AN AGENT-BASED MODELING APPROACH TO ITERATED PRISONER’S DILEMMA ON NETWORKS

A Thesis
submitted to the Faculty of the
Graduate School of Arts and Sciences
of Georgetown University
in partial fulfillment of the requirements for the
degree of
Master of Arts
in Communication, Culture, and Technology

By

Chen Shen, B.Eng.

Washington, D.C.
April 18, 2018
AN AGENT-BASED MODELING APPROACH TO ITERATED PRISONER’S DILEMMA ON NETWORKS

Chen Shen, B.Eng.

Thesis Advisor: Evan Barba, Ph.D.

ABSTRACT

Prisoner’s dilemma is a framework in game theory to explain why rational individuals would choose to not cooperate even at their own expense. In 1984, Axelrod’s computer tournament and its simple yet profound results brought iterated prisoner’s dilemma (IPD) to the view of the public and a myriad of research domains. Sociology, political science, economics, anthropology, evolutionary biology and computer science all used this framework to explore the cooperative behavior in their discipline and continued pushing forward the study of IPD to this day. One particular thriving branch is evolutionary dynamics, where the interaction and adaptation between multiple strategies are simulated to illustrate their robustness.

Nevertheless, the majority of studies done assumed a complete graph or grid-based topology for the strategies to inhabit. Admittedly, they have their own merits in revealing general patterns, yet neither is suitable to model actual social structures especially those based on the Internet, where spatial contiguity does not entail interaction. This study examines the evolution of strategies on different network models, including scale-free network, small-world network, and social circles. Due to their unique topology, the evolution on these networks exhibits distinct features comparing to existing literature. This research develops an agent-based model to generate different networks and simulate the evolution. The research proposes future studies and the model developed serves as a general-purpose model open for inquiries in the field of IPD.
ACKNOWLEDGEMENTS

The research and writing of this thesis are dedicated to everyone who helped along the way. I
would like to thank Evan Barba as the thesis advisor, Martin Irvine as a committee member,
Garrison LeMasters and Sarah Twose as an academic advisor. Thanks to the NECSI faculty for
education on complex systems and computational modeling. Thanks to the NetLogo group that
makes and maintains this fantastic simulation tool. At last, thanks to my family that always
support my research.

Many thanks,
Chen Shen
TABLE OF CONTENTS

Introduction..................................................................................................................................... 1

Chapter 1 The Iterated Prisoner’s Dilemma Framework .......................................................... 2
   Prisoner’s Dilemma ................................................................................................................... 2
   Iterated Prisoner’s Dilemma .................................................................................................. 4
   Real-life Examples of Iterated Prisoner’s Dilemma ............................................................... 5

Chapter 2 Literature Review ....................................................................................................... 7
   The Evolution of Cooperation ................................................................................................ 7
   Other Literature .................................................................................................................... 12

Chapter 3 Methodology ............................................................................................................ 14
   Agent-Based Modeling .......................................................................................................... 14
   Social Network Analysis ....................................................................................................... 20
      Nodes and Links ................................................................................................................ 21
      Structural Focus ................................................................................................................ 22
   The Combination of the Two Methods ................................................................................. 23

Chapter 4 Model Explanation ................................................................................................... 25
   Interface Introduction ............................................................................................................. 25
   Platform Introduction ............................................................................................................. 26
      Intuitive Visual Presentation ............................................................................................... 26
   Network Extension ............................................................................................................... 27
   Randomness .......................................................................................................................... 28
   BehaviorSpace and Data Analysis ....................................................................................... 29

Model Explanation .................................................................................................................. 29
   Main Process of the Model ................................................................................................ 29
   Main Process of BehaviorSpace ............................................................................................. 32

Network Metrics and Network Structure Models ................................................................. 33
   Undirected Network ........................................................................................................... 33
   Degree ................................................................................................................................ 33
   Average Path Length .......................................................................................................... 34
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clustering Coefficient</td>
<td>35</td>
</tr>
<tr>
<td>Degree Distribution</td>
<td>35</td>
</tr>
<tr>
<td>Scale-free Network</td>
<td>36</td>
</tr>
<tr>
<td>Small-world Network</td>
<td>38</td>
</tr>
<tr>
<td>Social Circles</td>
<td>40</td>
</tr>
<tr>
<td>Model Structure Justification</td>
<td>42</td>
</tr>
<tr>
<td>Chapter 5 Data Analysis</td>
<td>44</td>
</tr>
<tr>
<td>List of Strategies</td>
<td>44</td>
</tr>
<tr>
<td>The Effect of Population</td>
<td>45</td>
</tr>
<tr>
<td>Scale-free Network</td>
<td>45</td>
</tr>
<tr>
<td>Small-world Network</td>
<td>48</td>
</tr>
<tr>
<td>The Effect of Network Structure</td>
<td>51</td>
</tr>
<tr>
<td>The Effect of Noise Level</td>
<td>53</td>
</tr>
<tr>
<td>Other Ways to Deal with Noise and the Drawback</td>
<td>57</td>
</tr>
<tr>
<td>Adaptability</td>
<td>60</td>
</tr>
<tr>
<td>Chapter 6 Conclusion and Limitation</td>
<td>63</td>
</tr>
<tr>
<td>Conclusion</td>
<td>63</td>
</tr>
<tr>
<td>Limitation</td>
<td>64</td>
</tr>
<tr>
<td>Future Research</td>
<td>65</td>
</tr>
<tr>
<td>References</td>
<td>66</td>
</tr>
<tr>
<td>Bibliography</td>
<td>70</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

Figure 1. The dynamic population in an IPD tournament .................................................... 10
Figure 2. The graph of Lotka-Volterra equations ................................................................. 15
Figure 3. Simulation interface........................................................................................... 25
Figure 4. BehaviorSpace interface.................................................................................... 32
Figure 5. Average path length example ............................................................................. 34
Figure 6. Degree distribution example ............................................................................... 36
Figure 7. The preferential attachment process ................................................................. 37
Figure 8. Lattice experiment ............................................................................................ 39
Figure 9. Clustering coefficient and average path length .................................................. 40
Figure 10. Population effect on scale-free networks with 5% TFT ...................................... 46
Figure 11. Population effect on scale-free networks with 5% TFT, scatter chart ............... 46
Figure 12. Population effect on scale-free networks with 5% TFT and 45% All-C ............. 47
Figure 13. Population effect on scale-free networks with 25% TFT and 25% All-C .......... 48
Figure 14. Population effect on small-world networks with 5% TFT ............................... 49
Figure 15. Population effect on small-world networks, with 5% and 10% TFT ............... 49
Figure 16. Population effect on small-world networks, with 5%, 10% TFT, and 5% TFT plus 45% All-C ................................................................. 50
Figure 17. Neighbors parameter effect with p = 0 ............................................................ 51
Figure 18. Neighbors parameter effect with p = 0.08 ....................................................... 52
Figure 19. Rewiring parameter effect on small-world networks with k = 3 ....................... 52
Figure 20. Performance decrease in noisy environments with payoff matrix 5-3-1-0 ....... 54
Figure 21. Performance decrease in noisy environments with payoff matrix 5-4-1-0 ....... 55
Figure 22. Population change in an initialization of 5% TFT and 5% GTFT ..................... 55
Figure 23. 5% GTFT mutation in a TFT population .......................................................... 56
Figure 24. 5% GTFT invades TFT .................................................................................... 56
Figure 25. Population change in a population including 5% initial TFT, with 5% GTFT mutation introduced later ................................................................. 57
Figure 26. Four phases of an uncooperative invasion in a population mixed with TFT ..... 59
Figure 27. Effect of 5% unadaptable All-C ................................................................. 60
Figure 28. Effect of 10% unadaptable All-C ................................................................. 61
Figure 29. Effect of 15% unadaptable All-C ................................................................. 61
Figure 30. Effect of 30% unadaptable All-C ................................................................. 62
Introduction

This paper develops an agent-based model to simulate the evolution of Iterated Prisoner’s Dilemma strategies on different network models. We start in Chapter 1 by introducing prisoner’s dilemma and iterated prisoner’s dilemma. In Chapter 2 we review the essential works in the field. Chapter 3 introduces the primary methods used in this research, namely agent-based modeling and social network analysis. Then in Chapter 4 we explain the agent-based model we designed and some fundamental network science concepts used in this study. We analyze the simulation results in Chapter 5, then conclude with limitations and future studies in Chapter 6.
Prisoner's Dilemma

Prisoner's Dilemma (PD), initially framed by Merrill Flood and Melvin Dresher in the 1950s', has long been a famous example in game theory to analyze why two individuals choose not to cooperate even if the payoff would suggest otherwise. The "prisoner" in the title originated from the context of the game formalized by A. W. Tucker. The payoff matrix in the classic description of PD has negative rewards as they represent imprisonment, and individuals are motivated to play towards less negative results. Since the negative items can sometimes be confusing, we adopt a positive matrix which is essentially the same in the sense that players choose to cooperate or not to get points as high as possible. The rule is described as below:

