is quite likely for a strategy to not win for a whole round. This is because a strategy can dominate quite severely. It's not unlikely for a certain strategy to dominate a population by more than 95 percent. As a consequence there aren't many wins, because most agents are using the same strategy. A solution would have been to make the amount of agents much larger and have the round take longer. This would be to increase the chance of a strategy to score points, but it would have made the program very slow and also technically wouldn't prevent the possibility extinction.. Extinction in models is always an interesting phenomenon, but in this case it is unwanted. If a strategy gets extinct it could never return and this would prevent the cyclical pattern I want to simulate.

After a round all strategies are counted and displayed in a plot. The percentage of a certain strategy in the population of agents is used to determine its popularity. In the beginning of a new round all scores are reset and the simulation starts over again.

2.5 Results RPS-Model

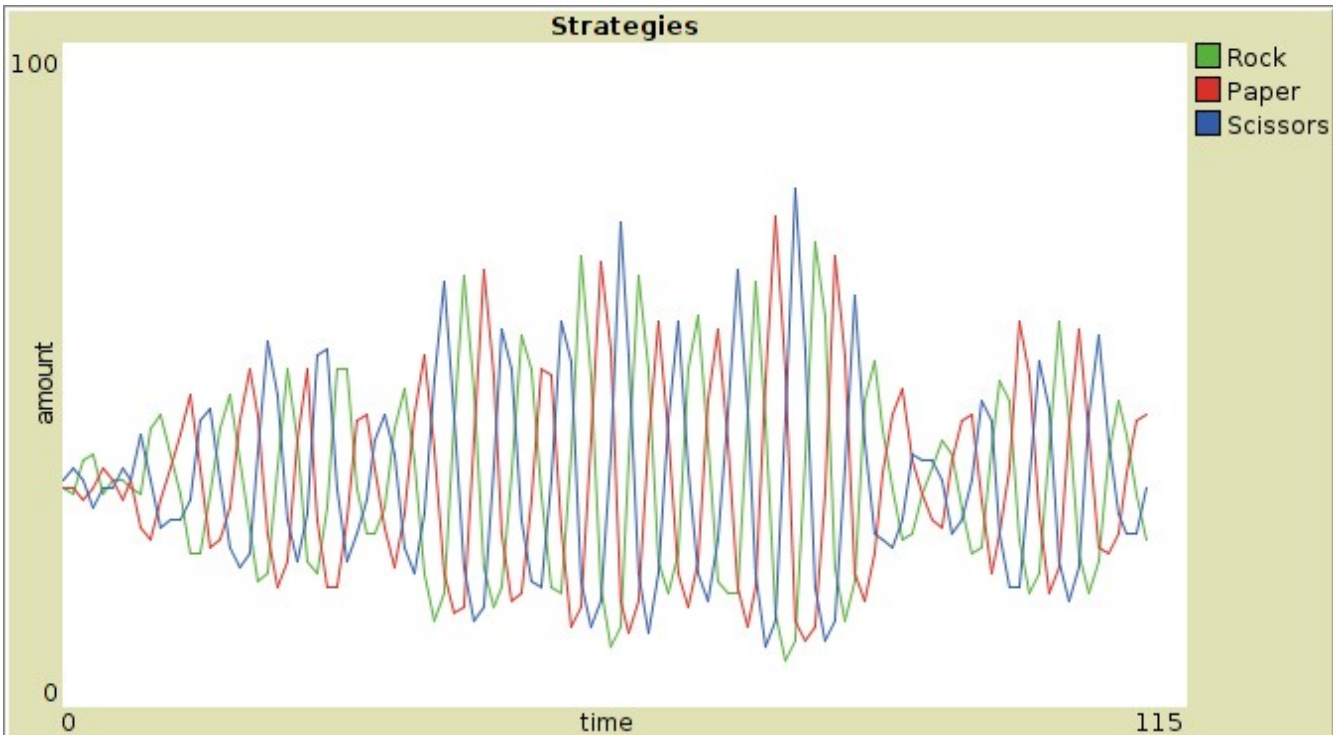


Illustration 2: Results after about a hundred rounds

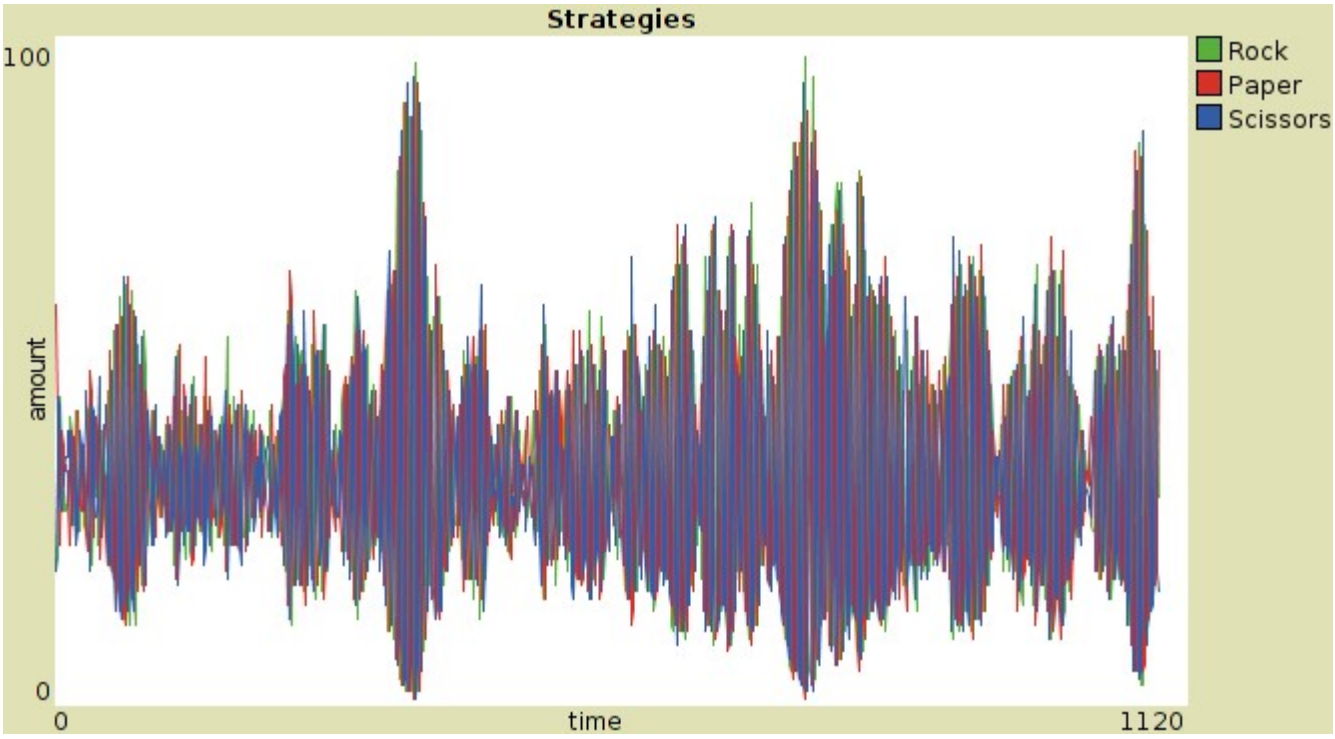


Illustration 3: Results after about a 1000 rounds

2.6 Explaining the Results

2.6.1 Cyclical Pattern

As you can see the simulation follows a cyclical pattern. The most popular strategies always occur in the same order. Rock is followed by Paper, Paper is followed by Scissors, Scissors is followed by Rock and then the pattern repeats. This behaviour is easily explained. For example, take a population were there are a lot of Rocks and a few Papers and Scissors. The only one who can exploit this competition is Paper. Rocks need Scissors to score points, but there are very few Scissors so it's difficult to score for Rocks. The same yields for Scissors. Scissors needs Papers to score points, but there are very few Papers so it's difficult to score for Scissors. The only strategy that scores is Paper. Paper needs Rocks to score and there are a lot of Rocks. So in this round Paper will score a lot of points in comparison to the other strategies and therefore in the next round Paper will be the most popular strategy.

This pattern will always occur, even when the popularity of strategies are equal. When all strategies are equal, all strategies can exploit the competition equally well. But due to the randomness of the game some strategies will score better then others by chance. These differences will be exaggerated when the amount of rounds grows until one strategy becomes dominant and the above described pattern emerges. Because of the self-enforcing nature of the pattern, once the pattern starts it persists. Illustration 2 shows a session were all strategies were equal in the beginning and as you can see the cyclical pattern emerges.

2.6.2 Bloating and Shrinking Pattern

In the long run there is a bloating and shrinking pattern as shown in illustration 3. When differences in percentages between strategies are small, the differences tend to bloat. When the differences are large they tend to shrink. The differences getting larger is explained by the small difference getting exaggerated over time and making the differences in percentage bigger. But how do the differences get smaller again? When the difference between the strategies becomes extreme, the amount wins per strategy converges. The strategy which can exploit the environment may have so few agents that its score is comparatively low. The strategy which is dominant has a comparatively high score. There aren't a lot of agents which are beaten by the dominant strategy, but the dominant strategy has a large working force which ensures it gets a reasonable amount of wins. The amount of wins in these rounds are also low, because most fights are between agents of the dominant strategy which end in a tie. The consequence is that the free win every strategy gets has a comparatively higher impact. These two factors will make the differences between the strategies shrink.

