STRONG RECIPROCITY, SOCIAL STRUCTURE, AND THE EVOLUTION OF COOPERATIVE BEHAVIOR

by

Shade Timothy Shutters

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

ARIZONA STATE UNIVERSITY

December 2009

STRONG RECIPROCITY, SOCIAL STRUCTURE, AND THE EVOLUTION OF

COOPERATIVE BEHAVIOR

by

Shade Timothy Shutters

has been approved

July 2009

Graduate Supervisory Committee:

Ann Kinzig, Co-Chair
Berthold Hölldobler, Co-Chair
Jürgen Liebig
Charles Perrings
Charles Redman

ACCEPTED BY THE GRADUATE COLLEGE

ABSTRACT

The phenomenon of cooperation is central to a wide array of scientific disciplines. Not only is it key to explaining some of the most fundamental questions of biology and sociology, but it is also a cornerstone of understanding and successfully overcoming social dilemmas at multiple scales of human society. In addition, cooperation is considered crucial to any hope of long-term sustainable occupation of the earth by humans. Drawing on a broad interdisciplinary literature from biology, sociology, economics, political science, anthropology, law, and international policy analysis, this study uses computational methods to meld two disparate approaches to explaining cooperation – individual incentives and social structure. By maintaining a high level of abstraction results have broad applicability, ranging from colonies of social amoebae or ants to corporations or nations interacting in markets and policy arenas. In all of these cases, actors in a system with no central controller face a trade-off between individual goals and the needs of the collective. Results from evolutionary simulations of simple economic games show that when individual incentives, in the form of punishment, are coupled with social structure, especially complex social networks, cooperation evolves quite readily despite traditional economic predictions to the contrary. These simulation results are then synthesized with experimental work of others to present a challenge to standard, narrow definitions of rationality. This challenge asserts that, by defining rational actors as absolute utility maximizers, standard rational choice theory lacks an evolutionary context and typically ignores regard that agents may have for others in the local environment. Such relative considerations become important when potential interactions of a society's individuals are not broad and random, but are governed by

emergent social networks as they are in real societies. Finally, analysis of the implications of these findings to efficacy of international environmental agreements suggest that conventional strategies for overcoming global social dilemmas may be inadequate when other-regarding preferences influence national strategies.

for a demon-haunted world

ACKNOWLEDGEMENTS

It is with great humility that I thank my advisor, mentor, and friend, Dr. Ann Kinzig, who invited me into her lab with only a desire to aid what she perceived to be a lost but capable student. She and her students, Kris Gade, Bethany Cutts, Steven Metzger and Maya Kapoor, were my second family during the long years of this research. This is true too of the extended lab group Ann built around us including Jason Walker, Brad Butterfield, Gustavo Garduño-Angeles, and Elisabeth Larson.

In addition to committee members Bert Hölldobler, Jürgen Liebig, Charles Perrings, and Charles Redman, I was guided and encouraged by many others that gave selflessly of their time, especially J. Marty Anderies. David Hales (University of Delft, Netherlands) and Kim Hill gave critical feedback on this work. Sebastiano Alessio DelRe (Università Bocconi, Italy) and John Murphy (University of Arizona) provided assistance with the simulation program. For much-needed support I give special thanks to Christofer Bang, Cyd Hamilton, and Mary Laner.

For providing both a unique intellectual atmosphere and limitless administrative support, I am deeply indebted to ASU’s IGERT in Urban Ecology, Center for Social Dynamics and Complexity, and School of Life Sciences, each with more staff, faculty, and students to whom I owe gratitude than I could possibly list here. Long-standing relationships with the Math & Cognition Group, the Social Insect Research Group, and the ecoSERVICES Group also helped solidify ideas developed in this dissertation.

Finally, this dissertation would never have reached completion without the infinite patience and unwavering support of my loving wife, Callen Shutters.

This work was supported through generous fellowships from Arizona State University's Integrative Graduate Education and Research Traineeship (IGERT) in Urban Ecology (NSF Grants # 9987612 and # 0504248) and the U.S. National Science Foundation. Additional funding was provided by Indiana University-Bloomington, University of Bologna, Italy, University of Washington-Friday Harbor, University of Alaska-Fairbanks, The Air Force Office of Scientific Research, and both the Graduate College and School of Life Sciences at Arizona State University.

Any opinions, findings, conclusions or recommendations expressed in this material are mine alone and do not necessarily reflect the views of the National Science Foundation.

TABLE OF CONTENTS

Page

CHAPTER Page

CHAPTER Page

LIST OF TABLES

Table Page

Table | Page

LIST OF FIGURES

ABBREVIATIONS

CHAPTER 1

INTRODUCTION

> government…though composed of men subject to all human infirmities, becomes, by one of the finest and most subtle inventions imaginable, a composition which is in some measure exempted from all these infirmities
> - Mancur Olson (1965)

The history of life is punctuated by the periodic emergence of new hierarchical levels of organization, among them eukaryotic cells, multi-cellular life, eusociality, and institutions (Michod 1997). Like the human institution of government described by Olson above, these emergent levels are made of populations of individuals which become entities in their own right, often existing far longer than the life spans of the individuals of which they are comprised. These emergent entities frequently have global attributes and an evolutionary trajectory not predictable from observation of the component individuals. At the heart of these emergent levels of living organization lies the phenomenon of cooperation (Maynard Smith and Szathmáry 1997, Michod 1997), a mechanism that can facilitate collective action by individuals and make it possible for the collective to become itself an individual.

This dissertation is an investigation into cooperative behavior – a phenomenon that remains largely unexplained by science and whose evolution is one of the greatest questions facing evolutionary biologists (West et al. 2007). A broadly applicable explanation of cooperation remains elusive despite years of theoretical and empirical investigation. Some researchers even suggest that the current state of sociobiologic inquiry is in disarray (Wilson and Wilson 2007). This is due partly to the recurring debate over the units of selection and simple semantics of social behavior (West et al. 2007), partly to a resistance by theoreticians to move beyond formal mathematical models

(Bedau 1999, Griffin 2006), and partly to the slow pace of incorporating new insights from emerging fields such as evolutionary economics, social network theory, and complexity science (Sawyer 2005).

In addition, researchers are hindered by a narrow definition of cooperation wherein agents subordinate their individual self-interest to that of a larger group. In other words, the quest for an explanation of cooperation is often confused with a quest for an explanation of altruism (West et al. 2007). While not denying the existence of seemingly altruistic acts, such as a soldier jumping on a grenade to save his comrades, the focus of this dissertation is on behaviors that allow individuals to benefit a larger group while also being evolutionarily beneficial to the individual. Though it may seem an individual is behaving in the interest of a collective and not himself, those acts may nonetheless increase the evolutionary fitness of the individual.

Experimental research on cooperation through the use of laboratory games generally assumes that participants will attempt to maximize payoffs in a game if the participants are rational. This is a subtle but important departure from rational choice theory, in which actors maximize utility, not payoffs (Mas-Colell et al. 1995). In this study I explore the implications of equating payoffs and utility in this manner. In addition, I incorporate the biological concept of relative fitness which further confounds predictions of economic experiments.

Though fundamental questions of cooperation remain unanswered there are other pressing reasons that justify its investigation. Many environmental problems today require coordinated, collective action among nations of the world if they are to be solved (Oldero 2002, Rees 2002, Kaul and Mendoza 2003, Beddoe et al. 2009). As societies, driven by a desire to avoid their own "tragedy of the commons" (Hardin 1968), increasingly push for sustainable management of the earth's resources, international cooperation is required. Given that sustainability is a desirable policy course for a growing number of nations (Beddoe et al. 2009), and that cooperation is often a prerequisite for coordinated global sustainable management, a fundamental understanding of when and under what conditions cooperation will emerge becomes imperative.

As defined in this dissertation, cooperation results when individuals act in a manner that produces some beneficial collective or social outcome, even though those individuals may have an incentive to cheat or act otherwise. Currently there is no broadly compelling theory that explains why unrelated individuals choose to cooperate. This is equally true at the international level where the actors are nations of the global community that may be called upon to cooperate in multinational initiatives to the detriment of their own national self-interests. But as Sandler (2004, p. 260) points out, "nations will sacrifice autonomy only in the most desperate circumstances."

A long tradition of western philosophy holds that cooperation is obtainable only through top-down coercion by a central authority. As Hobbes ([1651] 1946) asserted in his classic work *Leviathan*, "there must be some coercive power, to compel men equally to the performance of their covenants, by the terror of some punishment." However, a coercive central power is often the least desirable solution to social dilemmas (Ostrom 1990). Furthermore, for environmental problems requiring international cooperation, a central power does not currently exist that could carry out such enforcement (Sandler 1999, Wagner 2001, Barrett 2003b, 2005, Hodgson 2009)[1].

On the other hand, overwhelming evidence from both case studies and experiments does not support the Hobbesian notion that a Leviathan is the only path to collective action. Instead it shows that cooperation can emerge in the absence of a central controller despite predictions of economic theory that selfish individuals, left to their own devices, will fail to cooperate. Attempts to understand these empirical results have recently led to the concept of strong reciprocity – the idea that cooperation persists because individuals inflict costly punishment on cheaters and bestow rewards on cooperators, and in either case receive no benefit in return.

Like cooperation, punishment is ubiquitous among social organisms. Where cooperating individuals have an incentive to cheat, punishment mechanism often exist to deter cheating (Frank 1995). This includes restricting cancer cell growth through preprogrammed death or senescence (Sharpless and DePinho 2005), toxin release by colonial bacteria that affects only non-cooperators (Travisano and Velicer 2004), the destruction of eggs laid by workers in social insect colonies (Foster and Ratnieks 2001), and enforcement of mating and dominance hierarchies in non-human mammals (Clutton-Brock and Parker 1995, Dugatkin 1997b). Even the process of cellular meiosis can be viewed as a form of policing selfish genes (Michod 1996). In humans, punishment and policing are common across societies and many cultural groups (Marlowe et al. 2008)

---

[1]Abbott et al (2000) argue that the closest thing currently to an effective global government is the World Trade Organization.

and neurological research suggests this behavior is to some degree genetically coded in humans (Sanfey et al. 2003, de Quervain et al. 2004, Spitzer et al. 2007).

In this dissertation I examine the ability of punishment mechanisms to induce a society or group to act cooperatively. In particular I explore the effects of punishment under different ratios of costs between the punisher and punishee and examine the role that social network structure plays in the ability of a society to cooperate. In addition, I examine the effects of both retaliation by a punished agent and punishment of those who refuse to punish cheaters, both of which are typically ignored in punishment studies (Clutton-Brock and Parker 1995, Nikiforakis 2008). The primary method of investigation used in this work is agent-based computer modelling, a computational tool that incorporates genetic algorithms, heterogeneity, mutation, and selection to simulation evolutionary trajectories of decision making behavior.

It is important to distinguish between promoting cooperation in expectation that it will lead to the provision of a public good and promoting cooperation merely for cooperation's sake. Though often implied that cooperation is always desirable, the costs required to facilitate cooperation may outweigh its benefits. In this dissertation I present evidence to support this point, especially when punishment is the mechanism used to induce cooperative behavior.

The structure of this dissertation is as follows. Chapter 2 outlines background information on the problem of cooperation and presents theoretical fundamentals on a variety of topics meant to facilitate understanding of this study by a wide audience. Chapters 3 and 4 present empirical results from computer simulation experiments showing that in structured societies, punishment can increase contributions to a public good but that it may also lead to detrimental side-effects. Chapter 5 presents simulation results demonstrating that, unlike cooperation, fairness, defined as a roughly equal division of resources, is not induced by a combination of punishment and social structure. Chapter 6 develops a theoretical framework to explain experimental results from this dissertation and other sources and shows that a model in which agents make decisions based, in part, on the decisions of others, best explains results. Chapter 7 applies this theoretical framework to the design of international environmental treaties, primarily those intended to promote sustainability and the provisioning of global public goods, and shows that treaties may not work as intended when nations are concerned with relative position. Finally, Chapter 8 reviews major findings of this dissertation and discusses possible future research related to this work.

This study not only advances understanding in evolutionary biology, theoretical sociology, and behavioral economics, but also has practical applicability to transboundary environmental management and policy science.

CHAPTER 2

COOPERATION: PROBLEM DISCUSSION AND BACKGROUND

Before any discussion on cooperation can proceed it is necessary to address years of semantic confusion on the topic and to define the terms being discussed. Due in large part to the multidisciplinary nature of inquiry into cooperative behavior, many terms have been used interchangeably in various literature. This is especially true of the terms cooperation, altruism, mutualism, symbiosis, and reciprocity. These terms are often confused by the same author at different points in his or her career and at times even in the same literary piece (West et al. 2007). In advocating a clear distinction between cooperation and altruism, West et al (2007) define altruism as a behavior that is costly, in terms of biological fitness, to the individual performing the act but beneficial to another, while they define cooperation as behavior by an individual that benefits others, in terms of fitness, and which is evolutionarily selected for *because* of the benefit it bestows (see also Travisano and Velicer 2004).

Though West and colleagues present an attempt to clarifying the confusing semantics of social behavior, they do so almost exclusively from a biological perspective. For applicability to a broader audience, especially those in social sciences, cooperation in this dissertation is defined as behavior by an individual that produces a beneficial collective or social outcome, even though those cooperating individuals may have an incentive to cheat or act otherwise.

Early Work on the Evolution of Cooperation

Among researchers vexed by cooperative behavior was Charles Darwin who could never, to his own satisfaction, rectify observations of seeming altruism with his own theory of natural selection (Sulloway 1998). In his seminal work Darwin fretted that explaining altruism ubiquitous in social insects was “to me insuperable, and actually fatal to my whole theory” ([1859] 1996, p. 192). Subsequent researchers did not afford the question a high-priority since it could easily be explained by the classical notion of group selection. However, when the theory of group selection was largely discredited in the 1960s (Olson 1965, Williams 1966, Hagen 1992) scientific interest in the topic of cooperation was renewed (Axelrod and Hamilton 1981).

Two theories emerged at this time as extensions of neo-Darwinian evolution that were thought to explain most instances of cooperation: Hamilton’s (1964) theory of kin selection, which was thought to explain cooperation between related individuals (especially between non-human animals), and Trivers’ (1971) theory of direct reciprocity, which was thought to explain cooperation between unrelated individuals. These theories were so influential that many today still assume, though incorrectly, that nearly all instances of cooperation and altruism can be explained by these two theories (West et al. 2007).

Following pioneering work of Robert Axelrod in 1981, cooperative phenomena became a major focus of computational research. In a series of computer simulation tournaments between various prisoners dilemma strategies, it was shown that cooperative strategies could be evolutionarily stable despite the ever present incentive to cheat (Axelrod and Hamilton 1981). An immense response from researchers followed in which various parameters and settings of Axelrod’s original model were altered (see Dugatkin 1997a for a detailed review of this work). Subsequent experiments led to a number of important insights that have helped move the field toward a broadly applicable theory of the evolution of cooperation.

One modification was the introduction of stochasticity. When strategies were executed probabilistically instead of deterministically, cooperative outcomes became much less likely (Nowak 1990). The same was found to be true if agents probabilistically made mistakes in the execution of their strategies (Hirshleifer and Coll 1988).

A more important modification of Axelrod's model was the introduction of space. Axelrod's original work grew out of evolutionary game theory, in which techniques of population biology are used to explore evolutionary stability of game situations (Maynard-Smith 1982). The technique, however, is limited to exploration of equilibrium points within an infinite, homogeneous, and well-mixed population (Killingback and Doebeli 1996). Nowak and colleagues were among the first to include spatial explicitness and to demonstrate that it could lead to qualitatively different outcomes in terms of cooperation (Nowak and May 1992, Nowak et al. 1994).

Other Theories

The work of Axelrod and his successors is generally classified as a direct reciprocity theory of cooperation (described below). This is only one of a group of theories that has emerged in an effort to understand cooperative behavior. Below are brief descriptions of important theories regarding the evolution of cooperation, along with major criticisms of each.

Inclusive fitness

Inclusive fitness theory, now often called kin selection, redefines an individual's fitness as a product of how well that individual's genes are propagated, regardless of who carries the genes (Hamilton 1964). In other words, an individual acting altruistically toward close relatives can pass on copies of genes through those relatives' offspring in addition to its own offspring. An initial requirement that individuals share a common ancestry was thought to limit the theory's applicability and later researchers broadened the definition of related individuals to include those that share particular genes of interest, regardless of ancestry (West et al. 2007). Despite this broader definition and its explanatory power regarding social insects and certain animal groups, inclusive fitness remains unsatisfactory for explaining cooperative behavior that is common between unrelated individuals in human societies (Di Paolo 1999, Abbot et al. 2001, Wilson 2005).

Direct reciprocity

The idea of direct reciprocity is embodied in the phrase "you scratch my back now, I'll scratch yours later". This theory, formerly (and often still) referred to as reciprocal altruism, asserts that when a population of agents reciprocates each other's cooperative behavior, that population will resist invasion by a selfish strategy (Trivers 1971). Axelrod and Hamilton's (1981) work with agents playing the iterated prisoner's dilemma is one example. However, because its underlying requirements and assumptions are so restrictive, direct reciprocity has fallen out of favor as a general theory of cooperation and few researchers still believe it is has applicability beyond certain situations involving humans (Dugatkin 1997a, West et al. 2007). In addition, direct reciprocity typically requires long-term repeated interactions and cannot explain cooperation in anonymous one-shot interactions – a phenomenon growing ever more prevalent in human societies (Nowak and Sigmund 2005).

Indirect reciprocity

In contrast to direct reciprocity, the idea of indirect reciprocity can be summarized as "you scratch my back, I'll scratch someone else's" (Nowak and Sigmund 2005). According to this theory, after an agent unconditionally confers a benefit on a 2nd agent, this 2nd agent will at some later time confer an unconditional benefit on a 3rd agent and so on (Leimar and Hammerstein 2001). Like direct reciprocity, indirect reciprocity requires a lengthy period of repeat interactions, though unlike direct reciprocity, these interactions must only be within the same group and not with the same individual. This excludes indirect reciprocity also as an explanation of cooperation in anonymous one-shot interactions.

A related concept is that of tag recognition, or the so-called "green-beard" phenomenon, in which a benefit is unconditionally conferred on another, but only to an interaction partner that exhibits the proper trait or signal (Macy and Skvoretz 1998, Ostrom 1998, Riolo et al. 2001). This mechanism may lead to the development of reputation, which has been shown to promote cooperative acts between repeatedly interacting individuals, both human (Nowak and Sigmund 1998b, Nowak and Sigmund 1998a, Suzuki and Toquenaga 2005) and non-human (Zehavi and Zahavi 1997). However, the ability of reputation to induce cooperative behavior requires the reliable reception and interpretation of signals by an individual, which may not happen despite broadcast of signals by another individual

Multi-level selection

Multi-level selection theory, also know variously as demic selection, intrademic selection, trait-group selection, or new group selection, asserts that a trait's frequency can increase in a population because it confers a benefit on a group of individuals, not on the individuals themselves (West et al. 2007). In contrast to classical group selection, which was discredited by both evolutionary ecologists and political economists during the 1960's (Olson 1965, Williams 1966, Hagen 1992), the contemporary theory of multi-level selection is a more general version of classical group selection (Wilson 2007) and has provided explanatory power for many social insect phenomena as well as cultural patterns in isolated human populations (Wilson and Hölldobler 2005, Hölldobler and Wilson 2009).

Multi-level selection theory was initially met with resistance because it required that selection forces be easily parsed into within-groups and between-groups forces (Price 1970). This, in turn, required that some members of a group have alternating periods in its life history in which it is at one time within a clearly demarcated group and at another time well-mixed with those of other groups (Wilson 1975). Consequently, multi-level selection was not widely accepted as an explanation of cooperation among organisms that do not form clearly defined groups, including dynamic human societies. Theories incorporating the idea of population viscosity sought to remedy this, but with mixed results (Queller 1992, Mitteldorf and Wilson 2000).

Eventually researchers dispensed with the requirement of clearly defined groups, which has turned multi-level selection into a powerful theory in several fields. This broadening of the theory is best summarized by Wilson and Wilson (2007), who assert that "groups need not have discrete boundaries; the important feature is that social interactions are *local*, compared to the size of the total population." Through this broader definition this dissertation contributes to the theory of multi-level selection by identifying complex social networks as the substrate that delivers the required local interactions. This is discussed further in the conclusion in Chapter 8.

Others have criticized a broader definition of groups on the grounds that it blurs the distinction between multi-level selection and inclusive fitness, creating confusion and negatively affecting the ability to execute and interpret research on the evolution of social behavior (West et al. 2007). Whether this broadening of multi-level selection theory has been beneficial or counterproductive continues to be a source of contentious debate (Wilson 2007, West et al. 2008).

Strong Reciprocity

Theories described above have been variously in and out of favor since Darwin first raised the issue of cooperation. However, the quest for a fundamental understanding of cooperation through use of simulations and experimental games during the past three decades has increasingly focused on the concept of strong reciprocity – the idea that individuals reward others who cooperate and punish those who do not (Gintis 2000, Bowles and Gintis 2004). These acts of rewarding and punishing are performed altruistically in that the agent conferring the reward or punishment incurs a cost but obtains no material benefit in return. Though West et al (2007) assert there is nothing altruistic about the punishment and rewards that comprise strong reciprocity, I retain the terminology here to be consistent with contemporary literature.

Altruistic punishment, in particular, has been shown empirically to induce cooperative outcomes in social interactions (Fehr and Gächter 2000, 2002, Boyd et al. 2003, Fehr and Fischbacher 2004, Gardner and West 2004, Fowler 2005, Fowler et al. 2005). This finding is echoed by those engaged in statecraft, who assert that punishment mechanisms are prerequisites for successful international environmental agreements (Barrett 2003a, b).

Though composed of two principles – punishment and reward – strong reciprocity research has been dominated by work on punishment, and it is now well-established that altruistic punishment can increase contributions in public goods games (see below). Researchers seem sufficiently sure of punishment's ability to induce cooperation that they have moved to advocating its use by policy makers, both at local scales, in institutions governing common pool resources (Ostrom et al. 1992, Ostrom et al. 1994, Dietz et al. 2003), and at global scales, where non-compliance with environmental treaties must be deterred without the aid of an independent enforcement authority (Barrett 2003a, b).

An important parameter governing the mechanism of altruistic punishment, and one that will be referenced throughout this dissertation, is the ratio of costs incurred by the punishing party to those of the party being punished (Casari 2005). Letting c = the cost which an individual incurs to punish another, cM is then the fee or sanction imposed on the punished party where M is a parameter of the model referred to as the punishment multiplier. In an evolutionary context, when costs and benefits represent fitness, as M becomes arbitrarily large there should be some point at which it could no longer be considered altruistic to provide punishment but is instead evolutionarily beneficial. M, therefore, becomes an important parameter in understanding outcomes of punishment experiments.