<table>
<thead>
<tr>
<th></th>
<th>B Cooperates</th>
<th>B Defects</th>
</tr>
</thead>
<tbody>
<tr>
<td>A Cooperates</td>
<td>A gets 3 points, B gets 3 points</td>
<td>A gets 0 points, B gets 5 points</td>
</tr>
<tr>
<td>A Defects</td>
<td>A gets 5 points, B gets 0 points</td>
<td>A gets 1 points, B gets 1 points</td>
</tr>
</tbody>
</table>

If A and B both cooperate (C), they both get the reward (R) of 3 points. If one of them cooperates and the other defects (D), the defector will get a high temptation (T) of 5 points, while the cooperator gets a sucker (S) point of 0. If they both defect, they both get a punishment point (P) of merely 1. So the payoff matrix for A (the column player) can be represented as below:
By observing the matrix, imagine we are player A, we can make two, and only two assumptions:

(1) B will cooperate, or (2) B will defect. In assumption (1), if we cooperate, we get 3 points while defecting will net 5. In assumption (2), defecting also has 1 point as opposed to 0 by cooperating. So either way, to maximize our utility, we will choose to defect. We can reason in the same logic for player B. Hence both players will choose to defect and get one point each, which is much less than that from mutual cooperation. Moreover, if we take the two players as a whole, their total gain is only 2, far from the optimal result of 6. Rational individuals would get less than they can have, this explains the "dilemma" in PD. The points do not have to be identical to the matrix above, for a game to qualify as PD, the payoff only has to meet the following inequities:

(1) \( T > R > P > S \)

This requirement ensures defection will get more points regardless of the opponent’s choice (\( T > R \), and \( P > S \)). It also stipulates two defecting a worse result than two cooperation (\( R > P \)).

(2) \( R > \frac{T + S}{2} \)
This requirement means that on average, cooperating has a better outcome than alternating between defecting and being defected.

So, PD can be formalized as a game wherein the payoff matrix is as below:

\[
\begin{bmatrix}
(R, R) & (S, T) \\
(T, S) & (P, P)
\end{bmatrix}
\]

wherein \( T > R > P > S \) and \( R > \frac{T+S}{2} \)

**Iterated Prisoner’s Dilemma**

The framework of Prisoner’s Dilemma introduces us to a depressing worldview that rationality would incubate selfish agents who refuse to cooperate for a better outcome. For cooperation to emerge, we have to make a little modification to the PD game, so that two players will play multiple times (turns) and accumulate their scores, thus called Iterated Prisoner’s Dilemma (IPD). In an IPD game, players can have memories of the past moves, which enables reciprocity and revenge. If the payoff suggests that a long-term mutually beneficial relationship is more rewarding than a chain of retaliation caused by an initial defecting, the rational agents will behave accordingly. So, in the IPD framework, we do not assume any moral standards for players to facilitate cooperation, they are driven by exactly the same logic that makes them defecting in PD: to maximize the utility of their own.

Traditionally, there are some requirements for IPD:

1. The number of turns is unknown to the players

   Suppose both players know they are going to interact \( N \) (a finite number) turns. Because what they choose to do on the Nth move will have no further consequences, so on that
last move they will both defect to maximize their score. Knowing the opponent can reason in the same logic, in other words knowing opponent’s defection at the end is inevitable, the penultimate turn would have no bearing on the last turn’s choice. This means both players will defect on the $N - 1$ turn, and the reasoning can chain back to the very beginning, which effectively made all choices moot, and the interactions a chain of mutual betrayal.

As a result, for the players to cooperate to avoid unnecessary retaliation and the possible score loss, they cannot know in prior the number of turns they will interact.

2. There is a discount parameter $w$ to adjust the outcome of each turn to make the present more important than the future.

Mathematically, this defines a variable $w < 1$ so that the actual outcome on the $N$th turn is multiplied by $w^N$. For example, in a chain of $N$ mutual betrayals, instead of the ideal outcome of $N$ for both players, they only get $w^0 + w^1 + w^2 + \cdots + w^N$ (which is less than $N$ since $w < 1$).

The introduction of this discount parameter is due to the belief that presence is more valuable than an unwarrantable future, or, a bird in the hand is worth two in the bush. Cooperative behavior predicates on future encounters of the two players, but if the chances are very slim for them to meet again, they will put less consideration in the possible future accordingly.

Real-life Examples of Iterated Prisoner's Dilemma

IPD has a broad range of real-life examples especially in international relations and trades.
The arms race is a typical example that manifests the structure of IPD[1]–[4]. For example, the Soviet-US arms race had been commonly modeled as an IPD game[5], [6]. In such scenario, the two choices are to arm or to disarm. Despite the choice of the opponent, arming is the more favorable option. However, both arming is a less desirable outcome than both disarming. As a result, arms race can be models and analyzed with the IPD framework.

In economics, international economics and trade war are other examples of IPD[7]. They all exhibit the same structure: defection beats cooperation, but mutual cooperation is more favorable than mutual defection.
Chapter 2 Literature Review

The Evolution of Cooperation

It is almost impossible to talk about IPD without mentioning Axelrod’s revolutionary work on the subject. His book *The Evolution of Cooperation* (*Evolution* for short)[8] systematically described the characteristic of IPD, extensively explored the successful and robust strategy of Tit-for-Tat (TFT) and its situational optimality, and used TFT to explain how real-life cooperative behavior emerged and spread. Other than an academic nova, the writing was clear and accessible to a broader audience. It successfully brought the scholastic topic to people’s everyday life, not to mention demonstrated how computation and simulation could vastly contribute to social sciences. This study is based on the insights of Axelrod’s book so that we will cover its central ideas.

The book *Evolution* serves as a summary of his computer tournament. To find the best strategy of IPD, Axelrod invited experts in psychology, economics, political science, mathematics, and sociology to submit programs of IPD strategy. 14 strategies enrolled in the tournament, together with RANDOM - a strategy that has 50-50 chance between cooperate and defect. It was a round-robin tournament wherein each strategy competes against all other strategies, as well as a copy of itself. The IPD game they played used the standard payoff of 5-3-1-0, had no discount parameter, and lasted approximately 200 turns for the encounter.

The crown of this tournament was surprisingly won by the simplest program of all. It is a strategy called Tit-for-Tat (TFT) submitted by Professor Anatol Rapoport of the University of Toronto. TFT starts with cooperation, then does exactly what the opponent did in the previous turn. It means when interacting with a strategy that always cooperates (All-C, not enrolled in the
tournament), TFT behaviors like “always cooperating”(All-C) and they each gain a chain of 3 points for rewards. When interacting with a strategy that always defects (All-D, not enrolled in the tournament), TFT would suffer the first sucker of 0 points, and retaliate till the end, which is 5 points short of its opponent.

The result of this tournament revealed something unpredicted. For example, the one feature that decisively determined the performance of a strategy is nice, which means to start with cooperation and never defect until being betrayed. All nice strategies performed amazingly well in the tournament, in fact, the top eight were all nice strategies, and the rest were all non-nice. However, this cannot top the surprising fact that the simplest strategy of all won a tournament, which is unprecedented. Traditionally it was the complexity of a strategy that leads to excellence, not simplicity.

From the tournament, people also realized all strategies depend heavily on the demographics of the tournament. For example, to imagine a tournament participated mainly by non-nice strategies, nice strategies like TFT may have a difficult time to find cooperative opponents to reward its benignity. Was it only fortuitous that TFT won because nobody expected such a simple strategy? With this question in mind, Axelrod conducted the second tournament. One significant difference is all participants in the second tournament got a detailed analysis of the first one, which enabled them to study the winning strategies and submit countering ones.

The second tournament received programs from a wider ranger, 62 strategies were accepted. TFT enrolled again as the shortest program and won again. It was when everyone knew TFT’s exceptional performance and tried to outwit it. Some contestant seemed quite intelligent and tricky. For example, TRANQUILIZER, which behaved like TFT initially - nice yet provokable.
However, once it succeeded luring the opponent into a mutual trust relationship, it would occasionally try to defect and observe its opponent’s reaction. If its betrayal went unpunished, TRANQUILIZER would defect even more frequently to exploit the benignity and gullibility. However, it still failed to dethrone TFT.

After the second tournament, the optimality of TFT seemed established. However, based on the competition system of the tournament, one question remained: was TFT robust (which means it would perform well in a variety of environment)?

The method of evolutionary game theory is used to answer this question. It simulated the dynamic evolution process of strategies as if they are different species: in a shared population, all strategies started with a same percentage of the whole. However, after each generation which was substantially equivalent to a single computer tournament, their population ratio changes according to the performance. The successful strategies would have more “offspring” in the next generation, making them more prevalent and the unsuccessful ones would gradually go extinct.