3 Theoretical Framework

3.1 RPS-like models

First to put RPS in a theoretical framework, I will have to formalise the model. This formalisation will also includes the variations I will discuss in chapter four of this thesis. Therefore the use of the name RPS-like models.

A RPS-like model contains a group of strategies. The relationship between all strategies are defined. The relationship between to strategies defines which strategy is beaten by which strategy. A strategy ties against itself. In RPS-like models a strategy is represented by a group of agents. The different groups of agents compete against each other during one round. This is done by letting the agents randomly play against another agent. This can be against any agent from any group. In my RPS model this randomness is achieved by letting all of the agents wander in an environment and letting them fight when they meet. The size of the different groups representing a strategy is determined by how well the different strategy performed in the previous round. The more wins a strategy achieved in the previous round, the larger the group of agents. In my RPS model the size is determined by using the formula stated in section 2.4. The size of the groups in the first round is arbitrary.

3.2 Minority Games.

The model I have created has a lot in common with minority games. In a minority game agents have to choose between two strategies. The goal of the game is to choose the least popular strategy. The dynamics of minority gains are best explained by the canonical El Farol problem. The El Farol Problem is created by Brian Arthur[5]. El Farol is a bar in Santa Fé, New Mexico. Agents can choose between two strategies. Going to the bar or staying at home. The problem is that if to many people choose to go to the bar the bar becomes to crowded and the agents would have been better of staying at home. So there are two options and the goal is to choose the least popular options. If it's popular to stay at home, you should go to the bar and have a good time. If it's popular to go to the bar, it's better to stay at home because you would have had better time staying at home then when you would have gone to an overcrowded bar. At the same time, agents don't have any information of the choices of the other agents. They can base there choice only on the crowdedness of previous visits to the bar.

5 W.B. Arthur. (1994) Inductive reasoning and bounded rationality; the El Farol problem. American Economics Association Papers and Proceedings 84, 406 411.

3.3 Crowd-Anticrowd Theory

These kinds of games were formalised mainly by Neil F. Johnson[67]. It is called Crowd-Anticrowd Theory. Crowd-Anticrowd Theory is a framework for describing Minority Game-like models. But Crowd-Anticrowd Theory is also applicable for kinds of complex models, as stated by Johnson himself. I will now give an informal description of Crowd-Anticrowd Theory.

In this framework a group of agents has to choose between two options every round. These options are represented by 1 and -1. After the of the agents have made their choice between the two possibilities, the crowd and anti-crowd are formed. The crowd is defined as the group of agents that has chosen the most popular possibility of the two. The anti-crowd is defined as the group of agents that has chosen the least popular possibility of the two.

The choice a particular agent makes, is decided using the B-A-R (Binary Agents Resource) System. The B-A-R system is based upon the works of Challet and Zang[8]. In this system the choice of an agent is determined by the outcome of previous rounds. How many rounds the agents will look back, can be adjusted in the the history space. Meaning an agents with a history space of two will take the results of the two previous outcomes into account with deciding the option for the next round. The strategy of a particular agent is represented in a list of all possible histories with their corresponding choices for the next round. This strategy is represented using a binary string.

The benefits of this framework is the volatility, meaning the likeliness of agents to switch between options, can easily be measured. Using a binary string to represent a strategy allows for an easy and clear way to compare different strategies. A binary string also allows for easy implementation of evolutionary algorithms for developing strategies.

6 Neil F johnsons and Micheal L. Hart,(2003) Crowd-Anticrowd Theory of Multi-Agent Minority Games, arXiv:cond-mat/0212088v2

7 Neil F.Johnsons and Pak Ming Hui (2003) Crow-Anticrowd Theory of Collective Dynamics in Competitive Multi-Agents Populations and Networks, arXiv:cond-mat/0306516v1

8 D. Challet and Y.C. Zhang ., Physica A 256, 514 (1998)

3.4 Relevance for RPS-like models

Also important to address, is that the strategy in the Crowd-Anticrowd Theory is something different than the strategy in the RPS-like models. In Crowd-Anticrowd Theory, the strategy is the binary code of an agent that decides which option it is to choose next. In the RPS-like model, the strategy is one of the options agents have choose. So the 'options' in Crowd-Anticrowd Theory are comparable to the strategies in RPS-like models.

At first glance these two models might seem irrelevant to each other. The 'strategies' work on different levels. In Crowd-Anticrowd Theory the strategies are concern with which option to choose. In the RPS-like models this choice is arbitrary. Options, called strategies, are directly chosen according to their performance in the previous round. The goal of the RPS-model is to understand how these strategies compete in an environment. In the Crowd-Anticrowd Theory this is irrelevant because both options are equal in each and every way. What both kind of strategies do have in common is that the effectiveness of a particular strategy is directly affected by the presence of other strategies. In Crowd-Anticrowd Theory the goal is to choose different from the crowd, therefore the effectiveness of a strategy is inherently determined by the other strategy present in the environment. This also yields for RPS-like models.

Ideally, in the great scheme of understanding complex systems, both models will need to be merged. The strategies for predicting the choice of other agents in minority games will need to be combined with RPS-like models, where there are more than two options. It is therefore important to understand both aspects before they can be combined and understood. Crowd-Anticrowd Theory is concerned with how agents make the decisions based on a certain history space in Minority Games. Just like in Minority Games, the best option in RPS-like models is based on the strategies of the other agents. But in RPS-like models the options are more complex. The are more options and options also affect each other. Ff Crowd-Anticrowd Theory and RPS-like models would be combined, this complexity will affect the strategies of the agents when choosing their option,

4 Variations of RPS-like models

In this section I will explore some different RPS-like models. All models can be seen as variations of the RPS model. The difference is in the amount of strategies and the relationships between these strategies.

All these variations use the exact same model and implementation as RPS does. The only thing that is different, is the amount of strategies and their relationships .

4.1 RPS-5

The first variation I tried is a variation of RPS which is called Rock-Paper-Scissors-Lizard-Spock. RPSLS has two extra strategies. Lizard which hand sign is holding the thumb and index finger against each other. Spock is based on the Star Trek character and it's corresponding hand sign is the famous Vulcan greeting gesture. The rule list of the game is

- Rock blunts Scissors
- Rock crushes Lizard
- Scissors cuts Paper
- Scissors decapitates Lizard
- Paper wraps Rock
- Paper disproves Spock
- Lizard eats Paper
- Lizard poisons Spock
- Spock smashes Scissors
- Spock vaporises Rock

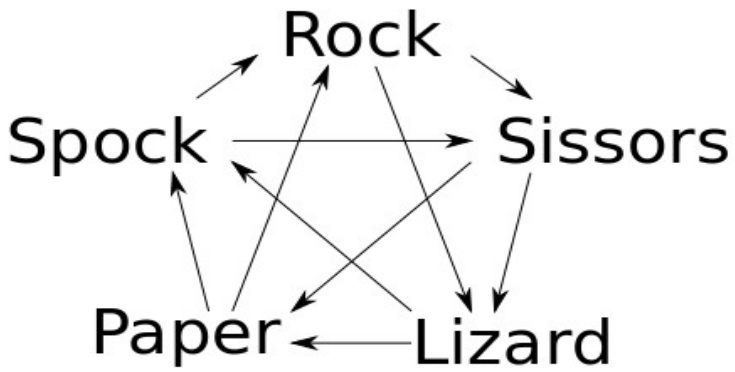


Illustration 4: RPSLS game in diagram

4.1.1 Description

The RPSLS game is just an extension of RPS with five strategies, therefore the name RPS-5. As you can see in the pay-off, the game is balanced. Each expected strategy has the same expected outcome. Therefore all strategies are equally good. Just like in standard RPS.

	Rock	Paper	Scissors	Lizard	Spock
Rock	0	-1	1	1	-1
Paper	1	0	-1	-1	1
Scissors	-1	1	0	1	-1
Lizard	-1	1	-1	0	1
Spock	1	-1	1	-1	0

Table 2: Pay-off matrix of RPSLS game

$$\text{Expected outcome} = \frac{1}{5}(-1) + \frac{1}{5}(-1) + \frac{1}{5}(0) + \frac{1}{5}(1)\frac{1}{5}(1) = 0$$

The difference between previous games is that a certain strategy doesn't have a monopoly any more on exploiting a certain group of agents. For example, previously in standard RPS, Rock was the only strategy that could exploit a competition with a high percentage of Scissors. But in the RPSLS Rock competes with Spock in exploiting a Scissors dominant competition.