As noted, altruistic punishment is only one aspect of strong reciprocity, and though it has dominated research on the topic, the phenomenon of altruistic rewarding should not be ignored. Those that advocate punishment, such as Ostrom (1994) and Dietz (2003), briefly discuss the benefits of rewards or incentives in social dilemmas but list only sanctioning mechanisms in their recommended institutional solutions. One explanation for less attention to rewards may be that experiments have demonstrated the threat of punishment leads to higher contributions in public goods games than the promise of rewards (Sefton et al. 2002, Andreoni et al. 2003). In addition, case studies of successfully managed common pool resources typically credit punishment instead of rewards (Ostrom 1990, Ostrom et al. 1992, Ostrom et al. 1994), though this may be a product of researcher preferences or bias.

Economic theory does not predict that reward systems should be inferior to punishment systems. However, from a biological perspective the disparity is not unexpected. As stated above, there should be times when punishment is an evolutionarily beneficial strategy (Shutters 2009). On the other hand, a reward given will always reduce the fitness of the rewarder relative to the agent receiving the reward.

Rational Choice Theory

Cooperation is the collective result of individual behavior and therefore the result of a series of individual choices. To begin to understand cooperation requires first an adequate understanding of theories of choice. By far the dominant paradigm for explaining how individuals choose among several alternatives is that of rational choice theory. This theory argues that individuals choosing from a vast set of options first rank those options in order of preference and then choose the option that is most preferred, given constraints on the ability to acquire those choices. To be rational is to have the following properties[2] with respect to preferences:

1) preferences are *complete* – given the set of all available consumption choices X, an individual can consistently rank his preferences for any two choices $c_1, c_2 \in X$ so that one of the following is true: $c_1 \succ c_2$ (read c_1 is preferred to c_2), $c_1 \prec c_2$, or $c_1 \sim c_2$ (read c_1 is indifferent to c_2 , or the individual is indifferent between c_1 and c_2);
2) preferences are *transitive* – given three choices c_1, c_2, and $c_3 \in X$, if $c_1 \succ c_2$ and $c_2 \succ c_3$, then $c_1 \succ c_3$; and if $c_1 \sim c_2$ and $c_2 \succ c_3$, then $c_1 \succ c_{3;}$
3) preferences are *non-satiable* – given c = consumption of some good and ε = some incremental consumption of the same good, $(c + \varepsilon) \succ c$ and $u(c + \varepsilon) > u(c)$. In other words, consuming more of a good is always better in terms of utility (Mas-Colell et al. 1995).

[2] Though only properties 1 and 2 are requirements of rational choice theory, property 3 is included as an important corollary customarily listed as a component of rationality.

Under the preceding restrictions all choices confronting an individual can be ranked in order of preference. For purposes of formalized models it is preferred to examine choices in terms of an individual's utility, which may be defined simply as an individual's satisfaction or happiness (Rayo and Becker 2007). Formally however, utility is a function of consumption $u(c)$ that quantifies preference rankings such that if $c_1 \succ c_2$, then $u(c_1) > u(c_2)$ and if $c_1 \sim c_2$, then $u(c_1) = u(c_2)$. Therefore, a rational decision maker, in choosing the most preferred consumption alternative, maximizes his utility. Note that in standard rational choice theory the utility function of one individual is assumed to be independent of preferences and consumption of others, though most economists would now agree that this is simply a best first approximation of behavior. This point will be discussed in detail in Chapter 6.

Game Theory

A common methodology for testing theories of choice and rationality is the use of controlled laboratory games. The economic theory of games provides a highly simplified framework for analyzing decision making under constraint (see Binmore (1992) or Osborne (2004) for a comprehensive introduction to game theory). Competitive situations can be made sufficiently abstract that they become mathematically tractable, are applicable across species or entities, and are readily simulated by computer applications.

Games may be either one-shot (single stage) games or multi-stage games. A one-shot game requires strategic reasoning while multi-stage games require that the agent reason based on what it learns through repeated interactions (Weirich 1998). As learning models are beyond the scope of this dissertation, simulations are restricted to one-shot games. Agents' strategic choices are independent of both their own past choices and of the past choices of other agents in the population. However, because an agent's strategy in any given period is a result of the cumulative effects of evolution in an environment with other agents, one may validly argue that any agent's strategy is indirectly driven by past choices of itself and others.

Games may also be classified as either normal form or extensive form. An extensive form game describes a series of game plays in advance and thus applies only to multi-stage games. On the other hand, a normal form game describes a one-shot strategy and requires the strategy be causally independent of those against which it plays (Weirich 1998). As stated previously, experiments presented in this dissertation consist only of one-shot games. Therefore, only normal form games are used throughout.

Games may be further classified as either cooperative games or non-cooperative games. The designation of a game as cooperative or non-cooperative defines whether or not agents may make binding coalitions before strategies are played and should not be confused with whether or not the game has a cooperative outcome. In non-cooperative games agents may not make binding agreements before game play – agreements may be made in advance but they are non-binding and, therefore, not enforceable. Cooperative games, on the other hand, permit binding agreements prior to play which allows coalitions to form. These games have essentially two stages – a decision of whether to join a coalition or not and then a play of strategy. Experiments in this dissertation use only non-cooperative games, as a goal of this study is broad applicability to a variety of social species and at different hierarchical levels of society. In particular, attention is focused on the international level of human societies where actors are nation states. Both in non-human societies and at the international level of human societies there is no effective mechanism or institution that facilitates binding agreements. This is true in human society despite the existence of the World Court and the United Nations, institutions considered ineffectual for the purposes of enforcing binding agreements (Barrett 2003b). Accordingly, games which allow for binding coalitions are excluded in this dissertation. Following are descriptions of common experimental games relevant to this study.

The public good game

A public good game consists of n players. Each player i is given an endowment and then contributes a portion of that endowment x_i to a public good pool but keeps the remainder. Choice of x_i by each agent is made strictly independently of the choices of other agents. In this dissertation initial endowments for players in all games are standardized to 1 unit so that

[2.1] $$x_i \in [0,1].$$

A public good G is created by summing contributions from the n players and multiplying by some factor r that represents the synergistic effect of cooperation

[2.2] $$G = r\sum_{i=1}^{n} x_i \ .$$

To make the game meaningful for studying social dilemmas, r must be greater than 1 or individuals have no incentive to contribute to a public good. Likewise r must be less than

n or individuals have no incentive to retain their endowments. Accordingly,

[2.3] $$r \in (1, n) .$$

The public good G is then distributed evenly to all n players so that i's payoff p_i equals what the player did *not* contribute plus i's share of the public good:

[2.4] $$p_i = (1 - x_i) + \frac{G}{n} .$$

Substitution of [2.2] into [2.3] yields

[2.5] $$p_i = 1 + x_i\left(\frac{r}{n} - 1\right) + \left(\frac{r}{n}\sum_{j \neq i}^{n} x_j\right).$$

Since [2.3] requires that $r < n$, it follows that

[2.6] $$\left(\frac{r}{n} - 1\right) < 0 \ .$$

and any positive value for x_i in [2.5], ceteris paribus, will decrease player i's payoff.

In other words, for any given set of contributions by all other participants, an individual's payoff is maximized by contributing 0 to the public good. In contrast total social welfare, measured here as the sum of all payoffs, is maximized when every individual contributes its entire endowment to the public good.

The prisoners dilemma

The prisoners dilemma is a reduced version of the public good game played between only 2 players. Unlike the public good game, it is customary in the standard prisoners dilemma to limit strategies to those of full defection ($x = 0$) or full cooperation ($x = 1$), so that each agent simply faces a binary choice – cooperate or defect. Despite this simplification, the expected outcome of the prisoners dilemma is the same as that of the public goods game – rational agents attempting to maximize their own individual benefit will experience the least desirable social outcome.

A hybrid between the public good game and the standard prisoners dilemma is the continuous prisoners dilemma (CPD). In this version of the prisoners dilemma the game is still restricted to two players, but the set of allowable strategic choices is expanded to the entire interval [0, 1] as in the public good game. This allows for a richer set of possible outcomes while maintaining the analytical simplicity of a 2-person interaction (see Chapter 3 for a detailed description of the CPD).

The ultimatum and dictator games

Bargaining games comprise another class of experimental games. This category includes the ultimatum and dictator games, which are primarily used to understand the evolution of fairness.

The ultimatum game is played between two players *i* and *j* and is structured so that *i*, the proposer, is given an endowment from which a portion x_i must be offered to *j*, the responder. The offer may be any portion from 0% to 100% of the endowment. It is customary when possible to standardize the endowment to 1 so that $x_i \in [0, 1]$. The responder may accept the offer, in which case each player receives her agreed upon share, or may reject the proposal, in which case both players receive nothing. Economic theory predicts that a rational responder will accept the smallest possible positive fraction, and that a rational proposer, knowing this, will offer the smallest possible positive fraction (Binmore 1992).

In a simplification of the ultimatum game known as the dictator game, the responder has no ability to react. i simply gives a portion of its endowment to j and the game ends (Osborne 2004). In this case the economic expectation is that i will offer 0. This game is often used as a control case for comparisons to ultimatum game results.

Experimental games and rationality

In the context of controlled economic laboratory games it is customary to discuss experimental predictions and results in terms of the utility of a subject's payoffs $u(p)$ instead of consumption $u(c)$. Since monetary payoffs normally translate easily into purchasing power, this is a reasonable assumption and will be adopted throughout this text.

It is an altogether different matter that in nearly all experimental economics it is implied, if not expressly stated, that rational agents are expected to maximize their payoff p, not their utility from that payoff $u(p)$. This is a subtle but very important distinction. Experimenters can generally ignore this distinction by assuming that utility is a monotonic transformation of payoffs so that if $p_1 > p_2$, then $u(p_1) > u(p_2)$. In other words, agents will maximize utility $u(p)$ if they simply maximize their payoffs p. This assumption is valid only if payoffs are ordered the same as preferences for those payoffs.

If they are not, then there is no basis to predict game outcomes using strict rational choice theory.

This becomes a problem in light of numerous studies from experimental economists that demonstrate human behavior in laboratory games is not always consistent with payoff maximization. Instead, evidence suggests that a player's utility is a function, at least in part, of other players' payoffs. This phenomenon, termed interdependent preferences, is examined in detail in Chapter 6.

Contributing to this semantic confusion are the ways in which different authors use the terms *payoff* and *utility*. In the majority of behavioral literature cited in this study, expected payoff is the explicit quantified value (usually monetary) of a possible game outcome. For example, if choice A in a game earns a player 5 dollars, we would say the player's expected payoff for choice A is 5 dollars. Yet, in David Morrow's textbook *Game Theory for Political Scientists* (1994, p. 351) for example, the author defines the term payoff as "A player's utility for an outcome of a game." In other words, payoffs equal the utility of payoffs, or $p = u(p)$. Is a payoff the expected value of a game outcome or the expected utility of that outcome? Without loss of generality I consistently interpret payoffs in this dissertation as the expected value of a game and not the expected utility. This allows investigation of a given payoff matrix under different utility functions, including functions that incorporate interdependent preferences or predispositions for fair allocations discussed above.

Social Network Theory

The term “social structure” is increasingly confusing to those attempting to understand individual behavior by examining explicit connections between members of a society. The term is often used with models that segregate a society into smaller groups that are merely well-mixed subpopulations, especially multi-level selection models (e.g. Fryxell et al. 2007). This is still systems-level thinking with multiple, linked pools. However, the explicit connection pattern between individuals within these societies is still ignored. Here a narrower definition of social structure is proposed in which the connections of every member of a society to every other member are explicitly defined. In other words the social network of the society is defined.

Social network theory emerged from attempts by social scientists to understand social phenomena in terms of the connection pattern among a society’s individuals (see Wassermann and Faust (1994) for a comprehensive text on social networks and their analysis). At the same time but in other academic circles, the mathematical theory of graphs developed to understand quantitative properties of network structures. This has unfortunately led to alternate vocabularies and techniques to address what are effectively equivalent properties and phenomena of social networks. Throughout this dissertation the following groups of terms from social network theory and graph theory may be used interchangeably:

(a) social network, network, graph;

(b) connection, link, edge;

(c) agent, node, actor.

A social network is easily represented by a square matrix in which each node is uniquely identified by its column and row. The existence or absence of a link can then be listed between every possible pair within a society at the appropriate index site. Such a matrix is defined as an adjacency matrix. The simplest such matrix contains only binary information; $a_{ij} = a_{ji} = 1$ if agents *i* and *j* are linked and $a_{ij} = a_{ji} = 0$ if they are not (Figure 2.1). Index values may also contain information other than simply 0's and 1's such as the strength of a link, the cost of maintaining or using a link, probability of interaction, or any number of other meaningful types of information describing the relationship between two members of a population.

In addition, the simplest adjacency matrix represents what is known as a non-directed graph (Figure 2.1a). In a non-directed graph if node *i* is linked to node *j*, then *j* is linked to *i*. This need not always be the case. In some circumstances it is important to distinguish between the link from *i* to *j* and the link from *j* to *i*. This is done by means of a directed graph (Figure 2.1b). Simulation experiments in this dissertation use only simple binary, non-directed graphs.

A further designation of graphs is whether they are connected or non-connected. A connected graph is a network in which every node is reachable by every other node, regardless of how many links it may take to reach each other (Figure 2.2a). If there exists any node that is not reachable by every other node, then the network is not a connected graph (Figure 2.2b). As this dissertation is concerned with ability of behavior to propagate throughout a society or population only connected graphs are considered.

Social network metrics

There exist a myriad of quantitative descriptors of social networks, many of which require laborious algorithms to compute. Among them are at least four that are more relevant to the work of this dissertation.

Degree - The measure of degree can be confusing because it can describe both an individual node and an entire network. The degree of any node i in an undirected network is simply the number of direct links from i to other nodes. In other words it is a count of how many neighbors to which i is directly connected in a non-directed network. In contrast, the degree of a network (not just a single node) is the average degree of all nodes in the network. In other words if d_i = number of neighbors to which i is linked, then the degree of the entire network with n nodes can be represented as

[2.7] $$\overline{d} = \frac{1}{n}\sum_{i=1}^{n} d_i \ .$$

Geodesic and Eccentricity - The geodesic of two nodes is the shortest distance between them. When links are unweighted, so that they are all of equal length or value, the geodesic is simply the least number of links required to travel between two nodes. The geodesic ($d_{i,j}$) is synonymous with the popular notion of *degrees of separation* between nodes i and j. A related measure is a node's eccentricity, which is simply the largest geodesic between that node and all other nodes in the network.

Diameter of a graph - The diameter of a graph *D* is the largest eccentricity value among the nodes of the graph (and therefore the largest geodesic among the nodes). This can be a useful measure of how close the members of a society are to each other.

Clustering coefficient - The clustering coefficient is a measure of the density of local connections around a node. Specifically it measures how many neighbors of a node are connected to other neighbors of the same node. This measure is standardized by the number of possible connections between neighbors.

Classification of social networks

The following are descriptions of important classes of social networks or graphs used in this dissertation along with descriptions of how they are generated. Graphical representations of networks used in this research are presented in Figure 2.3 and Figure 2.4. The actual computer algorithms used to generate networks for this dissertation were written in the Java programming language and are presented in Appendix B.

Random networks - A random network, also known as an Erdős-Rényi graph, is simply generated by starting with a fixed number of nodes and then connecting pairs of nodes at random until the desired total number of links in the network is reached (Figure 2.4a). In this dissertation an extra step is added after a random network is generated to ensure that the graph is connected (see above). If the random generation process results in a non-connected graph, additional links are added until it is connected, resulting in an insignificant variation in total number of links among these networks. Random networks have the feature of low degrees of separation or network diameter, but also have low clustering coefficients.

Regular network - A regular network is often represented as a grid structure. All nodes have the same degree, or number of neighbors, and are arranged in a regular repeating pattern. In addition, such structures are torroidal, meaning that they have no edges but instead loop around onto themselves such as the surface of a sphere. Regular networks typically have high clustering coefficients but also high degrees of separation.

The two most commonly used regular networks in simulations are the von Neumann graph (Figure 2.3c), in which every node is connected to four neighbors in a torroidal grid, and the Moore graph (Figure 2.3e), in which every node is connected to eight neighbors in a torroidal grid. Hexagonal networks (Figure 2.3d) are also used but with less frequency.

Regular networks may also be one-dimensional instead of two-dimensional. In this case the resulting network is linear and, because it is also torroidal, is known as a ring (Figures 2.3a & b).

Random regular networks - Random regular networks have elements in common with both regular grid structures and randomly generated graphs. Like regular networks, all nodes in a random regular network have the same degree. However, links are generated randomly instead of in a regular, repeating pattern (Figure 2.4f). Like generation of an Erdős-Rényi, network pairs of nodes are linked at random but only until a node reaches some predetermined degree. At that point, the node becomes ineligible for further links.

Small world networks - Also known as a Watts-Strogatz graph, small-world networks show features of both random networks and regular networks (Figures 2.4d & e) and represent the dominant interaction pattern observed in human societies (Watts and Strogatz 1998). Watts-Strogatz graphs are generated by starting with a regular network and then randomly cutting and relinking ties in the network with a probability *r*. Watts and Strogatz found that over a certain range of *r*, networks are generated that exhibit features long sought by social scientists of both low degrees of separation and high clustering coefficients.

In this dissertation small world networks are generated by starting with a population structured in a linear ring (Figure 2.3a). This is known as the ring substrate method of generation.

Scale-free networks - Also known as a Barabási-Albert graph, a scale-free network is characterized by a power law distribution of nodal degrees and are ubiquitous in the real world from metabolic pathways to river drainage patterns to the hyperlink patterns of the world wide web (Barabási and Albert 1999). Scale-free networks are generated by growth through preferential attachment (Figure 2.4b & c). That is, the network is "grown" by adding new nodes one at a time until the desired number of total nodes is reached. As a new node is added, the point within the existing network at which it is attached is probability-based. The probability that any one node in the existing network will be the point of new attachment is proportional to the number of existing links that node already has (for a discussion of preferential attachment based on factors other than number of existing links see Ko et al. 2008).

Though scale-free networks as described above are sometimes classified as simply another form of small world network (e.g. Buchanan 2002), in this dissertation I draw a sharp distinction between the two. While scale-free networks may have the feature of low degrees of separation in common with small world networks, the clustering coefficient of a scale-free network is rarely high and is more reflective of random networks.

Agent-based Modelling

Because we live in a world of continuous change, standard theories of behavior are of little use in understanding dynamic behavior (North 2005). Not only are traditional mathematical methods of understanding dynamic systems limited to a small number of special cases, but those methods may have little potential for understanding how individual behavior affects emergent properties at higher scales (Anderies 2002, Harrison and Singer 2006). Therefore, argues North (2005), researchers must dispense with a quest for elegant mathematical equilibria if their goals are to truly understand behavior in a changing world (see also Janssen 2002).

In some cases the overzealous pursuit of mathematically simple models has led to unrealistic representations of a system (Harrison and Singer 2006). In the case of early models of group selection, simplifying assumptions made to allow formalized treatment led to vehement rejection of group selection theories and may have set back unbiased research on the topic for over 30 years (Wilson and Wilson 2007).

Prior to the availability of sufficient computational power, systems of interacting parts were often analyzed through sets of difference or differential equations. As computing power grew, methods of systems dynamics emerged in which computers were used to iteratively solve complex systems of equations. However, systems level analysis is still limited to discovering and describing macro-level, aggregate phenomena, and are not concerned with heterogeneous attributes of the individual components of a system (Sawyer 2005). While aggregate mathematical models have worked remarkably well for systems such as gas molecules, they may create particularly unrealistic representations of biological systems (Bedau 1999). This is especially true of societies (Sawyer 2005, Harrison 2006).

Since the early 1990's agent-based modelling has emerged as a preferred method of investigating such dynamic processes when analytical methods become intractable and it is ideal for creating models in which social structure is explicitly acknowledged or individuals are not atomistic clones of one another (Drogoul and Ferber 1994, Bedau 1999, Janssen 2002, Sawyer 2005, Wilson and Wilson 2007). In hierarchical terms, the modelled unit moves from the system to the individual entities that comprise the system. Thomas Schelling's (1971) Sugarscape model, which exhibited aggregate social patterns not predicted by equations, is often acknowledged as the first fruitful use of agent-based modelling in the social sciences. Nowak and May further demonstrate the shortcomings of systems level models by showing that restricting agents to a defined structure of possible interactions led to very different outcomes than when the same system was modelled as a well-mixed aggregate (Nowak and May 1992).

Agent-based modelling has achieved acceptance in part from a growing realization of the limitations of reductionist science (Sawyer 2005) and by renewed attempts to use holistic or organic approaches to understanding a social system as a whole

(Sawyer 2005, Harrison 2006). To fully explore the evolution of cooperation in systems of social agents, computational social simulation, or agent-based modelling, is used as the primary method of investigation in this dissertation. Unlike laboratory experiments the use of agent-based simulations allows careful control over factors that may confound empirical studies such as emotion, reputation, visual cues, anonymity, or cultural influences (Cederman 2001). This control allows researchers to single out cultural and other factors that may be most important to facilitating cooperation and allows almost unlimited creativity in designing virtual experiments. More importantly, agent-based models go beyond the capabilities of mathematical analysis to allow investigation of dynamic systems far from stable equilibrium points. In particular agent-based modelling is used in this study to explore behavior in populations of agents that play dynamic, evolving strategies in various economic game situations.

However, it should be noted that this dissertation does not advocate simulations of networked populations as a panacea for understanding the evolution of social behavior. For instance, certain social aspects of eusocial insect colonies are described well by systems of differential equations (Reeve and Hölldobler 2007) even though there is evidence that colony interaction patterns exhibit characteristics of social networks (Fewell 2003). Instead the explanatory power of this dissertation is most applicable to those superorganisms whose internal structure is clearly described by complex social networks.

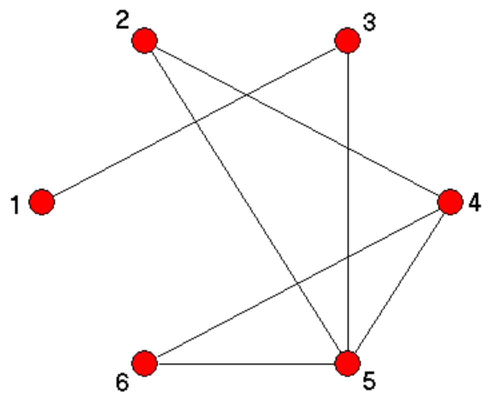
One criticism of agent-based computer simulations is that their results are often produced from proprietary programs or cannot otherwise be independently verified (Bedau 1999). The appropriate response to this is to verify that results are independent of software platform and that results are reproducible by sufficiently informed colleagues. This can be accomplished by supplying colleagues with only pseudo-code, or a concise description of a simulation, and confirming that they are able to reproduce results (Edmonds and Hales 2003). When possible, simulation programs developed for this dissertation have been independently recreated by fellow researchers[3] with sufficient expertise in agent-based modelling using only pseudo-code as a guide.

The simulation algorithms developed for this dissertation[4] will continue to have research uses well beyond those of this project. In addition to follow-up questions that may arise from this research, seemingly unrelated questions about the evolution of cooperation may be explored easily once models such as these are standardized.