The simulated population is shown in the graph below (image from *The Evolution of Cooperation*).
Figure 1. The dynamic population in an IPD tournament

The evolutionary process provided new insights of the dynamics between strategies. The curve labeled 8 in the graph above represents the strategy of HARRINGTON, the only non-nice strategy in the top 15. In the first 200 generations, its population increased dramatically along with other successful strategies. However, the difference is that HARRINGTON, as an exploitative strategy, is expanding at the expense of the exploitable population. It was not a sustainable increase. When the gullible strategies became extinct, the very foundation of HARRINGTON’s success was eroded by itself. So after 200 generations, HARRINGTON peaked and began to plummet, and finally at around generation 1000, vanished. As quoted from *Evolution*, “the ecological analysis shows that doing well with rules that do not score well themselves is eventually a self-defeating process.”
In the evolutionary competition, TFT, unsurprisingly, won again. So why is this simple strategy so successful? Axelrod summarized its predominance as four properties: nice, retaliatory, forgiving, and clear.

Unlike non-nice strategies, nice ones are highly compatible with themselves, giving them a substantial collective advantage. Retaliatory means being able to punish the opponent’s defection without which a strategy cannot survive in a hostile environment. Retaliation also impelled its responsive opponent to go back to cooperating for its own sake, which in turn protected the benefit of TFT. Forgiving means being willing to restore mutual trust as long as the opponent showed signals of confession, forgiving would maximize the ratio of mutual cooperation in all turns. The last one, clear, is particularly important. People tend to regard complex strategies as more competent; this assumption is generally correct in zero-sum games like chess, where complexity can conceal one’s intention and use unpredicted moves to strike the opponent. However, in a non-zero-sum game like IPD, it is wrong to predict the players assuming they would try their best to undermine opponents’ benefit. In IPD, complicated strategies, as opposed to TFT’s direct reaction to each move, would cause opponents’ cooperation go unreciprocated, or defections go unpunished. This confusion in turn seriously undermined the opponent’s will to establish mutual trust. When maximizing a strategy’s complexity, it would seem to the opponent as unresponsive and unpredictable as a random strategy. Moreover, when dealing with random strategies, always defecting is the optimal choice. So complexity induces the opponent to defect. By contrast, the simplicity of TFT made it highly recognizable, and as long as opponents identified it as TFT and realized their actions would inevitably be mirrored back at themselves, they were dissuaded from defection.
The book also discussed the definition of “invade” and “stable.” If in a population of strategy A, a newcomer with strategy B, by interacting with population of A can get higher score than the native population by interacting with themselves, B is said to be able to invade A. For example, in a population of All-C strategy, let discount = 1 and IPD last 200 turns, the native population’s expected payoff is 3 * 200 = 600. However, a newcomer of All-D will interact with All-C and accumulate an average score of 5 * 200 = 100, so All-D can invade All-C. Meanwhile, If there is no strategy can invade strategy A, it is called collectively stable. For example, the strategy of All-D is collectively stable. Because any strategy other than All-D would have at least one move of cooperation, but in a population of All-D, this move would get no reward and thusly lower the expected payoff. This conclusion seems very depressing, but All-D is not impregnable. When a group of cooperative newcomers instead of one try to invade a population of All-D, it is possible.

Other Literature

IPD currently is an interdisciplinary topic in a variety of domains. We will briefly discuss some major extensions of the traditional IPD framework relevant to this study. Spatial interaction is one of the branches[9]–[12]. It usually positions individuals onto a lattice-based world, and make them play with IPD with Von Neumann neighborhood (four direct neighbors) or Moore neighborhood (four direct and four diagonal neighbors). Then by continually adapting to better rules, strategies will form spatial patterns to represent community or territory. Spatial IPD focuses on the characteristics of the patterns emerged from such interactions.
Another variation is IPD with choice and refusal (IPDCR). The main difference of IPDCR from IPD is that each can choose whether or not to interact with another based on previous memories. It reflects the fact that in social life, once the infamy as a defector or exploiter is established for an individual, others would avoid interacting with him/her to protect their interest. This refusal is even applicable to other members of the same strategy. Kitcher[13], Stanley[14], and Ashlock[15] extensively explored this framework. Smuch[16] discussed the social network emerged from such choices.

Noise is another topic. It is defined as the form of random errors in implementing a choice in real-world interactions[17]–[19]. As an overall robust strategy, TFT’s performance in noisy environments rapidly decreases, leaving room for further improvement.
Chapter 3 Methodology

This paper mainly uses two methodologies, namely agent-based modeling and social network analysis. We will discuss them separately and then explained why they are used together.

Agent-Based Modeling

Agent-Based Modeling, or ABM, is a methodology that uses computational models to simulate the entities in a given system and studies this system by observing the entities’ interactions and resulting patterns. These entities are individually represented in the model, each with their properties, actions, rules to interact, and sometimes purposes. These features give the entities the sense of being autonomous, and they are thusly called agents. However, this does not necessarily mean ABM is only for systems consisted with sentient individuals, as long as we can explicitly express the laws governing the actions of said entities, like the Newtonian mechanics in a particle system, they can be modeled with ABM.

ABM as a modeling method, differs from the traditional mathematical model, or equation form, in some ways. We can consider the example of a simple predator-prey model, where prey feeds on renewable resources provided by the environment, and predators feed on prey. The dynamic change of population of both species is well studied and often described in the following Lotka-Volterra equations:

\[
\frac{dx}{dt} = ax - \beta xy \\
\frac{dy}{dt} = \delta xy - \gamma y
\]
For the sake of concise, we will skip the definition of terms in the equation. Moreover, even without definition, we get the basic sense that the population of two species is a nonlinear, differential equation based on each other. One advantage of the equation-based model (EBM) is we can graph variables based on the equations, giving an intuitive representation of how things evolve in time. The predator-prey model can be graphed as follow. From the graph, we can see both populations exhibit oscillation with a phase difference, and in general, this oscillation would last forever with dynamic peaks.

![Graph of Lotka-Volterra equations](image)

**Figure 2. The graph of Lotka-Volterra equations**

Here we give a general idea of how ABM handles this phenomenon and leave the technical details for later. We create an environment (the grassland) with parameters (the growth rate of grass), we create a group of sheep with properties relevant in this predator-prey scenario, as well as a group of wolves. We program them how to interact with other agents (eat, reproduce, hunt), then begin the simulation and observe how these agents evolve in this artificial world.
From the simple explanation above, we can see some key features of the two methods. For equation-based models, individuals from the same species are regarded as identical; species population is a numerical description (rather than an actual count) and the interactions are continuous, and we can determine the population of any given time as long as the initial parameters are given. We can summarize these three key features of EBM as being homogenous, continuous, and deterministic. They also mark the distinctions between EBM and ABM, wherein ABM can be described as heterogeneous, discrete, and indeterministic.

- **Heterogeneous**

In ABM, every agent is instantiated with a computational entity. These entities can have a different range for the same property, for instance, there can be fast and slow (it can be multivalued) wolves, old and young sheep. Moreover, they will behave and interact differently accordingly.

- **Discrete**

Discrete in twofold in ABM. Populations are discrete because we tally the number of each species to the whole, so the population can only be a nonnegative integer, whereas the range of populations in EBM is positive real. Second, since ABM is mainly carried out on computers, the interactions in ABM is typically calculated stepwise, in time ticks.

- **Nondeterministic**

In ABM, nondeterministic is also twofold. When we instantiate the whole simulation, the properties of agents, (e.g., the speed of wolves) can be a random variable based on a given possibility distribution function, so every time we reset the model, the individuals will be different. After we created all the agents, their actions can also be decided in a probabilistic
manner. As a result, every run of the model will lead to different result. It is the general pattern emerged out from the seeming chaos blob of randomness that ABM cares about.

From the comparison above, we can see some significant difference between ABM and the traditional EBM. Believing that both model method has their unique merits, we will not give a generalization which one is better but discuss which one is more suited for a given scenario. Moreover, we should first address some drawback of ABM. First being the computation constraint, because every single agent in a system is stored and continuously calculated, when the population is high it can be very computationally intensive. Moreover, when the number of constituent elements in the simulated system comes to this scale (typically reaching an order of magnitude of $10^6$), EBM would be generally more feasible than ABM. Second is the level of detail. Comparing to EBM wherein a wolf is but a number, the virtual wolf in ABM with lots of parameters are more realistic. However, this is a relative term. How realistic does the model have to be? How many properties should modelers take into account for the simulation? These can usually be a troublesome question with arbitrary answers. A model tries to capture every little aspect of the real world would be stuck in the bog of details and fail to produce any meaningful insight of the phenomenon modeled.