4.1.2 Results of simulation

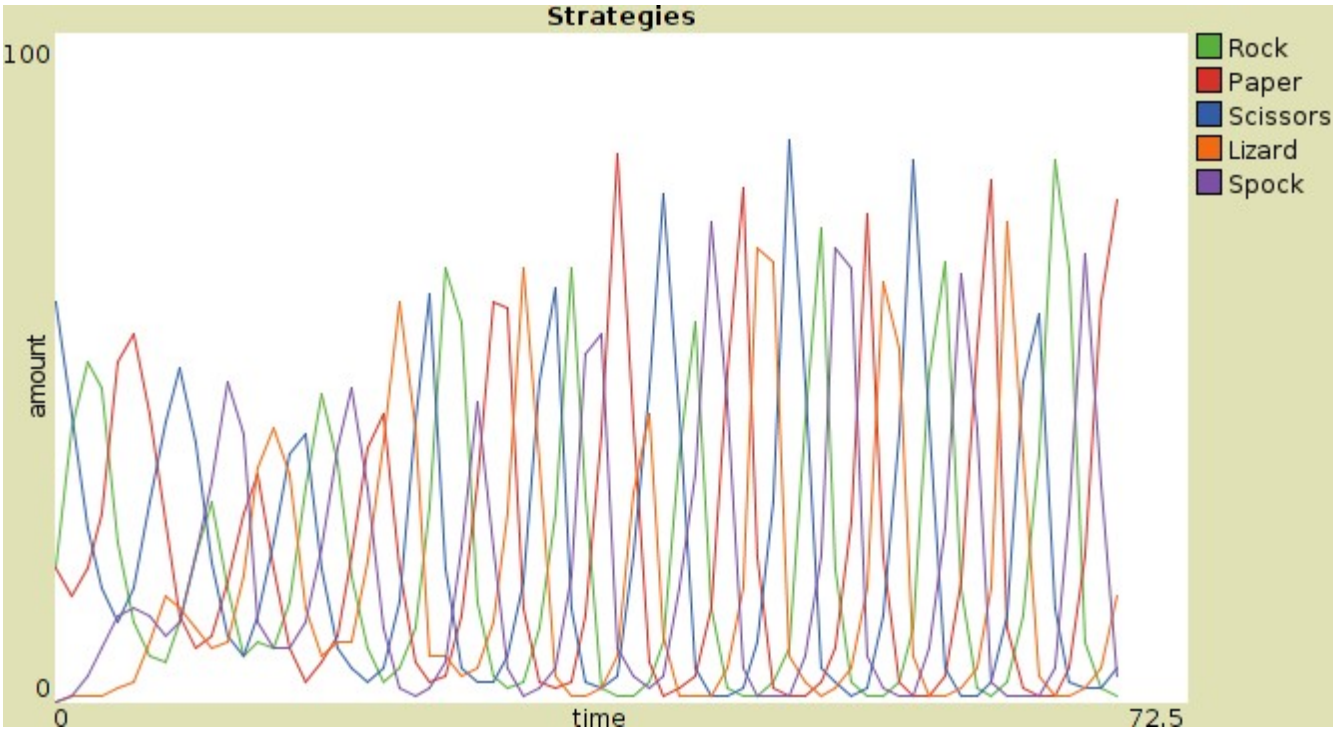


Illustration 5: Results of RPSLS or RPS-5

4.1.3 Explanation Results

As you can see a pattern arises after the model stabilizes. The strategies always dominate the environment in the same order. Namely: Paper, Lizard, Scissors, Rock, Spock. To explain this patterns lets first answer the question: what happens when two strategies can both exploit a group of agents equally well? You would expect that the two strategies would score equal and in the next round would be equally dominant, but as you can see in the results this doesn't happen. Why? Until now when explaining the performance of a certain strategy, we have only taken in consideration the amount of agents of a strategy in the competition which it can beat. The higher the amount of agents, the better the strategy would perform. But now we also have to consider the amount agents of the strategy itself. When a certain strategy has more agents, it has more opportunities to fight, thus also more opportunities to win. For example, take a population of a 100 agents. 70 agents are Rocks, 20 agents are Spocks and 10 agents are Papers. Both Spocks and Papers can exploit the competition but the Spocks are twice as abundant and therefore the Spocks will perform a lot better than the Papers.

But for this to explain the pattern, it has to be the case that always one of the competing strategies is more abundant, or else it wouldn't be the case that one strategy would be dominant. This is the case, because of the way the model is set up. For example, take a competition with a lot of Papers. The competition will be exploited by Scissors and Lizards. In the next round there will be a lot of Lizards and Scissors. This time Scissors can again exploit the competition because there are a lot of Lizards. Also it performs the best of all the strategies because it is the most abundant. As you can see a strategy can always exploit the competition for two rounds in row. In the first round it will be in an disadvantage against its competing strategy because this strategy could also exploit the previous environment. The second round the strategy has an advantage against its competing environment because it is more abundant than the competing strategy. This is because it could exploit the previous competition. A graphical representation of this underlying pattern can be seen in illustration 6.

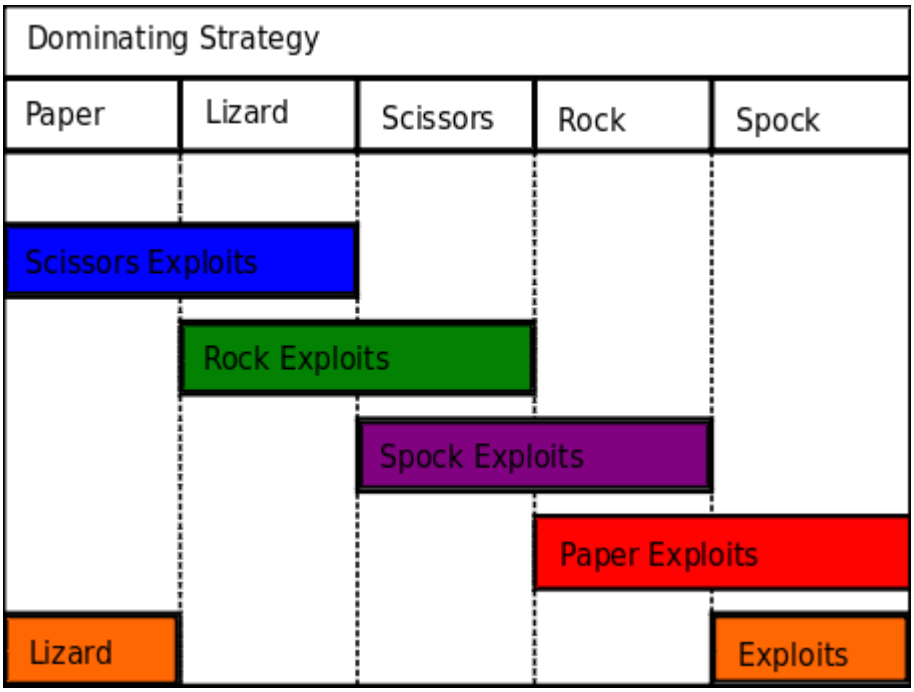


Illustration 6: Graphical Representation of the underlying pattern in RPSLS

If you look at illustration 5, you can see that a strategy sometimes is skipped in the pattern. The reason for this is that a strategy couldn't get a high enough advantage in the first round, so he could not dominate its competing strategy in the second round. The reason for this is chance. The strategy just got unlucky and didn't acquire enough wins. The interesting part is that the pattern isn't hampered in anyway and persists. In the next round the expected strategy dominates. This shows that that there is an underlying pattern and that the dominance of certain strategy is just a consequence of the pattern not the patten itself.

4.2 RPS-n

RPS-5 is an extension of RPS. I could also make an extension with 7 or more strategies. Some enthusiast even made a version with 101 versions[9]. We could generalise RPS for any number of strategies.

9 http://umop.com/rps101/rps101chart.html This chart is huge it's almost impossible to layout properly on a sheet of paper

4.2.1 Definition

This is the formal definition of RPS-n

M_n is defined as a model with n strategies $\quad (n \in \mathbb{N} \mid n > 3, n = odd)$
$\bar{S}_n$ is defined as the set af strategies belonging to model M_n
$\bar{S}_n = (S_1 + S_2 + \ldots + S_n)$

$S_x B\ S_y$ is defined as the relationship between S_x and S_y wich defines that strategy S_x beats strategy S_y.
R_x is the set of relationschips of strategy S_x

$$R_x = \sum_{i=1}^{\frac{1}{2}(n-1)} (S_{(x)} B\ S_{(x+i)\%n})$$

P_n is defined as the pattern that repeats itself in model M_n
$P_n = S_n \rightarrow \cdots \rightarrow S_2 \rightarrow S_1$

D_n is defined as the number of rounds a strategy is exploiting in a row in M_n

$$D_n = \frac{1}{2}(n-1)$$

4.2.2 Balance in RPS-n

Even if a RPS-n model contain hundreds of strategies, there is only one possible balanced distribution of the relationships. This is because for the model to be balanced, all strategies have to be equal. This means that all strategies have lose from 0.5(n-1) strategies and win from 0.5(n-1) strategies.

You can construct a balanced distibution in different ways. For example:

M_3
$\bar{S}_3 = (S_1 + S_2 + S_3)$

$R_1 = (S_1 B\ S_2)$
$R_2 = (S_2 B\ S_3)$
$R_3 = (S_3 B\ S_1)$
or
$R_1 = (S_1 B\ S_3)$
$R_2 = (S_2 B\ S_1)$
$R_3 = (S_3 B\ S_2)$

But these kinds of variations are isomorphisms of each other and I don't consider them different distributions.

