

國立交通大學

資訊科學系

博士論文

突現行為之研究：  
人工社會中公利與私益之衝突

A Study of Emergent Behavior:  
The Public Good vs. Private Interest Conflict in Artificial Societies



研究生：吳旭智

指導教授：孫春在 教授

中華民國九十三年六月

突現行為之研究：人工社會中公利與私益之衝突  
A Study of Emergent Behavior:  
The Public Good vs. Private Interest Conflict in Artificial Societies

研究生：吳旭智

Student：Hsu-Chih Wu

指導教授：孫春在 教授

Advisor：Prof. Chuen-Tsai Sun

國立交通大學  
資訊科學系  
博士論文



Department of Computer and Information Science  
College of Electrical Engineering and Computer Science  
National Chiao Tung University  
in partial Fulfillment of the Requirements  
for the Degree of  
Doctor of Philosophy  
in

Computer and Information Science

June 2004

Hsinchu, Taiwan, Republic of China

中華民國九十三年六月

# 突現行為之研究： 人工社會中公利與私益之衝突

學生：吳旭智

指導教授：孫春在教授

國立交通大學資訊科學系

## 摘 要

本論文探討人工社會當中，因為公利與私益之間的衝突，所造成的困局。我們利用『囚犯困境』此數學模型，作為研究的基礎模型。本論文提出一套分析架構，根據人工社會當中代理人之間的互動模式，來分析兩個以上的代理人之間的關係。此分析架構可以呈現一策略集合當中，所有策略之間的關係。此關係包含策略侵佔其他策略的能力，以及同種策略之間形成聚集的強度。以此分析架構以及另一例子說明，我們提出在研究方法上，於模型的模擬及執行之前，進行模型分析的重要性。本論文更進一步探討人工社會當中，環境因素對代理人行為以及策略運用的影響。

# A Study of Emergent Behavior: The Public Good vs. Private Interest Conflict in Artificial Societies

Student : Hsu-Chih Wu

Advisors : Dr. Chuen-Tsai Sun

Department of Computer and Information Science  
National Chiao Tung University

## ABSTRACT

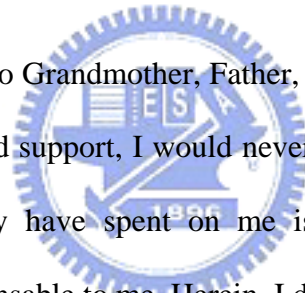
The author reports on his investigation of the conflict between public good and private interest in artificial societies, using the classical model known as the Prisoner's Dilemma. The proposed analytical framework can be used to analyze relationships between and among agents in an artificial society based on their interaction patterns. Framework success relies on (and emphasizes the importance of) a model analysis being performed prior to any simulation or execution of an agent-based computation. A secondary issue discussed in this report is the effect of environment on agent actions. The results indicate that agents benefit greatly from acknowledging environmental factors when determining subsequent moves.

## Acknowledgements

I am deeply indebted to my advisor, Professor Chuen-Tsai Sun, whose patient teaching and continuous support during my graduate research period not only make me progress a lot in the professional field I major in, but also paved the way for my future direction.

Thanks to all professors, the members of my dissertation committee, for their kind comments and suggestions. I appreciate my dear friends in the Laboratory for Learning Sciences and Technologies, who make up a creative studying environment.

I owe my achievements to Grandmother, Father, Mother and all my family members. Without their endless love and support, I would never be what I am. I would do my best to let them know what they have spent on me is worthwhile. In the future, their encouragement is still indispensable to me. Herein, I dedicate this dissertation to them.

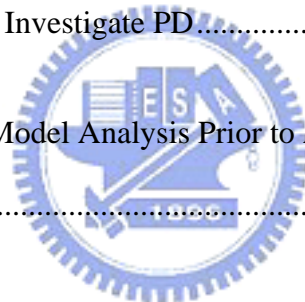


# Contents

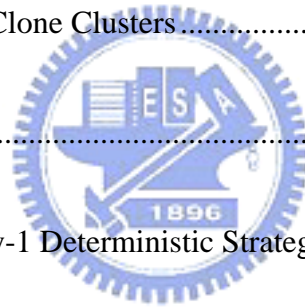
Abstract (in Chinese).....	i
Abstract (in English).....	ii
Acknowledgements .....	iii
Contents.....	iv
List of Tables.....	viii
List of Figures.....	x
Chapter 1. Introduction.....	1
1.1 Motivation.....	1
1.2 Study Importance .....	4
1.3 Study Outline .....	5
Chapter 2. Related Work .....	6



2.1 Artificial Society .....	6
2.1.1 Primary Components of Artificial Societies .....	6
2.1.2 Popular Artificial Societies for Problem Solving.....	9
2.1.3 The Public Good/Private Interests Conflict in Artificial Societies .....	11
2.2 The Prisoner's Dilemma (PD) .....	12
2.2.1 PD and IPD (Iterated Prisoner's Dilemma) .....	12
2.2.2 Research Topics on IPD .....	14
2.2.3 Methods Used to Investigate PD .....	15
Chapter 3. The Importance of Model Analysis Prior to Artificial Society Simulation and Execution .....	18
3.1 Why Analyze before Model Simulation and Execution? .....	20
3.1.1 Defining Model Scope .....	20
3.1.2 Reducing Model Complexity .....	22
3.1.3 Choosing Appropriate Model Instances .....	24
3.2 Case Study .....	24
3.2.1 Model Description.....	25
3.2.2 An Analytical Approach to the Model .....	27

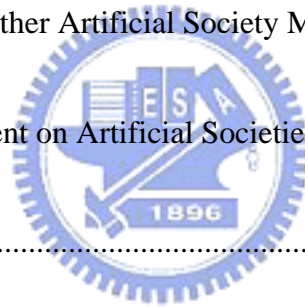


3.2.3 Discussion of This Example.....	35
3.3 Conclusion .....	38
Chapter 4. The Analytical Framework for IPD Models .....	40
4.1 Definitions .....	40
4.2 Behavior Characteristics of Infinite Duration.....	47
4.3 Two Criteria for Investigating Strategy Properties.....	55
4.3.1 Ability to Exploit Others.....	55
4.3.2 Ability to Form Clone Clusters.....	57
4.4 Conclusion .....	59
Chapter 5. Analysis of Memory-1 Deterministic Strategies in IPD.....	60
5.1 Exploitation Relation .....	60
5.1.1 Strategy Classification .....	61
5.1.2 Exploitation Chains .....	62
5.1.3 The Draw Case .....	64
5.2 Clustering Relation .....	65
5.3 Properties of Memory-1 Deterministic Strategies .....	68
5.3.1 Relation among Memory-1 Deterministic Strategies.....	68





5.3.2 Discussion of Well-Known Strategies .....	70
Chapter 6. Application of This Framework.....	74
6.1 Cyclic Relation among Three Strategies.....	74
6.2 Framework Generalization.....	77
6.2.1 Analyses of Other Deterministic Strategies .....	77
6.2.2 Analyses of Other $2 \times 2$ matrix games.....	79
6.2.3 Application of Spatial and Biased Selection IPD Models.....	80
6.2.4 Application on Other Artificial Society Models .....	80
Chapter 7. Effect of Environment on Artificial Societies .....	82
7.1 Introduction.....	82
7.2 Model Description .....	83
7.3 Results.....	86
7.4 Conclusion .....	93
Chapter 8. Conclusion .....	94
Bibliography .....	96
Appendix A .....	109
Vita .....	111



# List of Tables

Table 2.1:	Artificial Societies Commonly Used for Problem Solving.....	10
Table 2.2:	List of Artificial Societies, their Goal Levels, and Conflict between Individual Agent and Societal Goal.....	12
Table 2.3:	Prisoner’s Dilemma Payoff Matrix and Constraints .....	13
Table 2.4:	Some Commonly Used Payoff Matrices for the Prisoner’s Dilemma .....	17
Table 3.1:	Moves of $y$ and $X$ from Round 1 to Round $n$ .....	28
Table 3.2:	Potential Scores of $y$ for Each Combination of $(a_x, a_y)$ . .....	29
Table 3.3:	$B_m$ Scores for $m=1$ to $(n-1)$ .....	31
Table 5.1:	Exploitation Relations between Paired Memory-1 Strategies.....	61

Table 5.2:  $B(S_i|S_i)$  and  $E(S_i|S_i)$  for Memory-1 Deterministic Strategies ..... 66

Table 5.3: Some Commonly Discussed Memory-1 Deterministic Strategies ..... 69



# List of Figures

Figure 3.1: String Combinations Can Be Represented as $B_m$ Patterns Connected by Consecutive 0's.....	30
Figure 3.2: Mathematical Procedure for Obtaining Optimal $y$ Score.....	34
Figure 4.1: The Interaction between Strategies (C, D, C, D) and (C, D, D, C).....	47
Figure 5.1: Exploitation Chains of Memory-1 Deterministic Strategies.....	63
Figure 5.2: The Exploitation Relation among Memory-1 Deterministic Strategies.....	64
Figure 5.3: The Ability to Form Clone Cluster for Memory-1 Deterministic Strategies.	67
Figure 6.1: Cyclical Exploitation Relationships among 3 Strategies. ....	74

Figure 6.2: $3^2$ Possible Combinations of Strategy Relationships between A, B, and C, Given $A \rightarrow B$ .	75
Figure 7.1: Agent Representation in a Two-Dimensional Matrix in a Spatial IPD Model.	84
Figure 7.2: Amount-versus-Generation Graph for 16 Memory-1 Deterministic Strategies.	87
Figure 7.3: Amount-versus-Generation Graph for Memory-1 Deterministic Strategies and [15, 6, 6, 6, 0].	88
Figure 7.4: Amount-versus-Generation Graph for Memory-1 Deterministic Strategies and [15, 5, 6, 7, 1].	89
Figure 7.5: Distribution of Strategy Number versus Payoff Values.	90
Figure 7.6: Amount-versus-Generation Graph for Memory-1 Deterministic Strategies and [0, 6, 6, 6, 0].	91
Figure 7.7: Amount-versus-Generation Graph for Memory-1 Deterministic Strategies and [15, 6, 6, 6, 15].	92

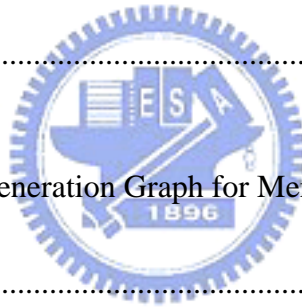


Figure 7.8: Amount-versus-Generation Graph for Memory-1 Deterministic Strategies and  
[15, 6, 6, 6, 15]..... 92



# Chapter 1.

## Introduction

### 1.1 Motivation



Many computer science researchers are currently focusing on the concept of “society.” In addition to “real world” society, they are dealing with an increasing number of societies created with computer technologies. Examples include multi-agent systems that are available for problem solving and goal achievement. Multiple User Dungeon (MUD) systems and online games allow users to fight battles, solve puzzles, converse, and establish long-term relationships such as virtual marriages [1, 2], and virtual markets in the form of online auction systems are very common. Participant behavior in these societies—described by many as mere abstractions or simplifications of human societies—is attracting attention not only from computer science researchers, but also from sociologists and scholars in other social sciences.

It is important to make a distinction between virtual and artificial societies. Virtual environments (e.g., MUDs) are constructed with computer technologies, but they are inhabited by individuals who are either human or controlled by humans [3]. In contrast, all entities that exist in an artificial society are artificial—both the environment and individuals within the environment. Human interference is impossible in artificial societies.

Unlike virtual societies (whose primary purpose is to provide alternative means for human interaction [3] or learning [4, 5]), the motivations for creating an artificial society are two-fold: to simulate real-world phenomena [6-9] or to solve problems within a societal context. The first motivation is a reflection of the difficulty of performing experiments and making observations in real-world societies [6-10]. Since artificial societies make it possible to simulate and observe certain phenomena, they are often referred to as artificial societies for simulation. The second motivation involves more complex issues—for instance, the use of multiple software-driven agents programmed to cooperate in order to solve prescribed problems. Artificial intelligence (AI) methods that incorporate such societies to solve problems include genetic algorithms, ant colony systems, artificial immune systems, classifier systems, and artificial neural networks. These kinds of societies are referred to as artificial societies for problem solving.



In artificial societies created for problem-solving purposes, solutions are arrived at by achieving either individual or global (system-level) goals. An example of the first is a genetic algorithm whose success is defined in terms of finding a chromosome that optimally satisfies specified problem constraints. An example of the second is a multi-agent system in which the agents are programmed to work in a cooperative manner. Most artificial societies contain a highly correlated mix of individual and global goals. Ideally, artificial society design should emphasize consistency between individual and global goals, but this ideal is sometimes ignored as systems become more complex.

For example, in genetic algorithms, the main goal of a chromosome is to achieve a higher level of fitness in order to increase the probability of being selected for the next generation. On the other hand, if it outperforms other chromosomes too quickly or by too much, it increases the danger of premature convergence [11, 12]. In a multi-agent system in which all agents share the same system resources, it is reasonable for an agent to hold onto a resource as long as possible (since doing so allows it to do more work to achieve its goal), but at the expense of adequate resources for other agents. This scenario is advantageous in the short term for the agent holding onto the resource, but long-term system performance may suffer from the inability of other agents to finish their tasks. This conflict between public good and private interest is a central issue for designers who wish to improve the efficiency of artificial societies.

## 1.2 Study Importance

The public good vs. private interest topic has also attracted real-world interest from sociologists, economists, and ecologists. It is usually expressed in terms of a mathematically simple but analytically intractable model known as the Prisoner's Dilemma (PD). Since the research focus is on the same conflict in artificial societies, the PD model will also play a central role in this investigation. The primary difference between the analytical framework presented in this dissertation and previous PD research is an emphasis on interactions between individuals rather than the equilibrium analyses and evolutionary simulations that dominate the current PD literature.

In addition, I will emphasize the importance of model analysis before running simulations or models. This is an important topic that is often overlooked by researchers who use artificial societies for solving problems or running simulations. A central argument to be presented in this dissertation is that appropriate model analysis facilitates simulation efficiency. A detailed example will be presented to illustrate that under certain circumstances, an analytical approach can outperform evolutionary or agent-based computations.

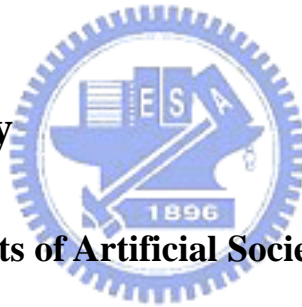
## 1.3 Study Outline

A review of related studies is presented in the following chapter. In addition to listing popular artificial societies for problem solving and describing them in terms of the public good/private interest conflict, I will discuss the development of the PD model and common approaches to PD investigations. In Chapter 3, I will present a simple case study outlining the importance of performing model analysis before running a simulation or solving a problem. In Chapter 4, a description of the analytical framework for analyzing PD strategy relationships will be given. An application of the framework to memory-1 strategies will be described in Chapter 5—including evidence that the framework is useful in identifying important PD strategies and simulation phenomena. Chapter 6 contains a discussion of framework extension and generalization to models whose strategies are encoded in other forms, in spatial PD models, and in biased selection PD models. Also in Chapter 6, I will discuss framework methodology and its application to other kinds of artificial society analyses. A discussion of the effects of strategic PD environments is presented in Chapter 7, and conclusions are given in Chapter 8.

# Chapter 2.

## Related Work

### 2.1 Artificial Society

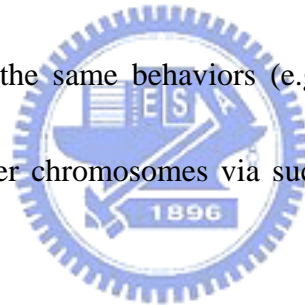


#### 2.1.1 Primary Components of Artificial Societies

According to Actikis and Pitt [13], all artificial societies include a) a set of agents, b) a set of societal constraints, c) a language for communication, d) a set of agent roles, e) a set of states of affairs that hold at each time at the society, and f) a set of agents (sometimes referred to as owners). Tim Doran [14, 15] defined an abstract model of an artificial society as a network (or graph) with nodes called *sites* and links called *channels*. At any single moment, a network is filled with many agents and items located at individual sites.

From the two definitions, the three primitive components of an artificial society are:

*a) Agents (individuals).* Considered basic units of an artificial society, agents have their own characteristics and express their own behaviors. Depending on the level of sophistication of an artificial society, it can be categorized as homogeneous or heterogeneous. Homogeneous agents have similar or equivalent characteristics—for example, chromosomes in the simplest form of a genetic algorithm (GA). Even though a set of chromosomes may have different gene combinations and fitness values, they have the same format and express the same behaviors (e.g., they are evaluated by the same function and interact with other chromosomes via such shared mechanisms as crossover operators).



Agents are said heterogeneous if they have completely different characteristics, including behaviors, goals, or encodings. Well-known examples of heterogeneous artificial societal agents are agents in multi-agent systems. They are purposefully designed with completely different goals and behaviors, cooperating and interacting in order to achieve system goals.

*b) Environment.* Artificial societies must provide environments in which agents can perform their functions. In its simplest form, an environment is a set of agents—for

instance, the set of all cells in a cellular automata [16, 17]. The actions of a cell are determined by its current status as well as the statuses of its neighboring cells—no other factor is involved, unlike the situation for more complex artificial societies. An example of a more complex scenario is a classifier system, which consists of four basic parts: input interface, classifiers, message lists, and output interface [18]. Messages are read from an input interface and added to a message list that all classifiers use to determine which messages satisfy their conditions. Results are sent to the output interface. In this example, messages are considered the environment that affects all classifiers.

There are two categories of environments: agent viewpoint and global environment. For most artificial societies, these two are distinctly different. Since information can decay or be lost through propagation, and since each agent has a limited perception of its environment, any environment is usually considered incomplete from an agent's viewpoint. It is important that users know if the environmental information they are working with is global or from an individual agent's viewpoint.

*c) Interaction.* The most important component that determines the complexity and characteristics of an artificial society is interactions between the basic entities of agent and environment. In their absence, artificial societies simply constitute a set of static data.

Interactions in artificial societies can be categorized as between agents or between agents and environments. The first category is more common. An example is the crossover operator in GAs, in which two chromosomes interact via gene exchange. In multi-agent systems, agents can negotiate with, take information from, or give information to other agents. Interactions between agents and environments are more common in more sophisticated artificial societies—for instance, ant colony systems [19-21]. Ants secrete pheromones into their environments, and their actions are determined according to the pheromones they encounter. Their interactions are said to be limited to their environment instead of with other ants.



### **2.1.2 Popular Artificial Societies for Problem Solving**

A list of artificial societies commonly used for problem solving is presented in Table 2.1, categorized according to their primary component characteristics.

GAs are considered simple artificial societies that contain chromosomes as homogeneous agents; no other environmental factor is involved. Classifier systems are more complex than GAs in that they accept external input and generate output, both of which can be viewed as environmental factors whose statuses affect the system. Furthermore, classifiers can interact with those environmental factors as well as with other classifiers. As stated in the preceding section, an ant colony system is considered a special

type of artificial society in that there is no interaction between agents, only between agents and their environment. Neural networks are considered a particular type in that interactions between agents (nodes) are fixed; interaction between two nodes is impossible in the absence of a connecting link.

**Table 2.1: Artificial Societies Commonly Used for Problem Solving**

	<b>Agent</b>	<b>Environment</b>	<b>Interaction</b>
<b>Genetic Algorithm</b> [22]	Homogeneous	N/A	A-A <sup>1</sup>
<b>Ant Colony System</b> [19-21]	Homogenous	Simple	A-E <sup>2</sup>
<b>Classifier System</b> [18]	Homogenous	Complex	A-A, A-E
<b>Neural Network</b> [23]	Homogenous	Complex	A-A, A-E, but fixed
<b>Artificial Immune System</b> [24-28]	Heterogeneous	Complex	A-A, A-E
<b>Multi-Agent System</b> [29, 30]	Heterogeneous	Complex	A-A, A-E

<sup>1</sup> interactions between agents.

<sup>2</sup> interactions between agents and environments.



### **2.1.3 The Public Good/Private Interests Conflict in Artificial Societies**

A common real-world conflict is that between the public good and private interests. However, it remains to be seen whether this conflict also exists in artificial societies that are not inhabited by humans or other living beings—especially artificial societies whose primary goal is to solve problems, instead of simulating real-world phenomena. As mentioned in Chapter 1, in problem-solving artificial societies the society concept is used to solve problems via either individual achievement or societal achievement of a system-level goal. Regardless of the approach, the public good/private interest conflict makes an appearance sooner or later. A list of common artificial societies used for problem solving is presented in Table 2.2, along with a summary of the goals and issues raised by this conflict.

From Table 2.2, it is clear that even in the artificial societies for problem solving, the conflict between private good and public interest is also a crucial issue. I believe that the investigation of this conflict will be helpful for efficient use of those models.