However, comparing to EBM, when the system modeled is made up by heterogeneous constituents that interact in a discrete and non-deterministic way, ABM has some advantages due to the fact its basic assumptions agree more with the universe of discourse. We can naturally reach the observation that ABM fits very well in social science, wherein the majority of the subject is a heterogeneous population behave in a non-deterministic way.
Despite the fact that ABM is particularly suitable for social science studies, it has not been very long since ABM is widely adopted in the field. Initially, ABM had its root in computer scientist von Neumann’s idea of cellular automata, wherein every cell in a grid is an automaton that follows a series of simple instructions and interacts with the neighboring cells accordingly. Gradually, ABM found its place across computer science, cybernetics, complexity theory, game theory, and artificial intelligence. The first trial to use the idea of ABM in social science is conducted in the 1970s by the economist Thomas Schelling. In his research of how individual preference of neighbors can lead to global segregation, Schelling used coins on a checkerboard to represent individuals and individually calculate whether they want to relocate to a place where more neighbors of the same group are present. Due to the technical constraint of the time, this was done without a computer, but it already manifested many of the critical features of contemporary ABM: concurrent processing, decentralize decision making, and emerging global pattern. After that, ABM has been applied to more and more domains in social science, which we will talk about in more details in the literature review part.

In recent year, ABM has been a very exciting paradigm for many disciplines. With the growth of human knowledge and data, the problems people face are more complex than ever before. While traditional modeling methods like EBM embraced a top-down mindset, ABM is by contrast bottom-up. All global result in an ABM, like the segregation pattern in Schelling’s model, is a result of the local interactions of individual agents. Even starting from a set of elementary rules, with substantial iterations, it can generate highly complex patterns. Like Conway’s Game of Life, with two lines of rules to govern how each cell act, it is enough to create a vast world of endlessly drifting, interfacing, and changing patterns. One of the core assumptions behind
complexity theory is many complex phenomena can be explained by the interaction of simple local rules. Due to the non-linear nature of such systems, one cannot easily predict what global pattern would emerge. Moreover, the number of interactions within agents become too large to trace and capture into a set of equations. Hence traditional modeling method is having a hard time simulating highly complex systems and make accurate predictions as they do for non-complex phenomena.

Another essential property of complexity also gives rise to ABM, namely emergence. An emergent property in a system is the one that would only manifest on a certain level, but not predictable from levels below it. For example, no amount of knowledge about combustion engine is enough to explain traffic jam. On the other hand, emergence can usually be studied through the interactions of entities, which is precisely how ABM represent the universe of discourse. So, it is no surprising ABM is widely used in areas where emergency properties play a crucial role in a system.

Also, thanks to recent breakthroughs in artificial intelligence and automated reasoning, it becomes theoretically and computationally possible for the numerous agents in a simulation to have a “mind.” Interactions in a simulation can be easy as “does this cell have two or three living neighbors? “ in Game of Life, but can also be as sophisticated as “based on today’s index, what is the best choice for the stock market.” Currently, an agent in ABM is already able to adjust its strategies, not only behaviors, to the situation, and we call these agents adaptive. A system consists of adaptive individuals is called Complex Adaptive System, or CAS. While other paradigms are hard to capture the evolving aspect of the said system, ABM is also suitable for
CAS because between every tick we can perform computation and update the agents in our simulation, effectively making them adaptive to the environment and other agents.

By now, we can see that ABM is new kind of modeling paradigm that focuses on the individuals and their interactions.

**Social Network Analysis**

To understand social network analysis (SNA), we can start from social networks, which can be defined as a set of nodes connected by one or more types of social relations. SNA, in contrast to the methods that focus on the attributes of these individual actors, is a structural analysis that focuses on their connections and interactions instead of individual properties.

As ABM, SNA is a relatively young method. For a long time, when it comes to empirical data collection and analysis for social science, sample survey is the dominating method. As both natural and social science shifted from reductionism to a systematic view, people realize there are aspects in social behavior and phenomenon that cannot be explained by examinations of the individuals, no matter how thorough and exquisite the examination can be. As pointed out by Allen Barton fifty years ago, “using random sampling of individuals, the survey is a sociological meatgrinder, tearing the individual from his social context and guaranteeing that nobody in the study interacts with anyone else in it.”
**Nodes and Links**

There are two essential elements in a social network: nodes and links. Although usually represent individuals, nodes can stand for practically any type of entity because they are represented with little information of individual attributes. However, this is not to say all nodes in social networks are high-level abstractions that are completely interchangeable. A friendship network among individuals and one among nations, even if they have the same structure, would have distinct features because of the actors’ different nature.

While the nature of nodes in a social network has few theoretical constraints, the links are relatively more formalized. Borgatti [20] categorized the links in a social network into four types: similarities, social relations, interactions, and flows.

Similarities are the links between nodes that share attributes. In social networks, structural similarities are more frequently studies and examined. For example, in a bimodal network of 100 people attending ten clubs, if person A and B go to precisely the same clubs, they are structurally similar and have the similarity link (although the link between A and B might be implicit in the network and can only be revealed by mathematical analysis).

Social relations can be based on kinship (mother of) or other social roles (friend of). It can also represent affective (likes) or cognitive (knows) relations.

Comparing to social relations, social interactions are more behavior-based, like “help,” “talk to.” As we can see, the demarcation between these and affective relations are somehow fuzzy, and in practice, they have indeed been used interchangeably.

Flows, to quote Alexandra Marin, “are relations based on exchanges or transfers between nodes.” It usually refers to the transfer of information, belief, or other resources.
Structural Focus

SNA research may take in many different forms, but one thing they have in common is to shift focus from within individuals to between them. SNA is a manifestation of the belief that social behavior and phenomenon is mostly shaped by relations and patterns, which is to say, many phenomena can be explained through the way entities are structured, regardless of the type and nature of entities. For example, the outbreak of epidemics and the spread of innovation, though they differ entirely if we are looking at the actors or the transmission media. However, due to a similar manner how individuals are structured in these two networks, the spread phenomenon share a lot in common, like the penetration speed and the susceptibility of different patterns. Even when it comes to the same phenomenon, SNA will yield completely different insight than individualism analysis. For example, to understand the performance of an employee in a big corporation, the individual-oriented method would collect data about the employer’s skill set, working habit, previous working history, health and mentality and so on, to explain specific pros and cons in his or her work performance. The belief is the performance is mainly decided by the attributes within the employer. It also implies if switching to a similar position, the employer would have similar performance because the main factors remain.

SNA would, by contrast, collect data on the employer’s links with other units in the system. These links include interpersonal ones like “seeking guidance from,” “have lunch with,” as well as ones between individual and collectives, like “get informed from mailist A,” “involve in the project of team B.” They believe it is the position, surrounding, and the linking relations that mainly shape the employer’s behavior, or, the deciding factors are between instead of within.
The implication is, if the position switches an employer that remains the similar social ties, the employer would have similar performance.

Focusing on different aspects, the two methods are not entirely incompatible. An important factor in deciding the relevance of the two methods is scale. In an arm-wrestling, ignore the physical strength of a player is preposterous, as absurd as using sheer aggregated physical strength to explain a 100 on 100 game of tug of war. As the scale goes up, individual attributes play a less and less relevant role in deciding the outcome. It is partly the reason why SNA is relatively new and thriving recently. New technologies (especially Internet) continued to enable a larger number of individuals to engage in the same system, a collective behavior like Wikipedia is unimaginable 50 years ago, let alone systematic analysis for such behaviors.

**The Combination of the Two Methods**

As we talked about, ABM and SNA both have their origin in system science and complexity, they both value the principle that to understand the behavior of a system, it is not enough (or even relevant) to examine the properties of the constituent units. So they share a theoretical foundation.

With roots deep in graph theory, SNA is a computational method. After collecting and mapping out the network structure, a full range of quantitative metrics are calculated and examined. ABM is by nature a computational model as well, all information is handled and exchanged quantitatively. It also promotes the compatibility of the two. Their combination has been blooming in recent years, cover the domains of sociology [21], market research (Lee et al.,
2011), social epidemiology (El-Sayed et al., 2012), social media (Nasrinpour et al., 2016), operational research (Baber et al., 2013) and so on.

With system science at the core, SNA and ABM have a different perspective of approaching systems. ABM focus more on the temporal interaction between agents; it observes the system’s behavior and evolution through time. In one of his most famous quote, ABM herald M. Epstein said: “If you did not grow it, you did not explain it.” From this quote, we can see the emphasis on the word grow, which has the intrinsic connotation of “evolve in time.” However, the time axis is not essential for SNA; the structure manifests in a fixed timeframe. For chronological studies in SNA, researchers take snapshots of the network and analyze them accordingly.

Also, and agents in ABM are treated relatively homogeneously. In ABM, usually, it is the local interaction rule generating the resulting pattern. In SNA, nodes and links are more distinct not in the sense of individual attributes but their structural features. A bridge has a higher impact than other links, nodes positioned at the center of the network are influenced earlier in an epidemic, and so on.

As a result, ABM and SNA analyze a system from different perspectives while sharing the same set of principles. Moreover, we consider them a complementary duo to uncover broader and deeper facts and insights from a system.
Chapter 4 Model Explanation

In this part, we will go through the model. We start by introducing the main elements and platform of our model. Then we will discuss the process of simulation and justify some major design choices. Finally, we will examine a typical experiment cycle to generate and analyze data.