4.2.3 Example

An example using n=7

$$M_7$$
$$\bar{S_7} = (S_1 + S_2 + S_3 + S_4 + S_5 + S_6 + S_7)$$

$$R_1 = (S_1 B\ S_2 + S_1 B\ S_3 + S_1 B\ S_4)$$
$$R_2 = (S_2 B\ S_3 + S_2 B\ S_4 + S_2 B\ S_5)$$
$$R_3 = (S_3 B\ S_4 + S_3 B\ S_5 + S_3 B\ S_6)$$
$$R_4 = (S_4 B\ S_5 + S_4 B\ S_6 + S_4 B\ S_7)$$
$$R_5 = (S_5 B\ S_6 + S_5 B\ S_7 + S_5 B\ S_1)$$
$$R_6 = (S_6 B\ S_7 + S_6 B\ S_1 + S_6 B\ S_2)$$
$$R_7 = (S_7 B\ S_1 + S_7 B\ S_2 + S_7 B\ S_3)$$

$$P_7 = S_7 \rightarrow S_6 \rightarrow S_5 \rightarrow S_4 \rightarrow S_3 \rightarrow S_2 \rightarrow S_1$$
$$D_7 = 3$$

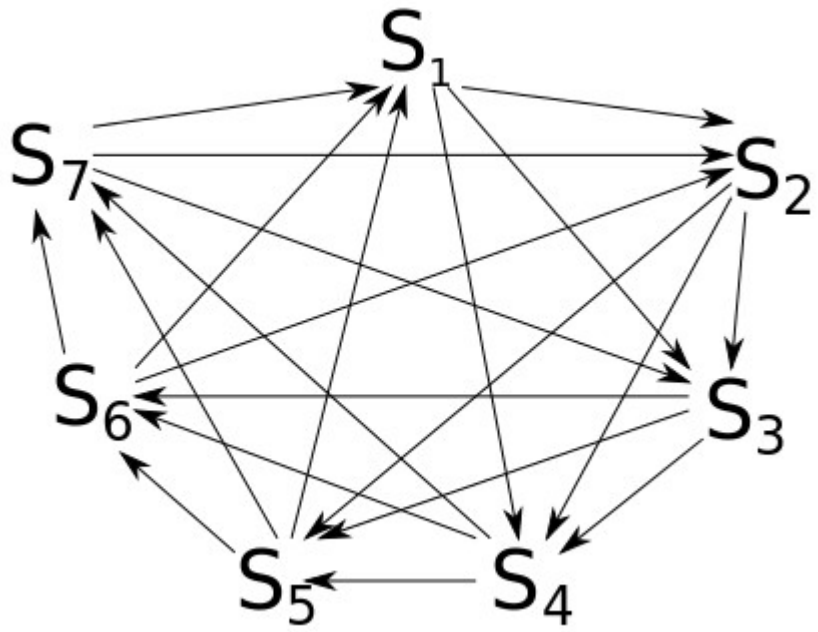


Illustration 7: Diagram of RPS-n game with 7 strategies

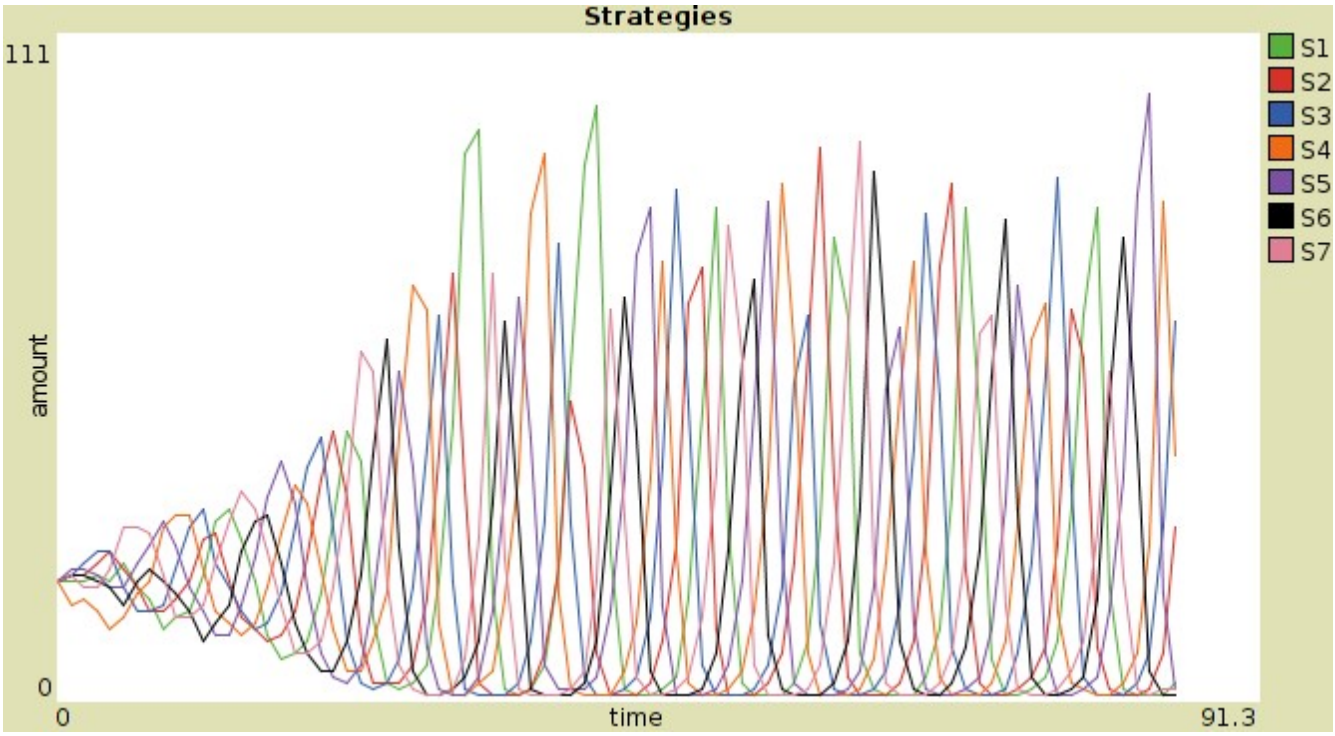


Illustration 8: Results of Simulation of RPS-7

As you can see in the results of the simulation the model clearly has the predicted pattern of:
$P_7 = S_7 \rightarrow S_6 \rightarrow S_5 \rightarrow S_4 \rightarrow S_3 \rightarrow S_2 \rightarrow S_1$

Take note that seven strategies is at the limit of the simulation. Most of the time strategies are unable to score during a round. If a model has more strategies the number of agents and the length of the rounds has to increase and thus also requires more computing power.

4.3 RPSW

The next variations is RPS with the added strategy of Well. The extra rules to this game are:

- Paper covers Well
- Scissors falls in Well
- Rock falls in Well

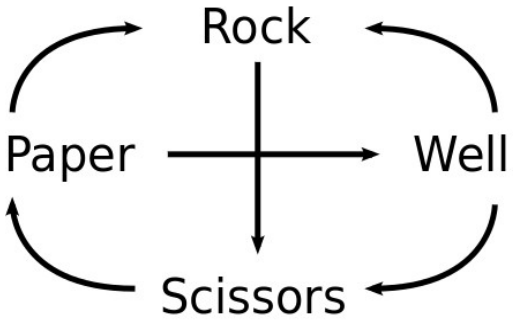


Illustration 9: Diagram of RPSW

4.3.1 Description

	Oppenent Rock	Oppenent Paper	Oppenent Scissors	Oppenent Well
Rock	0	-1	1	-1
Paper	1	0	-1	1
Scissors	-1	1	0	-1
Well	1	-1	1	0

Table 3: Pay-off Matrix for RPSW

Unlike the previous games RPSW is a unbalanced game.

$$\text{Expected outcome Rock} = \frac{1}{4}(0)+\frac{1}{4}(-1)+\frac{1}{4}(1)+\frac{1}{4}(-1)=-0.25$$

$$\text{Expected outcome Paper} = \frac{1}{4}(1)+\frac{1}{4}(0)+\frac{1}{4}(-1)+\frac{1}{4}(1)=0.25$$

$$\text{Expected outcome Scissors} = \frac{1}{4}(-1)+\frac{1}{4}(1)+\frac{1}{4}(0)+\frac{1}{4}(-1)=-0.25$$

$$\text{Expected outcome Well} = \frac{1}{4}(1)+\frac{1}{4}(-1)+\frac{1}{4}(1)+\frac{1}{4}(0)=0.25$$

As you can see Paper and Well are the preferred strategies in this game.