---

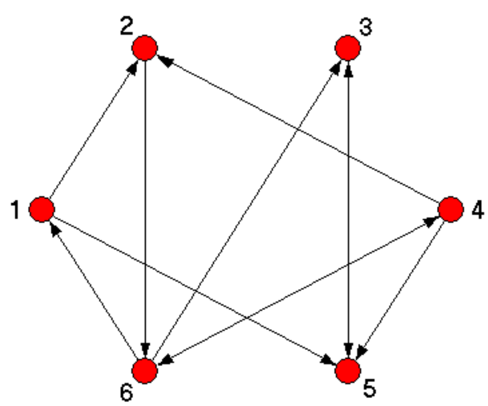
[3] Preliminary findings of this dissertation were successfully replicated by Dr. Francois Bousquet of CIRAD, France, using the CORMAS modeling platform (http://cormas.cirad.fr/indexeng.htm), and by S. Alessio Delre, of the University of Groningen, Netherlands, using the C programming language.

[4] Simulations were written in Java 1.5.0 (http://java.sun.com/) using the Eclipse 3.2 software development kit (http://www.eclipse.org/). See Appendix B for detail code.



	1	2	3	4	5	6
1-	0	0	1	0	0	0
2-	0	0	0	1	1	0
3-	1	0	0	0	1	0
4-	0	1	0	0	1	1
5-	0	1	1	1	0	1
6-	0	0	0	1	1	0

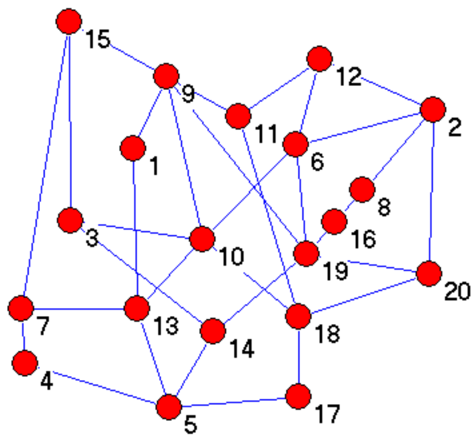
(a) Example of a non-directed graph with 6 nodes and its representation as an adjacency matrix. Note that the adjacency matrix of a non-directed graph is symmetrical about the main diagonal.



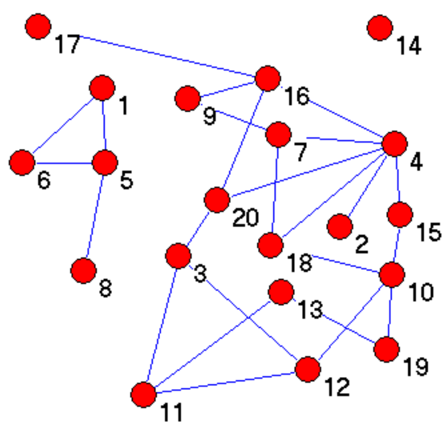
	1	2	3	4	5	6
1-	0	1	0	0	1	0
2-	0	0	0	0	0	1
3-	0	0	0	0	1	0
4-	0	1	0	0	1	1
5-	0	0	1	0	0	0
6-	1	0	1	1	0	0

(b) Example of a directed graph with 6 nodes and its representation as an adjacency matrix. The adjacency matrix of a directed graph need not be symmetrical.

Figure 2.1. Examples of directed and non-directed graphs and their adjacency matrices.



(a) A connected graph with 20 nodes. Every node is reachable by every other node.



(b) A non-connected graph. No node is reachable by every other node. Note that node 6 is reachable only by three other nodes, while node 14 is not reachable by any other node.

Figure 2.2. Examples of connected and non-connected graphs. Because this dissertation examines cases where interactions take place between agents, only connected graphs are used throughout.

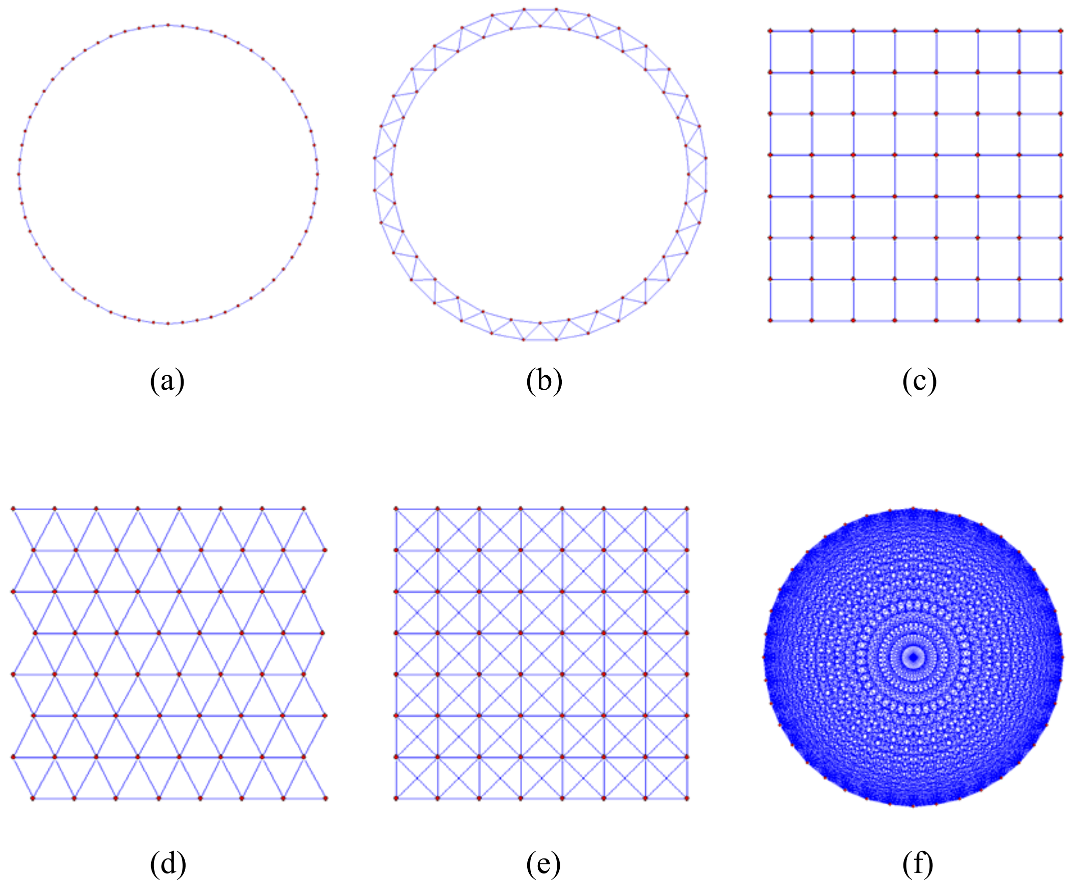
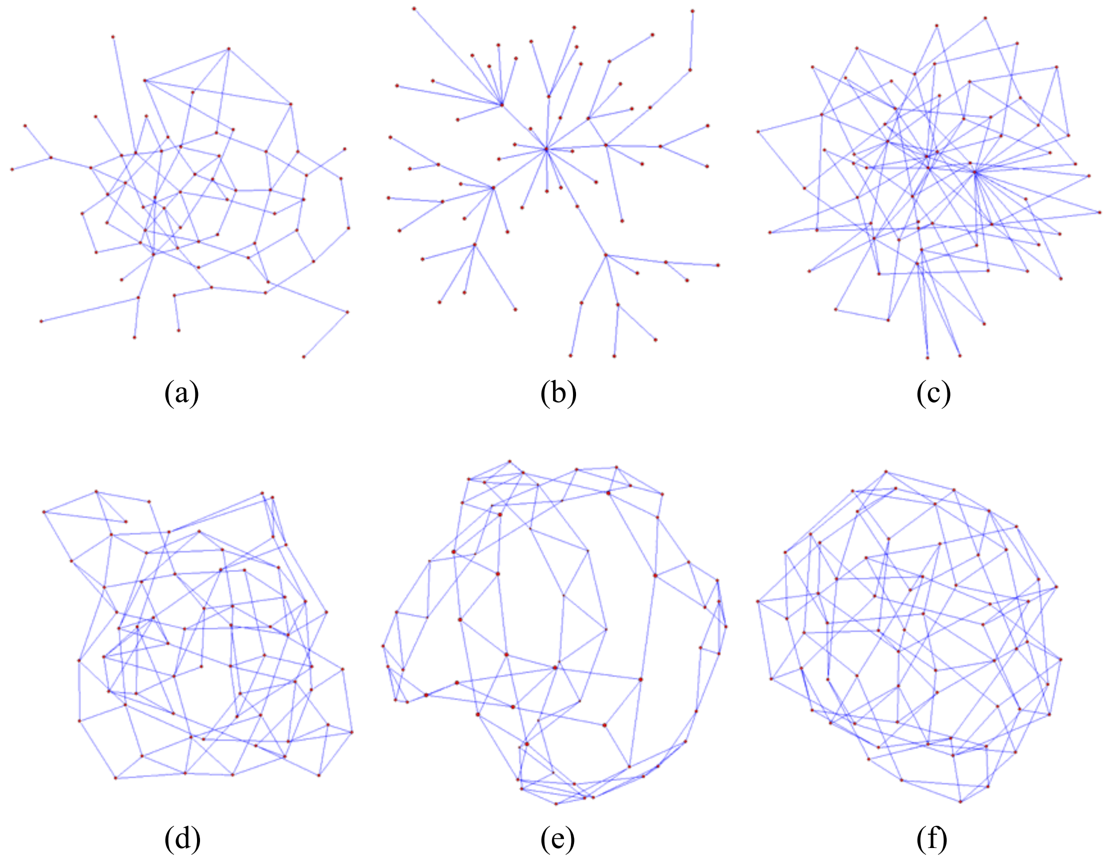


Figure 2.3. Examples of regular networks used in this dissertation. Each example network is composed of 64 nodes. (a) A ring with neighborhood radius = 1. (b) A ring with neighborhood radius = 2. (c) A von Neumann lattice. (d) A hexagonal lattice. (e) A Moore lattice. (f) A complete graph. Examples c, d, and e are known as regular lattices and are actually torroidal, meaning they bend back around on themselves to make a single surface with no edges.



(a) (b) (c)

(d) (e) (f)

Figure 2.4. Examples of non-regular networks used in this dissertation. Each example network is composed of 64 nodes. (a) A random network with probability of link = 0.20. (b) A scale-free network with one link per new node. (c) A scale-free network with two links per new node. (d) A small-world network using a ring substrate with neighborhood radius = 2 and probability of rewire = 0.2. (e) A small-world network using a ring substrate with neighborhood radius = 2 and probability of rewire = 0.05. (f) A random regular network with 4 links per node.

CHAPTER 3

PUNISHMENT AND SOCIAL STRUCTURE:

COOPERATION IN A CONTINUOUS PRISONERS DILEMMA

Introduction

The phenomenon of cooperative behavior remains unexplained in several branches of science. Though a number of mechanisms have been proposed to explain at least some observable instances of cooperation (Hamilton 1964, Trivers 1971, Axelrod and Hamilton 1981, Wilson and Sober 1994, Fehr et al. 2002, Foster et al. 2004) they invariably apply to limited cases or special circumstances (see Chapter 2, Alternate Theories for further discussion).

Arguably the leading contemporary explanation for the evolution of cooperation is the phenomenon of altruistic punishment. Altruistic punishment occurs when an individual incurs a cost to punish another without receiving any material benefit in return (Fehr and Gächter 2002). This mechanism has been shown repeatedly to induce cooperative behavior in laboratory experiments with humans, where subjects often pay to punish players that are not even in the same game as the punishee (Ostrom et al. 1992, Fehr and Gächter 2000, 2002, Andreoni et al. 2003, Gürerk et al. 2006).

From an economic perspective, however, the phenomenon of altruistic punishment is just as irrational as cooperation and as an explanation of cooperation in one-shot anonymous interactions it only shifts the question from “why should an individual cooperate?” to “why should an individual altruistically punish?” It therefore remains to demonstrate a causal mechanism for altruistic punishment if it to explain the evolution of cooperation. One mechanism currently proposed as the key to altruistic punishment is cultural group selection (Richerson and Boyd 2005, Hagen and Hammerstein 2006). This dissertation offers evidence for at least one mechanism leading to the evolution of altruistic punishment – social structure.

Like other explanations of cooperation, many theories of altruistic punishment are limited by the fact that they are framed in terms of evolutionary game theory (Maynard-Smith 1982) and fail to address the social structure governing interactions between actors (Jackson and Watts 2002). Such explanations assume a system is homogeneous or well-mixed and that members of the system interact randomly with each other with equal probability. These system dynamics models have limited applicability to groups of social organisms (Sawyer 2005, Griffin 2006, Harrison and Singer 2006).

Previous simulations have shown that adding simple two-dimensional space leads to very different behavior than simple well-mixed population models (Schelling 1971, Nowak and May 1992). Despite a growing tendency to include social structure in simulation models, there is still a bias for use of overly simplistic, regular two-dimensional lattices. Real-world network structures of cooperating agents are known to be far from well-mixed, yet neither do they conform neatly to the regular pattern of a lattice (Barabási and Albert 1999, Amaral et al. 2000, Dorogtsev and Mendes 2003).

Social scientists, on the other hand, have long acknowledged the association between complex social networks and cooperation (Oliver 1984, Marwell et al. 1988, Gould 1993, Chwe 1999, 2000). Yet only recently has the simulation and modelling community begun to move beyond regular lattice structures to explore the important role that complex social networks play in the evolution of cooperation (Santos and Pacheco 2005, Santos et al. 2006b, Chen et al. 2007, Olfati-Saber et al. 2007).

It remains then, to explore the combined effects punishment and social structure and whether their combination may lead to a broadly applicable explanation of cooperation. As Michael Chwe (1999) lamented,

> Collective action has been studied in two largely disjoint approaches, one focusing on the influence of social structure and another focusing on the incentives for individual participation. These approaches are often seen as competing or even opposed.

This chapter attempts to bridge these two approaches by focusing on altruistic punishment as the mechanism for the evolution of cooperation and by demonstrating how incorporating social structure may make punishment a viable mechanism for the evolution of cooperation. Agent-based computer simulations were conducted to test the ability of altruistic punishment to induce cooperation on a variety of social network structures.

The Simulation Model

To test the ability of altruistic punishment to induce cooperation, a punishment option was incorporated into simulations of the continuous prisoners dilemma played out on a variety of social networks. These networks included a complete graph, representing a well-mixed system, several regular lattices representative of some of the first computational simulations and often analogous to spatial explicitness, and more sophisticated complex social networks, such as scale-free[5] and small-world networks[6], representative of many real-world processes in physical, biological, and social systems.

---

[5] This simulation uses a Barabási-Albert (1999) type algorithm to create scale-free networks by preferential growth. That is, the network is “grown” by adding new nodes one at a time until the desired population level is reached. As a new node is connected, the point within the existing network at which a new node is attached is probability-based. The probability that any one node in the existing network will be the point of

The continuous prisoners dilemma (CPD)

In the classic prisoners dilemma players are limited to two choices - cooperate or defect. Here that requirement is relaxed and players are able to select a level of cooperation at any point on a continuum between full cooperation and full defection. This presents an arguably more realistic picture of choices facing those in social dilemmas (Sandler 1999, Killingback and Doebeli 2002). In this dissertation the set of contribution choices is standardized to the interval [0,1] so that 0 = full defection and 1 = full cooperation. This is known as the continuous prisoners dilemma (CPD).

The CPD can also be thought of as a simplified version of the public goods game described in Chapter 2 with only 2 players, *i* and *j*. When $n = 2$, [2.4] is modified so that *i*'s payoff becomes

[3.1] $$p_i = 1 - x_i + r(x_i + x_j)/2; \quad r \in (1,2).$$

The addition of altruistic punishment introduces a 3rd player to the game, the observer *k* and the possibility of punishment further modifies potential payoffs. The CPD payoff matrix used in this chapter is presented in Table 3.2.

new attachment is proportional to the number of existing connections that node already has. An important parameter to consider when growing these networks is the number of links that a new node makes to the existing network. Results in this study were obtained using scale-free networks grown by nodes that linked to two nodes of the existing network. Supplementary simulations with single-link attachment showed no appreciable difference in results.

[6] This simulation uses the Watts-Strogatz (1998) algorithm for creating small-world networks by random rewiring of a regular ring structure). That is, the algorithm begins with a simple circle of connected agents (a ring substrate) then randomly rewires links from an adjacent node in the circle to one selected at random anywhere in the population. An important parameter governing this algorithm is the radius of agents in the initial circle that are considered neighbors. Let r = the neighborhood radius which equals the number of links between an agent and what constitutes a neighbor. Given any agent A, when $r = 1$, only the two agents on either side of A are considered neighbors; when $r = 2$, two agents in each direction from A (four total) are considered neighbors, and so forth. Results presented in this study for small-world networks were obtained using a ring substrate with $r = 2$. Supplementary simulations using a ring substrate with $r = 1$ showed no appreciable difference in results.

Like the public goods game, for any given contribution by an opponent, an individual's payoff is maximized by contributing 0 to the public good. This is the expected rational choice or Nash equilibrium of the prisoners dilemma (Binmore 1992). The dilemma arises, however, because total social welfare is maximized when both individuals cooperate fully. Theory predicts that, given rational agents, each player in the CPD will contribute 0 to the public good and regardless of the amount of the agent's contribution, an observing neighbor will never pay to punish (Fehr and Gächter 2000).

Game play

A simulation run initiates by creation of a social network. Let $\mathcal{N}(\mathcal{V},\mathcal{E})$ be a connected network where $\mathcal{V}$ is the array of vertices or nodes and $\mathcal{E}$ is the array of edges or links. Each node is occupied by a single agent i consisting of strategy (x_i, t_i, c_i) where x_i = the contribution i makes to the public good when playing against j, t_i = the contribution below which the agent will punish another agent in a game being observed by i, and c_i = the cost that i is willing to incur to punish the observed agent when the observed agent's contribution is too low (Table 3.1). In other words t_i determines if agent i will punish and c_i determines how much agent i will punish. Each strategy component x_i, t_i, $c_i \in [0,1]$ and is generated randomly from a uniform distribution at the beginning of each simulation. To control for other factors that might contribute to the maintenance of cooperation, such as interaction history or reputation, the model does not allow recognition of or memory of other agents within the population. Every game is effectively one-shot and anonymous.

During a single CPD game an agent i initiates the encounter by randomly selecting j from its neighborhood, which unless otherwise indicated, consists of all nodes one link away from *i* in the given network type. Agents are given their endowment of one unit from which each simultaneously contributes a portion to a public good. Payoffs are then calculated using the payoff matrix in Table 3.2. The initiating player *i* then randomly selects a second neighbor *k*, who is tasked with observing and evaluating *i*'s contribution. If *k* judges the contribution to be too low ($x_i < t_k$), *k* pays c_k to punish *i* in the amount of $c_k M$, where *M* is the relative strength of punishment referred to here as the punishment multiplier. Each agent initiates three CPD games during a single generation of the simulation and each simulation run proceeds for 10,000 generations.

Each generation consists of three routines – game play, observation & punishment, and selection & reproduction. During each routine an agent interacts only with its immediate neighbors as defined by the network type and all interactions take place in parallel. The payoff variable for each agent *p*, tallies the costs and payoffs an agent experiences during a generation. Because this model depicts the elementary case in which parents do not differentially provision resources for their offspring, $p = 0$ for each agent at the beginning of a new generation[7].

Following game play and punishment agents compete with one another for the right to pass offspring to the next generation. During this reproduction routine each agent *i* randomly selects a neighbor *j* with which to compare respective payoffs accumulated during the generation. If $p_i > p_j$, *i*'s strategy remains at *i*'s node in the next generation. However, if $p_i < p_j$, *j*'s strategy is copied onto *i*'s node for the next generation. In the event that $p_i = p_j$, a coin toss determines the prevailing strategy. As strategies are copied

[7]Many species, especially social animals, do contribute to the success of their offspring through resource provisioning or parental care. See (Wilson 2000) for examples.

to the next generation each of the three strategy components of every agent is subject to mutation with a probability $m = 0.10$. If selected for mutation, Gaussian noise is added to the component with mean = 0 and std. dev. = 0.01. Should mutation drive a component's value outside [0,1] the value is adjusted back to the closer boundary value.

Simulation variables and output

The important parameter governing the mechanism of altruistic punishment is the ratio of costs incurred by the punishing party to those of the party being punished (Casari 2005, Shutters 2008). Defined above as the punishment multiplier M, this parameter is analogous to the strength or efficiency of punishment and, along with network type, is the independent variable in these simulations. The dependent variables of interest are the mean contribution and the mean payoff which evolve in a population after 10,000 generations. The mean contribution represents the population's level of cooperativeness while the mean payoff represents the population's social welfare.

Data were collected in two sets. In the first data set, 100 simulation replications were conducted at $M = 0.0$ and then at subsequent values of M in increments of 0.5, up to $M = 6.0$ (Table 3.2). This allowed for an analysis of variances in outcomes for a given simulation parameter set. In the second data set, a parameter sweep of M was conducted so that a single simulation was run at 0.0 and at subsequent values of M in increments of 0.01 up to $M = 6.0$ (Figure 3.1). This allowed for an analysis of the effect of M at higher resolution but at the cost of no replications.

Results

Control case: no social structure, no punishment

For control purposes the initial population was simulated on a complete graph with no altruistic punishment. As predicted by rational choice theory, the population evolved contribution rates of approximately 0. In the absence of both punishment and social structure no cooperation was exhibited.

Either punishment or social structure alone

In the second set of simulations, populations were subjected to alternate treatments of either social structure or punishment. First, with punishment disabled simulations were run on a variety of network structures (Table 3.4). Unlike similar experiments with fairness in the ultimatum game (Chapter 5), social structure alone did not drive outcomes from the Nash equilibrium (Table 3.2) and no cooperation evolved. Only in the anomalous case of scale-free networks did contributions deviate from the expected contribution rate of approximately 0.

Next, using a well-mixed complete graph analogous to no social structure, simulations were run in which punishment was enabled. Despite having the ability to punish each other populations lacking structure continued to evolve to the Nash equilibrium with increasing M (Figure 3.1a). Neither social structure alone nor punishment alone was sufficient to induce the population to evolve away from non-cooperative behavior. Again these results concur with rational expectations.

Punishment and social structure together

In the final round of simulations, the CPD was played using both structured populations and altruistic punishment for which the punishment multiplier *M* was systematically varied. Results are presented in Figure 3.1. With increasing *M*, punishment eventually led to nearly full cooperation on a Moore lattice and a small-world network. Under these network types as *M* increased, populations underwent a rapid transition from contributions ~ 0 to contributions ~ 1 (Figure 3.1b & 3.1c). This flip from nearly full defection to nearly full cooperation occurred also in supplemental simulations run on the following social structures: von Neumann lattice, hexagonal lattice, and linear (ring) structures (Table 3.4). Response curves to these structures were so similar to those of the Moore lattice and small-world network that their figures are excluded for the sake of brevity.

Interestingly, populations using scale-free networks neither evolved to the Nash equilibrium nor showed any significant response to the introduction of altruistic punishment. Though Figure 3.1d reveals a slight positive trend in mean contributions under a scale-free network with increasing *M* ($R^2 = 0.007$), the trend is not significant (Spearman rank order correlations, $p = 0.086$).