Interface Introduction

In this part, we will dissect the visual parts of the model and briefly introduce them. The interface of the model is shown as below.

Figure 3. Simulation interface

The main monitor in the middle is the dynamic visualization of the simulation. In the current screenshot, it is a social circles network 1,000 nodes. A line denotes a link between two nodes,
which are color coded to present strategies. A link is grey if its two ends are of the same strategy, and red when it connects two strategies, indicating possible adaption.

The buttons and slides on the left are controllers to determine a specific variable (e.g., the total number of nodes) or to carry out a command (e.g., ask the top 10% nodes with closeness centrality to change their current strategy to TFT). The monitor and plot on the right show global metrics (e.g., Average clustering coefficient of the current network) and track population change.

**Platform Introduction**

We built this model with NetLogo, a multi-agent programmable modeling environment developed by Uri Wilensky. As all ABM simulating environment, it is particularly suitable for simulating complex systems where heterogeneous agents interact with each other in a temporal way. Besides that, NetLogo has other traits that suit our purpose of the study:

**Intuitive Visual Presentation**

Being in the lineage of the graphical Logo programming language family, NetLogo is no outliner when it comes to intuitive and vivid visual presentation. In NetLogo, the major model elements are turtles. The name *turtle*, another inheritance from the Logo language, is a term for the agents that the programmer can directly control. Like the class in Object-Oriented Programming (OOP), turtles have data stored as properties, and functions stored as methods. Properties represent attributes of the turtle germane in the model. In our case, the agent's linked neighbor, its current strategy, its accumulated score, its spatial information are all properties. The methods are the behavior these agents can perform with themselves, with other agents, or with the environment.
For example, in our model, the agents can engage each other in an IPD game, can calculate its outcome from the games, can observe and compare its strategy with its neighbors, and can copy and adapt the strategy of its most successful neighbor.

The interface of NetLogo consists three tabs, one being code which is similar to standard Integrated Development Environment, code is the place programmer put into commands and data. The tab Info serves as a comprehensive instruction and documentation to the model. The third tab, namely Interface, is the most important feature that sets NetLogo apart from other programming languages. It translates the command in the code part to visual objects and shows their interaction and evolution frame by frame. During the process of simulation, we can dynamically change the visual properties (e.g., color, shape, size, orientation, location) of the agents according to their attributes. For example, we can assign colors according to their current IPD strategy, size proportional to their accumulated score, and placer agents with the same strategy into cliques spatially. All these visual elements contribute to the understanding of the simulated process in an intuitive way.

**Network Extension**

NetLogo also supports a series of extension to expand the capability and compatibility of the simulation. A particularly useful one for our research is the NW Extension, which supports many of the most common network types, including small-world network, scale-free network, and random network. The extension can generate the types above of network with adjustable parameters, like the “rewiring percentage” in a small-world network, which is very helpful when we are analyzing the effect of different factors on the optimality and evolution of strategies.
**Randomness**

In general, randomness in models can cancel out the effects of contingencies and ensure the robustness of results.

Randomness in NetLogo is threefold. First being all properties and parameters in methods can be randomized, and it supports a variety of Probability Distribution Functions (PDF). For example, we can use a normal distribution to generate the height of a thousand individuals or an exponential distribution of their wealth.

The second layer of randomness relates to the data structure of NetLogo. Anytime we call more than one agents to perform the same action, they are dynamically grouped into a data structure called agentsets. For example, when we ask the thousand simulated people to move one step forward, NetLogo calls the whole agentset, and every time the agentset is called, the sequence of which individuals are called is always randomized. It ensures no individual in the set has any priority (unless desired) over others and protects the result from unintended influences resulting from a fixed sequence.

Lastly, NetLogo has a feature called random seed, which can manually designate the random number generator to regenerate all the randomness in a specific run. So when we observe a particularly interesting or abnormal pattern, we can remake the initial structure and distribution of the network, then run the process with different parameters. It nets us with certainty in randomness, and make result comparable to each other because of an identical initialization.
**BehaviorSpace and Data Analysis**

NetLogo has an integrated tool called BehaviorSpace that enables NetLogo run the simulation in experiments. An *experiment* in NetLogo is a specific combination of all possible variables, options. BehaviorSpace will run such *experiments* multiple times without manual interference and output the measurements we required to a file for further data analysis. In BehaviorSpace, we can easily sweep across the parameter space to detect the impact of different factors on a variable. For example, we can tweak the number of neighbors in a small-world network, while maintaining all other parameters, to see how the does it influence the resulting pattern of strategies. BehaviorSpace also supports running *experiments* in batches, which means a large number of repetition for each variable combination, to get a more precise result and detect the variance.

NetLogo can further combine with the R. R is a programming language for statistical computing that is widely used in data science. By running the models of NetLogo in R, we can perform sophisticated and complex data analysis based on the data we generated from NetLogo, which means examining simulation results more systematically. For this paper, we are focusing on the general impact of different variables and network structures, so we do not use R in such a way. However, the combination opens a variety of possibilities for future research.

**Model Explanation**

*Main Process of the Model*

A complete run of the model consists the following phases:
1. Network Generating

We start with a given population and network parameter. We assign the population typically from 1-3000, and select a network structure for the current run, including “small-world network,” “scale-free network,” “random network.” Different network structures are usually used to represent different social construct. For example, we can generate a scale-free network with 2,000 agents to model the structure of a subgraph of the Internet, or we can generate a 200 agents small-world network to represent a close-woven community.

2. Strategy Initialization

After the agents are generated and linked, we initiate them with strategies to answer a particular inquiry. For example, to check does the conclusion in Axelrod’s research that a 5% population of TFT is about to invade an All-D territory still hold in networks, we can initiate all agents with 5% TFT strategy, and the rest 95% with All-D. To see how an equal amount of 4 strategies coevolve on a network, we can initiate each quarter of agents with a different strategy.

One thing worth noting is that the percentage is for each individual instead of the collective. Take the 5% TFT as an example, it means each agent has a 5% possibility to be TFT and 95% possibility to be All-D, instead of 5% of the whole population is TFT. We choose this individual possibility to avoid collective bias when rounding up or down a non-integer population, and we run each setting more than 100 times to average out random errors.

3. Strategy Interaction

Then the model will run by time ticks. In each time tick, all the agents will play an IPD game with all its linked neighbors. The agent accumulates its points from the games at the
end of a time tick, divided by its degree to get the average score, which represents the relative fitness of its strategy in its surroundings.

For the IPD game, we also have many variables. Including the terminate possibility of each turn, the discount parameter, and the level of noise. We will discuss them in detail in the data analysis section.

4. Strategy Evolution

When all agents get the average score at the end of a time tick, each agent will observe all its linked neighbors. If there exists a linked neighbor that has a higher average score than itself within this time tick, it will mimic the strategy of that neighbor and adapt to it in the next time tick. If multiple neighbors outperform itself, it will choose the neighbor with the highest score and adapt the strategy of that neighbor.

So after every time tick, all agents in the model tend to shift to a “better” strategy. However, the optimal strategy can be a relative choice due to a node’s place in a network, as well as the strategy its neighbors are choosing. So there are times we can observe an oscillating pattern between strategies. However, the general trend is an evolutionarily stable state that the population of strategy differs from their initialization.

As a result, from a complete run of the model, we can see the evolution trend of different strategies quantitatively.
Main Process of BehaviorSpace

The section above describes one run of the model. To gather aggregated data and analyze the general trend, we use BehaviorSpace to run the model multiple times. A typical experiment is shown as below.

Figure 4. BehaviorSpace interface

In this experiment, we are focusing on how the number of nodes in a scale-free network influence the resulting cooperative percentage. So we are locking all other parameters, but make population sweep from 5 to 2,000 with an increment of 5. For each population value, we run the model 100 times and record the ratio of cooperative strategies $p$ at the end. Then we analyze the relationship between population and $p$ with the data collected.
Network Metrics and Network Structure Models

In the experiment, we are focusing on three network models, namely scale-free network, small-world network, social circle. They represent different network structures found in real life. In this section, we will discuss their mathematical feature as well as social implications. However, to scientifically describe the characteristics of the models, we should first talk about some metrics for networks. SNA comes with a myriad of metrics, to describe and analyze networks from different perspective and context. Here we only introduce some of the basic ones germane to our experiment.

Undirected Network

An undirected network is one that can be represented by an undirected graph. In undirected graphs, in contrast to directed graphs, nodes are connected by bidirectional links. Which is to say, there is no difference between the two joint ends. Bidirectional links are usually used to symbolize mutual relationship in social life. For example, “is classmate with” is a bidirectional relationship, to say A “is classmate with” B is equivalent to say B “is classmate with” A. As opposed to a directional relationship that has a source and target, which distinguish the two nodes. Like “is the father of.”

In our model, links represent “is a neighbor of” or “plays IPD with,” which is bidirectional, so all the networks we generate are undirected.