**Table 2.2: List of Artificial Societies, their Goal Levels, and Conflict between Individual Agent and Societal Goal**

	<b>The Level of Goal (System/Individual)</b>	<b>Conflict between Agent's and Societal Goals</b>
Genetic Algorithm	Individual	Premature Convergence, [11, 12]
Ant Colony System	System	N/A
Classifier System	System	Credit Assignment [31, 32], Premature Convergence
Artificial Immune System	Individual/System	Selection Mechanism [24]
Neural Network	System	N/A
Multi-Agent System	Individual/System	Mechanism Design [33, 34], Action Selection [35], Resource Management [36], Robots Conflict Detection [37]

## 2.2 The Prisoner's Dilemma (PD)

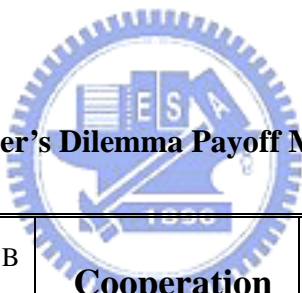
### 2.2.1 PD and IPD (Iterated Prisoner's Dilemma)

The Prisoner's Dilemma (PD) is a precise mathematical model that is frequently used by economists [38], sociologists [39, 40], political scientists [41], biologists [42], and psychologists [43] to analyze conflicts of interest [44]. It is considered a powerful tool for

explaining human and animal society organization and for promoting cooperation among agents in virtual environments [45]. Its simplicity is considered one of its outstanding qualities, yet it remains a surprisingly complex problem to analyze.

In a classic version of a PD game, two players decide whether each move they make should be one of cooperation or defection. A payoff is given to each player according to their combined moves. Table 2.3 shows a typical payoff matrix, including value constraints.

**Table 2.3: Prisoner's Dilemma Payoff Matrix and Constraints**



Player B	<b>Cooperation</b>	<b>Defection</b>
Player A	<b>Cooperation</b>	<b>Defection</b>
<b>Cooperation</b>	Reward Reward	Sucker's Temptation
<b>Defection</b>	Temptation Sucker's	Penalty Penalty

Note: **T** > **R** > **P** > **S**, and

$2 \times (\mathbf{R}) > \mathbf{T} + \mathbf{S}$ , or simplified:

$$\mathbf{T} > \mathbf{R} > \mathbf{P} > \mathbf{S}, \text{ and } 2 \times \mathbf{R} > \mathbf{T} + \mathbf{S}.$$

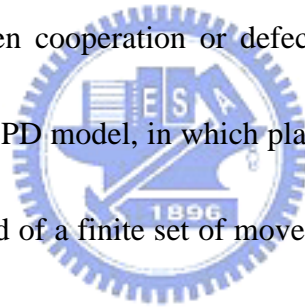
If both players know that they will play a PD game one time only, it is to their benefit to continually make defection moves in order to achieve a maximum outcome. If they know that they will play many games (a situation referred to as the Iterated Prisoner's Dilemma, or IPD), mutual cooperation is a better strategy for both players. Since most real-world dilemmas are iterated, the IPD is of great interest to researchers [46, 47].

### 2.2.2 Research Topics on IPD

IPD research falls into two broad categories:

a) Model Behavior Investigations. Despite its mathematical simplicity, IPD model behavior is surprisingly complex. Originally, IPD researchers were interested in understanding the behavioral characteristics of this model (e.g., dynamics of population [48, 49], and persistence of cooperation [50]), and of such common strategies as Tit-for-Tat (repeat what your opponent does in the previous round) and PAVLOV (a win-stay-lose-shift strategy) in various scenarios [46, 51, 52]. Other researchers reported on the strengths and weaknesses of strategies encoded in various formats, including memory- $n$  strategies [53, 54], finite automata strategies [55], rule-based strategies [56], and other formats [57-59]. Evolutionarily Stable Strategies (ESS) have also been the focus of numerous studies in terms of properties and their existence in various situations [60-64].

b) Model Sophistication. More recently, researchers have focused on IPD model sophistication in an attempt to apply it to a wider range of problems that are more representative of the real world. Several have introduced spatial IPD models in which agents are positioned in two-dimensional spaces; these agents are limited to interacting with neighboring strategies [65-70]. Others have reported that cooperative behavior is more likely in PD models where opponent selection is biased [45, 71-73]. In PD games with more than two players (known as N-player PDs), emergent cooperative behavior differs from that observed in two-person games [74, 75]. In N-choice PD, players may not be limited to choosing between cooperation or defection [71, 76, 77]. An extension of N-choice PD is the continuous PD model, in which player actions are viewed as an infinite set of continuous values instead of a finite set of moves [78-80]. For each of these models, the guiding goal is to determine if more realistic factors affect cooperative behavior.

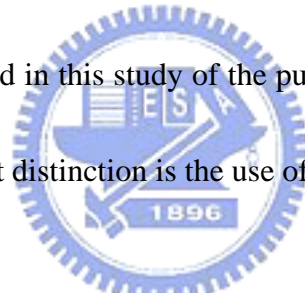


### **2.2.3 Methods Used to Investigate PD**

Currently the two predominant approaches to PD research are mathematical analysis [60, 62-64, 75, 81, 82] and evolutionary simulation [56, 65, 71, 74, 77, 78, 83-85]. Nowak [81] emphasizes the dynamic complexity and unpredictability of PD games despite their small number of strategies based on a simple set of rules. Besides, after using a spatial model to simulate the evolution of 2x2 matrix games, Lindgren and Nordahl [66]

concluded that limited computation resources produce simulation results that are heavily dependent upon the parameters involved. For these and additional reasons, PD games are usually analyzed under certain restrictions. For example, Nowak analyzed the dynamics of only three kinds of PD strategies [81]. In a separate project [82], he investigated a homogeneous population of strategies in an attempt to simulate strategy evolution.

Clear relationships among strategies (i.e., the specifics of player actions during a PD game) must be established prior to studying a PD problem. Furthermore, artificial society component categories underscore the importance of considering agent interactions. Although the IPD model is used in this study of the public good/private interest conflict in artificial societies, an important distinction is the use of an agent viewpoint of interactions.



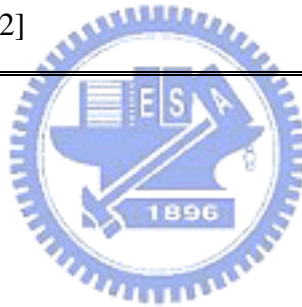
It is also important to note the important role played by simulations, regardless of the degree of understanding of model behavior or level of sophistication. Even in analytical studies, the majority of researchers have used simulations to collect comprehensive data on model behavior [76]. The framework described in this dissertation was used to analyze relationships between strategies—a crucial pre-simulation step that is often overlooked.

Most IPD studies make use of one of the payoff matrices shown in Table 2.4. The effect of payoff matrix choice on study results remains unclear. The model used in this study is based on the payoff matrix constraint shown in Table 2.3—in other words, it is

independent of the payoff matrix value.

**Table 2.4: Some Commonly Used Payoff Matrices for the Prisoner's Dilemma**

Payoff Matrix Values [R, S, T, P]	Reference
[3, 0, 5, 1]	[45, 46, 55, 65, 66, 68, 86]
[4, 0, 5, 1]	[84]
[1, -2, 2, -1]	[53]
[3, 1, 4, 2]	[87]



## Chapter 3.

# The Importance of Model Analysis Prior to Artificial Society Simulation and Execution



Performing a model analysis prior to model simulation and execution is important regardless of the intended purpose of the artificial society in question. In this chapter I will offer reasons why it is important and present an illustrative example to show how it facilitates the use of artificial societies for problem solving.

Regarding the model execution/simulation process, common methodologies include the following steps: a) model creation (constructing a model based on an existing theory, hypothesis, or empirical data); b) model execution (running a model to produce data); and



c) model verification (assessing a model's ability to operate as intended) and validation (analyzing data to ensure that a model is working as intended) [88].

Although it is rarely included in lists of model execution/simulation methodologies, theoretical and statistical model analyses play important roles. During the creation phase, model structures and relationships are mostly based on theories or hypotheses. Theoretical variables are defined and quantified, and relationships among them are encoded [89]. In the words of Hanneman and Patrick, any model being constructed is “one concrete realization of the prior theory” [10]. During the verification phase, the simulated results are statistically analyzed for purposes of interpretation and/or explanation [6].

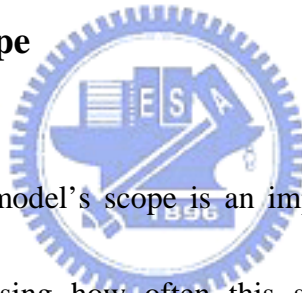


There are few discussions in the literature of useful analytical tasks to be performed after a model is created but before simulation begins. At this point, it is important to determine appropriate model parameters or parameter sets based on empirical experience or existing data. The importance of analysis at this phase is the focus of this chapter. We believe that pre-simulation model analysis can help reduce simulation complexity as well as assist in the identification of appropriate execution/simulation parameters.

## **3.1 Why Analyze before Model Simulation and Execution?**

The most important motivation for a pre-execution/simulation model analysis is the assumption that the more one knows, the easier it will be to properly simulate or run the model. In this section I will describe how a pre-execution/simulation analysis helps in defining model scope, reducing model complexity, and choosing appropriate simulation/execution parameters.

### **3.1.1 Defining Model Scope**



Even though defining a model's scope is an important first step toward increasing model efficiency, it is surprising how often this step is overlooked by researchers. Whenever an execution/simulation run provides significant findings, the data and the model clearly need to be inspected in terms of validity. But it is equally important to determine the conditions under which a particular model is successful, as well as the possibility of achieving success under other conditions.

Following model construction, concepts and entities are defined as parameters or variables. Prior to each new execution/simulation run, individual parameters must be set to specific values to satisfy some condition. An execution/simulation run is not equivalent

to a model. In this paper, a run is defined as an *instance* of the model. In theoretical terms, the comprehensive understanding of a model requires the execution/simulations of all possible model instances, but doing so is usually considered impractical.

A model  $M$  can be defined as

$$M=(P_1, P_2, \dots, P_n),$$

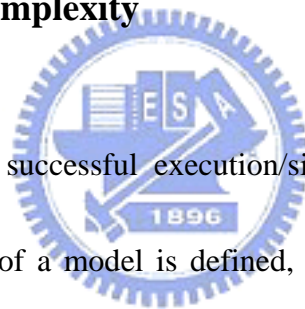
where  $P_1, P_2, \dots, P_n$  represent  $n$  parameters of  $M$ . Letting  $N$  denote the number of possible model instances and  $|P_i|$  denote the number of possible values of parameter  $P_i$ , then


$$N = |P_1| \cdot |P_2| \cdot \dots \cdot |P_n|$$

Each parameter has its own constraints. Examples of discrete parameters include the size of a population in a societal model and the number of nodes in a social network model [90]. Here the number of possible values is finite, but other parameters are considered continuous and infinite—for instance, tax rates in a simulation of tax and welfare systems. Most artificial society models contain both discrete and continuous parameters; even in simple models, the number of instances is usually large or infinite. Each instance represents a tiny part of the model.

Since it is impossible to execute/simulate all model instances, it is important to choose an appropriate single model instance or set of model instances. I want to emphasize the importance of knowing the number of potential choices before making what appears to be the most appropriate choice, since the success of one model instance implies overall model success, but the failure of one model instance does not imply overall model failure. It is easier to figure out the relationship between a model and a model instance once its scope is defined.

### 3.1.2 Reducing Model Complexity



The second step toward successful execution/simulation involves reducing model complexity. Once the scope of a model is defined, it is no longer necessary to run all possible model instances. Unnecessary instances should be avoided in order to make the execution/simulation process more efficient. The two types of model instances that can be skipped are:

a) *Unreasonable instances*, meaning that a parameter setting does not match real world conditions. These can be further divided into two categories: (1) instances with unreasonable parameter values, which are not under the constraints of the corresponding parameters; and (2) unreasonable parameter combinations, meaning that individual parameter values that are considered reasonable become unreasonable once they are

combined with other reasonable parameter values because of their correlational relationships.

*b) Equivalent instances*, meaning that instances may appear to be completely different but nevertheless produce identical simulation results, or have identical meanings from the perspective of the model. An analysis of equivalent instances can provide information about whether or not an instance should be executed or simulated. It may be unnecessary to execute/simulate reasonable or important instances in cases where the results from equivalent instances are identified.

Analyses of unreasonable or equivalent instances reduce the number of potentially appropriate model instances. Using the metaphor of a highway map, the scope of a model provides the number of possible ways to get from point A to point B, while reduced model complexity provides answers to questions such as: “Which routes will not get us from point A to point B?” and “Which individual routes lead to the same destination?” By reducing the numbers of unreasonable and equivalent instances, it becomes easier to choose the appropriate parameter settings for successful execution/simulation.

### 3.1.3 Choosing Appropriate Model Instances

The final analytical step before execution/simulation is to determine appropriate model instances that resemble most other instances or that otherwise have some significant importance. In the literature, most model instance selections are based on empirical data or the testing of hypotheses. Nevertheless, an execution or simulation instance with significant results must still be tested to determine if it is representative of other instances and produce identical or similar outcomes. Answering such questions becomes more difficult when model instance determinations are not based on theoretical or statistical analyses.



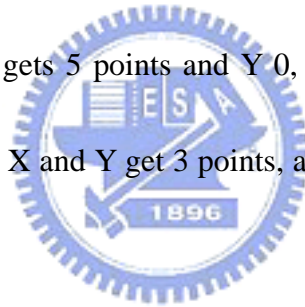
## 3.2 Case Study

The example presented in this section is based on Azuaje's [91] efforts to use a GA to evolve game strategies and cooperation. In my analytical approach the model is greatly simplified, which allows optimal solutions to be obtained more quickly and easily. Even though the robustness of artificial societies designed for problem solving (including evolutionary approaches) makes them popular among researchers in various disciplines, two important considerations are frequently overlooked: the importance of pre-run model

analysis, and the need to make a conscious decision between evolutionary and analytical approaches to individual problems based on their specific characteristics.

### 3.2.1 Model Description

In [91], Azuaje proposes an approach to the co-evolution of competing virtual creatures to model the emergence of cooperation in game strategy. His artificial life model considers two kinds of organisms, X and Y. Their individual decisions about whether or not to approach a food source are presented in the form of an IPD game. If X approaches a food source and Y doesn't, X gets 5 points and Y 0, and vice versa. If neither organism approaches a food source, both X and Y get 3 points, and if both approach the source, they each get 1 point.



Azuaje stated that Y is represented by a genetic code that determines its sequence of moves against X. For example, if a Y organism is encoded as:

001000,

that means it will make defection moves in the first two rounds, followed by a single cooperation move, followed by three defection moves. Furthermore, Y cannot recognize individuals or store (remember) previous events. In contrast, X is a more sophisticated

organism that can perform basic cognitive and memory functions; it follows a Tit-for-Tat strategy of mimicking each move that its opponent made in the preceding round.

Azuaje used GA approach to evolve his game strategies. In his experiment, all Y organisms were randomly produced, and the length of each Y was 30. Each Y organism interacted with an X organism to obtain its score, after which GA operators (including reproduction, crossover, and mutation) were used with the population of Y organisms. His results showed that it was possible for the evolved Y organisms to outperform the Tit-for-Tat strategy followed by the X organisms. After 50 generations, the most successful Y organism followed this code:

000000000000000000000000000001.



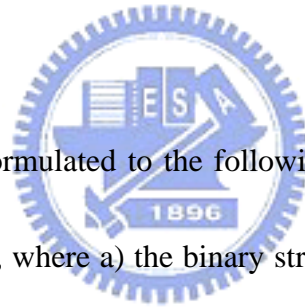
Its final point total was 92, versus 87 for the X organism.

According to this model, a) cooperative behavior can emerge from an evolutionary and unsupervised learning process, and b) evolving organisms are capable of achieving individual success by developing strategies that are more effective than Tit-for-Tat [91].



### 3.2.2 An Analytical Approach to the Model

I propose an analytical approach that is more efficient than the evolutionary approach. Y-type organisms evolve according to two basic genetic algorithm (GA) operators: crossover and mutation. Even though there are two kinds of organisms, only Y is subject to the forces of artificial selection. X, whose primary function is to evaluate Y's score, cannot evolve. It is unnecessary to select "100 fittest individuals from each type of organism to be included in the next generation" [91], that selection process can be limited to Y-type organisms.



Azuaje's model can be formulated to the following problem: find an optimal binary string  $y_{opt}$  that maximizes  $F(y)$ , where a) the binary string  $y$  represents a Y-type organism; b) 0's encoded in  $y$  represent "do not approach food" and 1's represent "approach food"; and c) a fitness function  $F(y)$  denotes a  $y$  score when it plays with X (the Tit-for-Tat strategy)  $n$  times, with  $n$  equal to the length of  $y$ .

Let  $y = s_1 s_2 s_3 \dots s_n$ .  $y$ 's moves from round 1 to round  $n$  are expressed as:

$$s_1, s_2, s_3, \dots, s_{n-1}, s_n$$

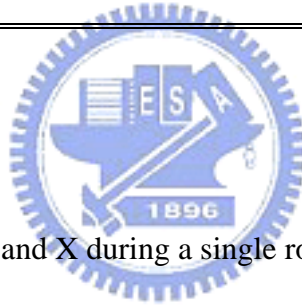
Since X simply repeats its opponent's preceding moves, its moves are expressed as:

$0, s_1, s_2, \dots, s_{n-2}, s_{n-1}$

Moves of  $y$  and  $X$  from round 1 to round  $n$  are shown in Table 3.1.

**Table 3.1: Moves of  $y$  and  $X$  from Round 1 to Round  $n$ .**

	<b>1</b>	<b>2</b>	<b>...</b>	<b><math>n</math></b>
<b>y</b>	$s_1$	$s_2$	$\dots$	$s_n$
<b>X</b>	0	$s_1$	$\dots$	$s_{n-1}$



The interaction between  $y$  and  $X$  during a single round is represented as:

$$(a_X, a_y),$$

where  $a_X$  stands for  $X$ 's move, and  $a_y$  stands for  $y$ 's move. There are four possible combinations for  $(a_X, a_y)$ . For each combination,  $y$  receives a score (in IPD terminology, a payoff)  $P(a_X, a_y)$ . Possible scores are listed in Table 3.2.

In Table 3.2,  $[R, S, T, P]$  denotes four IPD payoff values. In [91]'s model,  $[R, S, T, P] = [3, 0, 5, 1]$ .

**Table 3.2: Potential Scores of  $y$  for Each Combination of  $(a_x, a_y)$ .**

$(a_x, a_y)$	(0, 0)	(0, 1)	(1, 0)	(1, 1)
$y$ 's score $P(a_x, a_y)$	$R$	$T$	$S$	$P$

An interaction history of  $y$  and  $X$  moves from round 1 to round  $n$  is denoted as a sequence of  $(a_x, a_y)$  pairs:

$$(0, s_1), (s_1, s_2), (s_2, s_3), (s_3, s_4), \dots, (s_{n-1}, s_n)$$

The problem can be re-formulated as follows:

Find a binary string  $y = s_1 s_2 s_3 \dots s_n$ , to maximize

$$\sum_{m=1}^n P(s_{m-1}, s_m),$$



where  $s_0 = 0$ ,

$$P(0, 0) = 3,$$

$$P(0, 1) = 5,$$

$$P(1, 0) = 0, \text{ and}$$

$$P(1, 1) = 1$$

To solve this problem, define  $B_m$  as a pattern with  $m$  consecutive 1's sandwiched between two 0's, that is:

$$B_1 = 010,$$

$B_2=0110$ ,

...

$B_{n-1}=011\dots10$ , with  $(n-1)$  1's

The longest pattern is represented as  $B_{n-1}$ . Since  $n$  is the length of  $\mathbf{y}$ , after adding  $s_0$  at the beginning of  $\mathbf{y}$ , the maximum number of consecutive 1's must be  $(n-1)$  in order to satisfy the constraint that it starts and ends with 0.

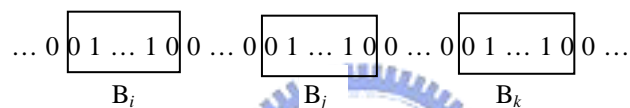
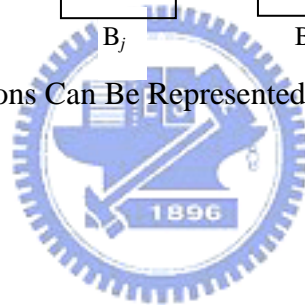


Figure 3.1: String Combinations Can Be Represented as  $B_m$  Patterns Connected by Consecutive 0's.



Let the number of  $B_1, B_2, \dots, B_{n-1}$  patterns in  $\mathbf{y}$  be  $b_1, b_2, \dots, b_{n-1}$ . Most strings with a random combination of 0's and 1's can be represented as the  $b_1, b_2, \dots$ , and  $b_{n-1}$  of  $B_1, B_2, \dots$  and  $B_{n-1}$  patterns connected by arbitrary numbers of 0's (Figure 3.1). Strings that start or end with 1 cannot be represented in this form. These cases will be addressed later in this section.

The reason for choosing  $B_m$  patterns to represent binary strings is to compute  $\mathbf{y}$ 's score. In cases of consecutive 0's, the  $\mathbf{y}$  score will be the sum of consecutive  $R$ 's. At the

appearance of the first 1, the score of that round changes from  $R$  to  $T$ , since  $(a_x, a_y)$  is  $(0, 1)$ .