Discussion

Cooperation in continuous versus discrete games

In treatments with social structure and no punishment, populations playing the continuous prisoners dilemma evolved contribution levels of approximately 0. This presents an interesting contrast to early simulation work of Nowak and May (1992), in which spatially arrayed populations played the standard prisoners dilemma. Recall that in the standard prisoners dilemma agents are restricted to only two choices – cooperate or defect (see Chapter 2 above). In other words the standard prisoners dilemma is the discrete choice counterpart of the continuous prisoners dilemma. Nowak and May found that under certain parameter settings populations evolved to an equilibrium mixture of cooperators and defectors. This may indicate an important difference between continuous and discrete games and should be explored further. However, Nowak and May also used a non-torroroidal, finite plane, which gives rise to edge effects that are not present in the current study. Furthermore, their simulations used initial populations of 90% cooperators whereas in this study's initial population uses a uniform distribution of public good contribution levels.

Localization of interactions and the evolution of altruistic punishment

Though altruistic punishment is now accepted as a mechanism for maintaining cooperation, it remains to explain the evolution of the punishment mechanism itself. This is true because, just as an agent contributing in the prisoners dilemma receives a lower payoff than those that do not contribute, an agent that punishes receive a lower payoff than those that do not punish.

Social structure, through its restriction of agents to local interactions, offers one possible explanation for the rise of the seemingly irrational phenomenon of altruistic punishment. Results are robust when moving from artificial social structures, such as regular lattices, to stochastically generated small-world networks which more realistically represent complex human interaction patterns. Furthermore, because social networks are not exclusive to human societies (Fewell 2003, Lusseau and Newman 2004, Flack et al. 2006) this finding may have broad applicability wherever social organisms engage in costly punishment.

Under the parameters of these simulations it is clear that punishment, as a mechanism for the evolution of cooperation, is only a viable explanation in the presence of structured populations. It would appear that Chwe is correct in his call for a melding of those exploring social structure and those studying individual incentives as mechanisms for collective action.

Cooperation and network density

These results also shed light on a contemporary debate among network scientists regarding the role that social networks play in facilitating cooperation. In particular there has been lively discussion on the role of "dense" networks, or what is defined in this study as complete (or nearly complete) networks. A long-held belief is that when a population is more densely connected the likelihood of cooperation increases (Marwell and Oliver 1993, Opp and Gern 1993, Jun and Sethi 2007). On the other hand, recent research suggests the opposite and shows that dense networks inhibit cooperation in a structured population (Flache and Macy 1996, Flache 2002, Takács et al. 2008). Results from this dissertation support the latter view. Simulations using the maximally dense complete network never evolved cooperation even when the punishment multiplier was set to the unrealistic value of M = 5,000. Instead, cooperation evolved only on sparsely linked networks (Table 3.4, Table 3.5).

The view that increasing network density adversely affects cooperation is further supported by the results from regular networks. Though full cooperation eventually evolved on each of the regular networks, the severity of punishment (measured as the magnitude of *M*) required to move the population from defectors to cooperators increased as the density of the network increased (Table 3.5). In other words, the more densely a network was linked, measured as the number of neighbors per agent in a regular lattice, the stronger the punishment required to evolve cooperation (Figure 3.2). This finding is in direct contrast a study by Jun and Sethi who conclude that "dense networks are more conducive to the evolution of cooperation" (Jun and Sethi 2007, p. 625).

Social dilemmas and their underlying social structure

Results from these simulations reveal that it may be possible to classify social dilemmas based on the social structure under which they occur. If so, it may give policy makers a new tool by creating a system of institutional recommendations for each structural class of social dilemma. For instance, social dilemmas occurring in a society characterized by a small-world network may be amenable to institutionalized punishment. Dilemmas characterized by interactions following a scale-free network or a highly dense network may require other institutional solutions.

The anomaly of scale-free networks

An unexpected simulation result was the response to punishment of populations embedded in scale-free networks. Unlike other social networks used in this experiment, populations on scale-free networks appear to be unresponsive to punishment even as *M* increases. To ensure that results were not due to an inadequate sampling of the model's parameter space, simulations were run on scale-free networks at $M = 5{,}000$ but again resulted in no convergence.

Another possible explanation for these results is that convergence on scale-free networks takes longer to emerge. Therefore, simulations were re-run at $M = 1.5$ but extended to 200,000 generations. However, even after extending the evolutionary period by 20 times, no convergence in contribution rates occurred.

These results suggest that there are features unique to scale-free networks that should be identified through further investigation. This is especially important given that scale-free architecture is common in nature and is known to exist in widely diverse organic systems, from cellular signal transduction pathways to the world wide web (Barabási and Albert 1999).

Evolutionary dynamics

Results presented thus far have consisted of a population's mean contribution at the end of 10,000 simulated generations. Before ending discussion it is important to examine the evolutionary trajectory through time. While it is true that one advantage of evolutionary computer simulations is the ability to store data of every interaction during every generation, the immense data processing and storage requirements that would be needed to completely analyze evolutionary dynamics was beyond the scope of this dissertation. However, a small subset of such trajectories is presented as examples of population dynamics over time. Complete generation-level data was collected for three individual simulations run on small-world networks. Simulations were selected to give examples of a population that evolved to full defection at low strength of punishment (*M* = 0.5), a population that evolved to full cooperation at higher strength of punishment (*M* = 3.0), and a population that evolved to an intermediate level of cooperation at *M* = 1.75, corresponding to the chaotic transition range in Figure 3.1c.

Results of this brief survey are presented in Figure 3.3. Simulations run at *M* = 0.5 and *M* = 3.0 converged to full defection and full cooperation respectively within the first 400 generations. In the intermediate range near the transition between defection and cooperation (*M* = 1.75), the population's mean contribution rate did not converge over time to either cooperation or defection but instead drifted in a random fashion. To ascertain whether the population was simply converging more slowly at punishment strengths near the transition point, the simulation run at *M* = 1.75 was extended to 100,000 generations but still exhibited no convergence in contribution rate.

Cooperation in other games

It is important acknowledge that the prisoners dilemma is but one of many 2-person games used to explore and understand social dilemmas. Though it is more commonly used than others, it is likely not representative of all social dilemmas. Manipulation of the payoff structure in the prisoners dilemma leads to several other games with alternative equilibria and expected outcomes. Hauert (2001) gives a comprehensive description of different games that arise when ordinality of payoffs changes. It is likely that all these alternate games have applicability to at least some real world social dilemmas.

One alternative game enjoying increased attention from scientists in recent years is the snowdrift game, also known as the chicken game or the hawk-dove game. Unlike the prisoners dilemma the snowdrift game has two Nash equilibria, neither of which is the least socially desirable outcome (defect-defect). In recent simulations of the snowdrift game on a regular (von Neumann) network, results showed that, in contrast to the prisoners dilemma, spatially structuring the population actually inhibits cooperation (Hauert and Doebeli 2004).

Future Directions

A limitation of this study is that comparisons were made between different networks based on a nominal classification scheme. While this does allow for a test of significance through ANOVA, it is less desirable than a general linear model in which CPD contributions could be regressed against one or more numerical descriptors of the underlying networks. As stated above, several such quantitative statistics exist to describe social networks (Wasserman and Faust 1994). However, the computational requirements to calculate such statistics for even a single randomly generated network are extensive and the exploration of evolutionary space required to generate a meaningful linear model would require the random generation and quantitative measure of hundreds or even thousands of networks. There is currently no feasible way to accomplish this high-volume quantitative analysis in addition to the computational requirements of the CPD simulations themselves. To move forward with evolutionary network science such as that presented in this dissertation, it is important that such a computational solution be developed.

Second, even though several nominal classifications of networks were used in this experiment the simulated worlds remain essentially flat and one-dimensional. An approach more representative of the complexities of real-world societies would be the use of multiple hierarchically nested networks. For instance, a model of metapopulations may place populations at each node of a network. However, each population may itself be made of multiple interacting actors arranged in their own network. The same is true for models of international relations networked nations are made of networks of people. In addition, there is no reason why an actor at one hierarchical level may not interact with an actor at another level. While adding multiple layers of complexity to such models it should also lead to a much richer array of outcomes for analysis and hypothesis testing.

Table 3.1

Strategy components used by agents in the continuous prisoners dilemma

Component	Description
x	contribution to public good
t	threshold for punishment
c	amount or cost of punishment

Note: x, t, $c \in [0, 1]$

Table 3.2

Payoffs *p* in the continuous prisoners dilemma between *i* and *j* with possible punishment of *i* by *k*

	$x_i \geq t_k$	$x_i < t_k$
k punishes *i*?	no	yes
p_i	$1 - x_i + r(x_i + x_j)/2$	$1 - x_i + r(x_i + x_j)/2 - c_k M$
p_j	$1 - x_j + r(x_i + x_j)/2$	$1 - x_j + r(x_i + x_j)/2$
p_k	0	$- c_k$

Note: see Tables 3.1 and 3.3 for description of variables

Table 3.3

Simulation parameters for the CPD and their values

Parameter	Values
The population size (*N*)	400
The number of generations a single simulation run	10,000
The number of games initiated by each agent in one generation	3
The range for strategy component values (*x*, *t*, *c*)	[0, 1]
The probability of strategy component mutation (*m*)	0.1
The (mean, standard deviation) of Gaussian noise added to a mutated strategy component	(0, 0.01)
The punishment multiplier (*M*)	0.0 to 6.0[a]
The public good multiplier (*r*)	1.5[b]
The probability of rewire for small-world networks	0.05
The number of links per new node in scale-free networks	2

[a] In increments of 0.01.

[b] An alternative representation of the public good multiplier *r* is to be standardized by the number of players per game. Stated in this manner *r* is bounded by $0.5 < r < 1$ for the prisoners dilemma and is fixed at $r = 0.75$ in all cases in this dissertation.

Table 3.4

Mean ending contributions on various networks with and without punishment

	Mean contribution (std. dev.)	
Network type	*M* = 0.0	*M* = 4.0
Complete graph	0.003 (0.001)	0.030 (0.010)
Regular graphs		
Moore	0.004 (0.001)	0.990 (0.017)
Hexagonal	0.005 (0.001)	0.997 (0.002)
von Neumann	0.005 (0.001)	0.998 (0.001)
Linear	0.006 (0.001)	0.996 (0.002)
Complex, real-world graphs		
Small-world	0.006 (0.001)	0.997 (0.001)
Scale-free	0.490 (0.310)	0.666 (0.232)
Other graphs		
Random	0.023 (0.012)	0.455 (0.284)
Random regular	0.005 (0.001)	0.999 (0.001)

Note: in each case number of replications = 100

Table 3.5

For regular networks, approximate value of *M* at which populations transitioned from low to high contributions in the continuous prisoners dilemma

Network type	Number of neighbors	Approx. transition value of *M* [a]
Linear	2	1.5
von Neumann	4	1.8
Hexagonal	6	2.2
Moore	8	2.8
Complete	$N - 1$	N/A[b]

[a] See Appendix A for method of approximating transition values.

[b] A transition did not occur on the complete graph with increasing *M*. This was true even at values as high as $M = 5{,}000$.

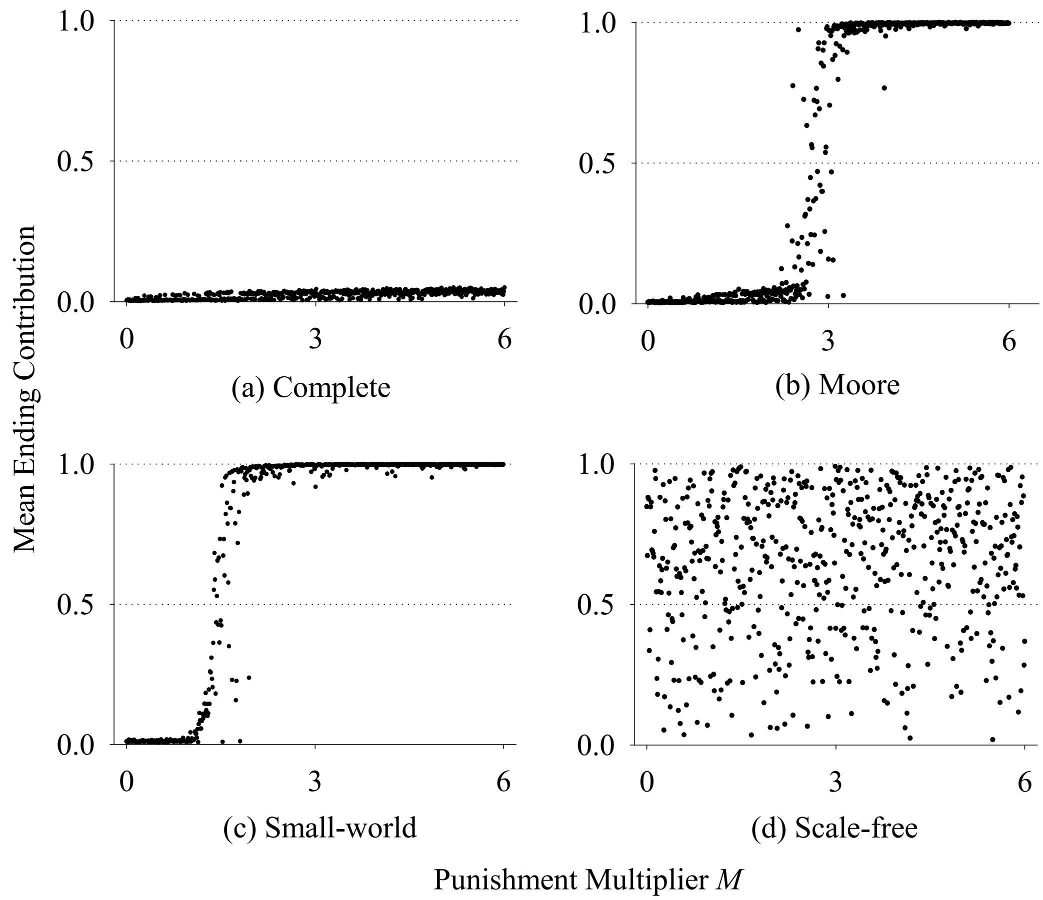
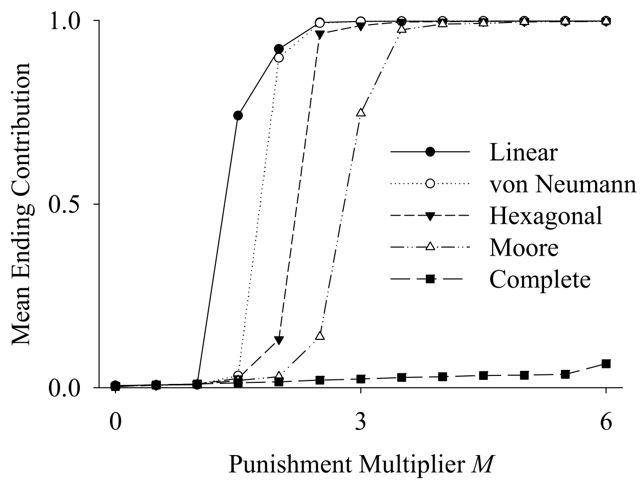
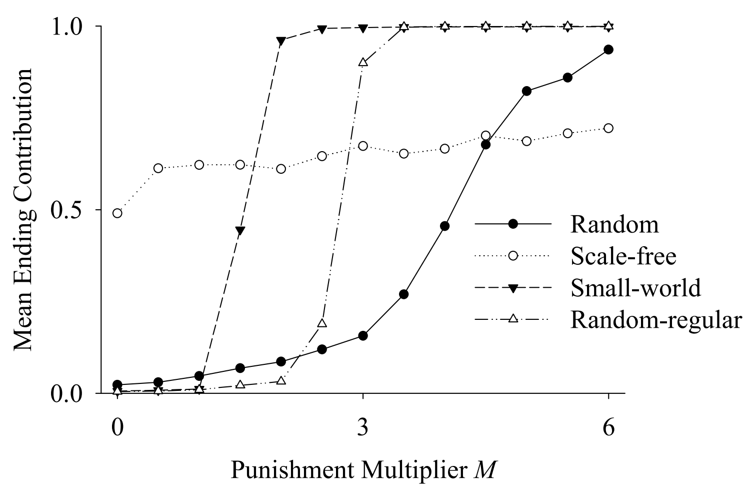


Figure 3.1 Results of the continuous prisoners dilemma on four different networks. Mean contributions vs. *M* are presented for populations on (a) complete network, (b) a regular (Moore) lattice, (c) a small-world network, and (d) a scale-free network. Each dot represents the population's mean contribution in the 10,000th generation of a single simulation run. Simulations on small-world networks clearly demonstrate a transition effect as *M* increases. Scale-free networks exhibit no such effect.



(a)



(b)

Figure 3.2. Response of mean ending contributions to increasing *M* in the continuous prisoners dilemma. (a) on regular network structures, and (b) on other networks. For networks that experience a rapid transition from low to high contributions, approximate transition values are listed in Table 3.5.

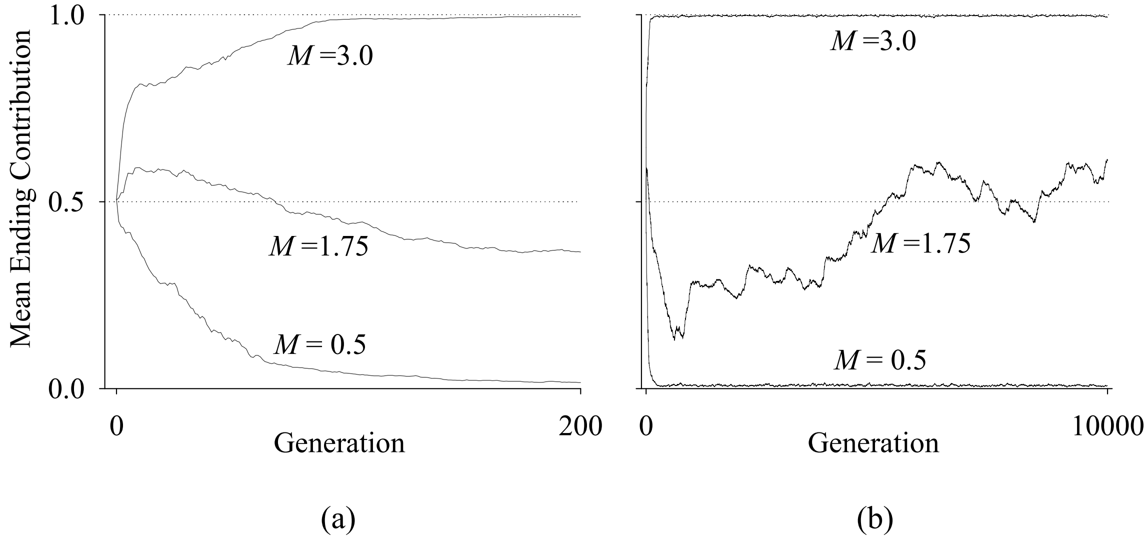


Figure 3.3. Evolutionary dynamics of the continuous prisoners dilemma on small-world networks. (a) Through 200 generations. (b) Through 10,000 generations. While simulations run at $M = 0.5$ and $M = 3.0$ converged to full defection and full cooperation respectively within 200 generations, the simulation run at $M = 1.75$ did not converge to any value even after 10,000 generations. The end point of each curve in (b) corresponds to a single point in Figure 3.1c.

CHAPTER 4

EXTENDING THE CONTINUOUS PRISONERS DILEMMA MODEL: THE AFTERMATH OF PUNISHMENT[8]

In this chapter, the continuous prisoners dilemma (CPD) simulation developed in Chapter 3 is modified to answer a serious of supplemental questions. These questions are related to the premise that in real world situations, or action arenas, the unilateral punishment of a non-cooperator is rarely the final interaction in a social dilemma. Therefore, experiments in this chapter investigate what happens after punishment takes place.

The Detrimental Side of Punishment

Thus far this dissertation has demonstrated that punishment can induce a structured population to cooperate, provided that the punishment multiplier is sufficient. However, it remains to examine whether such punishment-induced cooperation has a favorable impact on social welfare, measured as the sum of all individual payoffs in the population. Laboratory experiments have shown that even when punishment leads to increased contribution rates in a public good game, it may consistently decrease overall social welfare in the form of total payoffs (Sefton et al. 2002). This occurs because the fees collected from those wishing to inflict punishment, as well the sanctions collected from those being punished, are not redistributed by the experimenter and may be greater than the benefits from increased contributions to the public good.

Unfortunately this has led to ambiguity in the literature regarding the efficacy of altruistic punishment. For example, in a ground-breaking laboratory experiment by Fehr

---

[8] This chapter is based, in part, on (Shutters 2008).

and Gächter (2000), participants played the public goods game for 20 rounds and had the ability to punish others after each round. Treatments in which punishment was allowed led to higher contributions than when punishment was not allowed and the authors concluded that, since free-riding was deterred, punishment had facilitated cooperation. Yet in 18 of 20 rounds with punishment, average payoffs to all participants was actually lower than without punishment. So while punishment induced higher contributions to the public good it led to decreased social welfare.

Herrmann et al (2008) found that in some human societies, those contributing to a public good are punished just as frequently as non-contributors. This “antisocial punishment”, as the authors call it, can be so strong that it destroys the ability of punishment to facilitate cooperative outcomes.

Therefore, it is with great care and caution that scientists should approach policy makers to advocate the use of punishment as some have done (Ostrom et al. 1992, Ostrom et al. 1994, Barrett 2003b, a, Dietz et al. 2003). Using this argument, an oppressive and coercive central power that punishes those whose views do not concur could be considered a source of cooperation as long as it deters free-riding, but this may come at terrible cost to individual liberty (Marlowe et al. 2008).

Punishment versus payoffs

In the previous chapter, simulations of the CPD were used to examine the effect of punishment on contributions to a public good. However, it is prudent to also investigation the effect that punishment has on payoffs. Close examination of the rapid transition in contributions that occur in small-world and regular networks reveals that mean payoffs actually drop as *M* increases but before the transition occurs (Figure 4.1). This suggests that unless *M* is sufficiently high, altruistic punishment can actually lead to decreased social welfare. In the complete network, where there is no transition to high contributions, mean payoffs simply continue to decrease with increasing *M*.

Once the transition occurs to cooperative behavior, further increases in *M* beyond its transition value again decreases total payoffs (Figure 4.2). These results indicate that once a society achieves widespread cooperation some level of punishment persists, and suggest that there is an optimal strength of punishment at the point just beyond the transition to full cooperation. Any attempt to craft institutions that promote punishment as a mechanism for inducing cooperation will face a practical problem of attempting to find this optimal formula for punishment. At worst, a poorly crafted punishment regime will lead to worse payoffs than without punishment.

The 2^{ND} Order Free-rider Problem

To facilitate the provisioning of a public good it is often the case that institutional solutions are implemented to deter free-riding. However, these institutions are themselves public goods and the question then arises of how these institutions are maintained (Hodgson 2009). What deters free-riding in the provisioning of deterrence institutions? This is the essence of what is known as the 2^{nd} order free-rider problem (Okada 2008).

For example, it is common for human societies to employ police to enforce laws. A police force is tasked with detection and punishment of 1^{st} order free-riders. However, what incentives exist to ensure that members of a police force carry out their duties? Without deterrents and/or incentives it is expected that a rational enforcer would collect wages but then rely on fellow police officers to carry out enforcement of laws – a costly endeavor in terms of individual risk to the enforcement officer (Oliver 1980).