Degree

Degree is the number of links a node has. The degree of node A is usually denoted as deg(A).
In our model, in every time tick, each node A plays an IPD game with all its linked neighbors (there are $\text{deg}(A)$ of them) and gets the accumulated score $N(A)$. Since $N(A)$ is the aggregated score from $\text{deg}(A)$ games, the average performance of node A in this time tick is $\frac{N(A)}{\text{deg}(A)}$, we use this as an index of fitness and decide whether they adapt other strategies in the next time tick.

**Average Path Length**

Average path length is defined as the average length of the shortest paths between any random pair of nodes in a network. Different with the degree we just talked about which is a metrics for individual nodes, average path length $L$ is a global measurement for the whole network.

In the network below, there are 6 possible dyads. For dyad AB, BC, BD, CD, the shortest path length is 1; for dyad AC, the shortest path length is 2; for dyad AD, the shortest path length is 2 (though path ABCD’s length is 3, path ABD is length 2, we only consider the shortest one here). So $L = \frac{1 + 1 + 1 + 1 + 2 + 2}{6} = 1.33$

![Diagram of network](image)

**Figure 5. Average path length example**

Average path length describes the general accessibility and interconnectivity of a network. Take computer networks as an example, the common slogan “is only one click away” implies the shortest length of 1, which means highly accessible. However, maybe in some clunky
government site, it takes the user at least ten clicks to find some specific info pages from the main page, then that network has a high $L$, and is poorly connected.

**Clustering Coefficient**

Mathematically, clustering coefficient $C$ measures the possibility that a randomly chosen pair from the neighbors of node A to be neighbors themselves. It is also an individual metric for each node.

In the network above, node C has only two neighbors: A and D, which are neighbors. So there is 100% chance that a randomly chose pair from the neighbor of A to be neighbors, and we denote $C(C) = 1$. For node B, it has three neighbors A, C, D, so there are three possible dyads among its neighbors. However, only one pair of them, namely CD, is neighbors. So, $C(B) = 0.33$.

In a social context, nodes with high $C$ means their surrounding neighborhood is highly connected and vice versa. For example, in the setting of a classroom, hopefully, everyone befriends with everyone else. Then for a single node A, almost all its neighbors are also neighbors themselves, and A has a clustering coefficient close to 1. However, in real-life, large-scale network usually has a very low average clustering coefficient, for example, the mean $C$ of the Internet is 0.035.

**Degree Distribution**

Degree distribution is also a global measurement of the whole network. Degree distribution is the probability distribution of all possible degrees in a network. We still use the network above as an example, the possible degrees are 1(A), 2(C, D) and 3(B). The fraction $P(k)$ of nodes in the network having k links can be described as: $P( k=1 ) = 0.25 \ (1 / 4)$, $P(k = 2) = 0.5$, $P(k = 3) = 0.25$. So the probability distribution can be plotted as below:
The degree distribution is one of the most characteristic measurements of a network. Different networks in real-life, as well as the network models we will talk about next, have distinct degree distributions.

**Scale-free Network**

There are different ways to describe a scale-free network. From the degree distribution perspective, a scale-free network is one that the fraction function \( P(k) \) can be approximately described by the equation: \( P(k) \sim k^{-\gamma} \) where \( \gamma \) is typically between 2 and 3[22]. It means the possibility of a node in the scale-free network to have \( k \) links decrease exponentially as \( k \) increases. This descriptive definition of scale-free may be less intuitive and we can also look at it from a generative perspective.

We start with a network of only a pair of linked nodes: A and B. We add a new node C and attach it to a node with a possibility proportionally to its degree. Since \( \text{deg}(A) = \text{deg}(B) \), C has...
equal chance to attach to A or B. Let’s assume C is then linked to A, and we add another node D. At the moment, \( \text{deg}(A) = 2, \text{deg}(B) = \text{deg}(C) = 1 \), so D has a 50% chance to link to A, and 25% possibility to link to B or C respectively. If we keep adding nodes according to this rule, the result exhibits the Matthew effect that “the rich get richer,” and the resulting network is a scale-free network. Due to the way new nodes are attached to the existing network with a preference for the highly linked nodes, it is also called preferential attachment network. Below are some snapshots of this preferential attachment process.

![Figure 7. The preferential attachment process](image)

If we add new nodes and link them entirely randomly, the resulting network should have a normal distribution as its degree distribution. In scale-free networks, the distribution is highly right-skewed, and conforms to the so-called “power law.” The term “scale-free” was invented by physicist Barabási while examining the degree distribution of Internet. They found hubs that have degree unexplainable by a normal distribution, “almost as if we had stumbled on a significant number of people who were 100 feet tall, thus prompting us to coin the term ‘scale-free’”.

Researchers claimed many real-life networks to be scale-free[23]–[26]. On the other hand, scale-free networks have a fat-tailed distribution in degrees, means it can have extremely highly connected nodes. It contradicts with many social behavior-based networks, for example, the
friendship network, where links are not costless and take substantial resource and energy to maintain.

**Small-world Network**

Small-world network a type of network that exhibits two properties simultaneously: high clustering coefficient and low average path length. We use Watts-Strogatz model [27], [28] to explain the process to generate networks that have small-world property.

As stated in Watts and Strogatz original paper, the main idea is to adjust a rewiring parameter $p$ to tune the network between regularity and disorder. We start with $n$ unconnected nodes and position them in a circle, then connect each node with its $k$ neighbors clockwise and $k$ neighbors counterclockwise. If we do not rewire any existing link ($p = 0$), we get a regular ring lattice shown as the left in the picture below. In this network, clustering coefficient is relatively high (when $k = 2$, $C = 0.5$), but the average path length is also high ($L \sim \frac{N}{2k}$). If we imagine this to be the network of a community and links to be the relation “knows”, the community is fairly locally dense, in the sense that geographical neighbors tend to know each other. But for anyone to find a chain to connect to someone far away, it takes many steps, which is a form of low accessibility.
As shown in the image above (image from https://www.nature.com/articles/30918), we can begin to rewire the links with a possibility of $p$. As $p$ increases from 0 and approaches 0.1, short links in network emerge, and the average length path drops drastically, while the clustering coefficient remains high, this is shown in the middle graph in the picture above. When $p$ keeps increasing, the majority of local links are rewired, and the graph exhibit random. $L$ decreases but in a slower ratio than $C$, at this moment, the community is one that whether two people know each other has nothing to do with their geographical closeness. It is as possible for someone to know remote people as close neighbors as shown in the rightmost ring. The aim is to tune $p$ so that the network can enjoy the benefit of both low $L$ and high $C$, and this is achieved when $p$ is roughly between 0.1% and 10% as shown below (image from https://www.nature.com/articles/30918).
A more extensive range of real-life networks is claimed to manifest small-world properties\cite{29}–\cite{32}.

**Social Circles**

Social circles is a relatively new network model developed by Lynne Hamill and Nigel Gilbert\cite{33}, \cite{34}. The argument is that the basic network models we use currently do not match well against empirical observations of real-life networks. They summarized eight key characteristics that ideal large social network models should have. In those eight features, the small-world model does not have a fat-tail distribution of degree nor communities. Scale-free model lacks limited personal network size, high clustering, and short path lengths. Moreover, neither one has assortativity by the degree of connection\cite{35}.
So, they proposed a new model specially designed for agent-based modeling. The idea is based on the concept of social space and distance; it suggests if we map individuals onto a 2-dimensional space whose axes are defined by social measurements, on this social map, the proximity of individuals suggests social similarity[36] and social interactions[37]. Then, using the concept of social circles[38], every individual on this map has a social reach, indicating the maximum social distance they can establish links. Each has a circle centered on them with the radius of their social reach, demarketing a social space they can link to. For each pairwise individuals, if their distance is no greater than either of their social reach, which means they are within the social circle of each other, a link is added between them.

If all individuals share the same social reach, the degree of nodes tends to be a Poisson distribution[39]. The clustering coefficient is at least 0.39 (when two individuals are on the circumference of each other’s social circle). Most importantly, the assortativity of degree, which is a feature not fulfilled by any of the basic models, can also be achieved in social circles model. The model can be further extended to multiple groups of individuals with different social reach and population percentage. For example, in a social circle model, we can define three groups, with low, medium, and high social reach respectively, and with a different population. Then, the high reach group will change the degree distribution to a fat-tailed one. This multiple class structure adds variance and controllability to key measurements. We can manipulate the degree distribution, clustering coefficient, and average path length by tuning the relative social reach, and a population of groups, to better fit the desired network structure.
Model Structure Justification

There are two options for the overall structure of the simulation.

I. We can simulate everything on the fly in the simulation environment, including every choice of the agents on each turn, and then compute their scores from IPD game with all its linked neighbors.

II. We can simulate the IPD game between any two strategies outside the ABM and get an expected result for both players. Then simulate the interaction and evolution of multiple strategies inside ABM with the result as a reference.