The score of subsequent round will be  $S$  if that 1 is followed by a 0 and  $P$  if it is followed

by another 1.  $B_m$  pattern scores for  $m=1$  to  $(n-1)$  are listed in Table 3.3. They can be

formulated as:  $PP(B_m) = (T+(m-1)P+S)$ .

**Table 3.3:  $B_m$  Scores for  $m=1$  to  $(n-1)$ .**

$B_m$	$B_1$	$B_2$	$B_3$	...	$B_{n-1}$
<b>Encoding</b>	010	0110	01110	...	011...10
<b>Interaction History</b>	(0,1), (1,0)	(0,1), (1,1), (1,0)	(0,1), (1,1), (1,1), (1,0)	...	(0,1), (1,1), ... (1,1), (1,0)
<b>Score</b>	$T+S$	$T+P+S$	$T+2P+S$	...	$T+(n-2)P+S$

$y$ 's score can be computed by adding the scores of consecutive 0's and the summation of  $B_m$  patterns' scores from  $m=1$  to  $(n-1)$ . However, this kind of representation does not work when the string ends with 1, therefore two cases must be considered:

**Case 1:** Strings that end with 0 whose regular expressions are:

$$0\{1, 0\}^*0$$

**Case 2:** Strings that end with  $x$  number of consecutive 1's, whose regular expressions are:

$$0\{1, 0\}^*1^+$$

Note that the cases that strings start with 1 are skipped since  $s_0=0$  in problem formulation of this section.

The  $y$  score:  $\sum_{m=1}^n P(s_{m-1}, s_m)$  can be accumulated as follows:

**Case 1:** The string is divided into two parts: the  $B_1$  to  $B_{n-1}$  patterns and the consecutive 0's that connect them. Thus:



$$\begin{aligned} \sum_{m=1}^n P(s_{m-1}, s_m) &= \sum (B_1 \text{ to } B_{n-1} \text{ score}) + \sum (\text{Consecutive 0's score}) \\ &= \sum_{m=1}^{n-1} (PP(B_m) \times b_m) + ((n - \sum_{m=1}^{n-1} ((2 + (m-1)) \times b_m)) \times R) \\ &= (T + S) \times b_1 + (T + P + S) \times b_2 + (T + 2P + S) \times b_3 \\ &\quad + \dots + (T + ((n-1) - 1)P + S) \times b_{n-1} \\ &\quad + ((n - 2(b_1 + b_2 + \dots + b_{n-1})) - (b_2 + 2b_3 + \dots + (n-1-1)b_{n-1})) \times R \\ &= n \times R + (T + S - 2R) \times \sum_{m=1}^{n-1} b_m + (P - R) \sum_{m=2}^{n-1} ((m-1) \times b_m) \end{aligned}$$

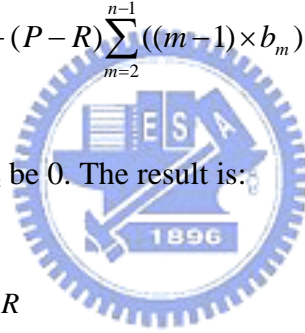
**Case 2:** The string includes an additional section consisting of the last  $x$  consecutive 1's, where  $x \geq 1$ . Thus:

$$\begin{aligned}
\sum_{m=1}^n P(s_{m-1}, s_m) &= \sum (B_1 \text{ to } B_{n-1} \text{ score}) + \sum (\text{Consecutive 0's score}) \\
&+ \sum (\text{Last consecutive 1's score}) \\
&= \sum_{m=1}^{n-1} (PP(B_m) \times b_m) + ((n-x - \sum_{m=1}^{n-1} ((2+(m-1)) \times b_m) \times R) + (T + (x-1) \times S)) \\
&= (T+S) \times b_1 + (T+P+S) \times b_2 + (T+2P+S) \times b_3 \\
&+ \dots + (T + ((n-1)-1)P + S) \times b_{n-1} \\
&+ ((n-x - 2(b_1 + b_2 + \dots + b_{n-1})) - (b_2 + 2b_3 + \dots + (n-1-1)b_{n-1})) \times R \\
&+ T + (x-1) \times S \\
&= n \times R + (T+S-2R) \times \sum_{m=1}^{n-1} b_m + (P-R) \sum_{m=2}^{n-1} ((m-1) \times b_m) \\
&+ (T + (x-1) \times S - x \times R)
\end{aligned}$$

Based on constraint of IPD:  $2R > T+S$  and  $R > P$ , maximizing

$$(T+S-2R) \times \sum_{m=1}^{n-1} b_m + (P-R) \sum_{m=2}^{n-1} ((m-1) \times b_m)$$

requires that  $b_1, b_2, \dots$ , and  $b_m$  be 0. The result is:



**Case 1:**  $\sum_{m=1}^n P(s_{m-1}, s_m) = n \times R$

**Case 2:**  $\sum_{m=1}^n P(s_{m-1}, s_m) = n \times R + (T + (x-1) \times S - x \times R)$

In Case 2, since  $[R, S, T, P] = [3, 0, 5, 1]$ ,

$$(T + (x-1) \times S - x \times R) = 5 - 3x$$

which is greater than 0 if  $x < \frac{5}{3}$ . Since  $x \geq 1$ , the optimal  $y_{\text{opt}}$  is obtained when  $x=1$ :

$$y_{\text{opt}} = 0000 \dots 001$$

with a score of

$$\sum_{m=1}^n P(s_{m-1}, s_m) = n \times R + (T - R)$$

If  $n=30$ , then:

$$y_{\text{opt}} = 00000000000000000000000000000001$$

with a score of

$$\sum_{m=1}^n P(s_{m-1}, s_m) = (30 \times 3 + (5 - 3)) = 92.$$

This result is exactly the same as that produced by Azuaje's evolutionary approach [91].

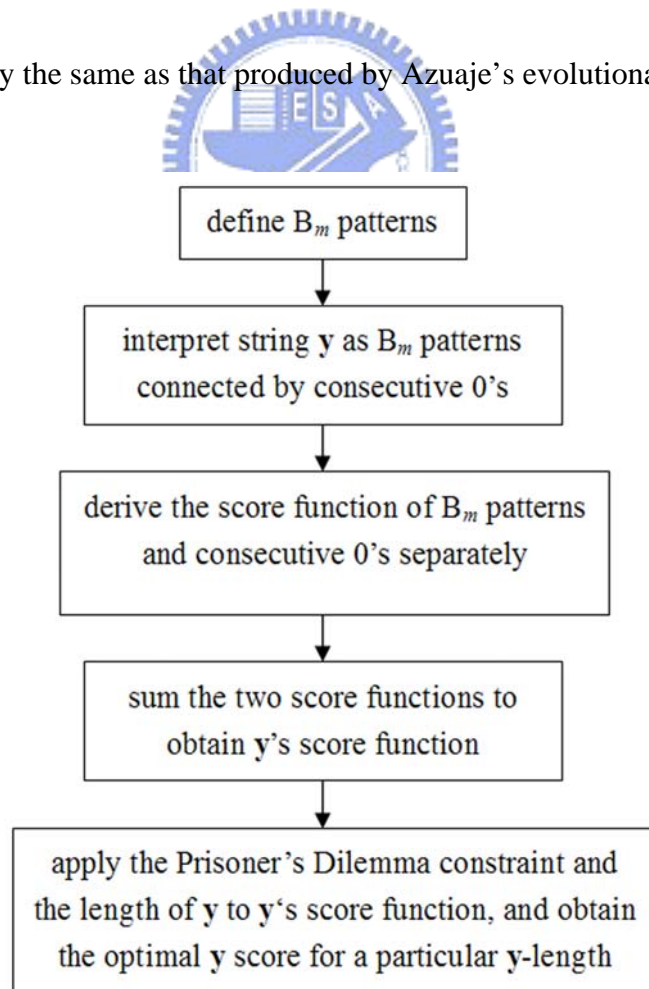


Figure 3.2: Mathematical Procedure for Obtaining Optimal y Score.



The mathematical procedure implemented to obtain optimal y score:  $\sum_{m=1}^n P(s_{m-1}, s_m)$  is

shown in Figure 3.2.

### 3.2.3 Discussion of This Example

My approach emphasizes an important question: What should be done before using evolutionary approach to solve a problem? I believe that too many researchers overlook the importance of model analysis. The model in the current example is complex on the surface because it contains two kinds of organisms. However, since X cannot evolve, it only serves as an evaluation tool for Y—that is, the problem actually addresses only one kind of organism. When using an evolutionary approach, only Y organisms need to be selected for the next generation, cutting the number of artificial selection operations in half and drastically reducing computation time. My main point here is that pre-run analytical work can increase one's understanding of problem scope, reduce model complexity, and help in the search for appropriate parameters—in short, increase the efficiency of a search for appropriate solutions.

My result was the same as [91]'s, but my analytical approach was faster and simpler. In this model, my approach is appropriate in cases with different Y lengths, but the constraints of the evolutionary approach dictate that a GA must be run for each change in the length of Y. I am not claiming that an analytical approach is always superior to an

evolutionary approach, since they clearly have complementary advantages and disadvantages. An analytical approach is much less effective than an evolutionary approach when problem spaces exceed a certain size or complexity threshold. This was not the case in [91]'s example.

Next, the model results are discussed from a game theorists' perspective. In [91], Azuaje states that Y was able to achieve individual success "by learning to approach the source at the end of a contest." This strategy was more successful than Tit-for-Tat. He also makes the claim that "information about the length of the games was not provided to the creature." I believe game length was implied in Y's encoding, since  $n$  is the length of Y and Y always plays with X  $n$  times. Thus, the important "shadow of the future" assumption of stable cooperation no longer holds; in Axelrod's words, "if you are unlikely to meet the other person again, or if you care little about future payoffs, then you might as well defect now and not worry about the consequences for the future" [46]. The behavior of an evolved solution for [91]'s model has already been discussed and verified in the literature [46].

Azuaje also wrote that "the emergence of cooperation did not require special assumptions about the individuals and the game environment." I suggest that the emergence of cooperation is actually determined by the X organism, which uses the

Tit-for-Tat strategy in his model. The strategy encourages the evolution of Y toward a strategy that is equal to or better than Tit-for-Tat, which in turn encourages mutual cooperation. Assuming that X follows an “always defect” strategy (ALLD), then Y will also defect, and cooperative behavior will not emerge.

Furthermore, it is generally accepted that no evolutionarily stable strategy (ESS) exists for traditional IPD games, meaning that no prevalent strategy exists for extended IPD interactions [62]. Game theorists are less concerned with finding a dominant IPD strategy than with investigating relationships among strategies [92] and identifying conditions under which strategies become evolutionarily stable [61]. [91]’s model would be very interesting if X used more than one strategy or if X were also capable of evolving. Either case would result in complex evolutionary dynamics, underscoring the weaknesses of the analytical approach and emphasizing the strengths of the evolutionary approach.

My analytical approach to the problem described in [91] is a faster and easier alternative to the evolutionary approach. I also emphasized two important considerations that are frequently overlooked by users of the evolutionary approach: the importance of pre-run model analysis, and the need to make a conscious decision between an evolutionary or analytical approach to solving a problem. Some game theory considerations regarding the Prisoner’s Dilemma and other two-person matrix games are

also addressed.

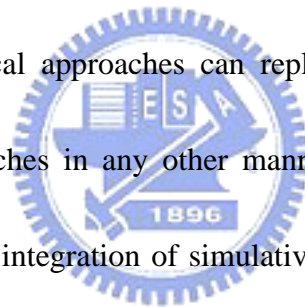
My approach can be expanded to solve more sophisticated problems. For example, in cases where X-type organisms use other kinds of strategies, y's score function may change; at a certain level of sophistication for X and y encoding, interactions between them may become too complex to be represented as a string. In such cases, a finite state machine representation may be useful for representing interactions between the two strategies [92]. Furthermore, my approach can be applied to other form of two-person matrix games (e.g., chicken games), and Prisoner's Dilemma derivatives (e.g., N-person or N-choice Prisoner's Dilemma). It is also important to investigate the extent to which the analytical approach is useful for non-deterministic strategies (strategies with slight chances of deviations in moves). For those sophisticated models, we believe a combination of analytical and evolutionary approaches may be more efficient than relying on either one alone. Further investigation is required to clarify the complementary properties of the two approaches.

### **3.3 Conclusion**

By analyzing various relationships among model components, model scope can be defined, model complexity reduced, and appropriate parameter settings identified—all

helpful in terms of increasing simulation efficiency. Theoretical analyses can reduce and/or complement the weaknesses of agent-based simulations and computations. For simple models, analysis is required to determine why a simulation is needed and why certain parameters should be chosen. For complex models, analysis reduces unnecessary work and guides the direction of a simulation toward discovery. An analysis of relationships between or among model components provides a global view of a model's scope, and helps to establish important strategies and appropriate parameter settings.

Although I want to emphasize the importance of pre-execution/simulation analysis, I am not claiming that analytical approaches can replace simulations, nor that they are superior to simulation approaches in any other manner. In each case, advantages and disadvantages are clear. The integration of simulative and analytical approaches will be an increasingly important topic in future execution/simulation studies for artificial societies.



# Chapter 4.

## The Analytical Framework for IPD Models

### 4.1 Definitions



In the framework described here, a finite state machine is used to represent interactions between deterministic memory- $n$  strategies. The finite state machines (sometimes referred to as finite automata) are dynamic systems that only change their behavior at discrete moments under consideration. The system consists of a finite set of internal states and a transition function. The transition function determines a subsequent system state as a function of the current state plus input. In [93], a formal definition of a finite state was given as:

**Definition 4.1** A finite-state machine (finite automata) consists of a 5-tuple  $(Q, \Sigma, q_0, \delta, A)$ ,

where

$Q$  is a finite set (whose elements we will think of as states),

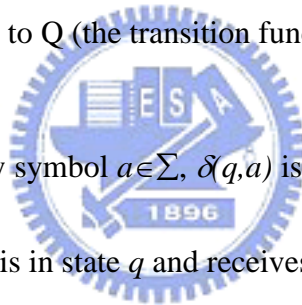
$\Sigma$  is a finite alphabet set of input symbols,

$q_0 \in Q$  (the initial state),

$A \subseteq Q$  (the set of accepting states), and

$\delta$  is a function from  $Q \times \Sigma$  to  $Q$  (the transition function).

For any element  $q$  of  $Q$  and any symbol  $a \in \Sigma$ ,  $\delta(q, a)$  is interpreted as the state to which the finite state machine moves if it is in state  $q$  and receives the input  $a$ .



A finite state machine can be represented in terms of a state transition diagram, described as:

**Definition 4.2** A finite state machine  $(Q, \Sigma, q_0, \delta, A)$  can be represented as a state transition diagram  $\{V, E\}$ , where

$V$  is a finite set of vertices, and each vertex represents a state in set  $Q$ ,

$E$  is a set of edges,

and  $(q_i, q_j) \in E$  if  $\delta(q_i, a) = q_j$ , where  $a \in \Sigma$ .

The diagram is a directed graph. The initial state  $q_0$  and the set of accepting states  $A$  are marked with specific notation. Each edge  $(q_i, q_j)$  is marked with an input symbol  $a$ , indicating that  $\delta(q_i, a) = q_j$ .

Regarding the memory- $n$  strategies, cooperative or defection moves are determined by the historical moves of two players. A memory- $n$  strategy determines a move in correspondence to moves made during the last  $n$  round, which is represented as a history string (HS). Formally defined,

**Definition 4.3** Let HS be the history string of  $S_i$  in its interaction with  $S_j$ , where  $S_i$  and  $S_j$  are memory- $n$  strategies. Then,



$$HS = h_{2n}h_{2n-1}\dots h_{k+1}h_k\dots h_2h_1.$$

The even and odd indices indicate the respective moves of a player and an opponent.

Here  $h_k \in \{C, D\}$ . C represents a cooperative move and D represents a defection move.

Note that in this case, HS' (the history string of  $S_j$ ) would be

$$HS' = h_{2n-1}h_{2n}\dots h_k h_{k+1}\dots h_1 h_2.$$

In other words, the two strategies have different history strings. In all, there are  $2^{2n}$  distinct history strings for a memory- $n$  deterministic strategy. For convenience, the



following formal definition of the order of history strings for a memory- $n$  strategy is offered.

**Definition 4.4**  $\{HS_{i/N} \mid 0 \leq i < N \text{ and } N=2^{2^n}\}$  is the set of all possible history strings for a memory- $n$  strategy, where  $HS_{i/N}$  is the  $i$ th string in the lexicographic order:

$$HS_{0/N} = CC \dots CC,$$

$$HS_{1/N} = CC \dots CD,$$

$$HS_{2/N} = CC \dots DC,$$

$$HS_{3/N} = CC \dots DD,$$

...

$$HS_{N-1/N} = DD \dots DD,$$



where  $|HS_{0/N}|=|HS_{1/N}|=\dots=|HS_{N-1/N}|=2n$ .

A memory- $n$  strategy is described in terms of the moves that are made according to their history strings.

**Definition 4.5.** A memory- $n$  deterministic strategy  $S$  is represented as:

$$S=(P_0, P_1, \dots, P_{N-2}, P_{N-1}),$$

where  $P_k \in \{C, D\}$ , and  $N=2^{2n}$ .  $P_k$  represents the move  $S$  corresponding to history string  $HS_{k/N}$ .

A memory-1 strategy is expressed as  $(P_0, P_1, P_2, P_3)$ , where  $P_0, P_1, P_2, P_3$  indicate moves when the results of the preceding round are CC, CD, DC, and DD, respectively. For example, strategy (C D C D) indicates a cooperative move if the preceding round is either CC or DC; a defection move is indicated in all other cases.

After finishing a round, the history strings of both strategies are updated by deleting the two leftmost characters, and adding the result of the next round to the right. For example, if a current history string is CCDDCD and the last round is DC, then the history string becomes DDCDDC. Interactions between player strategies can be viewed as history string transitions. Interactions between two strategies can be expressed as a finite state machine, where each state represents a specific history string, and where transitions between states are dependent upon transitions between the strategies' history strings.

**Proposition 4.1** Interactions between two memory- $n$  deterministic strategies can be represented as a finite state machine.

**Proof:**

Assume two memory- $n$  deterministic strategies,  $S_i$  and  $S_j$ , represented as

$$S_i = (P_0, P_1, \dots, P_{N-2}, P_{N-1}), \text{ and}$$

$$S_j = (P_0', P_1', \dots, P_{N-2}', P_{N-1}'),$$

where  $N=2^{2n}$  and  $P_k$  and  $P_k'$  represent the moves of  $S_i$  and  $S_j$  corresponding to history string  $HS_{k/N}$ .

Let  $FSM(S_i|S_j)$  be a finite state machine whose state transition diagram is:

$$D(S_i|S_j) = \{V, E\}.$$

The set of vertices  $V$  and the set of edges  $E$  are defined as

$$V = \{V_k \mid V_k \text{ is a vertex corresponding to history string } HS_{k/N}, 0 \leq k < N, N=2^{2n} \}, \text{ and}$$

$$E = \{(V_u, V_v) \mid V_u, V_v \in V,$$

where  $S_i$ 's history string may transit from  $HS_{u/N}$  to  $HS_{v/N}$  when interacting with  $S_j$ ,  
and  $0 \leq u, v < N \}$

In  $V$ ,  $V_k$  is the vertex corresponding to history string  $HS_{k/N}$ . Since there are  $N$  vertices, all possible history string permutations are contained.

In  $E$ , the edge  $(V_u, V_v)$  indicates that the history string of  $S_i$  will transit from  $HS_{u/N}$  to  $HS_{v/N}$ . All possible transitions are included.

Interactions between  $S_i$  and  $S_j$  can be represented by  $D(S_i|S_j)$  without loss of information. Thus, interactions between two memory- $n$  deterministic strategies can be represented as a finite state machine.

Note that the history strings in  $V$  and  $E$  refer to the history strings of  $S_i$ , which are different from those of  $S_j$ .  $D(S_i|S_j)$  represents the interaction between  $S_i$  and  $S_j$  from  $S_i$ 's perspective. The state transition diagram that represents the interaction between  $S_i$  and  $S_j$  from  $S_j$ 's perspective is denoted as  $D(S_j|S_i)$ .

□

**Definition 4.6** The finite state machine that represents interactions between two memory- $n$  deterministic strategies  $S_i$  and  $S_j$  from  $S_i$ 's perspective is denoted as  $FSM(S_i|S_j)$ . Its state transition diagram is  $D(S_i|S_j)$ . Thus,

$$D(S_i|S_j) = \{V, E\},$$

where

$$V = \{V_k \mid V_k \text{ is a vertex corresponding to history string } HS_{k/N}, 0 \leq k < N, \text{ and } N = 2^{2n}\}, \text{ and}$$

$$E = \{(V_u, V_v) \mid V_u, V_v \in V, \text{ where}$$

$S_i$ 's history string may transit from  $HS_{u/N}$  to  $HS_{v/N}$  when interacting with  $S_j$  for  $0 \leq u < N$  }.

In common finite state machines, state transitions depend on the current state and input symbols. However, the input symbol in  $FSM(S_i|S_j)$  is ignored; another way of saying this is that there is only one kind of input symbol, so the state transition depends only on the current state.