This scenario may continue for several levels, each with a new free-rider dilemma. If a police force should create an internal affairs department to ensure that its members are carrying out their enforcement duties, we then ask what incentives do internal affairs agents have to carry out their internal enforcement duties?

It is expected that individuals that cooperate but that do not punish others – individuals Heckathorn (1998) refers to as *private cooperators* – will have an evolutionary advantage over those that both cooperate and punish. This has been shown in experimental games where those that cooperate but do not punish receive the highest payoffs (Dreber et al. 2008). However, if punishers are responsible for cooperative outcomes but are evolutionarily inferior to those that do not punish, it is expected that they will evolve out of the population, taking any hope for general cooperation with them. Therefore, even if we conclude that cooperation is maintained in a society by the tendency of individuals to inflict costly punishment on non-cooperators, it remains to explain in an evolutionary context how these punishers could out-compete other cooperators that do not punish.

In addition to those that cooperate but do not participate in enforcement, there may also be individuals that cheat or defect with regard to provisioning a public good but participate actively in sanctioning other cheaters (*hypocritical cooperators*). This is a further complication that contributes to expected frailty of 2^{nd} order cooperation (Heckathorn 1998).

This 2nd order problem is of no concern once the society has reached a population of all cooperators, as punishers no longer reduce their fitness to punish. But as shown in Figure 3.3 even those societies that evolve to full cooperation pass through evolutionary periods in which members of the society contribute less than a fully cooperative amount. Yet the fact that populations with punishers do achieve full cooperation despite passing through periods of lower cooperation, indicates that social structure alone may create the feedbacks necessary to overcome the 2nd order (and higher) free-rider dilemma. This concurs with Hodgson (2009, p. 145) who states that to understand 2nd order institutions "explanations must ultimately devolve on individuals and their interactions." Full cooperation evolves despite the prospect that some cooperators may not contribute to the punishment of non-cooperators. Like Panchanathan and Boyd (2004) the 2nd order free-rider problem appears to have been solved without intervention.

Therefore, it remains to answer the question, what effect does 2nd-order punishment have in these simulations? It may be that the strength of punishment required to achieve cooperation under different social structures is affected by whether or not 2nd order free riders are subject to punishment. On one hand it is intuitive to predict that, since punishment of 1st order cheaters led to cooperative behavior, further punishment of 2nd order cheaters may lead to cooperation at even lower values of the punishment multiplier *M*. However, laboratory experiments with human subjects have demonstrated the opposite and suggest that 2nd order punishment can inhibit the emergence of cooperation (Denant-Boemont et al. 2007)

To test the effects of 2nd order punishment simulations were conducted of populations playing the CPD described in Chapter 3. The simulation was modified so that when a game is played between *i* and *j* and observer *k*, a new agent *l* simultaneously evaluates *k*'s punishment behavior (Table 4.1). To assess *k*'s general predisposition to punish, *l* compares its punishment threshold t_l to k's threshold t_k. If *k* is generally more lenient on low offers compared to *l* ($t_k < t_l$), *l* punishes *k*. In simpler terms, the newly introduced agent *l* is ensuring that the punisher is doing its job.

As in Chapter 3 a sweep of the parameter *M* was conducted on several networks to determine the effect of this additional 2nd order enforcement.

Results and discussion: 2nd order free rider simulations

Figures 4.3 and 4.4 present comparisons of simulations on several networks with and without punishment of 2nd order free riders. Contrary to expectations, the ability to punish 2nd order free-riders led to the requirement of higher *M* in order to induce cooperative behavior in a population. In other words, punishment needed to be more severe to achieve cooperation than in Chapter 3 when punishment of 2nd order free riders was not allowed (Table 4.2).

This is likely due to the fact that 2nd order punishment is not based on whether the punishment recipient was a cooperator or defector, but on whether the recipient was a punisher or not. In the presence of a 2nd order punishment institution, simply contributing to a public good is no longer sufficient to guarantee freedom from punishment. Many cooperative agents that would have otherwise helped moved a population to full cooperation in Chapter 3 may have been injured through sanctions in the present experiment, and would therefore decrease the overall effectiveness of punishment.

The Effect of Retaliatory Behavior

Another often unacknowledged drawback to punishment is the phenomenon of retaliation. Studies have shown that humans and other animals do not take kindly to being punished and often retaliate at a cost both to themselves and their punisher (Molm 1989a, b, Clutton-Brock and Parker 1995, Saijo and Nakamura 1995, Hopfensitz and Reuben 2005). This can inhibit the punishment of free-riding and ultimately negate the cooperative effects of punishment (Nikiforakis 2008). However, previous research on punishment has rarely considered the potential consequences of retaliation (Fon and Parisi 2005, Denant-Boemont et al. 2007).

In the previous chapter, simulation experiments revealed outcomes that may be achieved under a variety of social structures when altruistic punishment is allowed. However, the ability to punish was limited to a single act by a 3rd party. In the current experiment the CPD simulation used in Chapter 3 is modified to allow a punished agent to retaliate against its punisher.

To examine the effects of retaliation on cooperative outcomes, agent behavior was modified so that agents automatically retaliate after being punished by paying an amount $s \in [0, 1]$ to have its punisher sanctioned by an amount sM. Because the amount of retaliation s may be 0, agents may evolve so that they effectively do not retaliate, even when punished. Three different rules were implemented for calculating how much a punished agent should spend on retaliation:

(1) s equals the same amount the punished agent would have spent to punish a low contributor ($s = c$). This assumes that a single strategy component

dictates how much an agent will spend to punish another regardless of the reason for punishing.

(2) *s* is a new, independently evolving strategy component (*s* is independent). In this case acts of retaliation are assumed to be independent of other punishment acts by an agent.

(3) *s* equals the amount the agent contributes to the public good in the CPD ($s = x$). This reflects the idea that both punishment and retaliation are non-self interested behaviors, and so may be governed by the same strategy component.

Results and discussion of retaliation experiments

Using retaliation rule 1 ($s = c$) cooperation did not evolve on any network. The ability to retaliate led to the collapse of cooperation that evolved when there was no retaliation. Likewise with retaliation rule 2 (*s* is independent), full defection evolved on all social structures.

However, in simulations using retaliation rule 3 ($s = x$), results were more complex. As with simple punishment, simulations on networks other than the complete network underwent a rapid transition from low to high contributions with increasing *M*. However, contributions did not transition to full cooperation as before (Table 4.3) but instead plateaued at a value between full cooperation and full defection depending on the network (Figure 4.5). In addition, payoffs initially increased with increasing *M* but eventually evolved to levels even below the Nash equilibrium payoffs of the base CPD, in which there is no punishment or retaliation (Figure 4.6).

These results present a challenge to the explanation of cooperation based on punishment because humans often do retaliate after being punished (Hopfensitz and Reuben 2005, Nikiforakis 2008). However, results, at least under retaliation type 3, also do not result in full defection. While the presence of retaliatory behavior may present a barrier to full cooperation it does not preclude some intermediate level of contributions to a public good provided there is a sufficient strength of punishment.

Despite an intermediate level of public good contributions, results are ambiguous regarding cooperation. Though free-riding was partially deterred and contributions to the public good evolve to some positive level, social welfare eventually evolved to levels lower than the worst possible outcome in the absence of punishment and retaliation. In other words, populations with the option to retaliate fared worse than populations with no punishment at all, even though those with retaliation had a partially provisioned public good and those without punishment had none.

It is precisely this type of outcome that should lead policy makers to scrutinize punishment mechanism before they are incorporated into policies designed to foster cooperation. Their efforts may only result in the illusion of cooperation through increased compliance but at the cost of decreased social welfare. Perhaps this helps to explain the existence of institutional policies such as that of the United States Department of Labor, which implements methods for discouraging retaliation (USDL 2009).

Summary and Future Directions

This chapter demonstrates the potential danger of generalizing about the benefits of using punishment to induce cooperation. On the other hand it points to numerous potential questions that may be foci of future studies. First, while this study has thus far used a form of the prisoners dilemma, future studies should duplicate these type of simulations on a wide variety of 2-player games to ascertain a more general nature of punishment in social dilemmas and to inform policy makers of potential adverse affects of institutionalized punishment.

Second, as stated above, in experiments with punishment collected fees and fines are routinely removed from the experimental system without further consideration. It is more likely in real world situations that collected penalties and fines are redistributed, to some degree, to the society from which they are collected – either to those who did not defect or to all society members. Future simulations should include and explore a variable that allows redistribution of collected sanctions and fees.

In addition, simulations results presented here regarding 2^{nd}-order punishment should be coupled with laboratory experiments to validate the detrimental nature of such supplemental punishment.

Finally, a rich suite of questions is posed regarding retaliation. Three methods for determining how to retaliate are presented in this study. Others surely await discovery and testing.

Table 4.1

Payoffs *p* in the continuous prisoners dilemma between observer *k* and 2nd order punisher *l*

	$t_l < t_k$	$t_l \geq t_k$
p_k	$-c_l M$	0
p_l	$-c_l$	0

Note: see Tables 3.1 and 3.3 for description of variables

Table 4.2

Approximate value of *M* at which populations transitioned from low to high contributions in the continuous prisoners dilemma, with and without punishment of 2nd order free-riders

Network type (No. of neighbors)	Approx. transition value of *M* [a]	
	with no 2nd order punishment	with 2nd order punishment
Linear (2)	1.5	1.6
von Neumann (4)	1.8	2.8
Hexagonal (6)	2.2	4.1
Moore (8)	2.8	5.7
Complete ($N - 1$)	N/A[b]	N/A[b]

[a] See Appendix A for method of approximating transition values.

[b] A transition did not occur on the complete graph with increasing *M* under either treatment. This was true even at values as high as $M = 5{,}000$.

Table 4.3

Response of contribution rate to network type with and without retaliation in the continuous prisoners dilemma

	Mean contribution (std. dev.)	
Network type	Without Retaliation[a]	With Retaliation (type 3)[b]
Complete graph	0.030 (0.010)	0.036 (0.050)
Regular graphs		
Moore	0.990 (0.017)	0.065 (0.018)
Hexagonal	0.997 (0.002)	0.115 (0.035)
von Neumann	0.998 (0.001)	0.277 (0.064)
Linear	0.996 (0.002)	0.949 (0.015)
Complex, real-world graphs		
Small-world	0.997 (0.001)	0.525 (0.066)
Scale-free	0.666 (0.232)	0.644 (0.231)
Other graphs		
Random	0.455 (0.284)	0.126 (0.035)
Random regular	0.999 (0.001)	0.129 (0.040)

Note: in both treatments, punishment of low contributors is allowed

[a] Mean contribution over 100 runs at $M = 4$ (see table 3.4 above).

[b] Mean contribution over 100 runs at each $M = 10, 15, 20, 25, 30$ ($N = 500$).

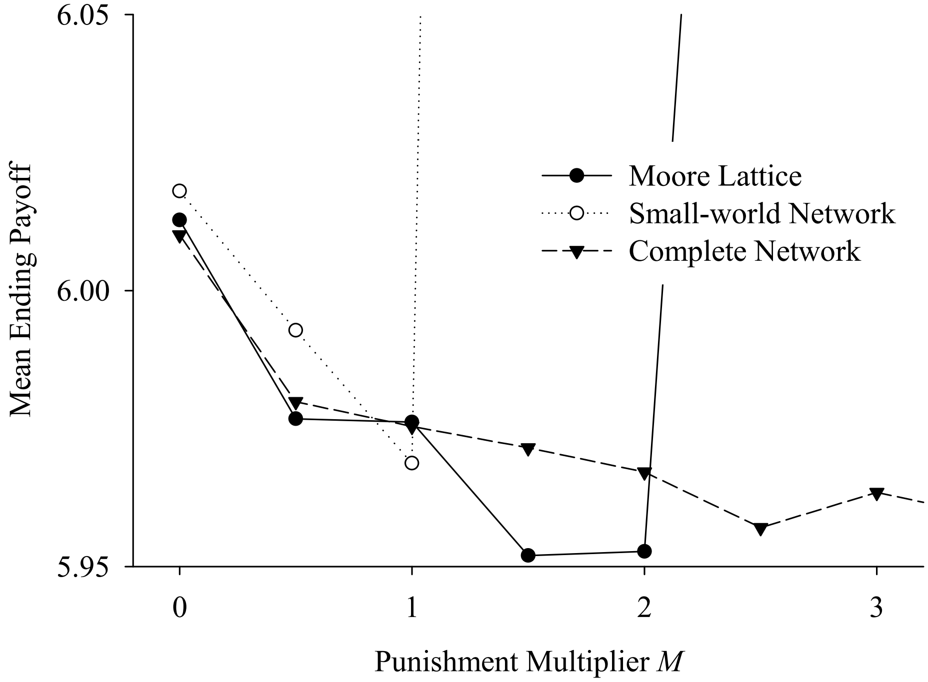


Figure 4.1. Prisoners dilemma payoffs vs. *M* at low values of *M*. The rational expectation for selfish individuals is that mean payoff = 6.0. As punishment is introduced, however, under both regular and small-world networks, payoffs fall below expectations. Payoffs continue to fall until the value of *M* reaches a threshold point (Table 3.5) at which payoffs jump to near the full cooperative value. Under a complete network, payoffs are always lower under a punishment regime than without.

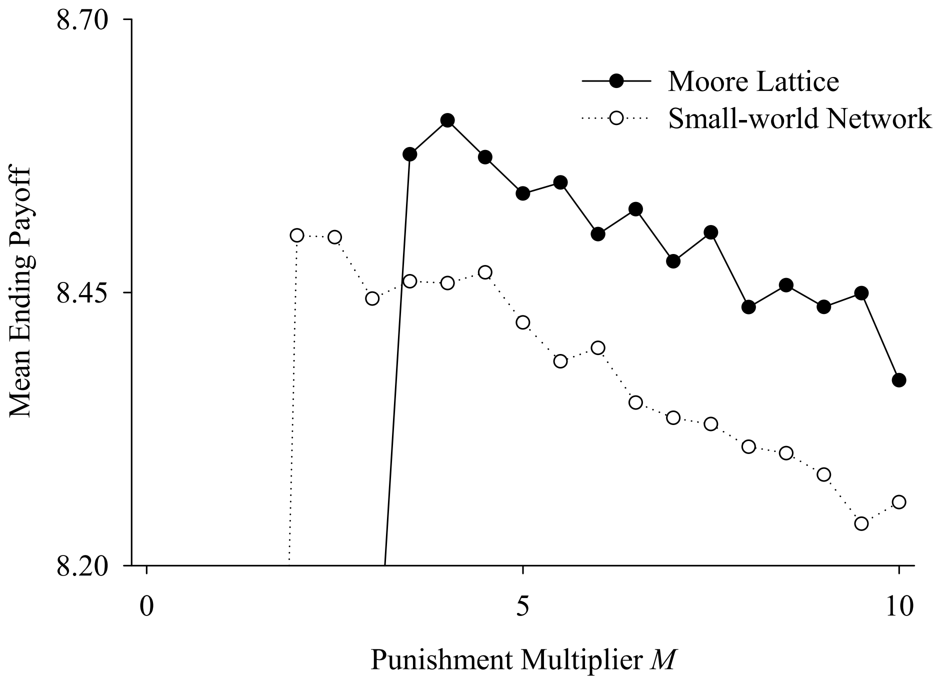


Figure 4.2. Prisoners dilemma payoffs vs. *M* at high values of *M*. The rational expectation for a population of fully cooperating individuals is that mean payoff = 9.0. However, having made the transition to cooperative contributions with increasing *M* (Table 3.5), payoffs steadily decline with increasingly potent punishment. This same trend is observed with increasing *M* before the transition to cooperative contributions (Figure 4.2) and suggests that crafting an optimal punishment institution may be difficult.

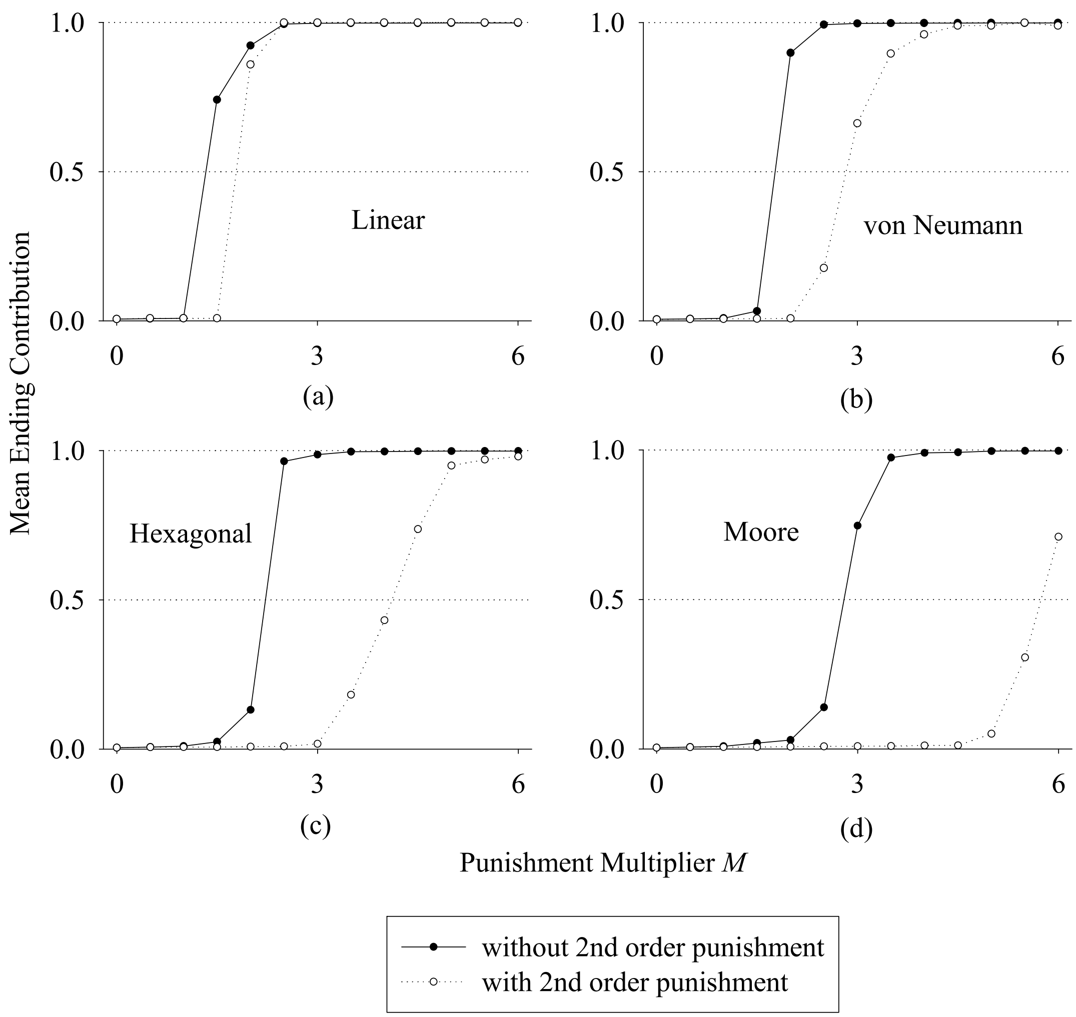


Figure 4.3. Effect of 2^{nd}-order punishment: lattice networks. Introducing punishment of 2^{nd}-order free-riders leads to less cooperative contributions at any given value of *M*. This effect becomes more pronounced as network density increases.

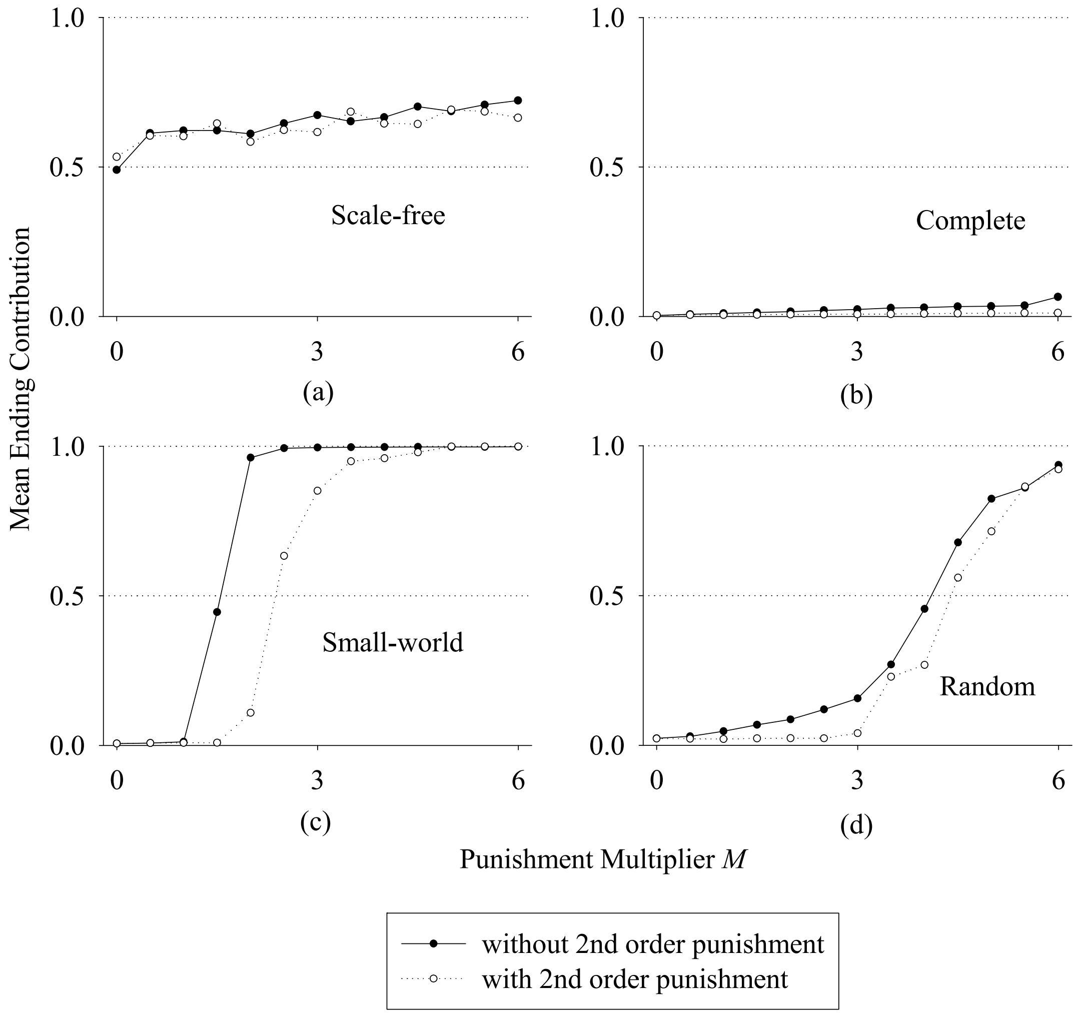


Figure 4.4. Effect of 2nd-order punishment: other networks. In cases other than scale-free networks, allowing punishment of 2nd-order free-riders leads to less cooperative contributions at any given value of *M*.