4.3.2 Results of Simulation

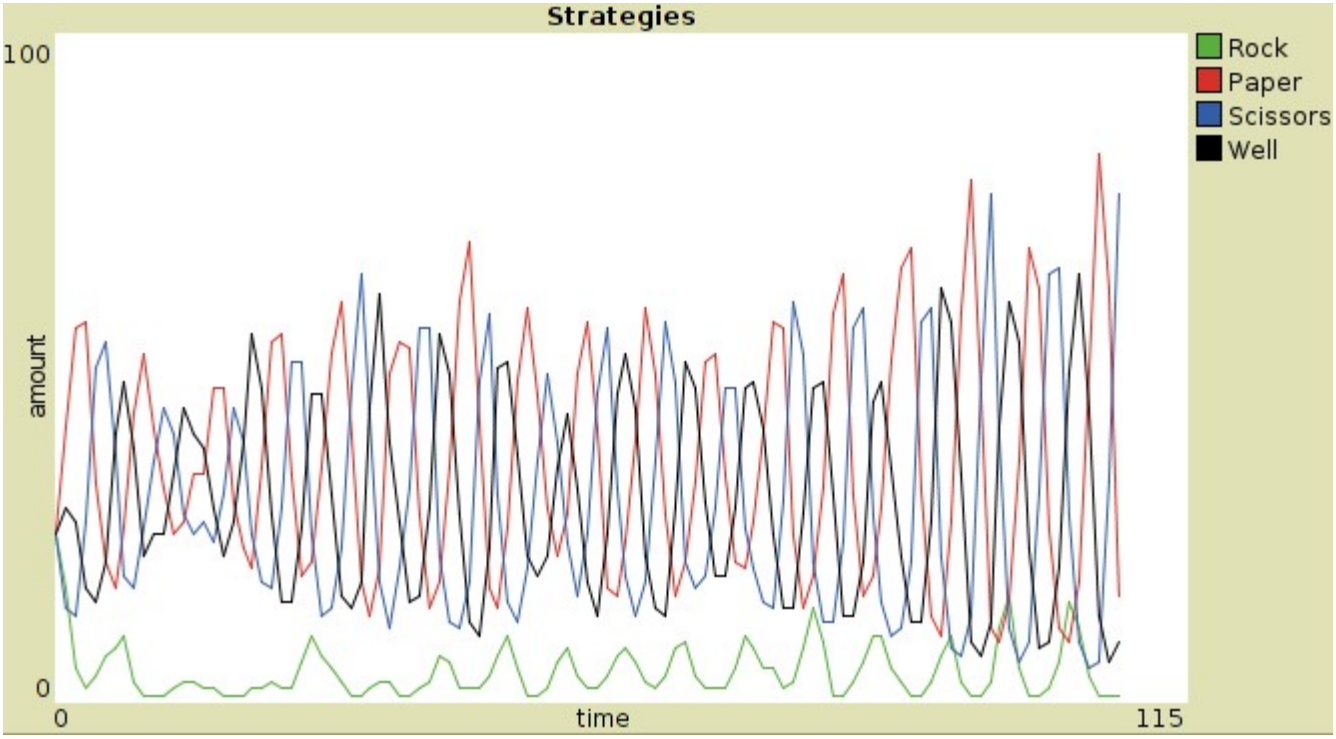


Illustration 10: Results of RPSW

4.3.3 Explanation Results

These result might look strange. According to the expected outcome of the strategies we would have expected that Well and Paper would fare significantly better than Scissors and Rock but this isn't the case. Rock doesn't fare well, but Scissors fares just as well as Well and Paper. This is because although Scissors only can exploit one kind of competition, namely a competition with a lot of Papers, it is the only strategy that exploit that particular competition. It has a monopoly on the competition. Paper can exploit three kinds of competition, one with a lot of Wells, one with a lot of Rocks and mix between the two strategies. Also it has the monopoly on Wells. It shares its dominance over Rocks with Wells. But Rock does so badly in the simulation that it isn't an advantage to be able to win from Rock. Therefore Paper is actually just as powerful as Scissors. Namely being able to exploit one environment. The same yields for Well but actually it is a bit weaker than Paper and Scissors. This is because it doesn't have a monopoly. It competes with Rock over dominating Scissors. In illustration 10 it shows that the high points of Wells are bit lower than the high points of Scissors and Papers. This is because Wells must share it environment with Rocks, which also is a bit higher after a round with a lot of Scissors.

4.4 RPS-5random

This next variation I created randomly. The goal of this variation was to test the explaining ability of my findings so far. I created this model by taking the RPS-5 model and randomly flipping two of the relationships. The changed relationships are marked with the a red box

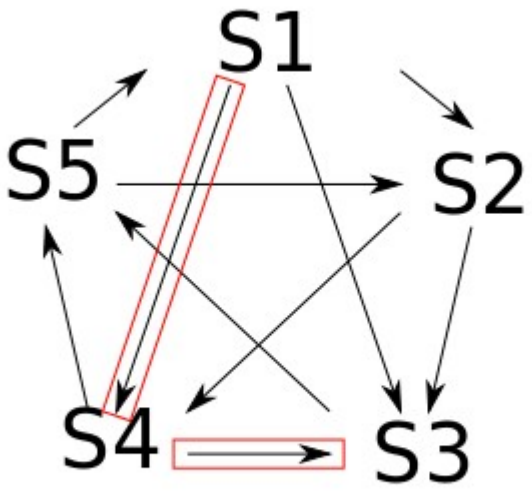


Illustration 11: Diagram of RPS-5random

4.4.1 Description

	S1	S2	S3	S4	S5
S1	0	1	1	1	-1
S2	-1	0	1	1	-1
S3	-1	-1	0	-1	1
S4	-1	-1	1	0	1
S5	1	1	-1	-1	0

Table 4: Pay-off Matrix of RPS-5random

$$\text{Expected outcome S1} = \frac{1}{5}(0) + \frac{1}{5}(1) + \frac{1}{5}(1) + \frac{1}{5}(1) + \frac{1}{5}(-1) = 0.4$$

$$\text{Expected outcome S2} = \frac{1}{5}(-1) + \frac{1}{5}(0) + \frac{1}{5}(1) + \frac{1}{5}(1) + \frac{1}{5}(-1) = 0$$

$$\text{Expected outcome S3} = \frac{1}{5}(-1) + \frac{1}{5}(-1) + \frac{1}{5}(0) + \frac{0}{5}(-1) + \frac{1}{5}(1) = -0.4$$

$$\text{Expected outcome S4} = \frac{1}{5}(-1) + \frac{1}{5}(-1) + \frac{1}{5}(1) + \frac{1}{5}(0) + \frac{1}{5}(1) = 0$$

$$\text{Expected outcome S5} = \frac{1}{5}(1) + \frac{1}{5}(1) + \frac{1}{5}(-1) + \frac{1}{5}(-1) + \frac{1}{5}(0) = 0$$

4.4.2 Results of the Simulation

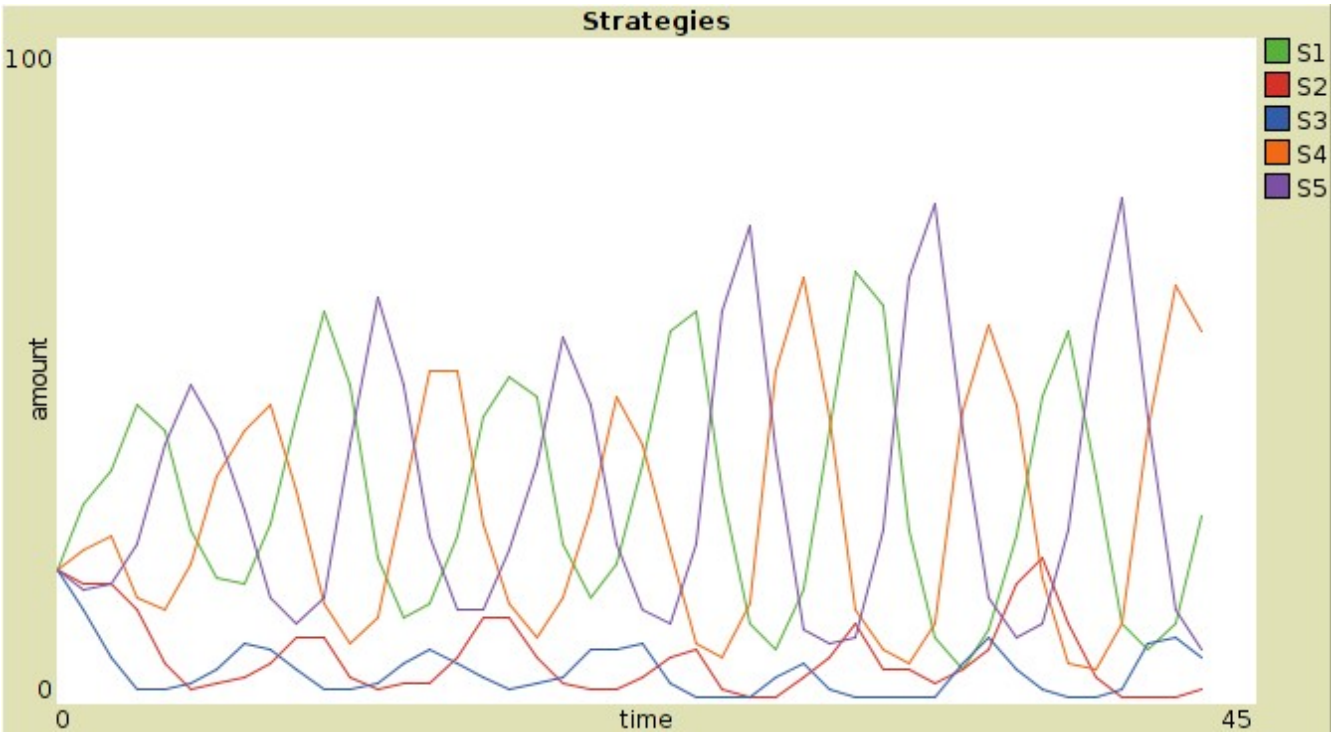


Illustration 12: Results of RPS-5random

4.4.3 Explaining the Results

As you can see in Illustration 12, The model exhibits a pattern in which First S1 becomes dominant then S5 and finally S4. S2 and S3 score poorly. S1 is the most effective strategy based on the expected outcome. Therefore It will always be the first dominating strategy, because it most effective in exploiting a mixed strategy. The next strategy that will become dominant is S5 because it is the only strategy that can exploit a competition with a lot of S1 agents. Next there are two strategy that can exploit a competition with a lot of S5 agents. S3

and S4. But S4 beats S3. Thus when the amount of S3 agents rise, S4 can again exploit the the competitions. This is similar like the dynamics we saw with RPS-5 in section 4.1.3. Next S1 becomes the most dominant because it can exploit a competition with a lot of S4 agents. S2 can also exploit S4 agents, but S1 is a lot more powerful strategy and S2 is no match.