The benefit of method I is we can change all parameters in real-time, even within one iteration of the simulation. The benefit of method II is that we offload part of the computation and avoid repetitive computation, which enables the ABM to handle much larger scale and more repetitions. We chose method II over I because in this research we are focusing on the evolution of strategies given different network structures and parameters, instead of the evolution given a shifting payoff matrix. Hence a changeable payoff is relatively irrelevant.

We calculate the payoff independently in another Java program we developed. The primary method takes in 5 parameters: strategy A, strategy B, discount parameter \( w \), terminate parameter \( t \), noise level \( n \). Then it simulates the interaction between strategy A and B (can be of the same strategy) in the scenario described by other parameters for 10,000 times, and output the mean score of both strategies.

Discount parameter \( w (0 < w \leq 1) \) works in such a way that the net gain is multiplied by \( w \) to the power of the current turn - 1. For example, for a string of mutual defecting, the score is \( w^0 + w^1 + w^2 + \cdots + w^N \). This represents the notion the present is more important.
Terminate parameter $t \ (t > 0)$ determines the possibility that an IPD game terminates after every single term. For example, let $t = 1,000$, then after each turn, we randomly generate a number between 1 and 1000 (inclusively) with a uniform distribution, if the random number generated is 1, we terminate the process. This guarantees no player knows in prior how many turns are there and being able to reason backward.

Noise parameter $n \ (0 \leq n \leq 1)$ represents the noise level in the process of communication. After the strategies choose their current action, we randomly generate a float number between 0 and 1. If it is smaller than $n$, the player will misinterpret the action chosen by the opponent, namely mistake C for D and vice versa. So, the bigger $n$ is, the higher possibility for errors to occur. It simulates the miscommunication happened in real-life scenarios.

We generate the payoff matrix in Java and load it in NetLogo. So as long as we are not simulating a dynamic IPD where the aforementioned parameters change within games, the pre-calculated scores are valid and so is the result. As for different scenarios (e.g., different behavior in high noise level), we recalculate the payoff and simulate with the new matrix. So, the simulation is still capable of handling different assumptions without sacrificing too much computation to simulate the interaction between strategies repetitively.
Chapter 5 Data Analysis

List of Strategies

All-C, a strategy that always cooperates, a.k.a. the selfish strategy
All-D, a strategy that always defects, a.k.a. the altruism strategy
Random, a strategy that has 50% chance to cooperate
TFT, a strategy that starts with cooperation, and then copies the opponent’s last move
GTFT: a variation of TFT. GTFT only retaliates after betrayed twice in a row.

Pavlov: a strategy that starts with cooperation. If the opponent cooperates, Pavlov will stay in the current choice (regardless of which it is). If the opponent defects, Pavlov will switch to the alternative[40].

In *The Evolution of Cooperation*, Axelrod pointed out, “and this (TFT outdo its always defecting neighbors) will be true even if only 5 percent of the interactions of the TIT FOR TAT players are with other TIT FOR TAT players”. However, this claim does not specify what topology are all these individuals in to reach the “5 percent of the interactions”. For example, if they are situated on a complete graph network, where everyone is connected to everyone else, this is equivalent to a 5% population of TFT. While in a lattice-based world where individuals are situated in a checkerboard and only interact with its four direct neighbors, this claim becomes “as long as 25% of the interactions (1 in 4 neighbors) is with other TFT players”. Obviously, it depends on the network topology to meet this criterion, so we will examine how to reach this “critical point” on different network models.
Another theme in *Evolution of Cooperation* is to find ways to promote cooperative behaviors, so is the purpose of this paper. As a result, we would use the percentage of enduring cooperation, p(EC), to measure the proliferation of cooperation. Moreover, examine the effect on p(EC) of different network parameters and conditions.

**The Effect of Population**

In this section, we will discuss how does the size of the network influence the p(EC).

*Scale-free Network*

For scale-free networks, we simulate a population of 5 to 2000 with an increment of 5 and run each population 100 times for a mean p(EC).

If we start with 5% TFT and 95% All-D as described in *The Evolution of Cooperation*, due to the low clustering coefficient of scale-free networks, few TFTs cannot find enough cooperative neighbors to sustain a higher score than All-Ds. So the p(EC) is relatively low as shown below. The first is a line chart with the average p(EC) as input; the second is a scatter chart showing all data points.
We can see from the plot that once the population reaches around 500, the population has no direct effect on p(EC). The initial increase in population is because of the way model generates strategies. When it has 5% TFT, it generates each node with a 5% chance to be TFT and 95% to be All-D. So at low population, it has a larger variance, which is shown from the scatter plot: on
the left side, there are high outliers. However, as the population grows, the initial number of TFT converges to 5% of the whole population, and the p(EC) has less variance.

Then we change the initial population to 50% All-D, 45% All-C, 5% TFT and 50% All-D, 25% All-C, 25% TFT. In these settings, there is 50% cooperative strategy to start with, so it is much easier for cooperation to spread. The plots are shown below. We can see the same trend as in the previous setting that once the population reaches a certain point (varied by initial population), p(EC) is no longer influenced by population. This conclusion agrees with the fact that scale-free networks are self-similar, in the sense that as long as the network model remains, one with a smaller population and one with a larger population exhibit similar behavior.

![Figure 12. Population effect on scale-free networks with 5% TFT and 45% All-C](image)
So for scale-free networks, when the size is large enough (usually about 500), population increase no longer affect the spread of cooperative behavior. The typical p(EC) is determined by the initial composition of strategies, cooperative strategies other than TFT (e.g., All-C) also contributes to the spread of cooperation.

**Small-world Network**

In small-world networks, cooperation spread much faster and more thoroughly than in scale-free network. We still start from 5% initial TFT and a standard small-world setting: rewiring \( p = 0.1 \) and connected neighbor \( 2k = 4 \) (connected to 2 clockwise and 2 counterclockwise). The plot below shows an asymptote to full invasion (100%) as the population grows.

**Figure 13. Population effect on scale-free networks with 25% TFT and 25% All-C**
Figure 14. Population effect on small-world networks with 5% TFT

If we keep all variables with the only exception of initial TFT population, we can see the more TFT to start with, the quicker it increases with population, and the faster it reaches saturation.

Figure 15. Population effect on small-world networks, with 5% and 10% TFT
From results above, we can see on small-world networks, a small percentage of TFT is enough to invade a population of All-D. Moreover, the invasion percentage increase with both initial TFT ratio and total population: it is easier to spread cooperation on larger scale networks. However, non-responsive cooperative strategies (e.g., All-C) have a very limited effect on the proliferation. This graph compares different initial compositions. The similarity between the blue curve and the green curve shows 5% TFT can do essentially the same as 5% TFT with 45% All-C. So the extra 45% All-C in the green curve does not affect the purpose. It also supports Axelrod’s proposition that TFT is the most effective in promoting cooperation in selfish population.

Figure 16. Population effect on small-world networks, with 5%, 10% TFT, and 5% TFT plus 45% All-C
The Effect of Network Structure

Now we examine how different parameters of small-world network influence the spread of cooperative strategies. From the discussion before, we know a small-world network can be described with two variables: connected neighbor number $k$ and rewiring percentage $p$ where $k$ reflects local density and $p$ stands for global reachability.

We simulate with $p$ from 0 to 0.12 with an increment of 0.04, and $k$ from 0 to 10 with an increment of 1. For each combination of $p$ and $k$, we test four different population level: 100, 400, 700, and 1,000. The reason we only simulate with 1,000 agents is from the last section we see that on small-world networks when the population exceeds 500, it is relatively easy for cooperation to spread even with a small initial population. The simulation results are shown below:

![Figure 17. Neighbors parameter effect with $p = 0$](image)
Firstly, we see the same result as the section above: when the population is high enough (the silver curve is 700, yellow 1,000), p(EC) is fairly high. If we lock $p$ and change $k$ as the variable, we see the number of connected neighbors has an obvious effect on p(EC), especially when $k < 4$. When $k$ is at 3, which indicates a clustering coefficient of 0.6, the small-world network is most suitable for cooperative strategies to spread. Further increase in $k$, or in clustering coefficient, doesn’t increase p(EC).
Alternatively, if we lock $k$ and change $p$ as variable, no noticeable effect on $p(\text{EC})$ is observed. From the comparison, we can see to increase cooperative behavior on small-world networks, the most effective method is still increasing the network size to approximately 500, then the marginal benefit of network size decreases. Another useful method is raising local connectivity ($k$), achieving a clustering coefficient between 0.5 and 0.6 ($k = 2, 3$ respectively). In the small-world lattice experiment, rewiring parameter $p$ catalyzes all the “magical” properties of small-world. However, in the scenario of promoting cooperative behavior, distant reachability ($p$) is not as important as local connectivity ($k$).