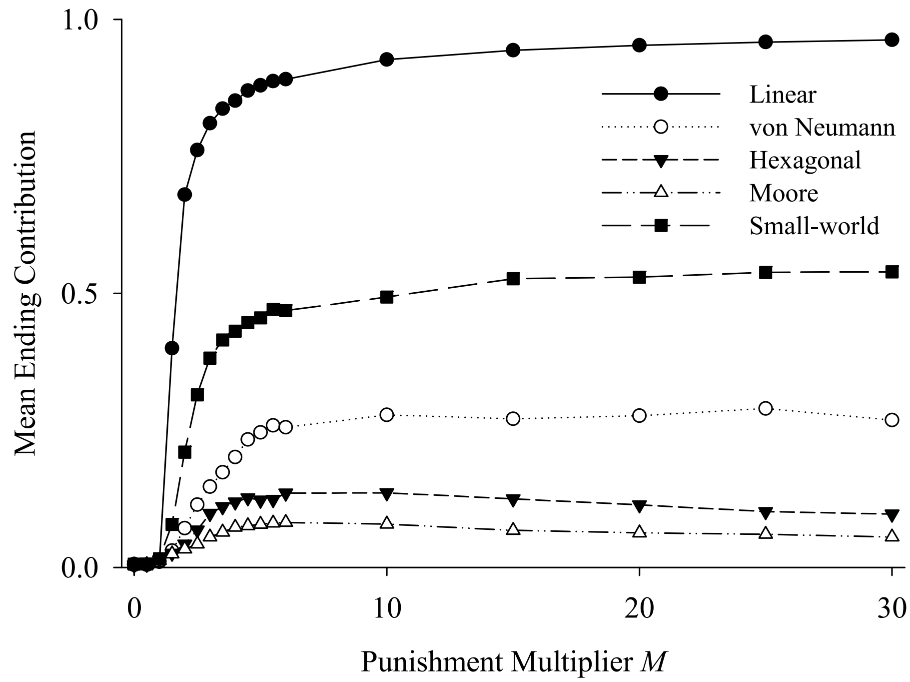


Figure 4.5. Effect of retaliation on mean ending contributions in the continuous prisoners dilemma. In populations given the option to retaliate, neither full cooperation nor full defection evolved in the above networks. These data come from simulations using retaliation type 3, in which the amount an agent spends on retaliation *s* is the same amount the agent contributes to the public good *x*. Using types 1 and 2 retaliation cooperation collapses completely and full defection evolves on all networks.

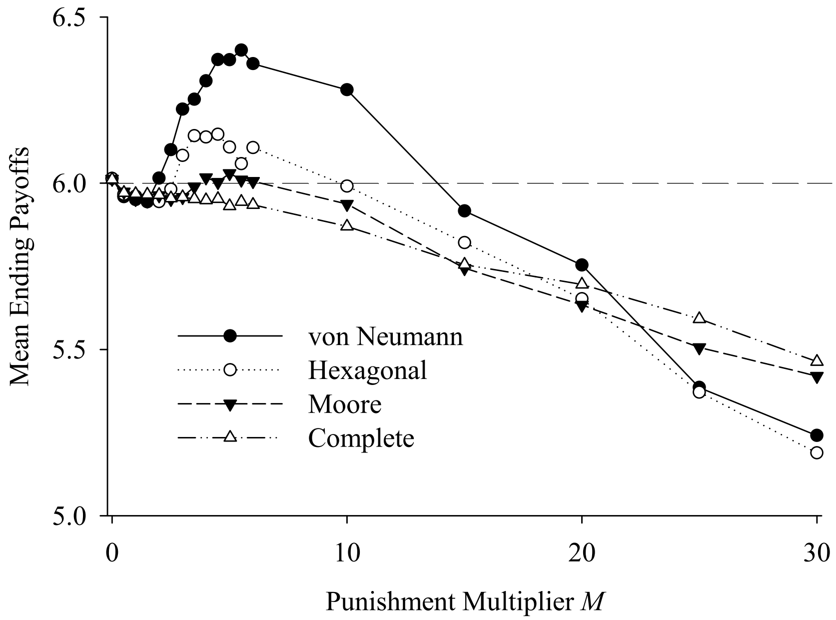


Figure 4.6. Effect of retaliation on mean payoffs in the continuous prisoners dilemma. In populations given the option to retaliate, mean payoffs eventually evolved, with increasing *M*, to levels lower than the least possible payoffs without punishment and retaliation (dashed line at $p = 6.0$). These data come from simulations using retaliation type 3, in which the amount an agent spends on retaliation *s* is the same amount the agent contributes to the public good *x*.

CHAPTER 5

PUNISHMENT AND SOCIAL STRUCTURE:

THE EVOLUTION OF FAIRNESS IN AN ULTIMATUM GAME[9]

Introduction

Kazemi and Eek (2008) assert there are two predecessors to cooperative outcomes of social dilemmas, the provisioning of public goods and the allocation of public goods. Whereas questions of provisioning are often associated with cooperation, questions of allocation are concerned with fairness. Though research on social dilemmas has been dominated by cooperation and provisioning questions (Kazemi and Eek 2008), to better facilitate the resolution of social dilemmas, research should encompass both antecedents of cooperative outcomes. Accordingly chapters 3 and 4 of this dissertation investigate cooperation through experiments with public goods provisioning, demonstrating that social structure, coupled with punishment, has an important influence on cooperative outcomes. In this chapter I present experimental results on the allocation step and discuss whether punishment and social structure similarly affect the ability of a population to evolve fairness behavior.

The question of fairness

Despite a voluminous literature addressing fair allocations, authors rarely attempt to define the term fairness. This is partly because concepts of fairness are often culturally contextual norms and may vary among individual groups (Kazemi and Eek 2008). However, a general definition is warranted to adequately discuss abstract questions of

[9] This chapter is a modified version of Shutters (2008).

fairness. Here I adopt the definition proposed by Varian (1974) in which a fair allocation is one that is both pareto efficient and equitable. Being pareto efficient an allocation cannot be altered without decreasing the payoff of at least one participant. Being equitable, says Varian, means no participant prefers the allocation of another participant. This definition may explain why many authors, while declining to explicitly define fairness, nevertheless typically imply that a fair allocation is one resulting in approximately equal shares to participating parties (Nowak et al. 2000, Henrich et al. 2001).

In the 2-player ultimatum game used in this chapter and described below, every allocation resulting from an accepted offer is pareto efficient. In other words, disregarding cases where an offer is rejected, the ultimatum game is a zero-sum game – no player can increase his payoff without decreasing the payoff of his opponent. This satisfies the first criterion for fairness. Regarding those allocations that are also equitable, all simulated agents used in this study begin each generation with the same resource endowment and compete in the same reproduction algorithm for the ability to pass offspring into the next generation. Therefore, an agent will benefit from the higher share of an allocation and, in this sense, will prefer an opponent's allocation if it is larger. The only point at which neither agent would prefer the other's allocation is when the allocation is an equal split so that in the following simulated ultimatum game, a fair allocation is one in which each player receives a 50% share.

Background

Since cooperation often means overcoming an incentive to cheat or free-ride, the emergence of cooperation among unrelated individuals remains largely unexplained in the life and social sciences. A long history of explanations includes kin selection (Hamilton 1964, Rothstein and Pierotti 1988, Wilson 2005), direct and indirect reciprocity (Trivers 1971, Nowak and Sigmund 1998a, Riolo et al. 2001, Killingback and Doebeli 2002, Nowak and Sigmund 2005), and multi-level selection (Wilson and Sober 1994, Goodnight 2005, Reeve and Hölldobler 2007). However, these explanations often require assumptions such as close genetic relationships, small populations, or repeated interactions in order for cooperation to evolve (Fowler 2005). Recent findings suggest that strong reciprocity – the altruistic punishing of cheaters and altruistic rewarding of cooperators – may provide an alternative and more general explanation. In particular altruistic punishment by third-party observers has been shown to play a positive role in maintaining cooperation (Fehr and Gächter 2000, Gintis 2000, Henrich and Boyd 2001, Fehr et al. 2002, Fehr and Gächter 2002, Boyd et al. 2003, Bowles and Gintis 2004, Jaffe 2004, Shinada et al. 2004, Fowler 2005).

The effectiveness of punishment as a mechanism for cooperation has long been debated, with some suggesting that it may simply lead to a destructive cycle of costly retaliation (Molm 1994). However, researchers now seem sufficiently sure of punishment's ability to induce cooperation that they have moved to advocating its use by policy makers, both at local scales, in institutions governing common pool resources (Ostrom et al. 1992, Ostrom et al. 1994, Dietz et al. 2003, Anderies et al. 2004), and at international scales, in agreements designed to provision global public goods (Sandler 1992, Wagner 2001, Barrett 2003b, a, 2005).

The punishment multiplier

An important parameter governing the mechanism of altruistic punishment is the ratio of costs incurred by the punishing party to those of the party being punished. Letting c = the cost that an individual incurs to punish another, cM = the fee or sanction imposed on the punished actor where M is the punishment multiplier. As M becomes arbitrarily large there should be some point at which it is no longer altruistic to provide punishment but is instead strategically beneficial. M, therefore, becomes an important parameter in understanding outcomes of punishment experiments. However, though any experiment that uses a punishment mechanism implies a value of M under which the experiment operates, explanations of how researchers set this parameter are largely absent. To my knowledge, even those studies in which researchers explicitly state their value of M, the authors rarely offer an explanation of the choice or demonstrate the effects of altering the parameter (e.g. Fehr and Gächter 2000, 2002, Andreoni et al. 2003, Boyd et al. 2003, Brandt et al. 2003, Gürerk et al. 2006).

For example, in their influential paper on altruistic punishment, Fehr and Gächter (2000) demonstrate that humans in anonymous, one-shot interactions will punish low contributors in a public-goods game. In their experiment $M = 3.0$ yet the authors offer no explanation for this choice. Likewise, Andreoni et al. (2003) set $M = 5.0$ in their experimental ultimatum games using punishment and rewards, but state only that they chose their ratio so as to ensure that punishment would take place.

Table 5.1 lists a number of recent studies on altruistic punishment and the values of *M* used in each study. Two of these studies, Fehr and Gächter (2000) and Masclet et al (2003), did not use a fixed value of *M* but instead used a non-linear function of the amount paid by the punisher to determine the amount deducted from the punishee. This leads to further confounding issues which are addressed in detail by Casari (2005).

Despite the fact that researchers often neglect discussion of their selections of *M*, policy-makers who choose to implement punishment mechanisms must include some definition of costs incurred by punisher and punishee. Even if polices are unable to directly set a common value of *M* for a crafted punishment institution, they may still be able to influence its value. Before researchers promote the application of such mechanisms to social dilemmas, a better understanding is warranted of how cooperative outcomes respond to punishment mechanisms under varying values of *M*. This is especially true since researchers have shown that if punishment is excessive, it can lead to worse outcomes, in terms of total payoffs, than if no punishment were present (Fehr and Rockenbach 2003).

To test the ability of altruistic punishment to elicit fair allocations simulations of the ultimatum game were conducted on a variety of network structures while systematically varying the parameter *M*. Perhaps for its sheer simplicity, the ultimatum game has grown in popularity until it has come to rival the prisoners dilemma as the preferred game-theoretical framework for studying cooperative phenomena (Nowak et al. 2000). The game is played by two agents *i* and *j* that must decide how to split an endowment. The proposer *i* initiates the game by offering a percentage of the endowment to the responder *j*. *j* then either accepts the division, in which case each agent collects its agreed upon share, or *j* rejects the division, in which case both agents receive 0. In either event the game ends. Economic theory predicts that, given rational agents, *j* will accept the smallest positive amount possible and that *i*, knowing this, will therefore offer the smallest amount possible. Tests of this economic expectation among non-humans have been inconclusive, with evidence both rejecting (Silk et al. 2005, Jensen et al. 2007) and supporting (Brosnan and de Waal 2003, Burkart et al. 2007) the existence of fairness behavior among unrelated primates. However, fairness behavior among humans is well-established and subjects across many cultures have shown a strong propensity to offer fair allocations (i.e. ~ 40-50% of the endowment) and reject unfair offers in experimental ultimatum games (Roth et al. 1991, Nowak et al. 2000, Henrich et al. 2001).

The Simulation Model

In simplest terms this model simulates a population of agents in a torroidal space playing the ultimatum game against one another. Under various parameter settings, agents are endowed with the ability to altruistically punish a neighbor after assessing the neighbor's game-play behavior. This punishment is accomplished through introduction of a third party to the game - the observer *k*. When in the role of *k* an agent observes a game being played by two other agents and may reduce its own fitness in order to punish what it perceives to be a low offer.

Each agent consists of a strategy (x, α, t, c) where x = the amount that an agent in the role of *i* will offer; α = the offer threshold above which an agent acting as *j* will accept an offer; t = the offer below which an agent acting as *k* will punish *i* in a game under observation; and c = the cost that an agent acting as *k* is willing to incur to punish *i* for offering too little (Table 5.2). Each of the four strategy components holds a value on the continuous interval [0,1] and is generated randomly from a uniform distribution at the beginning of each simulation. To control for other factors that might contribute to the maintenance of cooperation, such as interaction history or reputation, the model does not allow recognition of or memory of other agents within the population (though repeated interactions are possible since interactions are restricted to a small local neighborhood).

Following initialization a simulation proceeds through a number of generations, each of which consisted of three routines – game play, observation & punishment, and selection & reproduction. In each routine an agent interacts only with its immediate neighbors as defined by the network type (Table 5.3) and all interactions take place in parallel. In addition to its strategy, each agent is described by a fitness variable p, which is simply an accumulation of the costs and payoffs an agent experiences during a generation. Because only relative fitness is considered during selection and because this model uses the elementary case in which parents do not differentially provision resources for their offspring, $p = 0$ for each agent at the beginning of a new generation.

The present model most closely resembles the model of Page et al. (2000) who also demonstrated that a simulated population playing the ultimatum game could evolve fair allocations under a linear population structure (or what the authors refer to as a one-dimensional spatial ultimatum game). The authors did briefly discuss implications of a von Neumann neighborhood but the focus of the work was on the effect of varying the population size and the radius of an agent's neighborhood.

Another closely related model by Killingback and Struder (2001) produced results not in agreement with those of the current simulation. The authors simulated a modified ultimatum game in which fair allocations evolved when the population was structured on a hexagonal lattice. However, in an effort to model a "collaborator's dilemma," their modifications to the ultimatum game were extensive enough that it is unreasonable to expect outcomes similar to those from the standard ultimatum game.

During the game play routine, each agent *i* plays the role of proposer and randomly selects, with replacement, three responders from its neighborhood with which to play a game. After receiving a standardized endowment of 1 per game, *i* initiates each game by making its offer of x_i to *j* who then evaluates the offer. If the offer is above *j*'s acceptance threshold the offer is accepted and p_j increases by x_i while p_i increases by $1 - x_i$. If the offer is below the threshold it is rejected and both p_i and p_j remained unchanged.

After making offers to three neighbors, *i* selects three neighbors *k* to observe each of those games. These observers, chosen from the same neighborhood as responders, are selected with replacement and evaluate x_i in the observed game. Provided that the offer is not below *k*'s punishment threshold, p_i and p_k do not change. However, if the offer falls below *k*'s punishment threshold *k* punishes *i*. In so doing p_k is reduced by c_k while p_i is reduced by $c_k M$, where *M* is the punishment multiplier described above. Punishment of *i* is independent of whether x_i is actually accepted by *j* in the observed game. Payoffs for the ultimatum game are listed in Table 5.4.

Finally, each generation ends with a selection & reproduction routine during which each agent *i* randomly selects a neighbor *j* with which to compare payoffs. If $p_i > p_j$ the strategy occupying *i*'s node remains and passes to the next generation. If p_j is greater, *j*'s strategy is copied to *i*'s node. In the event that $p_i = p_j$ a coin toss determines which strategy occupies *i*'s node in the next generation. Once agents of the next generation are determined, each strategy component of every agent is subjected independently to mutation with a probability of $m = 0.1$. If selected for mutation a number randomly drawn from a Gaussian distribution with mean = 0 and standard deviation = 0.01 is added to the mutated trait. In the event that mutation causes the value of a trait to fall outside the interval [0,1] the trait is set to the closer endpoint (either 0 or 1). A single run continues in this manner for 30,000 generations and, as the model is not deterministic, is replicated 100 times to complete a single simulation.

For each network type, simulations were run starting with $M = 0$ and thereafter at increments of 0.5 until $M = 6.0$, by which point simulations that converged to a population-wide offer value had all done so. Model parameters are summarized in Table 5.5. The dependent variables of interest are the mean offer and the mean payoff which evolve in a population after 30,000 generations. The mean offer represents the population's level of fairness while the mean payoff represents the population's social welfare.

Results and Discussion

Results revealed three major trends worthy of discussion: (1) even without punishment, a negative correlation exists between the number of neighbors per agent and the mean offer rate to which a population evolves, (2) in simulations with some social structure an abrupt transition occurs from relatively low mean offers to offers of nearly 100% as M increases, and (3) a correlation exists between the number of neighbors per agent and the value of M at which the transition from low to high offers occurs.

Offer rates in the absence of punishment

It has been well demonstrated that spatial explicitness in computer simulations can lead to outcomes significantly different than when populations are unstructured or well-mixed (Nowak and May 1992, Nowak et al. 1994, Killingback and Doebeli 1996, Killingback and Studer 2001). However, there are a number of ways in which a population can be spatially explicit and it is important for researchers to show how different social structures can lead to different results. In this experiment care was taken to show not only the effect of introducing punishment but also to show how that effect differed among different spatial structures of the agent population. These spatial structures included a complete network in which every agent is a neighbor to every other agent in the population (see for example Riolo et al. 2001).

Initial simulations were conducted without any ability to punish by third-party observers. Agents simply played a standard ultimatum game under a number of different neighborhood structures (Table 5.6). Under these conditions the offer rate to which a population evolved was correlated with the neighborhood structure of that population. In particular, ending mean offers increased as the number of neighbors per agent decreased (Figure 5.1). Only under a linear population structure did offers approach fair allocations. This result is in agreement with Page et al. (2000) who found that agents playing the ultimatum game in a linearly structured population evolved approximately fair allocations.

However, though mean offers fell in response to increasing numbers of neighbors, offers remained significantly greater than the economic expectation of ~ 0. Only when populations lacked social structure (complete network), and agents could interact with any other agent in the population with equal probability, did ending offers approximate the Nash equilibrium. In fact, when additional simulation were run on a complete network with *M* as high as 5,000, offers did not deviate from the Nash equilibrium.

This result indicates that the structure of local interactions may matter more for cooperation than the overall population size and may be as important, if not more so, than punishment. More importantly it suggests that cooperative outcomes are possible even in a large anonymous population provided that local small-size clustering is allowed. This conclusion concurs with similar research highlighting the importance of small group or neighborhood size (Olson 1965, Page et al. 2000, Ifti et al. 2004).

Response of offer rates as M *increases*

By increasing *M* above 0 agents became endowed with the ability to punish one another. Following this introduction of punishment, agents initially evolved offer rates equal to those in the absence of punishment. In other words at relatively low values of *M*, punishment had no discernable effect on simulation outcomes. However, as *M* continued to increase populations on each network type, other than a complete network, eventually encountered a threshold value of *M* at which a rapid transition occurred in offer rates. In these transitions agents went from offering relatively low percentages to offering approximately 99% of their endowments in an attempt to be reproductively successful (Figure 5.2). Other than in the case of a linearly structured population discussed above, neither offers before nor after these transitions fell into a range that could be considered fair allocations. Instead those offers before the transition were below fair offers and those after were well above fair offers.

Despite several studies in which strong reciprocity is shown to induce cooperation, third-party punishment failed to lead to fair allocations in these simulations (Figure 5.2). This outcome is contrary to the experimental results of Fehr and Fischbacher (2003) in which anonymous human observers routinely paid to punish those who offered below 50% in a laboratory ultimatum game. A likely explanation for this difference is that cultural factors, which were explicitly excluded in the present simulation model, played a significant role in outcome of the Fehr and Fischbacher experiment. This concurs with conclusions drawn from ultimatum games played between chimpanzees that considerations of fairness are limited to humans (Jensen et al. 2007).

An important aspect of these transitions in mean ending offers is that it appears to flip between only two basins of attraction – those offers which evolve when $M = 0$ or relatively low and those offers of ~ 99% when M is relatively high. A possible explanation of this rapid transition is through consideration of agents' relative fitness. Because reproductive success is explicitly a function of payoffs in this model, a consideration of relative fitness means a consideration of relative payoffs. For a given agent i let p_i equal the agent's absolute payoff and let $\bar{p}$ equal the mean payoff of i's n neighbors. $n\bar{p}$ is then equal to the sum payoffs of agent i's neighbors. i's relative payoff before punishing another might be represented as

[5.1] $$\frac{p_i}{n\bar{p}}.$$

Using previous definitions of the punishment multiplier M and the cost of punishment c, the agent's relative payoff after a single instance of punishment can then be described as

[5.2] $$\frac{p_i - c}{n\bar{p} - cM}.$$

In order for punishment to be evolutionarily beneficial, it is expected that *i*'s relative payoff will be greater after an act of punishment so that

[5.3] $$\frac{p_i - c}{n\bar{p} - cM} > \frac{p_i}{n\bar{p}}$$

which simplifies to

[5.4] $$\frac{p_i}{n\bar{p}} > \frac{1}{M}.$$

From [5.4] it is clear that as M increases there should be some threshold value at which punishment switches from being a detrimental strategy to one that is beneficial. Once this condition is met an observer actually benefits, in terms of relative payoff, from punishing its neighbor and there is no reason mathematically to limit the amount of punishment. Therefore, as M increases the observed transitions in offer rates is not unexpected.

In general this result suggests that the ratio of costs between punisher and punishee can dramatically affect experimental outcomes and that researchers should be conscientious about the selection of M in both experiments and simulations. Because of cost constraints it is perhaps expected that laboratory experiments with punishment would not undertake a broader exploration of the parameter M. However, this is less of a concern with computer models and future studies that make use of simulations could easily include a sensitivity analysis of punisher/punishee cost ratios.

Furthermore, it is unlikely that such a punishment mechanism as that implemented in this simulation would be well-received as an institutional solution to promote cooperation among humans. Instead, this result supports the notion that punishment – taken too far – can be counterproductive (see Molm 1994).

Network type and the transition value of M

A further result of this experiment is that the approximate value of M at which a rapid transition in offers occurred was dependant on the neighborhood type. Transitions occurred at higher values of M as the number of neighbors per agent, or average degree, increased. Approximate values of these transitions are presented in Table 5.7.

A possible explanation for this trend is provided by extending the relative fitness model described in equation [5.4] to demonstrate not only that a transition is expected, but also at what value of M such a transition might occur. Given that the expected payoff of any agent drawn randomly from the population = $E(p)$, the expected relative payoff [5.1] for any agent with n neighbors can be simplified as

[5.5] $$\frac{E(p)}{nE(p)} = \frac{1}{n}.$$

Substituting the expected relative payoff of any agent [5.5] into equation [5.4] reveals that punishment is expected to become an evolutionarily beneficial strategy when

[5.6] $$M > n,$$

and that as punishment becomes a preferred strategy it will drive a rapid transition from low offers to high offers.