To give a simple example, if  $S_i = (C, D, C, D)$  and  $S_j = (C, D, D, C)$ , the interactions are represented as a finite state machine  $FSM(S_i|S_j)$ , whose state transition diagram  $D(S_i|S_j)$  is shown as Figure 4.1. If  $S_i$  and  $S_j$  both made defection moves in the previous round,  $S_i$  will make another defection move and  $S_j$  will make a cooperative move, resulting in a transition from state DD to state DC.

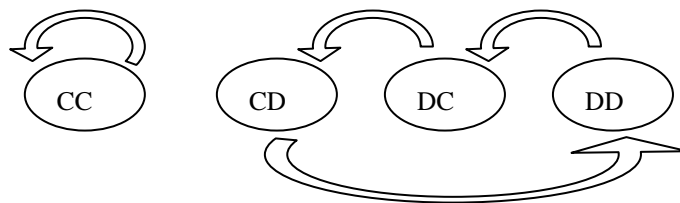


Figure 4.1: The Interaction between Strategies (C, D, C, D) and (C, D, D, C).

## 4.2 Behavior Characteristics of Infinite Duration

Finite state machines that represent interactions between two strategies have some interesting properties that are helpful in determining strategy behavior.

**Proposition 4.2** Let  $S_i$  and  $S_j$  be two memory- $n$  deterministic strategies. For the finite state machine  $FSM(S_i|S_j)$ , there is only one outgoing link for each state.

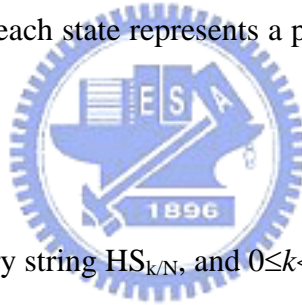
**Proof:**

Assume

$$S_i = (P_0, P_1, \dots, P_{N-2}, P_{N-1}) \text{ and}$$

$$S_j = (P_0', P_1', \dots, P_{N-2}', P_{N-1}'),$$

where  $N=2^{2n}$ . In  $FSM(S_i|S_j)$ , each state represents a particular history string. Let the set of states be



$$Q = \{q_k \mid q_k \text{ represents history string } HS_{k/N}, \text{ and } 0 \leq k < N \}.$$

The state transition is determined by the transition of history strings. If the current history string of  $S_i$  is  $HS_{k/N}$ , then

$$HS_{k/N} = h_{2n}h_{2n-1} \dots h_2h_1.$$

In the next round the history string of  $S_i$  will transit to

$$h_{2n-2}h_{2n-3} \dots h_2h_1P_kP_k'.$$

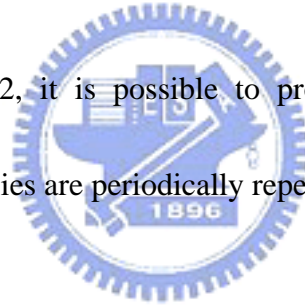
Since  $P_k, P_k' \in \{C, D\}$ , the history string  $h_{2n-2}h_{2n-3}\dots h_2h_1P_kP_k'$  is also an element in the set  $\{HS_{k/N} \mid 0 \leq i < N \text{ and } N=2^{2^n}\}$ . Assume that

$$HS_{m/N} = h_{2n-2}h_{2n-3}\dots h_2h_1P_kP_k'.$$

Each time  $S_i$  interacts with  $S_j$  and  $S_i$ 's history string is  $HS_{k/N}$ ,  $S_i$ 's history string will always transit to  $HS_{m/N}$ . This means that in  $FSM(S_i|S_j)$ , if the current state is  $q_k$ , the next state will always be  $q_m$ . Thus, there is only one outgoing link for each state in  $FSM(S_i|S_j)$ .

□

Following proposition 4.2, it is possible to prove that interactions between two memory- $n$  deterministic strategies are periodically repeated.



**Proposition 4.3** The behavior of a finite state machine with only one kind of input symbol and only one outgoing link for each state is periodically repeated for any initial state; the maximum loop length equals the number of total states.

**Proof:**

Assume that  $FSM$  is a finite state machine which has only one kind of input symbol and one outgoing link for each state. Let  $N$  be the number of states and  $q_s$  the initial state.

The sequence of state transition is

$$q_s q_{s+1} \dots q_{s+N-1} q_{s+N} \dots$$

According to the pigeonhole principle, since there are only  $N$  states, then  $q_{s+N} = q_{s+k}$  for some  $k$  which satisfies  $0 \leq k < N$ . The state transition sequence is

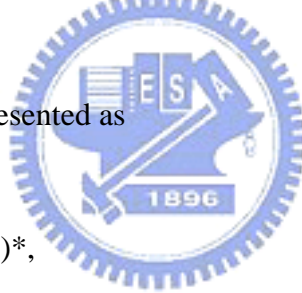
$$q_s q_{s+1} \dots q_{s+k} q_{s+k+1} \dots q_{s+N-1} q_{s+N} \dots$$

From Proposition 4.2, we have  $q_{s+k+1} = q_{s+N+1}$ ,  $q_{s+k+2} = q_{s+N+2}, \dots$ . The state transition sequence can therefore be expressed as

$$q_s q_{s+1} \dots q_{s+k} q_{s+k+1} \dots q_{s+N-1} q_{s+k} q_{s+k+1} \dots q_{s+N-1} \dots$$

The same sequence can be represented as

$$q_s q_{s+1} \dots (q_{s+k} q_{s+k+1} \dots q_{s+N-1})^*$$



where the asterisk indicates that the state transition is a repetition of  $(q_{s+k} q_{s+k+1} \dots q_{s+N-1})$ .

Thus, the machine behavior is periodically repeated.

Furthermore, by letting the length of the loop  $(q_{s+k} q_{s+k+1} \dots q_{s+N-1})$  be  $l$ , then  $l = N - k$ .

Since  $0 \leq k < N$ ,  $l$  satisfies  $0 < l \leq N$ , meaning that the maximum length of the loop is  $N$ .

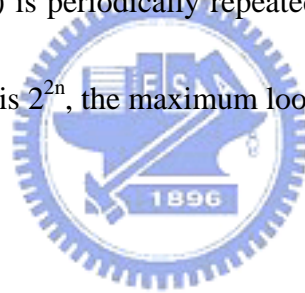
□



**Proposition 4.4** Interactions between two memory- $n$  deterministic strategies are periodically repeated for any initial condition; maximum loop length is  $2^{2n}$ .

**Proof:**

Let  $S_i$  and  $S_j$  be two memory- $n$  deterministic strategies.  $FSM(S_i|S_j)$  represents the interaction between  $S_i$  and  $S_j$  from  $S_i$ 's perspective. The input symbol in  $FSM(S_i|S_j)$  is ignored, or it can be said that there is only one kind of input symbol. According to Proposition 4.2, there is only one outgoing link for each state, and according to Proposition 4.3, the behavior of  $FSM(S_i|S_j)$  is periodically repeated. Furthermore, since the maximum number of states in  $FSM(S_i|S_j)$  is  $2^{2n}$ , the maximum loop length is also  $2^{2n}$ .



□

The next formal definition is for the loop that  $FSM(S_i|S_j)$  falls into for a specific initial state.

**Definition 4.7.**  $FSM(S_i|S_j)$  represents interactions between memory- $n$  strategies  $S_i$  and  $S_j$ . Its state transition diagram is  $D(S_i|S_j)$ .

Let  $L_k$  be the loop which  $D(S_i|S_j)$  will fall into if the initial vertex (initial state) is  $V_k$ .

Then,

$$L_k = (l_{k,1}, l_{k,2}, \dots, l_{k,|L_k|}) ,$$

where:

$|L_k|$  is the length of the loop, and

$$l_{k,m} \in V, \text{ for } 1 \leq m \leq |L_k|.$$

If the initial vertex is  $V_k$ , the machine will fall into the loop  $l_{k,1}, l_{k,2}, \dots, l_{k,|L_k|}, l_{k,1}$ .

If  $m \neq n$ , then  $l_{k,m} \neq l_{k,n}$ .

If the initial vertex is  $V_k$ , then after a sufficient period the probability that  $D(S_i|S_j)$  is at state  $l_{k,m}$  is  $1/|L_k|$ . Taking all initial states into account, the probability that it is at  $V_m$  at time  $t$  is



$$PS_m = \frac{\sum_{\forall k, V_m \in L_k} 1}{2^{2n}}, \text{ when } t \rightarrow \infty .$$

The traversal probability for each state is expressed as  $(PS_1, PS_2, \dots, PS_N)$ .

**Definition 4.8**  $FSM(S_i|S_j)$  is a finite state machine that represents interactions between strategies  $S_i$  and  $S_j$ . Its state transition diagram is  $D(S_i|S_j)$ . The traversal probability for each state is:

$$PS(S_i|S_j) = (PS_0, PS_1, \dots, PS_{N-1}), \text{ where}$$

$$PS_m = \frac{\sum_{\forall k V_m \in L_k} \frac{1}{|L_k|}}{2^{2n}}, \text{ when } t \rightarrow \infty,$$

and where  $L_k$  is the loop if the initial vertex (initial state) is  $V_k$ .

Recall that each state represents a specific history string whose final two characters represent the result of the previous round. Since these two characters indicate the payoff of the previous round, each state is mapped to one of the four payoffs. The traversal probability accumulates to derive the probability of each kind of payoff. That is,

$F_{CC} = PS_0 + PS_4 + \dots + PS_{N-4}$  is the probability that the result of the previous game was CC;



$F_{CD} = PS_1 + PS_5 + \dots + PS_{N-3}$  is the probability that the result of the previous game was CD;

$F_{DC} = PS_2 + PS_6 + \dots + PS_{N-2}$  is the probability that the result of the previous game was DC;  
or

$F_{DD} = PS_3 + PS_7 + \dots + PS_{N-1}$  is the probability that the result of the previous game was DD.

When  $S_i$  interacts with  $S_j$ , the *behavior characteristic* of  $S_i$  is defined as

**Definition 4.9** The behavior characteristic between  $S_i$  and  $S_j$  from  $S_i$ 's perspective is

$B(S_i|S_j) = (F_{CC}, F_{CD}, F_{DC}, F_{DD})$ , where

$$F_{CC} = PS_0 + PS_4 + \dots + PS_{N-4},$$

$$F_{CD} = PS_1 + PS_5 + \dots + PS_{N-3},$$

$$F_{DC} = PS_2 + PS_6 + \dots + PS_{N-2}, \text{ and}$$

$$F_{DD} = PS_3 + PS_7 + \dots + PS_{N-1}.$$

$(PS_0, PS_1, \dots, PS_{N-1})$  is the traversal probability for each state in  $FSM(S_i|S_j)$ .

From the behavior characteristic, we can see how the two strategies interact.  $F_{CC}$  indicates the probability that both will cooperate when they interact; if  $F_{CC}$  is high, so is the likelihood of cooperation between the two.



**Definition 4.10** The payoff characteristic between  $S_i$  and  $S_j$  from  $S_i$ 's perspective is

$$E(S_i|S_j) = F_{CC} \times R + F_{CD} \times S + F_{DC} \times T + F_{DD} \times P,$$

given that the payoff matrix is  $[R, S, T, P]$ .

The payoff characteristic is the expected payoff when two strategies interact. In previous studies, the expected payoff between stochastic strategies is usually derived via the Markov process. In stochastic strategies, initial conditions (initial moves) do not

affect the expected payoff derived by the Markov process. However, when dealing with deterministic strategies, initial moves do affect the expected payoff. The payoff characteristics of deterministic strategies take into account all possible initial moves. The payoff characteristics denote the interaction between two strategies without information loss.

### 4.3 Two Criteria for Investigating Strategy Properties

Based on the behavior characteristic  $B(S_i|S_j)$  and the payoff characteristic  $E(S_i|S_j)$ , two criteria were established for examining strategy properties: the ability to exploit others and the ability to form clone clusters.



#### 4.3.1 Ability to Exploit Others

Assume strategies  $S_i$  and  $S_j$ . Behavior characteristics between them are stated as:

$$B(S_i|S_j)=(F_{CC}, F_{CD}, F_{DC}, F_{DD}), \text{ and}$$

$$B(S_j|S_i)=(F_{CC}', F_{CD}', F_{DC}', F_{DD}').$$

Their payoff characteristics are:

$$E(S_i|S_j) = F_{CC} \times R + F_{CD} \times S + F_{DC} \times T + F_{DD} \times P, \text{ and}$$

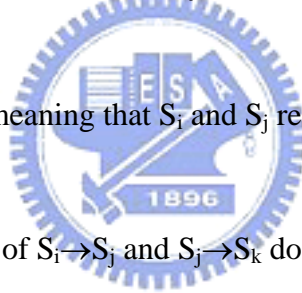
$$E(S_j|S_i) = F_{CC}' \times R + F_{CD}' \times S + F_{DC}' \times T + F_{DD}' \times P.$$

The ability to exploit others is determined by the relationship between  $E(S_i|S_j)$  and  $E(S_j|S_i)$ .

**Definition 4.11** When  $S_i$  interacts with  $S_j$ ,  $S_j$  is said to exploit  $S_i$  if  $E(S_i|S_j) < E(S_j|S_i)$ . This is expressed as:

$S_i \rightarrow S_j$  if  $E(S_i|S_j) < E(S_j|S_i)$ , meaning that  $S_j$  exploits  $S_i$ , and

$S_i = S_j$  if  $E(S_i|S_j) = E(S_j|S_i)$ , meaning that  $S_i$  and  $S_j$  receive the same payoff.



Note that the combination of  $S_i \rightarrow S_j$  and  $S_j \rightarrow S_k$  does not imply  $S_i \rightarrow S_k$ ; the relationship between  $S_i$  and  $S_k$  is determined by  $E(S_i|S_k)$  and  $E(S_k|S_i)$ . However,

$S_i \rightarrow S_j$  implies  $E(S_i|S_j) < E(S_j|S_i)$ , and

$S_j \rightarrow S_k$  implies  $E(S_j|S_k) < E(S_k|S_j)$ .

From these two inequalities only, nothing is known about  $E(S_i|S_k)$  and  $E(S_k|S_i)$ ; the relationship between  $S_i$  and  $S_k$  remains undetermined.  $S_i \rightarrow S_j$ ,  $S_j \rightarrow S_k$ , and  $S_i \rightarrow S_k$  denote  $S_i \rightarrow S_j \rightarrow S_k$ .

The *exploitation chain* is used to indicate relationships among more than two strategies.

**Definition 4.12.** An *exploitation chain* is expressed as:

$$S_{C_1} \rightarrow S_{C_2} \rightarrow \dots \rightarrow S_{C_k}$$

Strategy  $S_{C_i}$  exploits  $S_{C_j}$  if  $j < i$ .

### 4.3.2 Ability to Form Clone Clusters

The ability of strategies to form clone clusters is determined by their behavior characteristics with their clones—i.e.,  $E(S_i|S_i)$ . If  $E(S_i|S_i)$  is high, then two players using the same strategy  $S_i$  will receive relatively higher payoffs when they interact. This effect is more apparent in spatial models, in which strategies only interact with their neighbors [65-68].

According to Smith's [60] definition,  $S_i$  will invade  $S_j$  if

$$E(S_i|S_i) > E(S_j|S_i), \text{ or if}$$

$$E(S_i|S_i) = E(S_j|S_i) \text{ and } E(S_i|S_j) > E(S_j|S_j).$$

The invasion relationship between the two strategies is determined by  $E(S_i|S_i)$ ,  $E(S_j|S_j)$ ,  $E(S_j|S_i)$ , and  $E(S_i|S_j)$ . A rather complex analytical task results, since the following are all possible:

$S_i$  invades  $S_j$  and  $S_j$  invades  $S_i$ ,

$S_i$  invades  $S_j$  and  $S_j$  can't invade  $S_i$ ,

$S_i$  can't invade  $S_j$  and  $S_j$  invades  $S_i$ , or

$S_i$  can't invade  $S_j$  and  $S_j$  can't invade  $S_i$ .

Furthermore,  $S_i$ 's invasion of  $S_j$  and  $S_j$ 's invasion of  $S_k$  do not imply that  $S_i$  will invade  $S_k$ . As the number of strategies increases, relationships among them eventually become too complex to analyze.

According to the proposed framework, the relationships between  $E(S_i|S_j)$  and  $E(S_j|S_i)$  and between  $E(S_i|S_i)$  and  $E(S_j|S_j)$  are investigated separately. Complex invasion relationships can be divided into two categories, which makes it easier to clarify relationships among the strategies, and which helps investigators to find and analyze hidden properties.



## 4.4 Conclusion

The three most important properties of the framework are:

a) It remains independent of payoff matrix values as long as they satisfy the constraint of the Prisoner's Dilemma.

b) Interactions between strategies can be represented as a finite state machine. This kind of representation also provides an efficient means for deriving the expected payoff between deterministic strategies—a task that is difficult by using the Markov process because the result of deterministic strategies is sensitive to the initial state.

c) A strategy's ability to exploit others and to form clone clusters provides stepping-off points for further analysis and simulation. Using these two criteria, relations among strategies can be identified.



# Chapter 5.

## Analysis of Memory-1 Deterministic Strategies in IPD



### 5.1 Exploitation Relation

A memory-1 strategy is notated as  $(P_0, P_1, P_2, P_3)$ . The  $2^4$  deterministic strategies are named  $S_0, S_1, \dots, S_{15}$ , representing  $(C, C, C, C)$ ,  $(C, C, C, D)$ ,  $\dots$ , and  $(D, D, D, D)$ , respectively. The relation among strategies' ability to exploit others is known as the *exploitation relation*. Exploitation relations between paired memory-1 deterministic strategies are listed in Table 5.1.

Assume an element  $T_{ij}$  in row  $i$  and column  $j$ .

$T_{ij}=0$  if  $S_i=S_j$ ,

$T_{ij}=1$  if  $S_j \rightarrow S_i$ , and

$T_{ij}=-1$  if  $S_i \rightarrow S_j$ .

**Table 5.1: Exploitation Relations between Paired Memory-1 Strategies**

	$S_0$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$	$S_6$	$S_7$	$S_8$	$S_9$	$S_{10}$	$S_{11}$	$S_{12}$	$S_{13}$	$S_{14}$	$S_{15}$
$S_0$	0	0	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
$S_1$	0	0	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
$S_2$	1	1	0	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
$S_3$	1	1	1	0	1	0	0	-1	1	0	0	-1	0	-1	-1	-1
$S_4$	0	0	0	-1	0	0	0	-1	0	0	-1	-1	0	0	-1	-1
$S_5$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$S_6$	1	1	1	0	0	0	0	-1	1	1	0	-1	0	0	-1	-1
$S_7$	1	1	1	1	1	0	1	0	1	1	1	1	1	0	1	0
$S_8$	1	1	1	-1	0	0	-1	-1	0	-1	-1	-1	0	-1	-1	-1
$S_9$	1	1	1	0	0	0	-1	-1	1	0	0	-1	0	0	-1	-1
$S_{10}$	1	1	1	0	1	0	0	-1	1	0	0	-1	0	-1	-1	-1
$S_{11}$	1	1	1	1	1	0	1	-1	1	1	1	0	1	0	1	-1
$S_{12}$	1	1	1	0	0	0	0	-1	0	0	0	-1	0	0	0	-1
$S_{13}$	1	1	1	1	0	0	0	0	1	0	1	0	0	0	0	0
$S_{14}$	1	1	1	1	1	0	1	-1	1	1	1	-1	0	0	0	-1
$S_{15}$	1	1	1	1	1	0	1	0	1	1	1	1	1	0	1	0

### 5.1.1 Strategy Classification

**Definition 5.1.** Every strategy ( $S_k$ ) has three sets:

$$\text{WIN}_k = \{S_i \mid S_i \rightarrow S_k\},$$

$$\text{LOSE}_k = \{S_i \mid S_k \rightarrow S_i\}, \text{ and}$$

$$\text{DRAW}_k = \{S_i \mid S_k = S_i\}.$$

$\text{WIN}_k$  is a set of strategies exploited by  $S_k$ .  $\text{LOSE}_k$  is a set of strategies that may exploit  $S_k$ .  $\text{DRAW}_k$  contains strategies that result in equal payoffs with  $S_k$ .

The exploitation relations among these 16 strategies are constructed by  $\text{WIN}_k$ ,  $\text{LOSE}_k$ , and  $\text{DRAW}_k$  ( $0 \leq k \leq 15$ ). Strategies are said to be equivalent if they have the same  $\text{WIN}$ ,  $\text{LOSE}$ , and  $\text{DRAW}$  sets. There are three sets of equivalent strategies:  $\{S_0, S_1\}$ ,  $\{S_3, S_{10}\}$ , and  $\{S_7, S_{15}\}$ . The exploitation relations among them form an exploitation chain, shown as  $\{S_0, S_1\} \rightarrow \{S_3, S_{10}\} \rightarrow \{S_7, S_{15}\}$ .

**Statement 5.1**  $\{S_0, S_1\}$ ,  $\{S_3, S_{10}\}$ , and  $\{S_7, S_{15}\}$  are sets of equivalent strategies, and the exploitation chain they form is  $\{S_0, S_1\} \rightarrow \{S_3, S_{10}\} \rightarrow \{S_7, S_{15}\}$ .

### 5.1.2 Exploitation Chains

Based on Statement 5.1:

**Statement 5.2**  $\{S_0, S_1\} \rightarrow S_2 \rightarrow S_8 \rightarrow \{S_3, S_{10}\} \rightarrow S_{11} \rightarrow S_{14} \rightarrow \{S_7, S_{15}\}$ .