5. Conclusion

5.1 Findings

5.1.1 Exploiting the Competition

The most important factor is the way the agents can exploit the other agents present in the environment. When simulating the RPS model, it was shown the most popular strategy is the strategy that wins from the strategy that was dominant in the previous round. In previous round, the strategy won from most other strategies that were present in the environment in the previous round. It could exploit the best because it could ensure the most wins.

5.1.2 Abundance of the strategy

During the simulation of the RPS-5 it was shown there was also another factor determining the pattern. The way RPS-5 is set up a strategy always has to compete with another strategy in dominating the competition. The previous factor wasn't sufficient enough in predicting the outcome. Because one of the strategies would dominate more than the other although they both could exploit the competition equally well. The answer was that one of the strategies was more abundant than the other. Meaning, one strategy has more agents in the environment than the other. The abundant strategy had a larger 'working force' to ensure more wins. Of course this factor is a double-edged knife. If a strategy is too abundant it may undermine itself. If the strategies becomes larger it will push out the strategies it can exploit. Also it gives other strategies to exploit the abundant strategy itself. Therefore this factor is more subtle than the previous one.

5.1.3 Monopoly on a Competition

In the unbalanced model RPSW, it was shown that unfavourable strategies, namely a strategy that is beaten by more strategies than they beat themselves, could perform much better than expected because of the dynamics of the simulation. This was because it had a monopoly on a competition. This means that a dominance of a strategy occurred in the model, which only could be exploited by the given strategy with the monopoly. Favoured strategies could also underperform, because they exploited competition that never occurred during the simulation.

5.1.4 First Dominating Strategy.

In standard RPS it was shown that the first dominating strategy was different for each run. There was a probability factor deciding which strategy would first dominate the environment.

In RPS-5random this is not the case. S1 would always be the first to dominate the environment. The reason for this is that S1 had the highest expected outcome and could exploit the most strategies. Concluding, the first dominating strategy is a random pick between the group of strategies with the highest expected outcome. In RPS the group is: Rock, Paper, Scissors. In RPS-5random the group only contain one strategy: S1.

5.1.5 Initial conditions

During the simulations it was shown that the initial ratio of strategies had little influence on the patterns that emerged. It could only delay the pattern from emergings. The only real effect the initial ratio had was on the first dominating strategy discussed in section 5.1.4. The evidence for this is only empirical. I did not make an effort to prove this because it was beyond the scope of this thesis.

5.2 Further Research

5.2.1 Spatial Distance

In their research Sinervo&Lively it show that the distance between the lizards has an influence on the model. The distribution of the different strategies was not homogeneous, but contains clusters of one kind of strategy. In my model the spatial distance wasn't a factor, because agents could easily traverse the entire environment during a round. Taking the spatial in account should make RPS-like models more suitable for explaining models that occur in reality.

5.2.1 The RPS loop in unbalanced RPS-like models.

If you look at the two unbalanced RPS-like variations and if you eliminate the strategies in the model that preform poorly, the standers RPS model remains. In RPSW. if you eliminate Rock, which preforms poorly, RSW remains. If you look at illustration 9 you can see that RSW is equal to RPS model. The same yields for RPS-5random, when you remove S2 and S3 as you can see in illustration 13.

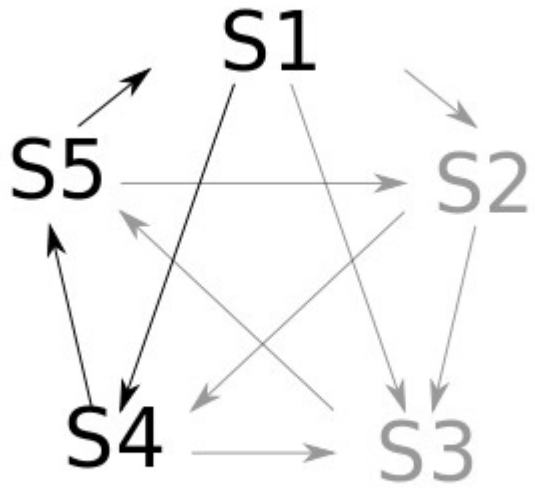


Illustration 13: The RPS loop in RPS5-random

This could be an important factor in predicting the pattern of unbalanced RPS-like models. But it requires more research. For example there are more possible RPS-loops in RPS-5random. Namely, S3S4S5. Why doesn't this RPS-loop emerge? A difference between the two RPS-loos is that the total expected outcomes of S1S4S5(0+0+0.4=0.4) is higher than S3S4S5(0+0+0=0). I'm not convinced that this is the determining factor. I discovered this phenomenon quite late in my thesis and couldn't research it further due to time constraints.

5.3 Final words

I think the next step in RPS-like systems is to try to find the building blocks that create the patterns that emerge. This would make it possible to predict the patterns that emerge during the simulations. To do this, models, that are larger en more complex, should be examined. The Netlogo implementation worked fine for this thesis but would be inadequate in further research if the models would become larger. The models are hard coded and changing to different kinds of models can be tedious. Also if the models would become larger and more complex, the way the results are now displayed would become too confusing.

Bibliography:

1. Hans Peters,(2008)Game Theory, a multi-leveled approach, Springer
2. Neil F johnsons and Micheal L. Hart,(2003) Crowd-Anticrowd Theory of Multi-Agent Minority Games, arXiv:cond-mat/0212088v2
3. Neil F.Johnsons and Pak Ming Hui (2003) Crow-Anticrowd Theory of Collective Dynamics in Competitive Multi-Agents Populations and Networks, arXiv:cond-mat/0306516v1
4. The rock-paper-scissors game and the evolution of alternative male strategies, B Sinervo & C.M. Lively, Nature vol 380, 21 March 1996
5. W.B. Arthur. (1994) Inductive reasoning and bounded rationality; the El Farol problem. American Economics Association Papers and Proceedings 84, 406 411.
6. D. Challet and Y.C. Zhang ., Physica A 256, 514 (1998)

Appendix A: source code of the RPS-model.

I decided to only include the source code of the standard RPS model. The other programs are very similar and this code could easily be modified into the other models. The main differences is in the amount of strategies that are handled and the way the function 'fight!!' works.