A cursory inspection of the approximate transition points from low offers to high offers under different neighborhood types (Table 5.7) indicates that equation [5.6] likely presents a highly oversimplified model. It is more likely that the term $n\overline{p}$, the part of relative payoff describing exactly to what an agent's payoff is relative, is much more complex than a simple sum of immediate neighbors' payoffs. A more general representation of the relative payoff of agent *i* would be

[5.7] $$\frac{p_i}{\sum \theta_j p_j};(i \neq j)$$

where θ is a weight indicating the degree to which the payoff of every other agent in the population affects agent *i*. For immediate neighbors this weight may be relatively high and for distant members in the population it may be 0. However, a benefit of this more general formulation of relative payoff is that it considers complex interactions such as the possibility that distant members of the population may also influence an agent or the possibility that immediate neighbors, despite their proximity, have negligible impact.

Future Directions

The different neighborhood structures used in this experiment (Table 5.3) were chosen because they have been used frequently in similar simulations for many decades. However, it is often overlooked that these symmetric and convenient neighborhood types are but a tiny subset of the vast number of possible ways that agents may be connected in a population. It is therefore prudent to embed research such as this in social network theory, with its many tools for analyzing a vast number of possible structures. This paper has merely explored possible relationships between the number of neighbors per agent and mean ending offers. However it is probable that a better explanation for observed correlations is much more complex and that results are driven by subtler elements of the network structure.

A prime candidate for extending this work into social network theory is a better determination of what constitutes P in the equation for relative fitness [5.1], or alternatively how each agent in a population is weighted in [5.7]. With a more accurate description of what exactly it is that an agent's fitness is relative to, the current model can better explain and predict the effects of altruistic punishment on fair allocations and cooperation.

This study also suggests questions for experimentalists. Though costs may be an inhibitive factor, it would be worthwhile to study the effects of systematically varying the punishment multiplier M in a controlled laboratory setting. A replication of this simulation with live subjects may isolate cultural or other factors that allow third-party punishment to induce fair allocations and may better guide policy makers in designing institutional solutions to social dilemmas.

In addition, laboratory ultimatum games may be designed to further explore the role of neighborhood structure on offer rates. Like the current simulation, this may be done without including 3rd party punishment. Participants may remain anonymous with the experimenter controlling the array of possible interactions between participants. Again, this would compliment the current study by helping to separate the effect that culture has on offer rates from effects due strictly to the way in which the population is structured.

Table 5.1

Some commonly-cited experiments using altruistic punishment and their values of the punishment multiplier *M*

Laboratory experiments	*M*	Simulations and models	*M*
Dreber et al. (2008)	4.0	Brandt et al. (2006)	1.2
Herrmann et al. (2008)	3.0	Fowler (2005)	2.0, 3.0
Gürerk et al. (2006)	3.0	Gardner and West (2004)	30.0
Fehr and Fischbacher (2004)	3.0	Jaffe (2004)	1.0
Shinada et al. (2004)	3.0	Brandt et al. (2003)	1.5
Andreoni et al. (2003)	5.0	Boyd et al. (2003)	4.0
Fehr and Fischbacher (2003)	3.0		
Masclet et al. (2003)	function		
Fehr and Gächter (2002)	3.0		
Fehr and Gächter (2000)	function		
Ostrom et al. (1992)	2.0, 4.0		

Table 5.2

Strategy components used by agents in the ultimatum game

Component	Description
x	amount offered to a responder
α	amount below which an offer is rejected
t	amount below which to punish an observed offer
c	amount to spend on punishment

Note: $x, \alpha, t, c \in [0, 1]$.

Table 5.3

Network types used in the ultimatum game

Network type	Description of neighbors	Number of neighbors
Linear	left, right	2
von Neumann	left, right ,up, down	4
Hexagonal	up, down, diagonals	6
Moore	left, right, up, down, diagonals	8
Complete	every other agent	$N-1$

Table 5.4

Payoff matrix of the ultimatum game with 3rd party punishment

	$x_i \geq \alpha_j$		$x_i < \alpha_j$	
	$x_i \geq t_k$	$x_i < t_k$	$x_i \geq t_k$	$x_i < t_k$
Proposer *i*	$1 - x_i$	$1 - x_i - c_k M$	0	$- c_k M$
Responder *j*	x_i	0	0	0
Observer *k*	0	$- c_k$	0	$- c_k$

Note: see Table 5.3 for an explanation of variables used in the payoff functions.

Table 5.5

Simulation parameters used in the ultimatum game

Parameter	Values
The population size (*N*)	625[a]
The number of generations a single simulation run	30,000
The number of runs	100
The number of games initiated by each agent in one generation	3
The range for trait values (x, α, t, c)	[0,1]
The probability of trait mutation	0.1
The (mean, standard deviation) of Gaussian noise added to a mutated trait	(0, 0.01)
The network type	see Table 5.2
The punishment multiplier (*M*)	0.0 to 6.0[b]

[a] When using a hexagonal lattice, $N = 676$.

[b] In increments of 0.5.

Table 5.6

Response of offers to network type in the ultimatum game without 3rd party punishment

Network type	Number of neighbors n	Mean ending offer[a]	Std. dev.
Linear	2	0.395	0.008
von Neumann	4	0.292	0.018
Hexagonal	6	0.253	0.024
Moore	8	0.160	0.028
Complete	$N - 1$	0.008	0.001

[a] Though customary to present ultimatum game offers as percentages of the initial endowments, absolute offer rates are presented here to be consistent with results of the continuous prisoners dilemma presented in Chapters 3, 4, and 6.

Table 5.7

Approximate value of *M* at which populations transitioned from low to high offers in the ultimatum game.

Network type	Number of neighbors	Approx. transition value of *M* [a]
Linear	2	1.6
von Neumann	4	2.1
Hexagonal	6	2.2
Moore	8	3.2
Complete	$N - 1$	N/A[b]

[a] See Appendix A for method of approximating transition values.

[b] A transition never occurred on the complete graph with increasing *M*. This was true even at values as high as $M = 5{,}000$.

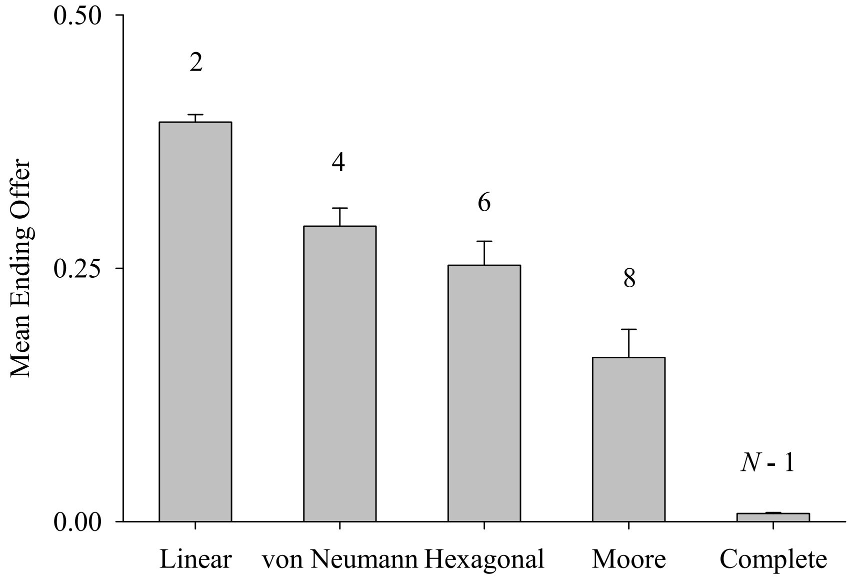


Figure 5.1. Response to regular networks of offers in an ultimatum game without 3rd party punishment. Bars represent +1 standard deviation. Numbers above bars indicate the number of neighbors each agent has in that network type.

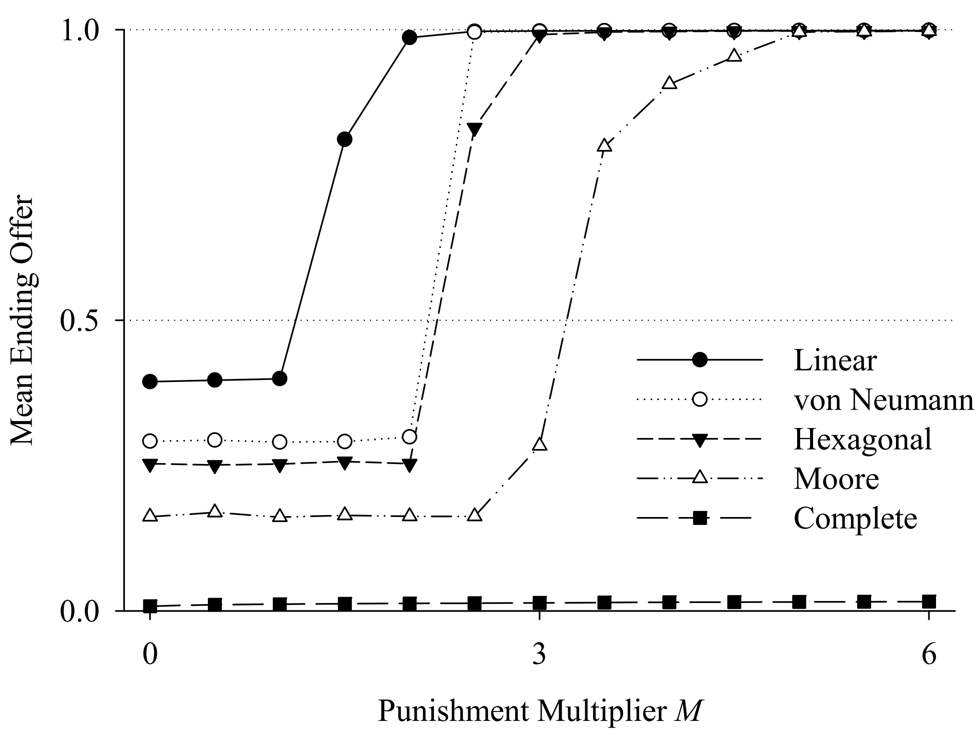


Figure 5.2. Response of ultimatum game offers to increasing *M* under regular network. All structures other that a complete network experience a rapid transition to offers of ~1.0 with increasing *M*. For networks exhibiting this transition from low to high offers, approximate transition values are listed in Table 5.7.

CHAPTER 6

PUNISHMENT, RATIONAL EXPECTATIONS, AND REGARD FOR OTHERS[10]

Previous chapters of this dissertation have synthesized original empirical data with a survey of relevant research by others. In this chapter I reanalyze those results, particularly from Chapter 3, to answer a separate set of questions and formulate a hypothesis to explain observed behavior, primarily in humans.

Introduction

Experimental economics has consistently revealed human behavior at odds with theoretically derived expectations of behavior by rational agents. This is especially true in laboratory games with costly punishment where humans routinely pay to punish others for selfish behavior even though the punisher receives no material benefit in return (Fehr and Gächter 2002). This phenomenon occurs even when interactions are anonymous and the punisher will never interact with the punishee again. However, costly punishment may not be inconsistent with Darwinian notions of relative fitness. This chapter presents an attempt to reconcile standard economic and evolutionary expectations of behavior and to evaluate the idea that when individuals make choices they do so with regard to behavior and status of others.

Agent-based modelling is used to simulate networked populations whose members play the prisoners dilemma while having the ability to altruistically punish one another. Results show that behavior evolving in structured populations does not conform to economic expectations of evolution driven by absolute utility maximization. Instead results better match behavior expected from a Darwinian perspective in which evolution

---

[10] This chapter is based on work presented in (Shutters 2009).

is driven by relative fitness maximization. Results further suggest subtle effects of network structure must be considered in theories addressing individual economic behavior.

Interdependent Preferences and Utility

Chapter 2 presented a review of the assumptions of standard rational choice theory, which stipulates that an individual's utility is a function of his consumption and only his consumption $u(c)$. Few scientists now believe in a strict interpretation of rational choice theory (Gould 1993) and a wide body of experimental evidence indicates that an individual's preferences change based on the consumption behavior of others (Pollak 1976, McAdams 1992, Oswald 1997, Solnick and Hemenway 1998, Easterlin 2001, Alpizar et al. 2005). This phenomenon, known as interdependent preferences (or other-regarding preferences), is addressed by theories asserting that an individual's utility function is based, at least in part, on the consumption behavior of others. These theories are especially concerned with explanations beyond the well-understood case of regard for kin, which is an extension of biological theories of kin selection.

The idea that individual choices are based in part on the behavior of others may seem obvious in our modern world of constant product promotion and may seem especially intuitive with regard to social creatures. But economists have been slow to incorporate the idea into theory. Though the concept of interdependent preferences was widely publicized by Veblen in 1899, it was for many decades largely ignored by mainstream economics. In 1949 Duesenberry complained that even though the phenomenon of interdependent preferences was empirically well-established, the field of economics had suffered a glaring failure by not incorporating the idea into new theories (Duesenberry 1949). This aversion, he believed, was due largely to analytical intractability of formal models attempting to incorporate interdependent preferences. Even simple two-person simulation models of interdependent preferences lead to chaotic behavior and inhibit prediction of aggregate demand (Rauscher 1992). And so still today, many economists take as their first approximation of rational behavior the assumption that an individual's utility is a function solely of the individual's absolute consumption (Hopkins and Kornienko 2004).

Duesenberry (1949) nevertheless attempted to formalize utility as a function of individual income (consumption potential) relative to income of others in society. Though there are virtually limitless ways in which preferences may be interlinked, Duesenberry's particular utility function may be represented in simple terms as

[6.1] $$u = u(c, c/\bar{c})$$

where $\bar{c}$ is the average consumption in a society and utility u is increasing in both c and $c/\bar{c}$ (Pollak 1976, Harbaugh 1996, Bastani 2007), though there is currently much debate regarding the relative importance of each argument (Alpizar et al. 2005). In any case, this formulation implies that any consumption by others in society will decrease the utility of individuals not partaking in the same consumption (Frank 2005b, Luttmer 2005).

Explanations of interdependent preferences

Despite difficulty formalizing interdependent preference models, several hypotheses exist to explain observations of the phenomenon. Following are some commonly cited explanations of interdependent preferences.

Conspicuous consumption – The concept of conspicuous consumption asserts that humans are concerned with their status in a social hierarchy and they consume certain goods to signal others in society that they are part of a specific social group (Knell 1999, Hopkins and Kornienko 2004, Johansson-Stenman and Martinsson 2006). Selecting which goods will send the intended signal depends on an assessment of the goods that others in the target status group are consuming. Thus, this explanation requires that a consumer's utility is a function of the visible consumption of others. Conspicuous consumption is normally limited to a class of goods known as "positional goods" (Knell 1999, Alpizar et al. 2005), which are those goods capable of signaling status.

The demonstration effect – In contrast to conspicuous consumption, the demonstration effect is not explicitly concerned with notions of status, nor does it require the consumption of a particular class of goods. Instead it argues that when a consumer is exposed to superior goods he is made aware that his own consumption choices are inferior (Duesenberry 1949, McCormick 1983). For instance, after viewing a program on a neighbor's high-definition television, a consumer may realize that his own analog television delivers an inferior signal. This realization that one is choosing inferior goods leads to a decrease in utility that can only be restored through a shift in consumption to the superior good. In this manner, consumption by others in society affects a person's utility without any necessary changes in the person's individual consumption.

Bounded rationality and heuristics – A similar idea to the demonstration effect is a byproduct of the concept of bounded rationality. Theories of bounded rationality assert that it is doubtful living entities have the unlimited cognitive capability to rank all choices they face by preference and to perform the utility maximization calculations required of standard rational agents. Instead, living agents weigh the cost of assessing available choices against the risk of making an inferior choice (Selten 2001). At a point when cognitive costs becomes too great, agents with bounded rationality dispense with further calculation and make choices based on incomplete information (Frank 2005a, Camerer and Fehr 2006). One method of increasing the likelihood of positive outcomes when making decisions with limited information is through the use of simple heuristics. Common heuristics include not only one's use of previously successful strategies, but also imitation of strategies that have been observed to result favorably for others (Boyd and Richerson 1985, Todd 2001, Richerson and Boyd 2005). In the latter case individual preferences are again influenced by the choices of others and the results of their choices.

Inequity aversion – the previous three explanations of interdependent preferences are simply redefined representations of self-interest. They all assume that in [6.1] u is increasing in $c/\bar{c}$. However, this is only one possible form of an interdependent utility function. There is ample evidence that in some circumstances humans also display a form of other-regarding preferences in which u decreases as $c/\bar{c}$ deviates further from 1. In other words humans exhibit an aversion to inequity or unfair consumption (Nowak et al. 2000, Fehr and Fischbacher 2003, Abbink et al. 2004, Fowler et al. 2005). Though controversial, there is evidence of the phenomenon in non-human primates as well (Brosnan and de Waal 2003, Vogel 2004).

These explanations have in common that an individual's utility depends at least in part on consumption choices of others in a society. While theories in this brief review attempt to explain apparent empirical instances of interdependent preferences, explanations should also be framed in a meaningful evolutionary context (Veblen 1898, Cordes 2007, Poirot 2007). To address this concern, scientists must explain how a mechanism of relative comparisons evolved in the first place. For this purpose we turn to evolutionary biology and a discussion of fitness.

Biological fitness and relativity

Organic evolution is driven by a number of processes, the most important of which arguably is natural selection. Natural selection acts on a quality of organic entities known as fitness – a composite of survivability, fecundity, mating ability, and other factors that determine the extent to which the entities' genes will be passed to future generations (Hedrick 2005). However, fitness drives selection only as it relates to the fitness of others in the same environment. In other words *relative* fitness, the differential ability to populate future generations, is the underlying quality upon which natural selection acts. Yet selection based on relative fitness is simply another way of stating that future representation depends on absolute fitness among a local neighborhood of competitors – precisely the situation facing agents embedded in a social network.

Consider a simplistic thought experiment in which the ultimatum game is played between the only two members of a society. If each is concerned only with maximizing consumption via payoffs then the expectation is that any offer greater than 0 will be accepted by the responding individual. Knowing this, the offering agent will offer the smallest positive fraction possible (Osborne 2004). But what if the agents are playing for a finite resource directly related to fecundity? In other words, what if the payoffs are measured in terms of fitness in an environment with finite carrying capacity? As selection favors those with higher relative fitness each agent will seek to win more fitness than the other leading to two possible equilibria – a 50/50 split or a refused offer resulting in 0 fitness for both players. Clearly the expectations are different when the prize is fitness instead of absolute wealth. Thus there is an inherent tendency towards outperforming others in game and conflict situations – strategies are interdependent.

Utility, payoffs and fitness: confusion in behavioral experiments

To an economist, behavior is driven by a desire of rational agents to maximize utility. To an evolutionary biologist, behavior is driven by natural selection of strategies that maximize relative fitness. However, in behavioral experiments, both economists and biologists customarily use payoffs as a surrogate for the variable they believe their agents are attempting to maximize. Yet if payoffs are truly equivalent to both utility and fitness, utility and fitness should also be equivalent to each other and behavior that maximizes utility should be the same behavior that maximizes relative fitness. But is this conclusion supported by empirical evidence? At least in modern humans, the answer is no. Several studies have revealed a significant inverse relationship between a society's per capita income and birth rate (Mulder 1998). In terms of biological fitness, maximization of payoffs, measured as income, would appear to be maladaptive in today's industrial world. At the same time, laboratory experiments have demonstrated that humans make utility maximizing decisions that do not necessarily maximize monetary payoffs (Minas et al. 1960, Henrich et al. 2001). It would appear that financial payoffs are an inadequate surrogate both for fitness and for utility.

In chapters 3-5 reproductive rate is directly and explicitly tied to strategic game behavior, demonstrating a possible genetic basis for cooperative behavior. Though some behaviors are thought to have evolved as an expression of genetic coding, at some point in evolutionary history, humans attained the ability to learn and imitate behaviors of others. At that point, behavioral strategies became, at least to some degree, divorced from genetic programming with the result that much social behavior was free to follow its own evolutionary trajectory via cultural selection (Boyd and Richerson 1985). As shown by evidence above, this trajectory may become biologically maladaptive. This is primarily true of humans where cultural evolution may favor strategies that maximize utility or income instead of biological fitness (Mulder 1998). In the context of simulations used in this study, agents are abstract to the degree that they could be considered not only biological agents but also cultural strategies that propagate via non-genetic means[11].

[11] Though not a perfect analogy, future research may benefit from modelling the propagation of human decision strategies as a coevolutionry host-parasite system. Strategies are dependent on human hosts and may exhibit periodic episodes of increased virulence (e.g. overconsumption) that have detrimental effects on host populations.

Relativity and Localized Interactions in Complex Networks

Regarding interdependent preferences it is important to note that when humans assess their status they do not use an entire society as a reference frame. Instead, concern with status is local - humans address their standing only among others of similar wealth and consumption comparisons are made only to a reference subset of the population (Harbaugh 1996). In other words, for purposes of utility calculations, humans are embedded in a social network where the local neighborhood consists of others of similar status. Depicting such comparison patterns is well-suited to the methodology of social network modelling.

Here we return to results from Chapters 3-5, in which social structure offers a plausible evolutionary explanation for interdependent preferences. Results presented in these chapters demonstrate that simulated agents rewarded with fecundity for the highest payoffs nonetheless evolve behavior that is inconsistent with maximizing payoffs in one-shot games. This behavior evolved only in societies structured in social networks. The limitation of interactions to a local neighborhood effectively creates the basis for an emphasis on relative comparisons. Regardless of the explanation one accepts as a proximate mechanism for the existence of relative considerations and interdependent preferences, behavior evolved in the context of social networks may provide an ultimate mechanism.

Altruistic Punishment and Rational Expectations

The mechanism of altruistic punishment is a leading candidate for explaining the evolution of cooperation. Altruistic punishment occurs when an individual incurs a cost to punish another but receives no material benefit in return (Boyd et al. 2003, Bowles and Gintis 2004, Fowler et al. 2005). It has been shown to induce cooperative behavior in numerous studies (Andreoni et al. 2003, Fehr and Fischbacher 2004, Gürerk et al. 2006). However, controversy regarding altruistic punishment lingers because, while it may explain many instances of cooperative behavior, the mechanism itself is seemingly irrational. Why should an individual expend fitness or wealth to punish someone with whom they will never again interact and when they receive no benefit from doing so? Yet despite the economic prediction that a rational agent will not pay to punish others, humans repeatedly do so in laboratory experiments (Ostrom et al. 1992, Fehr and Gächter 2000, 2002).

This economic expectation is based on the widely held premise that agents with independent preferences act to maximize their absolute payoffs. Previously a framework was presented for expectations when the evolution of punishment behavior is driven by relative payoffs instead of absolute payoffs (Shutters 2008). In this study, computer simulations were conducted to test this framework and to determine whether agents evolved behavior reflective of absolute payoff maximization or relative payoff maximization.