**Statement 5.3**  $S_4 \rightarrow \{ S_3, S_{10} \} \rightarrow S_{11} \rightarrow S_{14} \rightarrow \{ S_7, S_{15} \}$ .

**Statement 5.4**  $\{ S_0, S_1 \} \rightarrow S_2 \rightarrow S_8 \rightarrow \{ S_3, S_{10} \} \rightarrow S_{13}$ .

**Statement 5.5**  $\{ S_0, S_1 \} \rightarrow S_2 \rightarrow S_{12} \rightarrow S_{11} \rightarrow \{ S_7, S_{15} \}$ .

**Statement 5.6**  $\{ S_0, S_1 \} \rightarrow S_2 \rightarrow S_8 \rightarrow S_9 \rightarrow S_6 \rightarrow S_{11} \rightarrow S_{14} \rightarrow \{ S_7, S_{15} \}$ .

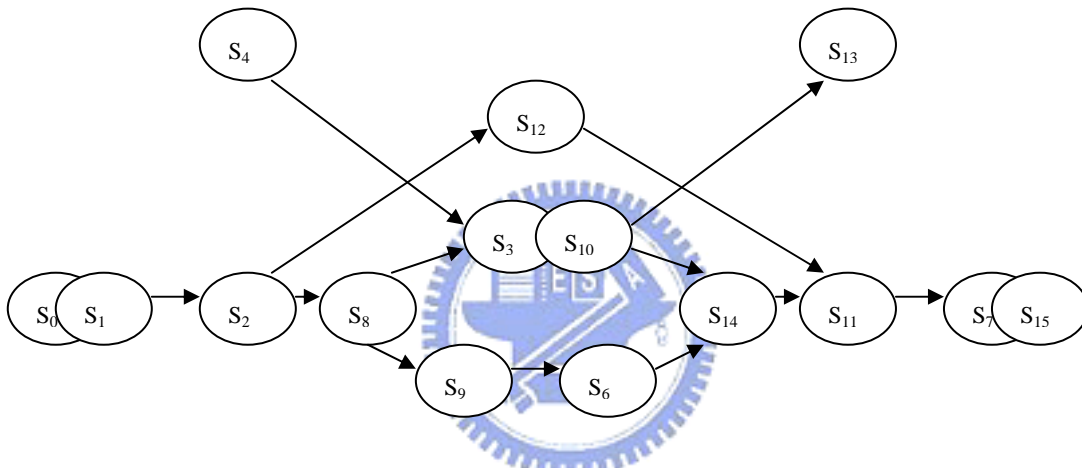
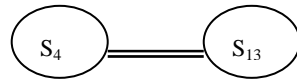


Figure 5.1: Exploitation Chains of Memory-1 Deterministic Strategies.

Relations among these strategies are shown in Figure 5.1. In most cases, if a path exists from  $S_i$  to  $S_j$ , then  $S_i \rightarrow S_j$ . There is one exception: a path exists from  $S_4$  to  $S_{13}$ , but  $S_4 \neq S_{13}$ . This exception is described in the next section.

### 5.1.3 The Draw Case

All relations not specified in the preceding section are considered draw relations, meaning that two interacting strategies will receive the same payoff if no path exists between them. A path does exist from  $S_4$  to  $S_{13}$ , but to describe the exception that  $S_4=S_{13}$ , a link must be added between the two strategies. In the case of  $S_i$  and  $S_j$ , that link is  $S_i=S_j$ . The exploitation relation between  $S_4$  and  $S_{13}$  is shown below.



$S_5$ , which receives the same payoff when it interacts with any other memory-1 deterministic strategy, is shown as an isolated vertex in Figure 5.2. This is the well-known strategy Tit-for-Tat.

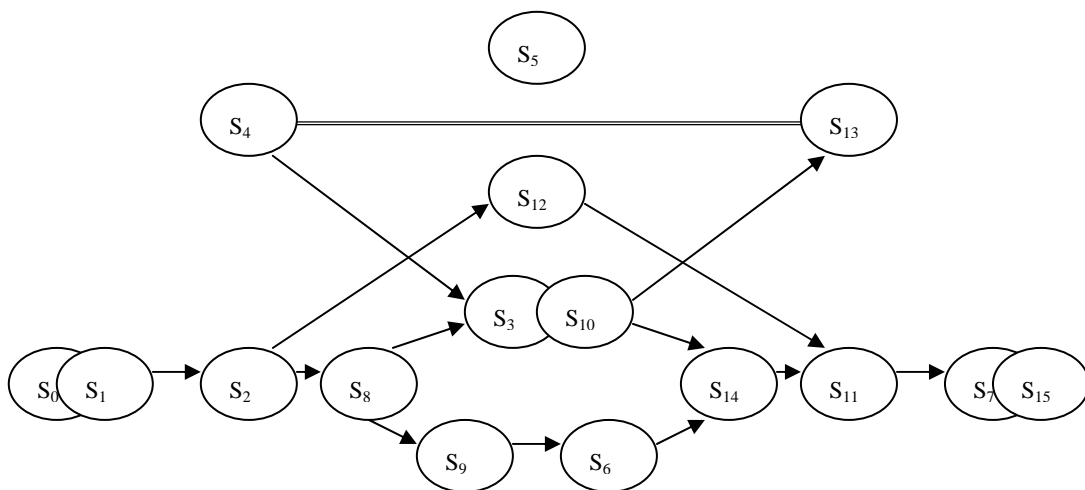


Figure 5.2: The Exploitation Relation among Memory-1 Deterministic Strategies.

The exploitation relation of all memory-1 deterministic strategies is shown in Figure

5.2. The principle for determining the ability of  $S_i$  and  $S_j$  to exploit others is:

$S_i \rightarrow S_j$  if there is a traversal path from  $S_i$  to  $S_j$ ;

$S_j \rightarrow S_i$  if there is a traversal path from  $S_j$  to  $S_i$ ; and

$S_i = S_j$  if there is no path from  $S_i$  to  $S_j$ , or if there is a '=' link between them.

## 5.2 Clustering Relation



The relation of the ability of strategies to form clone clusters is known as the clustering relation. Strategy  $S_i$ 's ability to cluster is determined by  $E(S_i|S_i)$ , the payoff characteristic resulting from interactions with its clones. Table 5.2 presents a list of  $E(S_i|S_i)$  for all memory-1 deterministic strategies.

Recall that  $R$ ,  $S$ ,  $T$ , and  $P$  in the payoff matrix of the Prisoner's Dilemma satisfy  $T > R > P > S$ , and that  $2R > S + T$ . According to this constraint, the clustering relations of all memory-1 deterministic strategies are:

If  $S + T > R + P$ :

$S_9, S_{15} < S_7 < S_{11}, S_{13} < S_{14}, S_8 < S_3, S_5, S_{10}, S_{12} < S_1 < S_2, S_4 < S_0, S_6.$

If  $S+T < R+P$ :

$S_9, S_{15} < S_{11}, S_{13} < S_7 < S_3, S_5, S_{10}, S_{12} < S_{14}, S_8 < S_2, S_4 < S_1 < S_0, S_6.$

If  $S+T = R+P$ :

$S_9, S_{15} < S_7, S_{11}, S_{13} < S_3, S_5, S_8, S_{10}, S_{12}, S_{14} < S_1, S_2, S_4 < S_0, S_6.$

**Table 5.2:  $B(S_i|S_i)$  and  $E(S_i|S_i)$  for Memory-1 Deterministic Strategies**

	$B(S_i S_i)$	$E(S_i S_i)$		$B(S_i S_i)$	$E(S_i S_i)$
$S_0$	$(1, 0, 0, 0)$	R	$S_8$	$(\frac{1}{2}, 0, 0, \frac{1}{2})$	$\frac{R+P}{2}$
$S_1$	$(\frac{3}{4}, 0, 0, \frac{1}{4})$	$\frac{3R+P}{4}$	$S_9$	$(0, 0, 0, 1)$	P
$S_2$	$(\frac{1}{2}, \frac{1}{4}, \frac{1}{4}, 0)$	$\frac{2R+S+T}{4}$	$S_{10}$	$(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4})$	$\frac{R+S+T+P}{4}$
$S_3$	$(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4})$	$\frac{R+S+T+P}{4}$	$S_{11}$	$(0, \frac{1}{4}, \frac{1}{4}, \frac{1}{2})$	$\frac{S+T+2P}{4}$
$S_4$	$(\frac{1}{2}, \frac{1}{4}, \frac{1}{4}, 0)$	$\frac{2R+S+T}{4}$	$S_{12}$	$(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4})$	$\frac{R+S+T+P}{4}$
$S_5$	$(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4})$	$\frac{R+S+T+P}{4}$	$S_{13}$	$(0, \frac{1}{4}, \frac{1}{4}, \frac{1}{2})$	$\frac{S+T+2P}{4}$
$S_6$	$(1, 0, 0, 0)$	R	$S_{14}$	$(\frac{1}{2}, 0, 0, \frac{1}{2})$	$\frac{R+P}{2}$
$S_7$	$(\frac{1}{4}, 0, 0, \frac{3}{4})$	$\frac{R+3P}{4}$	$S_{15}$	$(0, 0, 0, 1)$	P



The relations between  $\{S_1\}$  and  $\{S_2, S_4\}$ ,  $\{S_8, S_{14}\}$  and  $\{S_3, S_5, S_{10}, S_{12}\}$ , and  $\{S_7\}$  and  $\{S_{11}, S_{13}\}$  are determined by the value of (S+T) and (R+P). In this chapter, the focus is on relations that are independent of the value of the payoff matrix; thus,  $\{S_1\}$  and  $\{S_2, S_4\}$  are viewed as one set of strategies, as are  $\{S_8, S_{14}\}$  and  $\{S_3, S_5, S_{10}, S_{12}\}$ , and  $\{S_7\}$  and  $\{S_{11}, S_{13}\}$  (Figure 5.3).

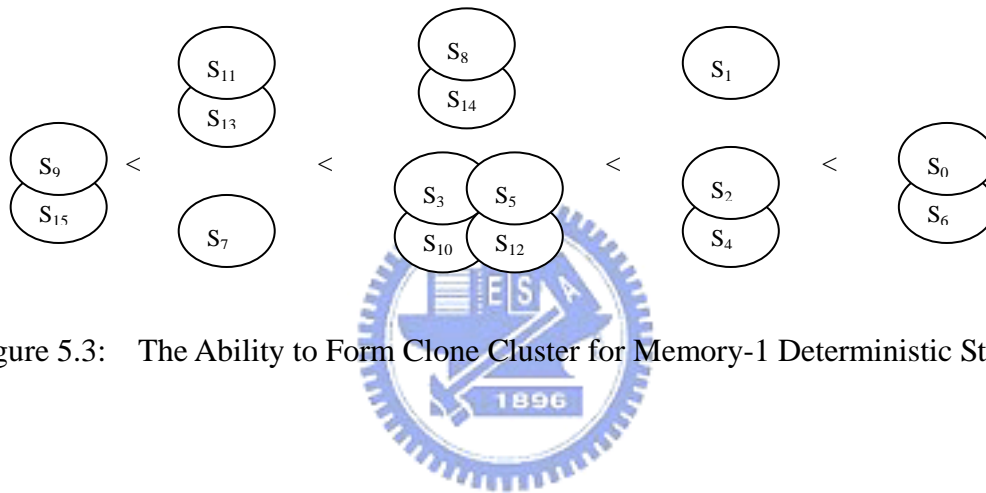


Figure 5.3: The Ability to Form Clone Cluster for Memory-1 Deterministic Strategies.

It can be argued that the use of a value-independent analysis contradicts one of our stated reasons for performing a pre-simulation analysis in Chapter 3—that is, defining model scope—since model scope may depend on the payoff matrix value. The payoff matrix values indeed may affect the analytical result of relation between strategies. However, I firmly believe that neglecting to perform an analysis of relations that are independent of payoff matrix values would make it difficult to determine whether results were the consequence of the IPD problem nature or payoff matrix values. In short, a

value-independent analysis highlights relations based on the native properties of the IPD model, while a value-dependent analysis emphasizes how different values affect exploitation and clustering relations.

## 5.3 Properties of Memory-1 Deterministic Strategies

### 5.3.1 Relation among Memory-1 Deterministic Strategies

Before examining the important strategies revealed in Figure 5.2 and 5.3, relations between the two criteria are considered. A good strategy should exploit others in order to receive a higher payoff, and its clone cluster should be strong enough to prevent invasion of others. These two criteria are equally important. Strategies that easily exploit others but fail to form strong clone clusters will likely spread throughout an environment, but when that environment is saturated with those strategies, they will become susceptible to invasion by other strategies. For example,  $S_{15}$  (always defects, regardless of the opponent's move, also referred as ALLD) is a typical strategy of this kind. Furthermore, strategies with a strong ability to form clone clusters but that are easily exploited by others cannot survive, since their being exploited by others usually occurs before their clone clusters are formed;  $S_0$  (always cooperates, regardless of the opponent's move, also referred as ALLC) is one of this kind of strategies.

The order of strategies in Figure 5.2 is almost perfectly opposite to that in Figure 5.3, showing that strategies that exploit others tend to have difficulty getting along with their clones; on the other hand, a strategy that gets along well with its clones has a higher chance of being exploited. In memory-1 strategies, strength in one criterion usually implies weakness in the other. Thus, mid-order strategies in both Figure 5.2 and 5.3 are considered important because they show a certain degree of strength in both criteria. These strategies have attracted the greatest attention from Prisoner's Dilemma researchers [53]; their names are listed in Table 5.3.



**Table 5.3: Some Commonly Discussed Memory-1 Deterministic Strategies**

<b>Strategy</b>	<b>Name</b>	<b>Representation</b>	<b>Description</b>
$S_3$	Stubborn	(C, C, D, D)	Repeats the first round move regardless of the opponent's move.
$S_5$	Tit-for-Tat	(C, D, C, D)	Repeats the opponent's previous move.
$S_{10}$	Bully	(D, C, D, C)	Defects against a cooperator, but cooperates if punished.
$S_{12}$	Fickle	(D, D, C, C)	Changes the strategy each round regardless of the opponent's move.

In addition to their greater survival potential, these mid-order strategies serve as a dividing line between nice and non-nice strategies. For instance,  $S_3$ ,  $S_5$ ,  $S_{10}$ , and  $S_{12}$  are located in the middle of the two orders shown in Figures 5.2 and 5.3. These strategies divide the two lists into perfectly symmetric parts. Most of the strategies on the left side of Figure 5.2 and the right side of Figure 5.3 will cooperate in instances where the last round is CC—an indication that they are “nice” strategies that will never be the first to defect [46]. Most strategies on the right side of Figure 5.2 and left side of Figure 5.3 are non-nice.

Some exceptions can be found in Figure 5.2 and 5.3.  $S_7$  is a nice strategy, but is located on the right side of Figure 5.2 and the left side of Figure 5.3.  $S_6$  and  $S_9$  are located in mid-order in Figure 5.2, but in Figure 5.3,  $S_6$  is on the right side and  $S_9$  the left. Because each of them has at least one special property, those exceptional strategies have become targets of numerous and extensive investigations.  $S_6$  is the well-known strategy PAVLOV [51], and  $S_7$  is the strategy known as RETALIATOR [65].

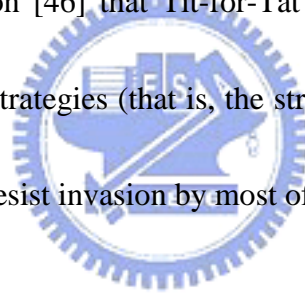
### 5.3.2 Discussion of Well-Known Strategies

Information concerning relations among memory-1 strategies is presented in Figure 5.2 and 5.3. In this section, I will discuss two memory-1 strategies that are located in particular positions within our proposed framework: Tit-for-Tat and PAVLOV. The

purpose of this section is to show that the properties of important strategies are explained well by the framework.

### **Tit-for-Tat:**

$S_5$  is the strategy: Tit-for-Tat. Its isolation from other strategies in Figure 5.2 means that it can't exploit other strategies, nor is it easily exploited by others. According to the proposed framework, the success of Tit-for-Tat strategy results from its a) ability to attain equal chances of survival with any other strategy, and b) strong ability to cluster. This is similar to Axelrod's conclusion [46] that Tit-for-Tat strategy can increase in frequency among predominantly ALLD strategies (that is, the strategy that always defects regardless of the opponent's move), and resist invasion by most of defective strategies (see also [86]).



### **PAVLOV:**

$S_6$  is the well-studied strategy known as PAVLOV [51]. Among all memory-1 deterministic strategies, it is the only one whose exploitation and clustering abilities are both higher than average. According to Nowak [51], PAVLOV's advantage over Tit-for-Tat is the result of two features: a) it can correct occasional mistakes and b) it resists invasion by strict cooperators. From the information in Figure 5.2 and 5.3, the first property is not observable, because occasional mistakes cannot occur in deterministic strategies. However, the second property is obvious because  $S_6$ 's ability to exploit others is

higher than that found in most nice strategies. At the same time, its ability to form clone clusters is the strongest among all memory-1 deterministic strategies, which explains why it can resist invasion by strict cooperators. Another important property for PAVLOV is that it loses against ALLD, which distinguishes it from Tit-for-Tat, which draws with all other strategies. The reason for this is that PAVLOV alternates between cooperation and defection [51]. This is explained well in Figure 5.2.

Figure 5.2 and Figure 5.3 shows significant information about the properties of memory-1 deterministic strategies. The properties of important strategies and particular evolutionary phenomenon are well explained. In the investigation of Prisoner's Dilemma, before further analysis and simulation is taken, the purposed framework should be used in advance to understand the properties of strategies, and the relationship among strategies.

The identification of important strategies and common phenomena by this proposed framework underscores the point of Chapter 3: some simulations are not necessary if appropriate analytical procedures are followed. This is not to say that analysis can completely replace simulation, which is required in order to verify predicted phenomena. The key word here is *verification* and not *discovery*, the difference being the amount of required simulation work. If the purpose of simulation is to verify certain properties, fewer model instances are needed; if the purpose is to discover unknown phenomena, more

model instances are needed to reflect as many conditions as possible. In either case, theoretical analysis helps reduce unnecessary simulation efforts and provides guidance toward anticipated results.



# Chapter 6.

## Application of This Framework

### 6.1 Cyclic Relation among Three Strategies



An important property of the framework proposed in this dissertation is its directed graph representation of exploitation relations. In this section I will discuss the relations among cyclical strategy sets.

Assume three strategies—A, B, and C; their cyclical exploitation relationships are shown in Figure 6.1

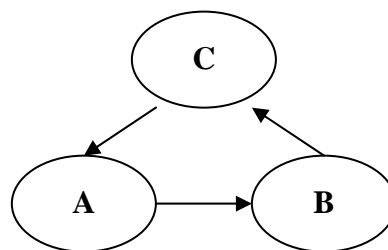


Figure 6.1: Cyclical Exploitation Relationships among 3 Strategies.



In environments consisting of A and B only, B will always exploit A. If C appears in the same environment, B will become less dominant because C can exploit it. Since A can exploit C, the arrival of C increases A's and decreases B's survival potential.

Note that no cycle exists within any subset of the 16 memory-1 strategies. However, the “=” exploitation relationship between certain strategies allows them to form various types of “semi-cycles.” For strategy  $A \rightarrow B$ , assume that for any other strategy C there are  $3^2$  possible combinations (Fig. 6.2).

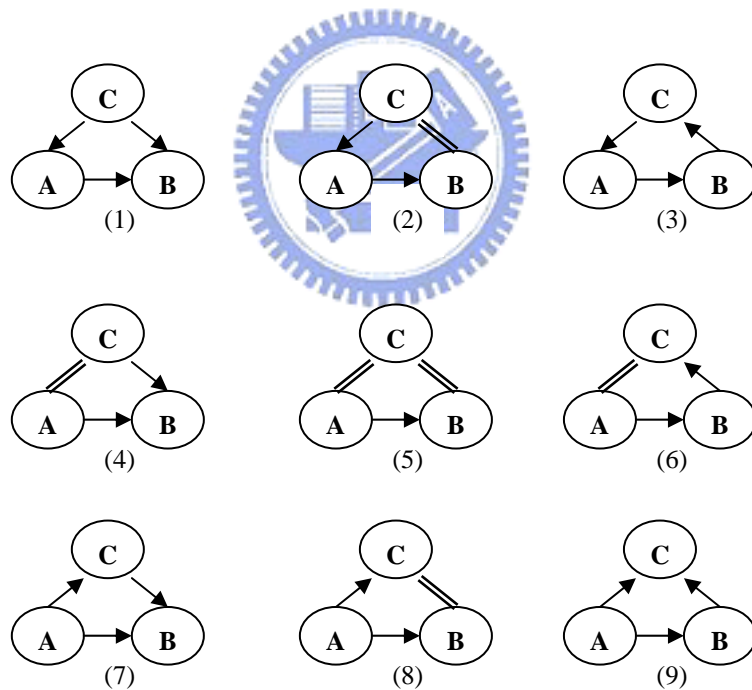


Figure 6.2:  $3^2$  Possible Combinations of Strategy Relationships between A, B, and C, Given  $A \rightarrow B$ .