The last part of the code is the remnant of the way Netlogo stores the GUI. Putting the code in a file and giving it the extension .nlogo should produce file which can be opened by the NetLogo program.

```
1.  breed [scissors scissor]     ;;each turtle represent a strategy
2.  breed [rocks rock]
3.  breed [papers paper]
4.  globals [n x roundx scorerocks scorepapers scorescissors totalscore]
5.
6.  to setup
7.    clear-all
8.    ask patches [set pcolor white]
9.    create-rocks #rocks
10.     [ set color green
11.       setxy random-xcor random-ycor]
12.   create-papers #papers
13.     [ set color red
14.       setxy random-xcor random-ycor]
15.   create-scissors #scissors
16.     [ set color blue
17.       setxy random-xcor random-ycor]
18.   resetscores
19.   setup-plot
20.   do-plotting
21. end
22.
23. to go
24.   while [roundx < 100][
25.     ask turtles
26.     [ rt random-float 10 - random-float 10  ;; wander around randomly
27.       fd 1 ]
28.     fight!!  ;; each time a strategy wins it's score gets raised
29.     set roundx(roundx + 1)
30.     tick
31.   ]
32.   clear-turtles ;; all old agents are deleted
33.   newpopulation
34.   resetscores
35.   tick
36.   do-plotting
37. end
38.
39. to fight!!
40.   ask rocks[ask papers-here [set scorepapers (scorepapers + 1)]  ;; an agent asks the agents it
```

```
   loses from to update their scores, although maybe a bit awkward it is the most simple solution
   codewise
41.     ]
42.   ask scissors[ask rocks-here [set scorerocks (scorerocks + 1)]
43.     ]
44.   ask papers[ask scissors-here [set scorescissors (scorescissors + 1)]
45.     ]
46.
47. end
48.
49. to newpopulation
50.   set totalscore (3 + scorerocks + scorepapers + scorescissors) ;; + 3 to prevent dividing by
   zero and to compensate the free win all strategies recieve
51. create-rocks (1 + scorerocks / totalscore * 100)  ;; +1 to prevent strategies from dying out.
52.     [ set color green
53.       setxy random-xcor random-ycor]
54.   create-papers (1 + scorepapers / totalscore * 100)
55.     [ set color red
56.       setxy random-xcor random-ycor]
57.   create-scissors (1 + scorescissors / totalscore * 100)
58.     [ set color blue
59.       setxy random-xcor random-ycor]
60. end
61.
62. to resetscores ;; resets globals for the next round
63.   set roundx 0
64.   set scorerocks 0
65.   set scorepapers 0
66.   set scorescissors 0
67.   set totalscore 0
68. end
69.
70. to setup-plot      ;; the code below sets up the plot
71.   set-current-plot "Strategies"
72.   set-plot-y-range 0 100
73. end
74.
75. to do-plotting      ;; plot strategies
76.   set-current-plot "Strategies"
77.   set-current-plot-pen "rock"
78.   plot count rocks
79.   set-current-plot-pen "scissors"
80.   plot count scissors
81.   set-current-plot-pen "paper"
82.   plot count papers
83. end
84.
85. to even ;;function to equalise the initail ratio
86. set #rocks 33
87. set #papers 33
88. set #scissors 34
89. end
90.
91. to randomize ;; randomizes initial ratio
```

```
92.    set n 0
93.    set #rocks 0
94.    set #papers 0
95.    set #scissors 0
96.    while [n < 10][
97.     set x random 3
98.     if x = 0 [set #rocks (#rocks + 10)]  ;; + 10 because if +1 the ratio would be near 1:1:1
   thanks to statistics
99.     if x = 1 [set #papers (#papers + 10)]
100.           if x = 2 [set #scissors (#scissors + 10)]
101.           set n ( n + 1)
102.           ]
103.
104.
105.       end
106.       @#$#@#$#@
107.       GRAPHICS-WINDOW
108.       274
109.       10
110.       634
111.       391
112.       12
113.       12
114.       14.0
115.       1
116.       10
117.       1
118.       1
119.       1
120.       0
121.       1
122.       1
123.       1
124.       -12
125.       12
126.       -12
127.       12
128.       1
129.       1
130.       1
131.       ticks
132.
133.       SLIDER
134.       12
135.       101
136.       261
137.       134
138.       #papers
139.       #papers
140.       1
141.       100
142.       33
143.       1
144.       1
```

```
145.       NIL
146.       HORIZONTAL
147.
148.       SLIDER
149.       12
150.       136
151.       261
152.       169
153.       #scissors
154.       #scissors
155.       1
156.       100
157.       34
158.       1
159.       1
160.       NIL
161.       HORIZONTAL
162.
163.       BUTTON
164.       12
165.       22
166.       79
167.       55
168.       setup
169.       setup
170.       NIL
171.       1
172.       T
173.       OBSERVER
174.       NIL
175.       NIL
176.       NIL
177.       NIL
178.
179.       BUTTON
180.       82
181.       22
182.       144
183.       55
184.       go
185.       go
186.       T
187.       1
188.       T
189.       OBSERVER
190.       NIL
191.       NIL
192.       NIL
193.       NIL
194.
195.       PLOT
196.       643
197.       16
198.       1315
```

```
199.        387
200.        Strategies
201.        time
202.        amount
203.        0.0
204.        35.0
205.        0.0
206.        100.0
207.        true
208.        true
209.        PENS
210.        "Rock" 1.0 0 -10899396 true
211.        "Paper" 1.0 0 -2674135 true
212.        "Scissors" 1.0 0 -13345367 true
213.
214.        SLIDER
215.        12
216.        66
217.        261
218.        99
219.        #rocks
220.        #rocks
221.        1
222.        100
223.        33
224.        1
225.        1
226.        NIL
227.        HORIZONTAL
228.
229.        BUTTON
230.        14
231.        175
232.        69
233.        208
234.        1:1:1
235.        even
236.        NIL
237.        1
238.        T
239.        OBSERVER
240.        NIL
241.        NIL
242.        NIL
243.        NIL
244.
245.        BUTTON
246.        75
247.        176
248.        184
249.        209
250.        NIL
251.        Randomize
252.        NIL
```

```
253.       1
254.       T
255.       OBSERVER
256.       NIL
257.       NIL
258.       NIL
259.       NIL
260.
261.       @#$#@#$#@
262.       WHAT IS IT?
263.       -----------
264.       This is a simulation of the rock paper scissor game in a population.
265.
266.       Each round a hundred persons choose one of three strategies: rock,paper or scissors. Rock
   beats scissors, scissors beats paper, paper beats rock.When in a population one stratey is
   effective, the strategy will become more popular. Next round more persons will choose this
   strategy.
267.
268.       This will produce a trend effect that in wich a strategy wil dominate and in the next
   round it's counter strategy will dominate. This is because last round the strategy was very
   effective because there were a lot of the strategy it counters.
269.
270.
271.
272.
273.       HOW TO USE IT
274.       --------------
275.       Setup and go.
276.
277.       It is possible to change the intitial ratio between the different strategies
278.
279.       HOW IT WORKS
280.       ----------------
281.       In the simulation the strategies walk around in the world. When they meet a strategy they
   beat (rock beats scissors, scissors beats paper, paper beats rock) the strategy gets a score.
282.
283.       After a round the strategies get updated. This done using this formula:
284.
285.       myscore/totalscore*100
286.
287.       myscore: is the score of a certain strategy
288.       totalscore: the summation off all the scores.
289.
290.       THINGS TO NOTICE
291.       ----------------
292.
293.       It is possible to have a larger population then 100 in the first round. I couldn't find a
   nice solution for it. Also it doesn't really have an effect on the simulation.
294.
295.       AUTHOR
296.       ----------------------
297.       Koen Klinkers 3213129
298.       k.klinkers@gmail.com
299.
```

```
300.
301.
302.        @#$#@#$#@
303.        default
304.        true
305.        0
306.        Polygon -7500403 true true 150 5 40 250 150 205 260 250
307.
308.        airplane
309.        true
310.        0
311.        Polygon -7500403 true true 150 0 135 15 120 60 120 105 15 165 15 195 120 180 135 240 105
    270 120 285 150 270 180 285 210 270 165 240 180 180 285 195 285 165 180 105 180 60 165 15
312.
313.        arrow
314.        true
315.        0
316.        Polygon -7500403 true true 150 0 0 150 105 150 105 293 195 293 195 150 300 150
317.
318.        box
319.        false
320.        0
321.        Polygon -7500403 true true 150 285 285 225 285 75 150 135
322.        Polygon -7500403 true true 150 135 15 75 150 15 285 75
323.        Polygon -7500403 true true 15 75 15 225 150 285 150 135
324.        Line -16777216 false 150 285 150 135
325.        Line -16777216 false 150 135 15 75
326.        Line -16777216 false 150 135 285 75
327.
328.        bug
329.        true
330.        0
331.        Circle -7500403 true true 96 182 108
332.        Circle -7500403 true true 110 127 80
333.        Circle -7500403 true true 110 75 80
334.        Line -7500403 true 150 100 80 30
335.        Line -7500403 true 150 100 220 30
336.
337.        butterfly
338.        true
339.        0
340.        Polygon -7500403 true true 150 165 209 199 225 225 225 255 195 270 165 255 150 240
341.        Polygon -7500403 true true 150 165 89 198 75 225 75 255 105 270 135 255 150 240
342.        Polygon -7500403 true true 139 148 100 105 55 90 25 90 10 105 10 135 25 180 40 195 85 194
    139 163
343.        Polygon -7500403 true true 162 150 200 105 245 90 275 90 290 105 290 135 275 180 260 195
    215 195 162 165
344.        Polygon -16777216 true false 150 255 135 225 120 150 135 120 150 105 165 120 180 150 165
    225
345.        Circle -16777216 true false 135 90 30
346.        Line -16777216 false 150 105 195 60
347.        Line -16777216 false 150 105 105 60
348.
349.        car
```

```
350.       false
351.       0
352.       Polygon -7500403 true true 300 180 279 164 261 144 240 135 226 132 213 106 203 84 185 63
    159 50 135 50 75 60 0 150 0 165 0 225 300 225 300 180
353.       Circle -16777216 true false 180 180 90
354.       Circle -16777216 true false 30 180 90
355.       Polygon -16777216 true false 162 80 132 78 134 135 209 135 194 105 189 96 180 89
356.       Circle -7500403 true true 47 195 58