**The Effect of Noise Level**

TFT is generally a very robust strategy because it can adequately defend against exploitative strategies and cooperate with other instances of itself. However, it has a critical weakness: vulnerable to a noisy environment. A noisy environment is one that a strategy’s choice can occasionally be misrepresented and led to the opposite outcome. It represents in real-life communication no information is guaranteed to be completely accurate. Different strategies suffer or benefit more or less from noise, yet TFT’s performance dramatically decreases in a noisy environment. It is because when two instances of TFT interact, in an ideal environment, they will cooperate till the end. However, in a noisy environment, once an error occurred, this error would be interpreted by the other TFT as defection, and it will retaliate, and this retaliation would further be regarded as defection, thus resulting in an everlasting chain of alternating
defects, until another error occurs and break them free. So the performance between 2 TFT instances would decrease quickly as noise level increases.

To solve this problem, different variations of TFT were developed, like GTFT and Pavlov. GTFT would only retaliate after two consecutive betrayals. This extra tolerance protects GTFT from falling into a deadlock of defection caused by noise. The graph shows the relative performance within TFT and GTFT at different noise levels.

![Graph showing performance decrease in noisy environments with payoff matrix 5-3-1-0](image)

**Figure 20. Performance decrease in noisy environments with payoff matrix 5-3-1-0**

From the picture, we can see TFT’s performance plummet to 80% even at merely 2% noise, while GTFT is very robust against the turbulence. The decrease of TFT can be even more dramatic if the payoff favors cooperation reward, for example, in an IPD payoff of T:5 R:4 P:1 S:0, TFT would suffer more from noise.
Figure 21. Performance decrease in noisy environments with payoff matrix 5-4-1-0

As a result, in a noisy environment, for example, 2% noise level, GTFT can invade TFT. However, the extra tolerance also makes GTFT more vulnerable against the exploit of All-D. So unlike a 5% initial TFT, it is very difficult for a 5% initial GTFT to survive and expand. If we start a simulation with 90% All-D, and 5% TFT 5% GTFT, GTFT would usually vanish before TFT turn the population to a more cooperative one as shown in the evolution population (GTFT is the cyan curve).

Figure 22. Population change in an initialization of 5% TFT and 5% GTFT
In this scenario, even TFT reached full invasion by the end of the simulation they are not optimal. If we generate 5% GTFT after TFT invaded All-D, this 5% GTFT can, in turn, invade TFT as shown below (though not a guaranteed 100% invasion).

Figure 23. 5% GTFT mutation in a TFT population

Figure 24. 5% GTFT invades TFT
Figure 25. Population change in a population including 5% initial TFT, with 5% GTFT mutation introduced later

This process shows the optimality of strategies is highly situational. Moreover, even in a stable situation, the dominant strategy may still benefit from mutation. Specifically, in this experiment, the network as a whole can still benefit from extra tolerance after the selfish agents are weeded out.

Other Ways to Deal with Noise and the Drawback

Pavlov is another way to deal with noise. Pavlov is also cooperative and 1-step-memory like TFT. If the opponent cooperates on the current turn, Pavlov will stay in the current choice. Otherwise, it will switch to the alternative. It means in noisy environments, a mutual defection turn is followed by a unilateral defection caused by noise, then they will both switch to cooperation. Also, if noise cause Pavlov to defect non-responsive cooperative strategies like All-
C, it can stay in this exploitative way for the higher score. In noisy setting, it is also able to invade TFT, as GTFT.

These alternatives have a common drawback: they are particularly vulnerable to certain uncooperative strategies. In the case of GTFT, it is highly vulnerable to random. Pavlov is susceptible to All-D. In a mono population of GTFT or Pavlov, they can perform better than one of TFT, but when we randomly mutate 5% of the population to their natural enemy (random or All-D), this mutation can quickly spread and collapse the cooperation population.

Instead, if we keep the cooperation population a mixed one with a small percentage of TFT (around 5%), even the global score will be slightly lower, the population is much more resilient against all kinds of uncooperative mutation. From the simulation, we can see when a mutation spread like a pandemic, once they contact the reserved TFT population, TFT will immediately push back, and turn back to GTFT or Pavlov. The process has typically 4 phases as shown below (we use Pavlov as an example, but the dynamics among GTFT, TFT, and random is the same):
Figure 26. Four phases of an uncooperative invasion in a population mixed with TFT

1. The coexistence of a Pavlov and TFT as minority
2. The spread of Pavlov’s natural enemy (All-D)
3. TFT eliminate this mutation
4. Back to phase 1, TFT may not eradicate the mutation based on random network structure.

   However, the majority of cooperation is preserved.

From the simulation, we can see some strategies can outperform TFT, but they have a specific enemy that can easily invade them. On the other hand, a small amount of TFT can serve as a
bastion to preserve cooperation. It means in noisy environments, to minimize defection, we may use alternatives to TFT for higher cooperation coverage. However, it is unwise to replace all TFT because the alternative is susceptible to certain invasions.

**Adaptability**

In this section, we will examine the importance of adaptability for the purpose of promoting cooperation. We introduce an unadaptable strategy called rigid altruism, defined as a strategy that always cooperates in one encounter, and does not adopt other better performing strategies between encounters. We start the simulation on small-world networks, with 5% TFT and an increasing population of rigid altruism.

![Figure 27. Effect of 5% unadaptable All-C](image)
From the graph we can see as population increases, rigid altruism begins to hinder the spread of cooperation. P(EC) is about 10% lower than that of 5% TFT with 5% adaptive altruism. The drop of p(EC) is amplified when we start with more initial rigid altruism as shown below.

**Figure 28. Effect of 10% unadaptable All-C**

**Figure 29. Effect of 15% unadaptable All-C**
From the simulation we can see at 30% initial rigid altruism, cooperation stops to grow at all. It shows unadaptable altruism can seriously prevent the spread of cooperation.
Chapter 6 Conclusion and Limitation

Conclusion

In Chapter 5, we analyzed some factors’ effect on spreading cooperation. The findings from the simulation can be summarized as follows:

- On small-world networks, as the population increase, \( p(\text{EC}) \) converges to 100%. On scale-free networks, the population has no stable effect on \( p(\text{EC}) \).

- On small-world networks, the neighborhood parameter \( k \) is more decisive in \( p(\text{EC}) \) than the rewiring parameter \( p \). A \( k \) value of 3 or 4 is optimal for the spread of cooperation.

- The optimality of strategies is circumstantial to the phase of evolution. In a noisy environment, though GTFT can perform better than TFT, due to its fragility to Random, it is not guaranteed to spread cooperation in a mixed population.

- In noisy environments, TFT is not optimal, strategies like GTFT, Pavlov can all outdo TFT. However, they are each vulnerable to a specific strategy. So a mono population of these strategies is highly unstable, and the cooperation can collapse. By mixing a small portion of TFT into the population, with a little sacrifice in global cooperation, the population is highly resistant to uncooperative mutations.

- Unadaptable cooperating, or rigid altruism, can seriously hinder the spread of cooperation to the extent of entirely stop.

Moreover, in general, due to the unique topological features, IPD games on networks exhibit different dynamics from the spatial IPD. While the majority literature of IPD focuses on the lattice-based world, we think the research in network IPD has great potentials.
Social circle, as a relatively new network model, is also very versatile in our research. It exhibits some similar features with small-world networks but has finer and wider range of parameters to tune to achieve the desired network. As a model uniquely designed for ABM, social circles have yet reached the potentials; we hope more social circles based ABMs can be built and discussed, to further this model.

**Limitation**

- NetLogo is a synchronous simulation environment. In our simulation, within one generation, each agent plays a series of IPD with linked neighbors, and observe who has the highest average score, then adopts that strategy simultaneously. It somehow contradicts with the asynchronous nature of human interaction.

- Possible errors introduced by the separation of computation between NetLogo and Java. In the model explanation part, we discuss the reason and benefit of this separation, yet to calculate payoff externally, we have to use the mean value of multiple PDs. If we simulate all the PD interactions within the ABM, the process can be more random and stochastic, and the result will be more robust.

- Only a selection of strategies is examined. In this research we are focusing on the spread of cooperation in selfish population, so we mainly talked about TFT and All-D. We discussed some remarkable variations of TFT, like GTFT and Pavlov, but there is still a large volume of strategies unchecked.

- Individuals do not differentiate their neighbors. In one generation, a node will interact with all its linked neighbors by the same strategy. Based on the opponents, its actual
choices may vary, but the strategy remains. While in reality, it is very common for individuals, or groups, to choose different strategies to cope with others based on past actions.

**Future Research**

As we discussed in the limitation part, the agents treat neighbors uniformly. We want to extend the model with choice and refusal, and the agents can keep memories from past generations so they can dynamically adjust the possibility and willing to play with its neighbors. Also, by choice and refusal, the network itself becomes temporal. In SNA, the triadic closure theory suggests the friends of the same individual have an increased chance to become friends. We want to incorporate this mechanism into the model so that more strategy-based communities can be formed.
References


[38] G. Simmel, “The number of members as determining the sociological form of the group. I,” Am. J. Sociol., vol. 8, no. 1, pp. 1–46, 1902.


Bibliography