The effect of C can be categorized as follows:

a) The appearance of C increases the possibility of A invading B. In the figure, examples of this type are cases 2, 5, and 6.

b) The appearance of C affects the amount of time required for the invasion, but it does not prevent B's invasion of A. Cases 1, 4, 7, 8, and 9 are examples of this type.

The three cases in the first category form “semi-cycles.” In terms of behavior, they share certain commonalities with strategies that form complete cycles, that is, invasion relationships between strategies are altered. Cases 2 and 6 appear to be intuitive. In case 2, A's survival capability increases due to its ability to exploit C. In case 6, B's survival capability is decreased because it can be exploited by C. One of the most well-known example in case 5 is the relation between ALLC, ALLD, and Tit-for-Tat. The relations between them are:  $ALLC \rightarrow ALLD / ALLD = \text{Tit-For-Tat} / \text{Tit-For-Tat} = ALLC$ . In the literature, it is well known that the appearance of Tit-for-Tat prevents ALLC from being invaded by ALLD. When dealing with strategies, it is important to know whether the appearance of another strategy will affect the invasion relation among strategies that already exist in the environment.

## 6.2 Framework Generalization

### 6.2.1 Analyses of Other Deterministic Strategies

The framework described in this dissertation can be applied to deterministic strategies encoded in other forms, such as finite automata [55, 74] and rule sets [56]. The key step is deriving an appropriate finite state machine to represent interactions between the two strategies.

**Proposition 6.1** Interactions between two deterministic strategies can be represented as a finite state machine whose behavior is periodically repeated.



**Proof:**

Deterministic strategies can be notated as  $(P_1, P_2, \dots, P_n)$ , where each  $P_k$  refers to moves corresponding to specific conditions. For example, history strings in the case of memory- $n$  strategies. Assume two strategies  $S_i$  and  $S_j$ ; then,

$$S_i = (P_1, P_2, \dots, P_n) \text{ and}$$

$$S_j = (Q_1, Q_2, \dots, Q_m),$$

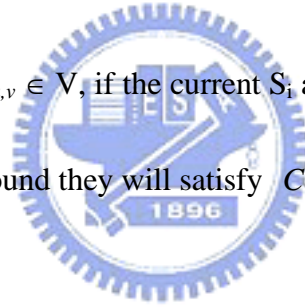
where  $P_k$  is the  $S_i$  move if its condition satisfies  $Cond_{P_k}$ , and  $Q_k$  is the  $S_j$  move if its condition satisfies  $Cond_{Q_k}$ . Since  $S_i$  and  $S_j$  are deterministic strategies,  $n$  and  $m$  are finite numbers.

For a finite state machine  $FSM(S_i|S_j)$  whose state transition diagram is  $D(S_i|S_j)$ ,

$D(S_i|S_j)=(V,E)$ , where

$V=\{V_{s,t} \mid V_{s,t}$  represents the condition that  $S_i$  satisfies  $Cond_{P_s}$  and  $S_j$  satisfies  $Cond_{Q_t}$ , when  $1 \leq s \leq n$  and  $1 \leq t \leq m \}$ , and

$E = \{(V_{s,t}, V_{u,v}) \mid V_{s,t}, V_{u,v} \in V$ , if the current  $S_i$  and  $S_j$  satisfy  $Cond_{P_s}$  and  $Cond_{Q_t}$ , respectively, then in the next round they will satisfy  $Cond_{P_u}$  and  $Cond_{Q_v} \}$ .

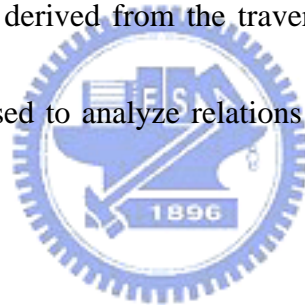


Since  $n$  and  $m$  are finite numbers, the number of vertices (states) in  $V$  is finite. The  $E$  set of edges indicates the transition between vertices; all possible transitions between conditions are included in  $E$ . When  $S_i$  interacts with  $S_j$ , it is possible to observe how  $S_i$  and  $S_j$  decide their moves from the state transition diagram  $D(S_i|S_j)$ . Interactions between deterministic strategies  $S_i$  and  $S_j$  can therefore be represented without loss of information. These interactions can also be represented as a finite state machine,  $FSM(S_i|S_j)$ , whose state transition diagram is  $D(S_i|S_j)$ .

Since  $D(S_i|S_j)$  represents interactions between  $S_i$  and  $S_j$ , the input symbol of  $D(S_i|S_j)$  is ignored—in other words, there is only one type of input symbol. Also, since  $S_i$  and  $S_j$  are deterministic strategies, each vertex has only one outgoing link. From Proposition 4.3, the behavior of  $D(S_i|S_j)$  is periodically repeated for each initial vertex (state).

□

According to Proposition 6.1, the traversal probability of each state can be derived once  $FSM(S_i|S_j)$  is constructed. Since each state represents one of the four payoffs, the behavior characteristic can be derived from the traversal probability of each state. The framework can therefore be used to analyze relations among deterministic strategies that are encoded in different forms.



### 6.2.2 Analyses of Other $2 \times 2$ matrix games

The framework can also be applied to other  $2 \times 2$  matrix games—for example, “chicken games” that are very similar to the Prisoner’s Dilemma except for slight differences in their payoff matrix value constraints (i.e.,  $T > R > S > P$ ).

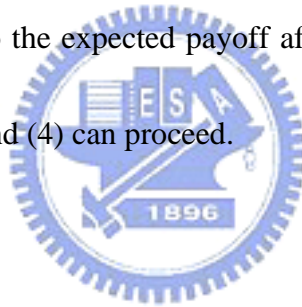
### **6.2.3 Application of Spatial and Biased Selection IPD Models**

The framework can also be applied to spatial IPD models, which have different emergence behaviors than non-spatial models due to an opponent selection bias—that is, players must be neighbors. Assume an environment restricted to ALLC and ALLD players. For all ALLC players in a non-spatial model, the probabilities of meeting another ALLC player are equal. In contrast, in a spatial model containing clusters of ALLC players, the probability of a player in the center of a cluster to meet an ALLC player would be 100%, since all of its neighbors would be ALLC players. The proposed framework addresses interaction patterns between two players regardless of the spatial/non-spatial status of the model in question. It can be applied to spatial models to better understand interaction patterns between strategies, but more research is required on how players contact and select their opponents.

### **6.2.4 Application on Other Artificial Society Models**

The framework's methodology is summarized as follows: (1) identify interaction patterns between two agents, (2) use these patterns to determine the expected payoff for each agent, (3) identify relationships between the agents according to their expected payoffs, and (4) construct strategy relations based on pair-wise relations. The primary

purpose of this methodology is: to analyze a model from the perspective of interactions between two individuals. While it can be applied to other types of agent-based and macro simulations, interaction patterns may be very different from those observed in the Prisoner's Dilemma. For multi-agent systems whose agents have deterministic behaviors, our finite state machine representation for pattern interaction is applicable. For non-deterministic multi-agent systems, it would be inappropriate (or impossible) to use our finite state machine representation at step (1). However, regardless of how two agents interact, the status of either agent changes following an interaction. The change of agent status may be transformed into the expected payoff after interaction. Once the expected payoff is identified, steps (3) and (4) can proceed.



# Chapter 7.

## Effect of Environment on Artificial Societies



### 7.1 Introduction

Recall Table 2.1 in Chapter 2, where popular artificial societies for problem solving are listed according to their characteristics of artificial societies' three primary components. With few exceptions, every type of artificial society is strongly affected by its environment: in classifier systems, agents take information from their environments, ants in ant colony systems leave information in their environments, and agents in multi-agent systems do both. In the absence of environmental considerations, it is impossible for problem-solving artificial societies to accomplish their goals.



Nevertheless, environmental factors are rarely mentioned in the IPD literature. Researchers who study the most popular strategies (e.g., Tit-for-Tat and PAVLOV) tend to limit their efforts to player status, thereby ignoring overall system or individual agent relationships with their environments when predicting or determining player moves. The effect of environment on IPD model strategy design is the focus of this chapter, based on the assumption that a better understanding of the relationship between environment and strategy design will be beneficial when applying IPD models in artificial societies used for problem solving.

## 7.2 Model Description



To address the issue of how environmental factors affect IPD strategy, I used a spatial PD model to compare the performances of strategies that do and do not acknowledge and consider environmental conditions. The spatial PD model consists of a  $N \times N$  matrix in which each cell contains an agent that can be represented by its position within the environment (e.g.,  $A_{i,j}$  and its neighbors, as shown in Figure 7.1). During each generation, an agent interacts with each of its eight neighbors  $T$  times; an agent score represents the sum of eight payoffs. If  $P_T(A_s|A_t)$  is defined as the average payoff of  $A_s$  from  $T$  interactions with  $A_t$ , the score of  $A_{i,j}$   $S(A_{i,j})$  can be expressed as:

$$S(A_{i,j}) = (P_T(A_{i,j}|A_{i-1,j-1}) + P_T(A_{i,j}|A_{i,j-1}) + P_T(A_{i,j}|A_{i+1,j-1}) + P_T(A_{i,j}|A_{i,j-1}) + P_T(A_{i,j}|A_{i,j+1}) + P_T(A_{i,j}|A_{i+1,j+1}) + P_T(A_{i,j}|A_{i,j+1}) + P_T(A_{i,j}|A_{i+1,j+1})) / 8$$

In a conventional spatial PD model, an agent A is replaced by its neighbors B if  $S(B) > S(A)$ .

$A_{i-1,j-1}$	$A_{i,j-1}$	$A_{i+1,j-1}$
$A_{i,j-1}$	<b><math>A_{i,j}</math></b>	$A_{i,j+1}$
$A_{i-1,j+1}$	$A_{i,j+1}$	$A_{i+1,j+1}$

Figure 7.1: Agent Representation in a Two-Dimensional Matrix in a Spatial IPD Model.

In our model, a player's neighbors are considered an environmental factor with the potential to influence the player's moves. For strategy  $A_{i,j}$ , the neighbor\_payoff of  $A_{i,j}$  is defined as:

$$\text{neighbor\_payoff}(A_{i,j}) = (S(A_{i-1,j-1}) + S(A_{i,j-1}) + S(A_{i+1,j-1}) + S(A_{i-1,j}) + S(A_{i+1,j}) + S(A_{i-1,j+1}) + S(A_{i,j+1}) + S(A_{i+1,j+1})) / 8$$

Next, a set of strategies was defined that considers the average payoff of neighbors as well as its previous moves and those of its opponent. The strategies are encoded as  $[V_0, V_1, V_2, V_3, V_4]$ , where:

if  $\text{neighbor\_payoff} < 1$ , the  $V_0$  strategy is followed;

if  $1 \leq \text{neighbor\_payoff} < 2$ , the  $V_1$  strategy is followed;

if  $2 \leq \text{neighbor\_payoff} < 3$ , the  $V_2$  strategy is followed;

if  $3 \leq \text{neighbor\_payoff} < 4$ , the  $V_3$  strategy is followed;

if  $4 \leq \text{neighbor\_payoff} \leq 5$ , the  $V_4$  strategy is followed, and

$V_0, V_1, V_2, V_3, V_4$  belong to the set of memory-1 deterministic strategies.

A GA was used to find strategies that survive the evolutionary process. The GA parameters include a  $N \times N$  matrix and a strategy residing in each cell in the matrix. Initially, strategies are randomly produced and distributed. For each generation, assume  $B$  is a neighbor of  $A_{i,j}$  and that it has the highest score among all of  $A_{i,j}$ 's neighbors.  $A_{i,j}$  is replaced by  $B$  if  $S(A_{i,j}) < S(B)$ . Crossover and mutation operators are applied after reproduction. Crossovers occur between an agent and one of its eight neighbors. For each strategy there is a slight chance that a gene will mutate to one of 16 memory-1 deterministic strategies.

Strategies with the potential to survive the evolutionary process were identified and inserted into the spatial PD model consisting of 16 memory-1 deterministic strategies. The performance of an individual strategy compared to memory-1 deterministic strategies was then observed.

### 7.3 Results

GA model execution consisted of 3,000 generations per run. During each run, the amount of each kind of strategy was recorded, and the 10 most dominant strategies were identified. Thus, 50 runs produced 500 strategies with the greatest potential to survive the evolutionary process. The following observations were made for those 500 strategies

- a)  $V_0$  strategies are mostly non-nice, especially  $S_{15}$  or  $S_{14}$ .
- b)  $V_4$  strategies are mostly nice, especially  $S_0$  or  $S_1$ .
- c)  $V_1$ ,  $V_2$ , and  $V_3$  are the combinations of  $S_5$ ,  $S_6$ , and  $S_7$ .

Next, strategy performance was verified and compared with memory-1 strategies. After inserting all of the memory-1 deterministic strategies into the environment, 10 runs of 100 generations each were executed. The results show that PAVLOV ( $S_6$ ), Tit-for-Tat

(S<sub>5</sub>), and Retaliator (S<sub>7</sub>) outperformed all other strategies, with PAVLOV being dominant in the later stages of each run (Figure 7.2).

Evolved strategies were placed in the spatial PD model consisting of 16 memory-1 deterministic strategies. Again, 10 runs of 100 generations each were executed. According to these results, the evolved strategies outperformed most of the memory-1 strategies, with a few outperforming the best memory-1 strategies. In the rest of this section I will give details on three examples.

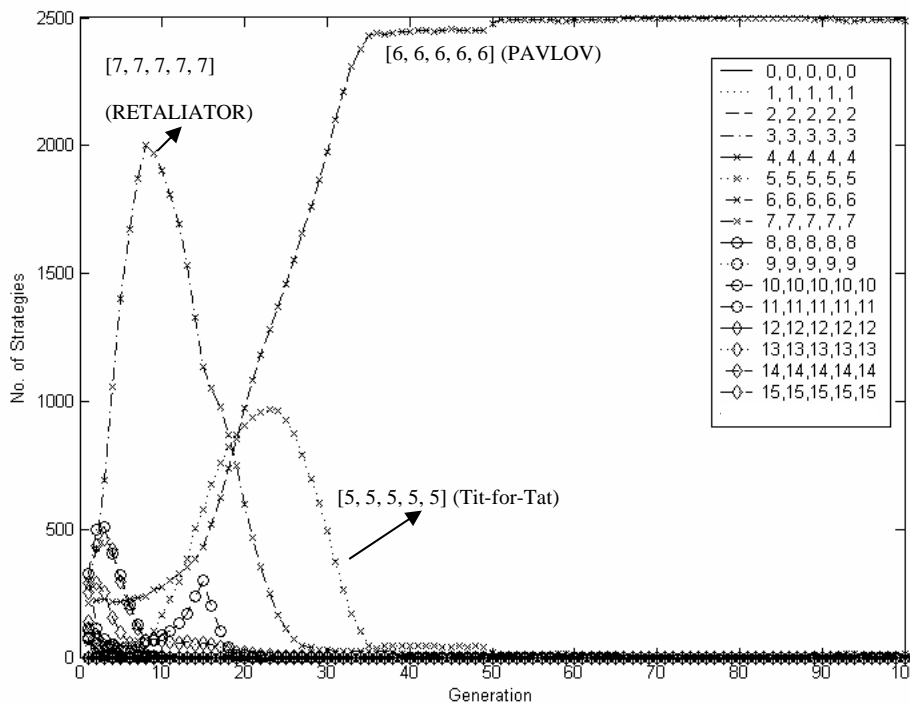


Figure 7.2: Amount-versus-Generation Graph for 16 Memory-1 Deterministic Strategies.

An amount-versus-generation graph for a spatial model consisting of [15, 6, 6, 6, 0] and 16 deterministic memory-1 strategies is shown in Figure 7.3. In this case, the evolved strategy [15, 6, 6, 6, 0] outperformed all memory-1 deterministic strategies. Even PAVLOV, considered the best of all memory-1 deterministic strategies in the model, could not invade the [15, 6, 6, 6, 0] strategy.

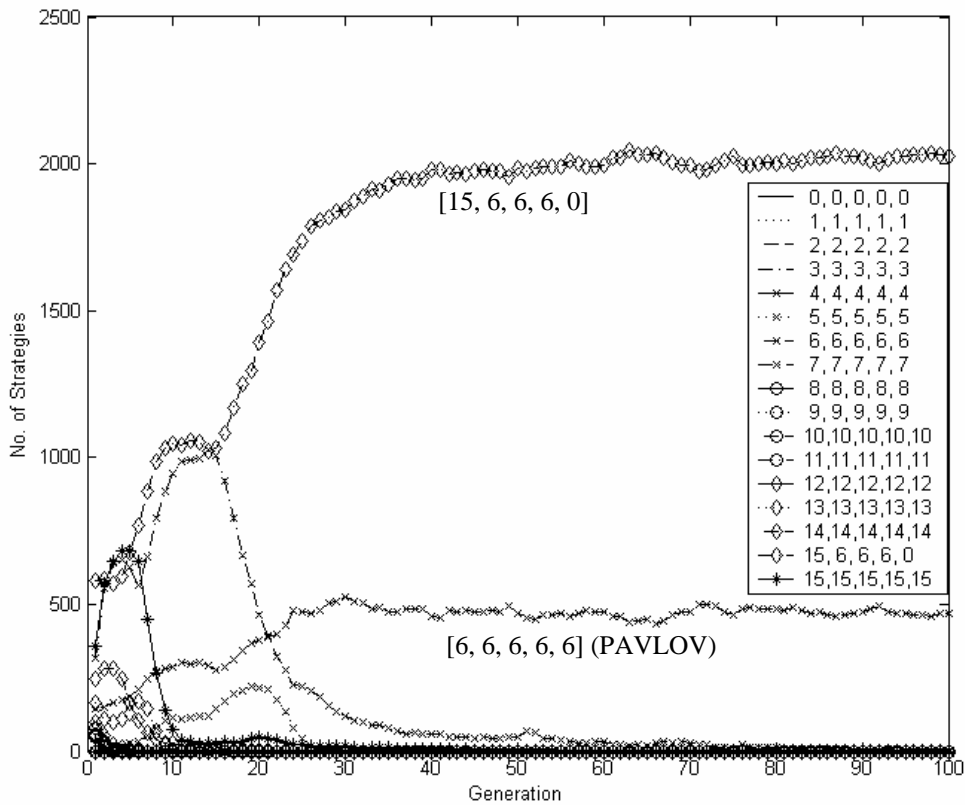


Figure 7.3: Amount-versus-Generation Graph for Memory-1 Deterministic Strategies and [15, 6, 6, 6, 0].

A slight change in strategy encoding affected the simulation results. Results from a simulation consisting of [15, 5, 6, 7, 1] and 16 deterministic memory-1 strategies are shown in Figure 7.4. The [15, 5, 6, 7, 1] strategy performed well during the early stages of evolution, but as its numbers increased within the environment, PAVLOV gained dominance because of its strong clustering capability. Even though the number of [15, 5, 6, 7, 1] strategies was larger than the number of PAVLOV strategies, they were susceptible to invasion

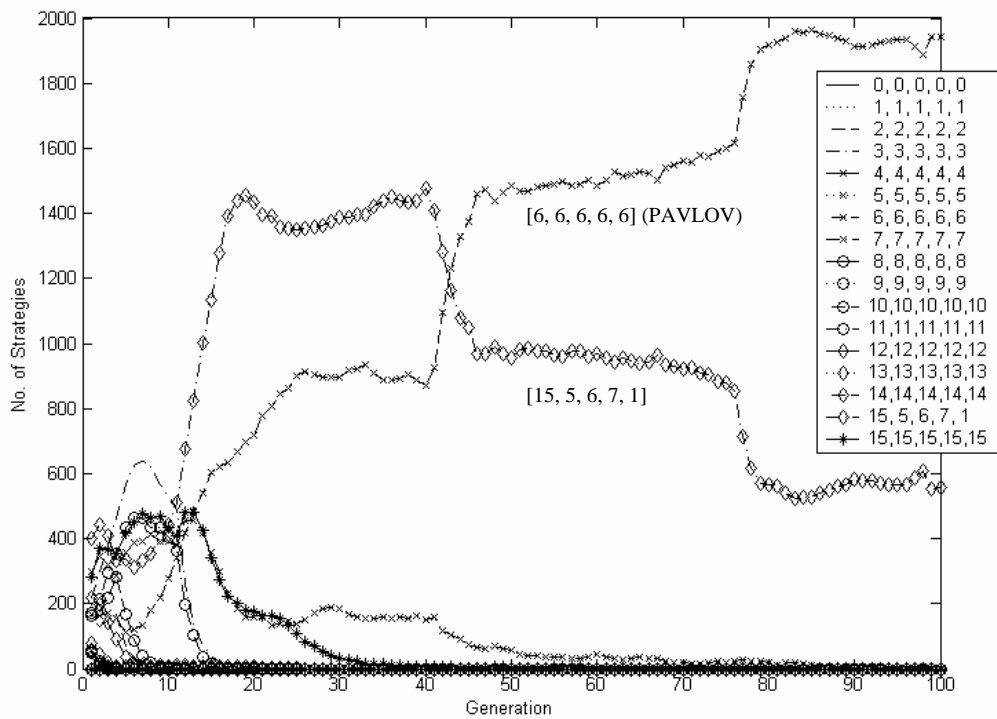


Figure 7.4: Amount-versus-Generation Graph for Memory-1 Deterministic Strategies and [15, 5, 6, 7, 1].