357.       Circle -7500403 true true 195 195 58
358.
359.       circle
360.       false
361.       0
362.       Circle -7500403 true true 0 0 300
363.
364.       circle 2
365.       false
366.       0
367.       Circle -7500403 true true 0 0 300
368.       Circle -16777216 true false 30 30 240
369.
370.       cow
371.       false
372.       0
373.       Polygon -7500403 true true 200 193 197 249 179 249 177 196 166 187 140 189 93 191 78 179
    72 211 49 209 48 181 37 149 25 120 25 89 45 72 103 84 179 75 198 76 252 64 272 81 293 103 285 121
    255 121 242 118 224 167
374.       Polygon -7500403 true true 73 210 86 251 62 249 48 208
375.       Polygon -7500403 true true 25 114 16 195 9 204 23 213 25 200 39 123
376.
377.       cylinder
378.       false
379.       0
380.       Circle -7500403 true true 0 0 300
381.
382.       dot
383.       false
384.       0
385.       Circle -7500403 true true 90 90 120
386.
387.       face happy
388.       false
389.       0
390.       Circle -7500403 true true 8 8 285
391.       Circle -16777216 true false 60 75 60
392.       Circle -16777216 true false 180 75 60
393.       Polygon -16777216 true false 150 255 90 239 62 213 47 191 67 179 90 203 109 218 150 225
    192 218 210 203 227 181 251 194 236 217 212 240
394.
395.       face neutral
396.       false
397.       0
398.       Circle -7500403 true true 8 7 285
399.       Circle -16777216 true false 60 75 60
```

```
400.       Circle -16777216 true false 180 75 60
401.       Rectangle -16777216 true false 60 195 240 225
402.
403.       face sad
404.       false
405.       0
406.       Circle -7500403 true true 8 8 285
407.       Circle -16777216 true false 60 75 60
408.       Circle -16777216 true false 180 75 60
409.       Polygon -16777216 true false 150 168 90 184 62 210 47 232 67 244 90 220 109 205 150 198
    192 205 210 220 227 242 251 229 236 206 212 183
410.
411.       fish
412.       false
413.       0
414.       Polygon -1 true false 44 131 21 87 15 86 0 120 15 150 0 180 13 214 20 212 45 166
415.       Polygon -1 true false 135 195 119 235 95 218 76 210 46 204 60 165
416.       Polygon -1 true false 75 45 83 77 71 103 86 114 166 78 135 60
417.       Polygon -7500403 true true 30 136 151 77 226 81 280 119 292 146 292 160 287 170 270 195
    195 210 151 212 30 166
418.       Circle -16777216 true false 215 106 30
419.
420.       flag
421.       false
422.       0
423.       Rectangle -7500403 true true 60 15 75 300
424.       Polygon -7500403 true true 90 150 270 90 90 30
425.       Line -7500403 true 75 135 90 135
426.       Line -7500403 true 75 45 90 45
427.
428.       flower
429.       false
430.       0
431.       Polygon -10899396 true false 135 120 165 165 180 210 180 240 150 300 165 300 195 240 195
    195 165 135
432.       Circle -7500403 true true 85 132 38
433.       Circle -7500403 true true 130 147 38
434.       Circle -7500403 true true 192 85 38
435.       Circle -7500403 true true 85 40 38
436.       Circle -7500403 true true 177 40 38
437.       Circle -7500403 true true 177 132 38
438.       Circle -7500403 true true 70 85 38
439.       Circle -7500403 true true 130 25 38
440.       Circle -7500403 true true 96 51 108
441.       Circle -16777216 true false 113 68 74
442.       Polygon -10899396 true false 189 233 219 188 249 173 279 188 234 218
443.       Polygon -10899396 true false 180 255 150 210 105 210 75 240 135 240
444.
445.       hex
446.       false
447.       0
448.       Polygon -7500403 true true 0 150 75 30 225 30 300 150 225 270 75 270
449.
450.       house
```

```
451.       false
452.       0
453.       Rectangle -7500403 true true 45 120 255 285
454.       Rectangle -16777216 true false 120 210 180 285
455.       Polygon -7500403 true true 15 120 150 15 285 120
456.       Line -16777216 false 30 120 270 120
457.
458.       leaf
459.       false
460.       0
461.       Polygon -7500403 true true 150 210 135 195 120 210 60 210 30 195 60 180 60 165 15 135 30
    120 15 105 40 104 45 90 60 90 90 105 105 120 120 120 105 60 120 60 135 30 150 15 165 30 180 60
    195 60 180 120 195 120 210 105 240 90 255 90 263 104 285 105 270 120 285 135 240 165 240 180 270
    195 240 210 180 210 165 195
462.       Polygon -7500403 true true 135 195 135 240 120 255 105 255 105 285 135 285 165 240 165 195
463.
464.       line
465.       true
466.       0
467.       Line -7500403 true 150 0 150 300
468.
469.       line half
470.       true
471.       0
472.       Line -7500403 true 150 0 150 150
473.
474.       molecule1
475.       true
476.       0
477.       Circle -7500403 true true 80 80 142
478.       Circle -1 true false 52 195 61
479.
480.       molecule2
481.       true
482.       0
483.       Circle -7500403 true true 78 79 146
484.       Circle -1 true false 40 41 69
485.       Circle -1 true false 194 42 71
486.
487.       molecule3
488.       true
489.       0
490.       Circle -7500403 true true 92 92 118
491.       Circle -1 true false 120 32 60
492.       Circle -1 true false 58 183 60
493.       Circle -1 true false 185 183 60
494.
495.       pentagon
496.       false
497.       0
498.       Polygon -7500403 true true 150 15 15 120 60 285 240 285 285 120
499.
500.       person
501.       false
```

```
502.       0
503.       Circle -7500403 true true 110 5 80
504.       Polygon -7500403 true true 105 90 120 195 90 285 105 300 135 300 150 225 165 300 195 300
    210 285 180 195 195 90
505.       Rectangle -7500403 true true 127 79 172 94
506.       Polygon -7500403 true true 195 90 240 150 225 180 165 105
507.       Polygon -7500403 true true 105 90 60 150 75 180 135 105
508.
509.       plant
510.       false
511.       0
512.       Rectangle -7500403 true true 135 90 165 300
513.       Polygon -7500403 true true 135 255 90 210 45 195 75 255 135 285
514.       Polygon -7500403 true true 165 255 210 210 255 195 225 255 165 285
515.       Polygon -7500403 true true 135 180 90 135 45 120 75 180 135 210
516.       Polygon -7500403 true true 165 180 165 210 225 180 255 120 210 135
517.       Polygon -7500403 true true 135 105 90 60 45 45 75 105 135 135
518.       Polygon -7500403 true true 165 105 165 135 225 105 255 45 210 60
519.       Polygon -7500403 true true 135 90 120 45 150 15 180 45 165 90
520.
521.       square
522.       false
523.       0
524.       Rectangle -7500403 true true 30 30 270 270
525.
526.       square 2
527.       false
528.       0
529.       Rectangle -7500403 true true 30 30 270 270
530.       Rectangle -16777216 true false 60 60 240 240
531.
532.       star
533.       false
534.       0
535.       Polygon -7500403 true true 151 1 185 108 298 108 207 175 242 282 151 216 59 282 94 175 3
    108 116 108
536.
537.       target
538.       false
539.       0
540.       Circle -7500403 true true 0 0 300
541.       Circle -16777216 true false 30 30 240
542.       Circle -7500403 true true 60 60 180
543.       Circle -16777216 true false 90 90 120
544.       Circle -7500403 true true 120 120 60
545.
546.       tree
547.       false
548.       0
549.       Circle -7500403 true true 118 3 94
550.       Rectangle -6459832 true false 120 195 180 300
551.       Circle -7500403 true true 65 21 108
552.       Circle -7500403 true true 116 41 127
553.       Circle -7500403 true true 45 90 120
```

```
554.        Circle -7500403 true true 104 74 152
555.
556.        triangle
557.        false
558.        0
559.        Polygon -7500403 true true 150 30 15 255 285 255
560.
561.        triangle 2
562.        false
563.        0
564.        Polygon -7500403 true true 150 30 15 255 285 255
565.        Polygon -16777216 true false 151 99 225 223 75 224
566.
567.        truck
568.        false
569.        0
570.        Rectangle -7500403 true true 4 45 195 187
571.        Polygon -7500403 true true 296 193 296 150 259 134 244 104 208 104 207 194
572.        Rectangle -1 true false 195 60 195 105
573.        Polygon -16777216 true false 238 112 252 141 219 141 218 112
574.        Circle -16777216 true false 234 174 42
575.        Rectangle -7500403 true true 181 185 214 194
576.        Circle -16777216 true false 144 174 42
577.        Circle -16777216 true false 24 174 42
578.        Circle -7500403 false true 24 174 42
579.        Circle -7500403 false true 144 174 42
580.        Circle -7500403 false true 234 174 42
581.
582.        turtle
583.        true
584.        0
585.        Polygon -10899396 true false 215 204 240 233 246 254 228 266 215 252 193 210
586.        Polygon -10899396 true false 195 90 225 75 245 75 260 89 269 108 261 124 240 105 225 105
    210 105
587.        Polygon -10899396 true false 105 90 75 75 55 75 40 89 31 108 39 124 60 105 75 105 90 105
588.        Polygon -10899396 true false 132 85 134 64 107 51 108 17 150 2 192 18 192 52 169 65 172 87
589.        Polygon -10899396 true false 85 204 60 233 54 254 72 266 85 252 107 210
590.        Polygon -7500403 true true 119 75 179 75 209 101 224 135 220 225 175 261 128 261 81 224 74
    135 88 99
591.
592.        wheel
593.        false
594.        0
595.        Circle -7500403 true true 3 3 294
596.        Circle -16777216 true false 30 30 240
597.        Line -7500403 true 150 285 150 15
598.        Line -7500403 true 15 150 285 150
599.        Circle -7500403 true true 120 120 60
600.        Line -7500403 true 216 40 79 269
601.        Line -7500403 true 40 84 269 221
602.        Line -7500403 true 40 216 269 79
603.        Line -7500403 true 84 40 221 269
604.
605.        x
```

```
606.        false
607.        0
608.        Polygon -7500403 true true 270 75 225 30 30 225 75 270
609.        Polygon -7500403 true true 30 75 75 30 270 225 225 270
610.
611.        @#$#@#$#@
612.        NetLogo 4.1.2
613.        @#$#@#$#@
614.        @#$#@#$#@
615.        @#$#@#$#@
616.        @#$#@#$#@
617.        @#$#@#$#@
618.        default
619.        0.0
620.        -0.2 0 0.0 1.0
621.        0.0 1 1.0 0.0
622.        0.2 0 0.0 1.0
623.        link direction
624.        true
625.        0
626.        Line -7500403 true 150 150 90 180
627.        Line -7500403 true 150 150 210 180
628.
629.        @#$#@#$#@
630.        0
631.        @#$#@#$#@
```