The distribution of strategy number versus payoff values during strategy evolution is presented in Figure 7.5. The figure underscores the rarity of agents with payoffs larger than 3 and smaller than 1, meaning that strategies  $V_0$  and  $V_4$  in a strategy set encoded as  $[V_0, V_1, V_2, V_3, V_4]$  will be used rarely compared to the  $V_1, V_2,$  and  $V_3$  strategies. However, Figure 7.3 shows that  $[15, 6, 6, 6, 0]$  outperformed PAVLOV (represented as  $[6, 6, 6, 6, 6]$ )—the best performing of all memory-1 deterministic strategies in the model. In other words, even though  $V_0$  and  $V_4$  are rarely used, they exert a critical influence on strategy survival.

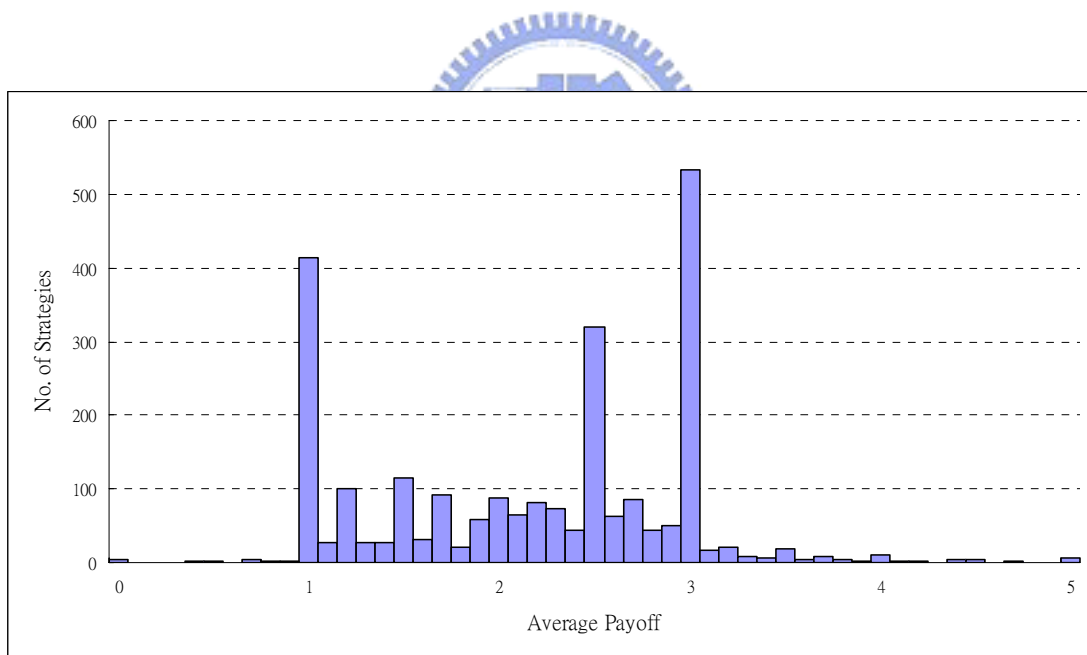


Figure 7.5: Distribution of Strategy Number versus Payoff Values.

The values of  $V_0$  and  $V_4$  were changed in an attempt to further validate their importance. Results of strategy  $[0, 6, 6, 6, 0]$  are shown in Figure 7.6. Although it



clearly outperformed most memory-1 deterministic strategies, it was incapable of invading PAVLOV strategies in the same manner as [15, 6, 6, 6, 1].

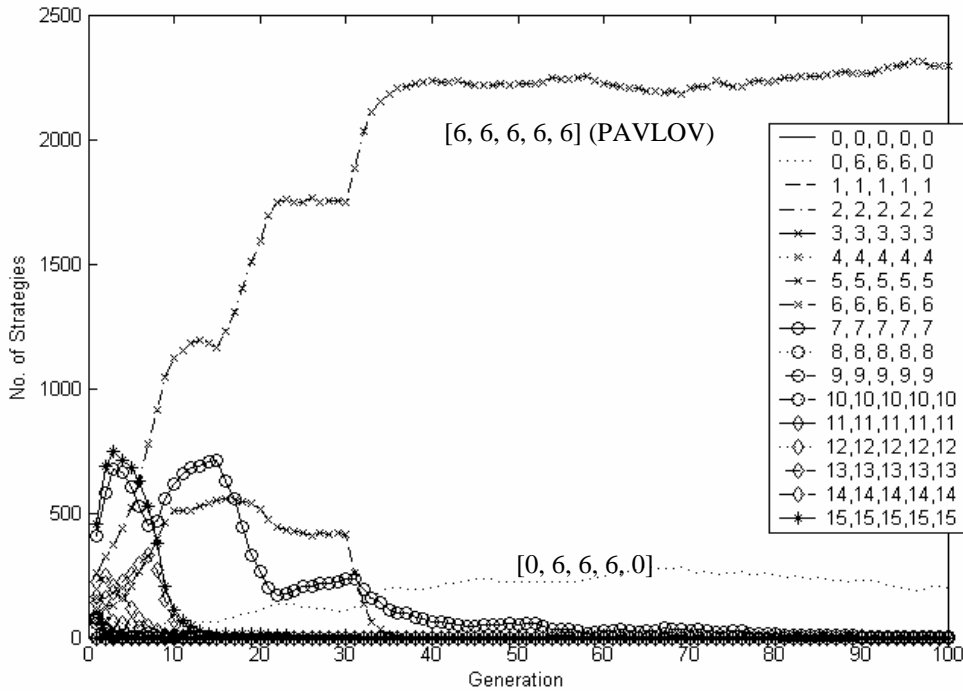


Figure 7.6: Amount-versus-Generation Graph for Memory-1 Deterministic Strategies and [0, 6, 6, 6, 0].

Figures 7.7 and 7.8 present results of strategy [15, 6, 6, 6, 15]. While it was capable of invading all of the memory-1 deterministic strategies (Figure 7.7), it could also be dominated by other memory-1 deterministic strategies (Figure 7.8). In other words, strategy [15, 6, 6, 6, 15] was relatively strong compared with memory-1 deterministic strategies, but it was not as strong as [15, 6, 6, 6, 1], which was capable of invading memory-1 deterministic strategies at any time.

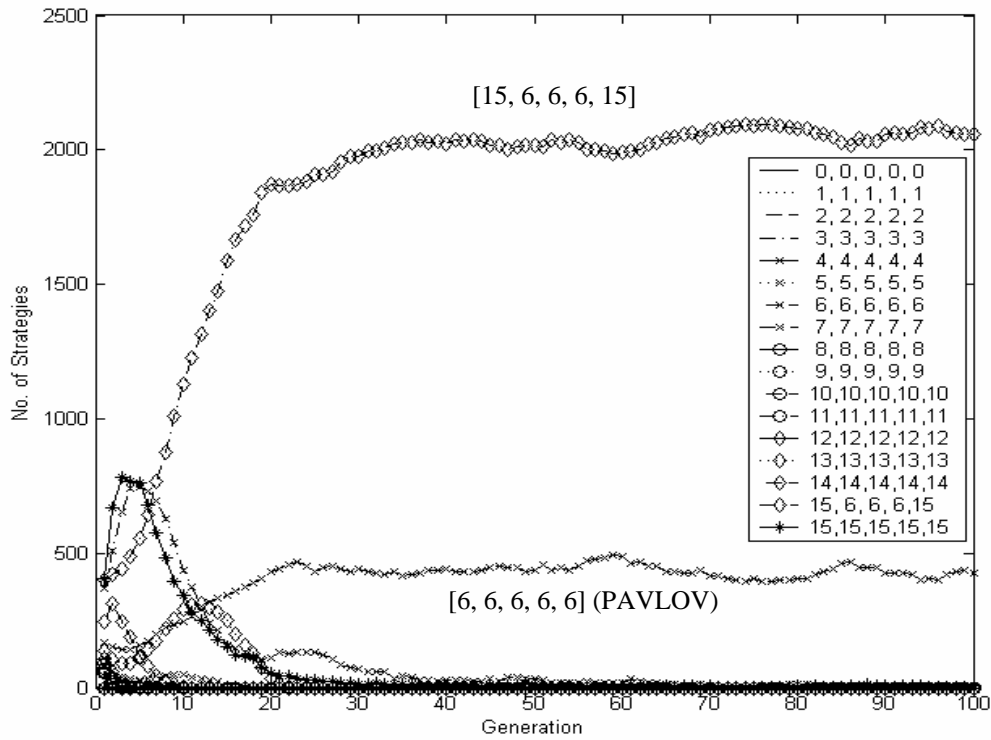


Figure 7.7: Amount-versus-Generation Graph for Memory-1 Deterministic Strategies and [15, 6, 6, 6, 15].

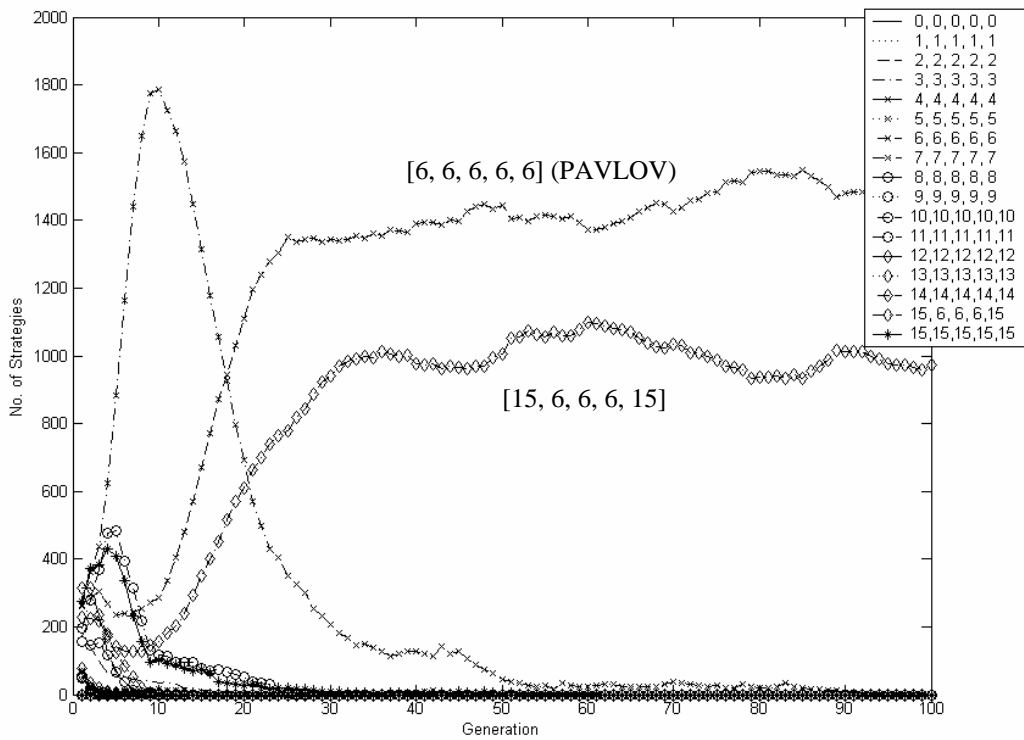
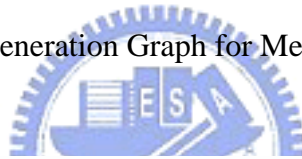


Figure 7.8: Amount-versus-Generation Graph for Memory-1 Deterministic Strategies and [15, 6, 6, 6, 15].

## 7.4 Conclusion

According to the results produced by the evolutionary model, strategies that take into account simple environmental factors (e.g., their neighbors' average scores) when deciding their subsequent moves have greater potential to outperform strategies that only consider the historical moves of two players. Three characteristics were observed in the evolved strategies:

a) When the average neighbor score ranges between 1 and 3, then PAVLOV, Tit-for-Tat, or Retaliator strategies should be used to improve chances for survival.

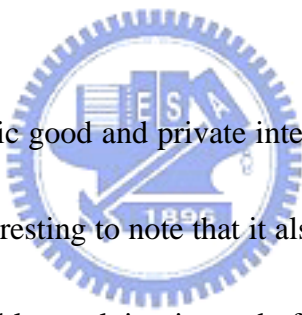
b) When the average neighbor score is below 1, the environment is assumed to be full of defecting agents, which encourages the use of non-nice strategies (e.g., ALLD).

c) When the average neighbor score is above 1, the environment is assumed to be full of cooperating agents, which encourages the use of nice strategies (e.g., ALLC).

One other important characteristic is that some decisions based on rare/unusual circumstances can have a critical impact on strategy survival. Even though average neighbor scores were rarely below 1 or above 3, the use of appropriate strategies under these circumstances should not be overlooked.

# Chapter 8.

## Conclusion



The conflict between public good and private interest is very common in both human and animal societies. It is interesting to note that it also occurs in artificial societies, even when they are designed for problem solving instead of real world simulation. The purpose of this dissertation was to investigate the private interest/public good conflict as it specifically occurs in artificial societies designed for problem solving.

The Iterated Prisoner's Dilemma (IPD) that was used as an abstract model is commonly employed in investigations of real-world simulations. The proposed analytical framework for investigating the IPD model made use of interaction patterns between agents to construct inter-agent relationships. Existing strategy properties and well-known phenomena were discovered without having to invest a large amount of simulation time or

resources. It was also useful in identifying interesting IPD model properties, such as the cyclic exploration relationships among strategies.

The proposed framework highlights an important methodological issue in simulations and the use of artificial societies for problem solving: what work should be done prior to execution? Although considered important, pre-simulation or pre-execution model analysis is generally overlooked by researchers in either area. A simple yet typical example involving GA was presented, and the proposed analytical framework provided further evidence in support of performing an initial model analysis. Furthermore, simulative and analytical approaches have complementary advantages and disadvantages; I expect that their integration will become an important topic in future research, and that the first studies will focus on clarifying their complementary properties.

Also discussed was the effect of environment on the IPD model—another important factor that has generally been ignored in the IPD model literature. The results show that strategies that take the environmental factor into account tend to perform well.

# Bibliography

- [1] P. Curtis and D. A. Nichols, "MUDs grow up: Social virtual reality in the real world," presented at The 1994 IEEE Computer Conference, 1994.
- [2] C. H. Hsieh and C. T. Sun, "MUD for Learning: Classification and Instruction," *International Journal of Instructional Media*, vol. 33, 2004.
- [3] C. Castelfranchi and Y.-H. Tan, "The Role of Trust and Deception in Virtual Societies," presented at Proceedings of the 34th International Conference on System Sciences, Hawaii, 2001.
- [4] J. Zola and A. Ioannidou, "Learning and teaching with interactive simulations," *Social Education*, vol. 63, pp. 142-145, 2000.
- [5] U. J. Christensen, D. Heffernan, and P. Barach, "Microsimulators in medical education: An overview," *Simulation and Gaming*, vol. 32, pp. 250-262, 2001.
- [6] R. Axelrod, "Advancing the art of simulation in the social science," in *Simulating Social Phenomena*. Berlin: Springer, 1997, pp. 21-40.

- [7] E. Brent, "Sociology: Modeling social interaction with autonomous agents," *Social Science Computer Review*, vol. 17, pp. 313-322, 1999.
- [8] E. Brent, A. Thompson, and W. Vale, "Sociology: A computational approach to sociological explanations," *Social Science Computer Review*, vol. 18, pp. 223-235, 2000.
- [9] N. M. Gotts, J. G. Polhill, and A. N. R. Law, "Agent-based Simulation in the Study of Social Dilemmas," *Artificial Intelligence Review*, vol. 19, pp. 3-92, 2003.
- [10] R. Hanneman and S. Patrick, "On the uses of computer-assisted simulation modeling in the social sciences," *Sociological Research Online*, vol. 2, 1997.
- [11] Y. Leung, Y. Gao, and Z.-B. Xu, "Degree of population diversity - a perspective on premature convergence in genetic algorithms and its Markov chain analysis," *IEEE Transactions on Neural Networks*, vol. 8, pp. 1165 - 1176, 1997.
- [12] G. Rudolph, "Self-adaptive mutations may lead to premature convergence," *IEEE Transactions on Evolutionary Computation*, vol. 5, pp. 410 - 414, 2001.
- [13] A. Artikis and J. Pitt, "A Formal Model of Open Agent Societies," presented at Proceedings of the Fifthe International Conference on Automous Agents, 2001.
- [14] J. Doran, "Computer Bases Simulation and Formal Modeling in Archaeology: a Review," in *Mathematics and Information Science in Archaeology*, vol. 3, A. Voorrips, Ed. Bonn: Holos., 1990, pp. 93-114.

- [15] J. Doran, "Social simulation, agents and artificial societies," presented at Proceedings on International Conference on Multi Agent Systems, Paris France, 1998.
- [16] M. Kutrib, R. Vollmar, and T. Worsch, "Introduction to the special issue on cellular automata," *Parallel Computing*, vol. 23, pp. 1567-1576, 1997.
- [17] T. Worsch, "Simulation of cellular automata," *Future Generation Computer Systems*, vol. 16, pp. 157-170, 1999.
- [18] L. B. Booker, D. E. Goldberg, and J. H. Holland, "Classifier Systems and Genetic Algorithms," *Artificial Intelligence*, vol. 40, pp. 235-282, 1989.
- [19] M. Dorigo, V. Maniezzo, and A. Coloni, "Ant System: Optimization by a Colony of Cooperating Agents," *IEEE Transaction on Systems, Man, and Cybernetics-Part B: Cybernetics*, vol. 26, pp. 29-41, 1996.
- [20] L. Wang and Q. Wu, "Further example study on ant system algorithm based continuous space optimization," presented at Proceedings of the 4th World Congress on Intelligent Control and Automation, Shanghai, China, 2002.
- [21] V. Maniezzo and A. Coloni, "The ant system applied to the quadratic assignment problem," *IEEE Transactions on Knowledge and Data Engineering*, vol. 11, pp. 769 - 778, 1999.



- [22] M. Mitchell, *An Introduction to Genetic Algorithms (Complex Adaptive Systems)*: MIT Press, 1998.
- [23] J. Sima and P. Orponen, "General-purpose computation with neural networks: A survey of complexity theoretic results," *Neural Computaiton*, vol. 15, pp. 2727-2778, 2003.
- [24] L. N. d. Castro and J. Timmis, *Artificial Immune Systems: A New Computational Intelligence Approach*. London: Springe, 2002.
- [25] A. Tarakanov and D. Dasgupta, "A formal model of an artificial immune system," *Biosystems*, vol. 55, pp. 151 - 158, 2000.
- [26] R. L. King, S. H. Russ, A. B. Lambert, and D. S. Reese, "An artificial immune system model for intelligent agents," *FUTURE GENERATION COMPUTER SYSTEMS*, vol. 17, pp. 335-343, 2001.
- [27] S. A. Hofmeyr and S. Forrest, "Architecture for an Artificial Immune System," *Evolutionary Computation*, vol. 8, pp. 443-473, 2000.
- [28] J. E. Hunt and D. E. Cooke, "Learning using an artificial immune system," *Journal of Network and Computer Applications*, vol. 19, pp. 189-212, 1996.
- [29] W. Jiao, M. Zhou, and Q. Wang, "Formal framework for adaptive multi-agent systems," presented at IEEE/WIC International Conference on Intelligent Agent Technology, Beijing, China, 2003.

- [30] M. Wooldridge, "Agent-based software engineering," *IEE Proceedings on Software Engineering*, vol. 144, pp. 26 - 37, 1997.
- [31] C. Zozaya-Gorostiza and D. R. Orellana-Moyao, "Modifications to the credit apportionment mechanism of a simple classifier system," *Lecture Notes in Artificial Intelligence*, vol. 1793, pp. 235-246, 2000.
- [32] N. M. Hewahi and H. Ahmad, "Credit apportionment scheme for rule-based systems: Implementation and comparative study," *Lecture Notes in Artificial Intelligence*, vol. 2358, pp. 435-449, 2002.
- [33] N. Nisan, "Algorithms for selfish agents," presented at Proceedings of 16th Symposium on Theoretical Aspects of Computer Sciences, 1999.
- [34] N. E. Kfir-Dahav, D. Monderer, and M. Tennenholtz, "Mechanism design for resource bounded agents," presented at Proceedings of the Fourth International Conference on MultiAgent Systems, Boston, MA USA, 2000.
- [35] P. Maes, "Modeling Adaptive Autonomous Agents," in *Artificial Life: An Overview*, C. G. Langton, Ed. Cambridge: The MIT Press, 1995, pp. 135-162.
- [36] M. R. Lauer, P. A. Mitchem, and R. A. Gagliano, "Resource optimization and self interest: variations on the game of life," presented at Proceedings of the 28th Annual on Simulation Symposium, Phoenix, AZ USA, 1995.

- [37] K. S. Barber, T. H. Liu, and S. Ramaswamy, "Conflict detection during plan integration for multi-agent systems," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 31, pp. 616 - 628, 2001.
- [38] F. Kofman and J. Lawarree, "A prisoner's dilemma model of collusion deterrence," *Journal of Public Economics*, vol. 59, pp. 117-136, 1996.
- [39] M. K. Surbey and J. J. McNally, "Self-deception as a mediator of cooperation and defection in varying social contexts described in the iterated prisoner's dilemma," *Evolution and Human Behavior*, vol. 18, pp. 417-435, 1997.
- [40] G. Rabow, "The social implications of nonzero-sum games," in *IEEE Technology and Society Magazine*, vol. 7, 1988, pp. 12 - 18.
- [41] K. Uno and A. Namatame, "Agent-based simulation for policy issue," presented at Proceedings of 1999 IEEE International Conference on Systems, Man, and Cybernetics, 1999.
- [42] S. Legge, "Cooperative lions escape the Prisoner's Dilemma," *Trends in Ecology & Evolution*, vol. 11, pp. 2-3, 1996.
- [43] C. Boone, B. D. Brabander, and A. v. Witteloostuijn, "The impact of personality on behavior in five Prisoner's Dilemma games," *Journal of Economic Psychology*, vol. 20, pp. 343-377, 1999.
- [44] W. Poundstone, *Prisoner's Dilemma*. New York: Oxford University Press, 1992.

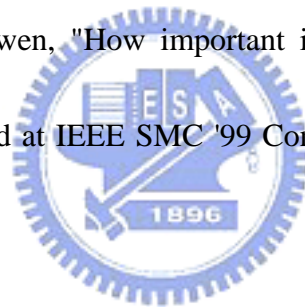
- [45] J. C. Oh, "Promoting cooperation: Using 'kin' biased conditional strategy in the iterated Prisoner's Dilemma game," *Information Sciences*, vol. 133, pp. 149-164, 2001.
- [46] R. Axelrod, *The Evolution of Cooperation*. New York: Basic Books, 1984.
- [47] R. Hoffmann, "Twenty Years on: The Evolution of Cooperation Revisited," *Journal of Artificial Societies and Social Simulation*, vol. 3, 2000.
- [48] M. Doebeli, A. Blarer, and M. Ackermann, "Population dynamics, demographic stochasticity, and the evolution of cooperation," presented at Proceedings of the National Academy of Sciences of the United States of America, 1997.
- [49] L. Hsu, T. Hsu, J. Mortimer, M. Panju, and S. Schroeder, "Dynamically stable multiple strategy states of the iterated prisoner's dilemma," *Physica D*, vol. 85, pp. 296-303, 1995.
- [50] E. Amann and C.-L. Yang, "Sophistication and the persistence of cooperation," *Journal of Economic Behavior & Organization*, vol. 37, pp. 91-105, 1998.
- [51] M. Nowak and K. Sigmund, "A strategy of Win-Stay, Lose-Shift that outperforms Tit-for-Tat in the Prisoner's Dilemma game," *Nature*, vol. 364, pp. 56-58, 1993.
- [52] C. Wedekind and M. Milinski, "Human cooperation in the simultaneous and the alternating Prisoner's Dilemma: Pavlov versus Generous Tit-for-Tat," presented at

Proceedings of the National Academy of Sciences of the United States of America, 1996.

- [53] D. P. Kraines and V. Kraines, "Natural selection of memory-one strategies for the iterated Prisoner's Dilemma," *Journal of Theoretical Biology*, vol. 203, pp. 335-355, 2000.
- [54] D. B. Neill, "Optimality under Noise: Higher Memory Strategies for the Alternating Prisoner's Dilemma," *Journal of Theoretical Biology*, vol. 211, pp. 159-180, 2001.
- [55] T.-H. Ho, "Finite automata play repeated Prisoner's Dilemma with information processing cost," *Journal of Economic Dynamics and Control*, vol. 20, pp. 173-207, 1996.
- [56] P. H. Crowley, "Evolving cooperation: Strategies as hierarchies of rules," *BioSystems*, vol. 37, pp. 67-80, 1996.
- [57] C. Hauert and O. Stenull, "Simple Adaptive Strategy Wins the Prisoner's Dilemma," *Journal of Theoretical Biology*, vol. 218, pp. 261-272, 2002.
- [58] E. A. Billard, "Evolutionary strategies of stochastic learning automata in the prisoner's dilemma," *Biosystems*, vol. 39, pp. 93-107, 1996.
- [59] S. X. Li, "Interactive strategy sets in multiple payoff games," *Computers & Industrial Engineering*, vol. 37, pp. 613-630, 1999.

- [60] J. M. Smith, *Evolution and the Theory of Games*. New York: Cambridge University Press, 1982.
- [61] J. P. Lorberbaum, D. E. Bohning, A. Shastri, and L. E. Sine, "Are there really no evolutionarily stable strategies in the Iterated Prisoner's Dilemma?," *Journal of Theoretical Biology*, vol. 214, pp. 155-169, 2002.
- [62] J. P. Lorberbaum, "No strategy is evolutionarily stable in the Repeated Prisoner's Dilemma," *Journal of Theoretical Biology*, vol. 168, pp. 117-130, 1994.
- [63] A. Iqbal and A. H. Toor, "Evolutionarily stable strategies in quantum games," *Physics Letters A*, vol. 280, pp. 249-256, 2001.
- [64] K. Binmore and L. Samuelson, "Can Mixed Strategies be Stable in Asymmetric Games," *Journal of Theoretical Biology*, vol. 210, pp. 1-14, 2001.
- [65] K. Brauchli, T. Killingback, and M. Doebeli, "Evolution of cooperation in spatially structured populations," *Journal of Theoretical Biology*, vol. 200, pp. 405-417, 1999.
- [66] K. Lindgren and M. G. Nordahl, "Evolutionary dynamics of spatial games," *Physica D*, vol. 75, pp. 292-309, 1994.
- [67] P. Grim, "Spatialization and greater generosity in the stochastic Prisoner's Dilemma," *BioSystems*, vol. 37, pp. 3-17, 1996.

- [68] H. Ishibuchi, T. Nakari, and T. Nakashima, "Evolution of neighborly relations in a spatial IPD game with cooperative players and hostile players," presented at Congress on Evolutionary Computation 2, Washington, 1999.
- [69] P. Grim, "The greater generosity of the spatialized prisoner's dilemma," *Journal of Theoretical Biology*, vol. 173, pp. 353-359, 1995.
- [70] M. Nakamaru, H. Matsuda, and Y. Iwasa, "The Evolution of Cooperation in a Lattice-Structured Population," *Journal of Theoretical Biology*, vol. 184, pp. 65-81, 1997.
- [71] X. Yao and P. J. Darwen, "How important is your reputation in a multi-agent environment," presented at IEEE SMC '99 Conference Proceedings, Tokyo Japan, 1999.
- [72] R. L. Riolo, M. D. Cohen, and R. Axelrod, "Cooperation without Reciprocity," *Nature*, vol. 414, pp. 441-443, 2001.
- [73] D. Hales, "Tag Based Cooperation in Artificial Societies," in *Department of Computer Science: University of Essex*, 2001.
- [74] M. Matsushima and T. Ikegami, "Evolution of Strategies in the three-person Iterated Prisoner's Dilemma Game," *Journal of Theoretical Biology*, vol. 195, pp. 53-67, 1998.



- [75] X. Yao, "Evolutionary stability in the n-person iterated prisoner's dilemma," *Biosystems*, vol. 37, pp. 189-197, 1996.
- [76] M. Frean, "The evolution of degrees of cooperation," *Journal of Theoretical Biology*, vol. 182, pp. 549-559, 1996.
- [77] P. J. Darwen and X. Yao, "Why more choices cause less cooperation in iterated prisoner's dilemma," presented at Proceedings of the 2001 Congress on Evolutionary Computation, Seoul South Korea, 2001.
- [78] P. S. S. Borges, R. C. S. Pacheco, R. M. Barcia, and S. K. Khator, "A fuzzy approach to the prisoner's dilemma," *Biosystems*, vol. 41, pp. 127-137, 1997.
- [79] P. G. Harrald and D. B. Fogel, "Evolving continuous behaviors in the Iterated Prisoner's Dilemma," *Biosystems*, vol. 37, pp. 135-145, 1996.
- [80] L. M. Wahl and A. A. Nowak, "The Continuous Prisoner's Dilemma: I. Linear Reactive Strategies," *Journal of Theoretical Biology*, vol. 200, pp. 307-321, 1999.
- [81] M. Nowak and K. Sigmund, "Game-dynamical aspects of the Prisoner's Dilemma," *Applied Mathematics and Computation*, vol. 30, pp. 191-213, 1989.
- [82] M. Nowak and K. Sigmund, "The evolution of stochastic strategies in the Prisoner's Dilemma," *Acta Applied Mathematics*, vol. 20, pp. 247-265, 1990.
- [83] K. Lindgren, "Evolutionary phenomena in simple dynamics," in *Artificial Life II*, C. G. Langton, Ed. Redwood: Addison-Wesley, 1991, pp. 295-312.



- [84] M. R. Frean and E. R. Abraham, "A voter model of the spatial Prisoner's Dilemma," *IEEE Transaction on Evolutionary Computation*, vol. 5, pp. 117-121, 2001.
- [85] B. A. Huverman and N. S. Glance, "Evolutionary games and computer simulation," presented at Proceedings of the National Academy of Sciences of the United States of America, 1993.
- [86] M. Nowak and K. Sigmund, "Tit for Tat in heterogeneous populations," *Nature*, vol. 355, pp. 250-252, 1992.
- [87] T. K. Moon, R. L. Frost, and W. C. Stirling, "An epistemic utility approach to coordination in the Prisoner's Dilemma," *BioSystems*, vol. 37, pp. 167-176, 1996.
- [88] C. Goldspink, "Methodological implications of complex systems approaches to sociality: Simulation as a foundation for knowledge," *Journal of Artificial Societies and Social Simulation*, vol. 5, 2002.
- [89] C. Jacobsen and R. Bronson, "Computer simulations and empirical testing of sociological theory," *Sociological Methods and Research*, vol. 23, pp. 479-506, 1995.
- [90] R. D. Stocker, G. Green, and D. Newth, "Consensus and cohesion in simulated social networks," *Journal of Artificial Societies and Social Simulation*, vol. 4, 2001.

- [91] F. Azuaje, "A computational evolutionary approach to evolving game strategy and cooperation," *IEEE Transaction on Systems, Man, and Cybernetics-Part B: Cybernetics*, vol. 33, pp. 498-502, 2003.
- [92] H.-C. Wu and C.-T. Sun, "An analytical framework for the Prisoner's Dilemma: Finite state machine representation for interactions between deterministic strategies," *Journal of Interdisciplinary Mathematics*, vol. 5, pp. 313-338, 2002.
- [93] J. C. Martin, *Introduction to Language and the Theory of Computation*. New York: McGraw Hill, 1996.



# Appendix A

Behavior Characteristics of memory-1 Deterministic Strategies:

	$S_0$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$	$S_6$	$S_7$
$S_0$	(1,0,0,0)	(1,0,0,0)	(3/4,1/4,0,0)	(1/2,1/2,0,0)	(1,0,0,0)	(1,0,0,0)	(1/2,1/2,0,0)	(1/4,3/4,0,0)
$S_1$	(1,0,0,0)	(3/4,0,0,1/4)	(3/4,1/4,0,0)	(1/2,1/4,0,1/4)	(1,0,0,0)	(3/4,0,0,1/4)	(1/4,3/4,0,0)	(1/4,1/2,0,1/4)
$S_2$	(3/4,0,1/4,0)	(3/4,0,1/4,0)	(1/2,1/4,1/4,0)	(1/4,1/2,1/4,0)	(1,0,0,0)	(1,0,0,0)	(3/4,1/4,0,0)	(1/4,3/4,0,0)
$S_3$	(1/2,0,1/2,0)	(1/2,0,1/4,1/4)	(1/4,1/4,1/2,0)	(1/4,1/4,1/4,1/4)	(1/2,0,1/4,1/4)	(1/2,0,0,1/2)	(1/4,1/4,1/4,1/4)	(1/4,1/4,0,1/2)
$S_4$	(1,0,0,0)	(1,0,0,0)	(1,0,0,0)	(1/2,1/4,0,1/4)	(1/2,1/4,1/4,0)	(1/4,3/8,3/8,0)	(1,0,0,0)	(1/4,3/8,0,3/8)
$S_5$	(1,0,0,0)	(3/4,0,0,1/4)	(1,0,0,0)	(1/2,0,0,1/2)	(1/4,3/8,3/8,0)	(1/4,1/4,1/4,1/4)	(1/4,1/4,1/4,1/4)	(1/4,0,0,3/4)
$S_6$	(1/2,0,1/2,0)	(1/4,0,3/4,0)	(3/4,0,1/4,0)	(1/4,1/4,1/4,1/4)	(1,0,0,0)	(1/4,1/4,1/4,1/4)	(1,0,0,0)	(1/4,3/8,0,3/8)
$S_7$	(1/4,0,3/4,0)	(1/4,0,1/2,1/4)	(1/4,0,3/4,0)	(1/4,0,1/4,1/2)	(1/4,0,3/8,3/8)	(1/4,0,0,3/4)	(1/4,0,3/8,3/8)	(1/4,0,0,3/4)
$S_8$	(1/2,0,1/2,0)	(1/2,0,1/2,0)	(3/8,1/4,3/8,0)	(1/4,1/2,1/4,0)	(1/3,1/3,1/3,0)	(1/3,1/3,1/3,0)	(0,1,0,0)	(0,1,0,0)
$S_9$	(1/2,0,1/2,0)	(3/8,0,3/8,1/4)	(3/8,1/4,3/8,0)	(1/4,1/4,1/4,1/4)	(1/3,1/3,1/3,0)	(1/4,1/4,1/4,1/4)	(0,1,0,0)	(0,3/4,0,1/4)
$S_{10}$	(0,0,1,0)	(0,0,1,0)	(0,1/4,3/4,0)	(0,1/2,1/2,0)	(1/3,0,1/3,1/3)	(1/4,1/4,1/4,1/4)	(1/4,1/4,1/4,1/4)	(0,1,0,0)
$S_{11}$	(0,0,1,0)	(0,0,3/4,1/4)	(0,1/4,3/4,0)	(0,1/4,1/2,1/4)	(0,0,1/2,1/2)	(0,0,0,1)	(0,1/4,3/8,3/8)	(0,1/4,0,3/4)
$S_{12}$	(1/2,0,1/2,0)	(1/2,0,1/2,0)	(1/2,0,1/2,0)	(1/4,1/4,1/4,1/4)	(0,1/2,1/2,0)	(0,1/2,1/2,0)	(1/4,1/4,1/4,1/4)	(0,1/2,0,1/2)
$S_{13}$	(1/2,0,1/2,0)	(3/8,0,3/8,1/4)	(1/2,0,1/2,0)	(1/4,0,1/4,1/2)	(0,1/2,1/2,0)	(0,3/8,3/8,1/4)	(0,1/3,1/3,1/3)	(0,0,0,1)
$S_{14}$	(0,0,1,0)	(0,0,1,0)	(0,0,1,0)	(0,1/4,1/2,1/4)	(1/3,0,1/3,1/3)	(0,1/3,1/3,1/3)	(1/3,0,1/3,1/3)	(0,1/2,0,1/2)
$S_{15}$	(0,0,1,0)	(0,0,3/4,1/4)	(0,0,1,0)	(0,0,1/2,1/2)	(0,0,1/2,1/2)	(0,0,0,1)	(0,0,1/2,1/2)	(0,0,0,1)

Behavior Characteristics of memory-1 Deterministic Strategies (Continued):

	<b>S<sub>8</sub></b>	<b>S<sub>9</sub></b>	<b>S<sub>10</sub></b>	<b>S<sub>11</sub></b>	<b>S<sub>12</sub></b>	<b>S<sub>13</sub></b>	<b>S<sub>14</sub></b>	<b>S<sub>15</sub></b>
<b>S<sub>0</sub></b>	(1/2,1/2,0,0)	(1/2,1/2,0,0)	(0,1,0,0)	(0,1,0,0)	(1/2,1/2,0,0)	(1/2,1/2,0,0)	(0,1,0,0)	(0,1,0,0)
<b>S<sub>1</sub></b>	(1/2,1/2,0,0)	(3/8,3/8,0,1/4)	(0,1,0,0)	(0,3/4,0,1/4)	(1/2,1/2,0,0)	(3/8,3/8,0,1/4)	(0,1,0,0)	(0,3/4,0,1/4)
<b>S<sub>2</sub></b>	(3/8,3/8,1/4,0)	(3/8,3/8,1/4,0)	(0,3/4,1/4,0)	(0,3/4,1/4,0)	(1/2,1/2,0,0)	(1/2,1/2,0,0)	(0,1,0,0)	(0,1,0,0)
<b>S<sub>3</sub></b>	(1/4,1/4,1/2,0)	(1/4,1/4,1/4,1/4)	(0,1/2,1/2,0)	(0,1/2,1/4,1/4)	(1/4,1/4,1/4,1/4)	(1/4,1/4,0,1/2)	(0,1/2,1/4,1/4)	(0,1/2,0,1/2)
<b>S<sub>4</sub></b>	(1/3,1/3,1/3,0)	(1/3,1/3,1/3,0)	(1/3,1/3,0,1/3)	(0,1/2,0,1/2)	(0,1/2,1/2,0)	(0,1/2,1/2,0)	(1/3,1/3,0,1/3)	(0,1/2,0,1/2)
<b>S<sub>5</sub></b>	(1/3,1/3,1/3,0)	(1/4,1/4,1/4,1/4)	(1/4,1/4,1/4,1/4)	(0,0,0,1)	(0,1/2,1/2,0)	(0,3/8,3/8,1/4)	(0,1/3,1/3,1/3)	(0,0,0,1)
<b>S<sub>6</sub></b>	(0,0,1,0)	(0,0,1,0)	(1/4,1/4,1/4,1/4)	(0,3/8,1/4,3/8)	(1/4,1/4,1/4,1/4)	(0,1/3,1/3,1/3)	(1/3,1/3,0,1/3)	(0,1/2,0,1/2)
<b>S<sub>7</sub></b>	(0,0,1,0)	(0,0,3/4,1/4)	(0,0,1,0)	(0,0,1/4,3/4)	(0,0,1/2,1/2)	(0,0,0,1)	(0,0,1/2,1/2)	(0,0,0,1)
<b>S<sub>8</sub></b>	(1/2,0,0,1/2)	(1/3,1/3,0,1/3)	(3/8,1/4,0,3/8)	(0,1,0,0)	(1/2,0,0,1/2)	(1/3,1/3,0,1/3)	(1/4,1/2,0,1/4)	(0,1,0,0)
<b>S<sub>9</sub></b>	(1/3,0,1/3,1/3)	(0,0,0,1)	(1/4,1/4,1/4,1/4)	(0,1/4,0,3/4)	(1/4,1/4,1/4,1/4)	(0,0,0,1)	(0,1,0,0)	(0,1/2,0,1/2)
<b>S<sub>10</sub></b>	(3/8,0,1/4,3/8)	(1/4,1/4,1/4,1/4)	(1/4,1/4,1/4,1/4)	(0,3/4,1/4,0)	(1/2,0,0,1/2)	(1/3,1/3,0,1/3)	(3/8,1/4,0,3/8)	(0,1,0,0)
<b>S<sub>11</sub></b>	(0,0,1,0)	(0,0,1/4,3/4)	(0,1/4,3/4,0)	(0,1/4,1/4,1/2)	(0,0,1/2,1/2)	(0,0,0,1)	(0,1/4,3/8,3/8)	(0,1/4,0,3/4)
<b>S<sub>12</sub></b>	(1/2,0,0,1/2)	(1/4,1/4,1/4,1/4)	(1/2,0,0,1/2)	(0,1/2,0,1/2)	(1/4,1/4,1/4,1/4)	(0,1/2,1/2,0)	(1/2,0,0,1/2)	(0,1/2,0,1/2)
<b>S<sub>13</sub></b>	(1/3,0,1/3,1/3)	(0,0,0,1)	(1/3,0,1/3,1/3)	(0,0,0,1)	(0,1/2,1/2,0)	(0,1/4,1/4,1/2)	(0,1/3,1/3,1/3)	(0,0,0,1)
<b>S<sub>14</sub></b>	(1/4,0,1/2,1/4)	(0,0,1,0)	(3/8,0,1/4,3/8)	(0,3/8,1/4,3/8)	(1/2,0,0,1/2)	(0,1/3,1/3,1/3)	(1/2,0,0,1/2)	(0,1/2,0,1/2)
<b>S<sub>15</sub></b>	(0,0,1,0)	(0,0,1/2,1/2)	(0,0,1,0)	(0,0,1/4,3/4)	(0,0,1/2,1/2)	(0,0,0,1)	(0,0,1/2,1/2)	(0,0,0,1)

## Vita

Hsu-Chih Wu was born on Jan. 1, 1976 in I-Lan, Taiwan, Republic of China. He received the B. D degree in the Department of Computer and Information Science from National Chiao Tung University, in 1997. Since Sep. 1997, he was a graduate student in the Institute of Computer and Information Science at National Chiao Tung University.

Since Sep. 1998, he is working for the Ph.D. degree in the Institute of Computer and Information Science at National Chiao Tung University. His current research interests include Evolutionary Computation, Information Visualization, Analysis of Strategic Games, and Social Simulation.