The Simulation[12]

In a series of agent-based computer simulations, social networks comprised of regular lattices were populated with agents that played the continuous prisoners dilemma

[12] The simulation model described here is also described similarly in Chapter 3 but repeated here so that dissertation chapters are self-contained for potential publications.

against one another. After each game, a 3rd party observer had the opportunity to pay to punish agents in the game if a player's contribution was deemed too low. Each component of an agent's strategy – how much to contribute in a game, when to punish, and how much to spend on punishment – were all independently evolving attributes of an agent.

The continuous prisoners dilemma

The prisoners dilemma is a commonly used framework for studying social dilemmas and consists of a two-player game with payoffs structured so that social goals and individual goals are at odds. Total social welfare is maximized when both players cooperate but a player's individual payoff is always maximized by cheating.
In the classic prisoners dilemma players are limited to two choices – cooperate or defect. In this study agents select a level of cooperation x at any point on a standardized continuum between full defection ($x = 0$) and full cooperation ($x = 1$). This is known as the continuous prisoners dilemma (CPD) and presents an arguably more realistic picture of the complexity of true social dilemmas (Killingback and Doebeli 2002). In a game between agents i and j, i's payoff p is

[6.2] $$p_i = 1 - x_i + r(x_i + x_j)/2; \qquad x \in [0,1]\ , \ r \in (1,2)$$

where r represents the synergistic effect of cooperating. In this chapter $r = 1.5$ in all cases. Total social welfare $p_i + p_j$ is maximized when $x_i = x_j = 1$, yet i's payoff is maximized when $x_i = 0$ regardless of the contribution made by j.

Introduction of altruistic punishment

The introduction of punishment to the CPD adds a 3rd party to the game – the observer. The decision of whether to punish or not is controlled by an attribute of the observer, the observer's punishment threshold, which is compared to a player's contribution in an observed game. If the player's contribution falls below the observer's threshold, the observer punishes the player. The amount the observer spends to punish c is also an agent attribute but is only invoked when the agent acts as a 3rd party observer of a neighboring game.

Letting c = the cost to the observer to punish, the amount deducted from the punished player is cM, where M is the punishment multiplier, a simulation parameter controlling the relative strength of punishment.

In agreement with many recent experiments, the introduction of punishment in structured populations led to cooperative behavior. However, the focus of this study is not on the cooperative outcomes of the prisoners dilemma but on the mechanism of altruistic punishment that induces those outcomes.

Social structure

Testing predictions of the relative payoff model required simulations in which agents had different numbers of neighbors. For this purpose social networks were used to structure populations so that interactions, both game play and observation, were restricted to a fixed number of immediate neighbors. To control for confounding effects due to variance in the number of neighbors among agents, networks were limited to regular lattice structures (Table 6.1) so that variance in number of neighbors = 0 in all simulations.

In addition to regular lattices, simulations were run on a complete network. In a complete network every agent is linked to every other agent and, like a regular lattice, has no variance in number of neighbors per agent. A complete network represents the social network version of a homogeneous well-mixed system.

The simulation algorithm

A single simulation initiates with the creation of the appropriate lattice network. At each of 400 nodes is placed an agent i consisting of strategy (x_i, t_i, c_i) where x_i = the contribution *i* makes to the public good when playing against *j*, t_i = the contribution limit below which *i* will punish another agent in a game being observed by *i*, and c_i = the cost that *i* incurs to punish the observed agent when the observed agent's contribution is too low. In other words t_i determines if agent *i* will inflict punishment and c_i determines how much punishment agent *i* will inflict. Each of the three strategy components holds a value on the continuous interval [0,1] and is generated randomly from a uniform distribution at the beginning of a simulation. To control for other factors that might contribute to the maintenance of cooperation, such as interaction history or reputation, agents have no recognition of or memory of other agents.

The simulation proceeds for 10,000 generations, each consisting of a game play routine, including observation & punishment, and a reproduction routine. During a single CPD game an agent *i* initiates a game by randomly selecting *j* from its neighborhood. Both agents are then endowed with one arbitrary unit from which they contribute x_i and x_j

respectively to a public good. Players' choices are made simultaneously without knowledge of the others' contribution. *i* then randomly selects a second neighbor *k*, which is tasked with observing *i*'s contribution. If *k* judges the contribution to be too low, *k* pays to punish *i*. After each game, running payoffs for *i*, *j*, and *k* are calculated as shown in Table 6.2. Each agent initiates 3 games during a single generation and all games in a generation are played simultaneously.

Following game play the reproduction routine runs in which each agent *i* randomly selects a neighbor *j* with which to compare payoffs for the generation. If $p_i > p_j$, *i*'s strategy remains at *i*'s node in the next generation. However, if $p_i < p_j$, *j*'s strategy is copied onto *i*'s node for the next generation. In the event that $p_i = p_j$, a coin toss determines which strategy prevails. As strategies are copied to the next generation each strategy component of every agent is subject to mutation with a probability $m = 0.10$. If a component is selected for mutation, Gaussian noise is added to the component with mean = 0 and std. dev. = 0.01. Should mutation drive a component's value outside [0,1], the value is adjusted to the closer boundary value.

The Model of Relative Payoff Maximization

Because evolution is driven by relative fitness as opposed to absolute fitness, it is important to consider the possibility that evolution of economic behavior is driven by relative payoffs. In other words, letting p = the payoff of agent *i* and P = the sum payoffs of the *n* neighbors of *i*, we should compare evolved behavior when *i*'s survival is driven by maximization of p (absolute payoff) versus maximization of p/P (relative payoff). At the risk of oversimplification, we can think of these two approaches as the standard economic and evolutionary economic viewpoints respectively.

Letting time step 0 be a point in time prior to an act of punishment and time step 1 be a point after punishment, the following describe the effects of a single punishment act:

[6.3a] $$p_1 = p_0 - c$$

[6.3b] $$P_1 = P_0 - cM.$$

For an act of punishment to be evolutionarily beneficial it should lead to an increase in i's relative payoff such that the following is true:

[6.4] $$\frac{p_1}{P_1} > \frac{p_0}{P_0} \quad \text{or} \quad \frac{p_0 - c}{P_0 - cM} > \frac{p_0}{P_0} \; .$$

When simplified and restated (see Chapter 5), this evolutionary economic model predicts that for an agent with n neighbors, punishment becomes an evolutionarily beneficial strategy when

[6.5] $$M > n$$

(compare to Ohtsuki et al. 2006).

In contrast, the standard economic expectation of behavior evolved through maximization of p is that i will never punish. As [6.3a] shows, $p_1 < p_0$ for any positive amount of punishment. Punishment is never a beneficial strategy when absolute payoff motivates strategic behavior.

To compare these predictions, the simulation described above was run while systematically varying the punishment parameter *M*, and were conducted on the lattices listed in Table 6.1 to vary the number of neighbors *n*. 100 replications of the simulation were run for each value of *M* starting at $M = 0.0$ and subsequently at increments of 0.5 until $M = 6.0$.

Experimental Results and Discussion

Results were mixed but generally favor the Darwinian perspective that the evolution of agent behavior is driven by maximizing relative payoffs.

Occurrence of punishment

Experimental results are presented in Table 6.3. As *M* increased, simulations run on all lattice networks eventually reached a value of *M* at which the population underwent a rapid transition from defectors to cooperators. Results from this dissertation previously demonstrated that punishment is the mechanism driving this flip to cooperative behavior and that the transition value of *M* indicates the point at which punishment becomes a beneficial strategy (Chapter 3). Only in simulations run on a complete network did cooperative behavior never evolve. This was true for complete networks even at $M = 5{,}000$.

This result suggests that the simple standard economic premise that agents maximize absolute payoff is not valid for populations embedded in discrete social structures (see also Granovetter 1985). Only in a complete network, which is analogous to a well-mixed, homogeneous system, did simulation results match predictions of the standard model. Instead results support an evolutionary model and the Darwinian notion that when sufficiently strong, punishment may become evolutionarily beneficial and, in turn, induce cooperative behavior in a population.

This result is especially significant given that human populations are not homogeneous and well-mixed, but are structured by complex interaction networks (Watts and Strogatz 1998).

Punishment and number of neighbors

On lattices where punishment behavior did evolve, the biological expectation was that punishment would proliferate when $M > n$ [6.5], where n is the number of neighbors an agent has. Table 6.3 shows that for each lattice type, the value of M at which punishment actually became prevalent was lower than predicted by this model.

This result indicates that subtle effects of the network structure are likely missing from the simplistic prediction of [6.5]. If researchers and policy makers are to better predict behavior of rational agents in structured populations, they must not only re-evaluate their assumptions of rationality, they must also begin to understand and quantify these network effects.

Among plausible reasons that punishment emerges at a lower value of M than expected is that a cheater in agent i's neighborhood is also a member of other neighborhoods and may be incurring punishment from agents other than i. In addition, neighbors of i may be lowering their payoffs by engaging in their own altruistic punishment. Both of these events decrease P_1 in equation [6.4], leading to a lower transition value of M.

Summary

Many problems facing policy makers today require tools that facilitate the prediction of human behavior. Since models of human behavior are typically premised on concepts of economic rationality, it is critical to evaluate the standard assumptions of rationality. Results of this study suggest that relaxing standard assumptions to include more a evolutionary perspective may lead to higher predictive accuracy. In particular, they suggest the roles of relative and absolute payoff maximization should be reviewed when making assumptions about the preferences of rational agents.

Over 100 years ago economist Thorstein Veblen bluntly asked, “Why is economics not an evolutionary science?” (Veblen 1898). The intent of this chapter is to continue Veblen’s quest by contributing to an ongoing dialog between evolutionary biologists and behavioral economists. The ultimate goal of such a dialog is, by modifying assumptions of rationality to incorporate Darwinian theory, to improve both understanding and prediction of social behavior.

Table 6.1

Lattice networks used for the continuous prisoners dilemma in this chapter

Network Type	Description of Neighbors	Number of Neighbors
Ring	Left, right	2
Von Neumann	Left, right, up, down	4
Hexagonal	Left, right, diagonals	6
Moore	Left, right, up, down, diagonals	8
Complete	All other agents	$N - 1$

Table 6.2

Payoffs *p* in the continuous prisoners dilemma between *i* and *j* with possible punishment of *i* by *k*

	$x_i \geq t_k$	$x_i < t_k$
k punishes *i*?	no	yes
p_i	$1 - x_i + r(x_i + x_j)/2$	$1 - x_i + \mathrm{r}(x_i + x_j)/2 - c_k M$
p_j	$1 - x_j + r(x_i + x_j)/2$	$1 - x_j + r(x_i + x_j)/2$
p_k	0	$- c_k$

Note: see Tables 3.1 and 3.3 for description of variables.

Table 6.3

Simulation results versus both standard economic and evolutionary economic predictions

Network Type	Value of M at which punishment becomes beneficial		
	Standard Prediction	Evolutionary Prediction	Simulation Result
Ring	Never	2	1.5
Von Neumann	Never	4	1.8
Hexagonal	Never	6	2.2
Moore	Never	8	2.8
Complete	Never	$N - 1$	Never

Note: $N = 400$ in all cases

CHAPTER 7

INTERDEPENDENT PREFERENCES AND THE EFFICACY OF INTERNATIONAL ENVIRONMENTAL AGREEMENTS

We now return to real-world applications described in Chapter 1. Regardless of one's definition of sustainability, there is customarily a requirement of those acting sustainably to subordinate their self-interest to that of some collective goal. This requirement often takes the form of restraint in the exploitation of some natural resource. In this sense, sustainable management of natural resources resembles a public good and its provisioning at a global level can be represented as an international public goods game. However, at this hierarchical level of society, where interactions take place between nations, there is no central authority that might implement a "command-and-control" policy to enforce sustainable practices (Sandler 1992, Dietz et al. 2003). Without a central enforcer, cooperative outcomes depend on the complex interactions of multiple, heterogeneous nations that play out this international game through the crafting of agreements. Of particular interest to this dissertation are the policy instruments known as international environmental agreements (IEAs).

Nations attempting to implement IEAs face the same dilemmas and incentives to cheat as participants in experimental economic games (Barrett 2005), particularly when agreements are multilateral. Laboratory experiments using public-goods games therefore have a clear application to the decision-making process in which nations engage when crafting IEAs. These interactions might be modelled as a cooperative game, in which the participants can form binding agreements and coalitions prior to executing their strategies (Binmore 1992). However, cooperative game theory assumes the existence of an external enforcement mechanism that allows for binding pre-choice agreements and, as previously stated, national actors at the global scale have no such exogenous enforcement. Therefore, international environmental dilemmas are more properly modelled as non-cooperative games (Wagner 2001, Barrett 2005) in which binding agreements prior to execution of strategies are not permissible (see Chander and Tulkens 2006 for a critique of this assertion).

A Model of International Conflict: The Standard Prisoners Dilemma

In contrast to the continuous prisoners dilemma (CPD) used in chapters 3 and 4, let us consider now the simpler, standard version of the prisoners dilemma (PD) in which two players each have but two options – cooperate or defect. Figure 7.1a displays the customary shorthand for each of the four possible combinations of strategies by players A and B: cooperate-cooperate = R (reciprocity), defect-cooperate = T (temptation), cooperate-defect = S (sucker's bet), and defect-defect = P (punishment). The standard PD is then defined as having the following two conditions:

[7.1a] $T > R > P > S$

[7.1b] $2R > (T+S)$

Typical payoffs of this simple game are often represented in a 2x2 payoff matrix as shown in Figure 7.1b. These variables are defined from Player A's viewpoint so that in the example PD of Figure 7.1(c) $R = 5$, $T = 8$, $P = 1$, and $S = 0$. Player B faces symmetrical outcomes. A dilemma arises because while T leads to the highest individual payoff for Player A, R leads to the highest joint payoffs and is therefore socially optimal.

As self-interested players attempt to effect outcome *T*, their joint defection instead results in the Nash equilibrium *P*, the least socially desirable outcome.

The standard PD sufficiently represents the most important aspects of international conflict and is often used as a framework to analyze global dilemmas (Sandler 1999). When modeling international conflict through the PD, resolution remedies typically call for the use of diplomatic tools to alter the payoffs such that socially optimal *R* is the new Nash equilibrium (Barrett 1999), or as Wagner (2001, p. 385) says, "an IEA must change the rules of the game". This may be done through a number of mechanisms used in IEAs such as side payments (Sandler 1999, Wagner 2001), trade and other forms of issue linkage (Wagner 2001, Barrett 2003b, Barrett and Stavins 2003, Ward 2006), and cost sharing (Boyle 1991, Sandler 2004). It may also include negative incentives such as trade restrictions and punitive financial policies (Barrett and Stavins 2003). A detailed discussion of each of these methods is not as important as the fact that their common intent is to restructure payoffs so that

[7.2] $$R > T > P > S.$$

Note the ordinality of *R* and *T* is reversed compared to [7.1]. This removes the dilemma so that the cooperative strategy leads to both the social and an individual optimum[13].

---

[13] In this particular case the game is converted to the Stag Hunt game which has two Nash equilibria, one of which is the socially optimal outcome. The Stag Hunt addresses another interesting set of questions, though these relate primarily to risk aversion instead of social dilemmas. See (Skyrms 2001, 2004) for further discussion on the Stag Hunt game.

Distinguishing Between Payoffs and Utility

Recall here that standard rational choice theory predicts a rational agent will make choices that maximize utility – not payoffs. This is a subtle but critical distinction and it implies that an unstated utility matrix is the true representation of predicted outcomes. Implicit in diplomatic strategies that restructure a dilemma's payoffs is an assumption that utility of outcomes will conform to the same ordinality as payoffs of outcomes so that presentation of both a payoff matrix and a utility matrix is redundant. In other words, it is generally assumed that reordering payoffs automatically reorders utilities so that when a game having $T > R$ and $u(T) > u(R)$ is restructured such that $R > T$, a result will be $u(R) > u(T)$. This is valid when utility is a simple linear transformation of payoffs.

Figure 7.1c shows how PD payoffs might be restructured to achieve this desired outcome. In this case the PD in Figure 7.1b has been restructured so that social and individual goals are aligned. If nations achieve satisfaction only from maximizing their own payoffs so that $u_A = u(p_A)$, the dilemma is solved and we would expect both nations in this example to cooperate.

But what if nations are motivated by their payoffs relative to those of their opponents so that $u_A = u(p_A, p_B)$? One possible form of such an interdependent utility function is structured so that satisfaction is derived solely from relative payoffs

[7.3] $$u_A = u(p_A / \bar{p})$$

where $\bar{p}$ is the mean payoff of all parties to the dilemma. In the case of a 2-player prisoners dilemma this becomes

[7.4] $$u_A = u[2p_A/(p_A + p_B)],$$

creating a matrix of utilities that is not a simple linear transformation of the payoff matrix. Using [7.4] to transform the payoff matrix in Figure 7.1c gives the utility matrix in Figure 7.1d. Despite restructuring the order of payoffs the ordinal utilities remain unchanged, $u(T) > u(R) > u(P) > u(S)$. Expected behavior is unaffected by techniques of statecraft in this case and the dilemma persists. One oversimplification of [7.4] is that it does not take into account the starting position of each player. It is assumed in this and similar models that players start from symmetric positions. This would indeed be a rare occurrence in real world situations (Hadjimichalis and Hudson 2006). It is intuitive that nations enter a dilemma with existing relative positions in terms of previously accumulated payoffs – relative not only to other nations engaged in the dilemma but to countries not affected by the current game. Sandler (1992) asserts that such asymmetry is related to further failure of collective action.

The examples above represent the two extremes of utility with regard to interdependent preferences: one in which only absolute payoffs matter and the other in which only relative payoffs matter. Let us now explore the area between these extremes beginning with the general interdependent utility function [6.1] and assuming that payoffs p and consumption c are interchangeable (Chapter 2). The general utility function can now be written

[7.5] $$u(p, p/\overline{p}) = (1-\omega)p + \omega(p/\overline{p})$$

where ω is a weight factor determining the relative importance of absolute and relative payoffs so that utility is a function only of absolute payoff when $\omega = 0$ and only of relative payoff when $\omega = 1$. Although quite rudimentary this model allows us to examine, under a range of intermediate values of ω, the expected behavior of participants in a dilemma both before and after payoffs have been altered through IEAs.

Of particular importance are outcomes *R* and *T* in Figures 7.1b and 7.1c and the ordinal rank of their utilities using [7.5]. To compare the utility of each of these game outcomes let

[7.6] $$u_\Delta = u(R) - u(T)\ .$$

The cooperative strategy C is expected when $u_\Delta > 0$ while defection D is expected when $u_\Delta < 0$. Substituting payoffs of outcomes *R* and *T* into [7.5], [7.6] can be expanded to

$$u_\Delta = [(1-\omega)R + \omega] - [(1-\omega)T + \omega(2T/(T+S))]\,.$$

When rearranged and simplified this gives

[7.7] $$u_\Delta = (1-\omega)(R-T) + (\omega/[S+T])(S-T)$$

where $(R - T)$ is the component influenced by absolute payoff and $(S - T)$ is the component influenced by relative payoff. The absolute component $(R - T)$ is negative in the PD but becomes positive when the game is restructured as in Figure 7.1c. Therefore, a player motivated only by absolute payoff ($\omega = 0$) will defect in the PD but cooperate in the restructured game. However, since $T > S$ in both the PD and the restructured game, the component influenced by relative payoffs is always negative and a player driven only by considerations of relative payoffs ($\omega = 1$) will defect even in a restructured game. These conditions are summarized in Table 7.1. For the more realistic case of a player that is motivated partly by absolute payoff and partly by relative payoff ($0 < \omega < 1$), it becomes impossible in this model to predict whether a dilemma restructured to promote cooperation will actually achieve its goal (Table 7.1).

Figure 7.2 plots u_Δ as a function of ω to demonstrate that when a player is at least partly motivated by relative payoffs, there will be a range of ω under which the expected outcome is joint defection even when a PD has been restructured specifically to promote cooperation. At some intermediate value $\omega = i$, a player in a restructured game is indifferent between cooperating and defecting. When $\omega \in [0, i)$ both players are expected to cooperate, but when $\omega \in (i, 1]$ the expectation is joint defection. The threshold value $\omega = i$ depends on attributes of the specific game and the players involved.

Nations As Agents

Chapter 3 and 6 showed that populations of agents embedded in complex social networks can evolve strategic behavior for the PD that is consistent with behavior motivated by interdependent preferences. In addition, this dissertation has shown that agents in networked populations can evolve an array of complex behaviors not predicted by standard economic theory, including not only cooperation and punishment, but also failure to respond positively to restructuring of dilemma payoffs (see also Gould 1993, Pacheco and Santos 2005, Santos and Pacheco 2006, Santos et al. 2006b, Chen et al. 2007, Hui et al. 2007, Tomassini et al. 2007). While this is an interesting academic exercise any attempt to apply these findings to real world global problems would be merely speculative without demonstrating that nation-states are similar to agents used in this study's simulations.

To validate the analogy it must first be shown that nations of the world interact through complex networks of connections and do not comprise a homogeneous, well-mixed system. This network of interactions may reflect trade linkage, geographical proximity, competition for a distant resource (e.g. Antarctica), military alliances, or any other number of factors that structure interactions between nations. Second, it must be demonstrated that nations, like individual humans, base strategic decisions, at least in part, on the behavior or status of other actors with which they interact.

Regarding the first requirement we turn to several recent studies of the ways in which countries are linked. Careful analysis of the pattern of international trade reveals that between-nation interactions are neither random nor well-mixed interaction networks, but instead match parameters of complex small-world networks (Castillo and Baeza-Yates 2003, Serrano and Boguna 2003, Freyberg-Inan 2006). Others have since shown that such international networks are not limited to trade relationships but extend to global-scale interactions of "groups of nations, NGOs and international agencies"(Ward 2006, p. 149), and even international terrorist networks (Moon and Carley 2007), and they have also shown that these network structures are instrumental in global cooperation (Bernauer et al. 2008).

Regarding the second requirement that national strategies depend in part on the behavior of other nations, the existence of such behavior has been acknowledged since at least David Ricardo's treatise on national comparative advantage (Ricardo 2001 [1817]), and has been well-established through numerous studies of the international demonstration effect (James 1987, 2000, Liu and Sun 2005).

Relative Payoffs and Implications for Successful IEAs

If we accept the empirical evidence that nations are embedded in complex interaction networks and act strategically based in part on interdependent preferences, we may conclude that the possibility exists that IEA remedies based on restructuring payoffs may bear little fruit in practice.

The possible ineffectiveness of restructured payoffs was long ago demonstrated in laboratory experiments between human subjects. In late 1950's Scodel and Minas conducted some of the first controlled laboratory experiments using the PD with monetary incentives. These experiments were extraordinary, not only for the implications their results have on the current topic, but also for the lack of subsequent notice they received in mainstream economics literature. Results were first gathered using the standard PD, adhering to the requirements of [7.1]. Not surprisingly defection occurred in nearly 86% of all games, agreeing well with expectations of rational choice theory (Scodel et al. 1959). However, in the second phase of experiments payoffs were altered to match [7.2] so that cooperate-cooperate offered the highest individual payoff to both players as well as the highest total payoff. The PD was effectively restructured into a game in which individual goals and social goals were supposedly aligned. Much to the surprise of the researchers, defection still occurred in a staggering 72% of the restructured games (Minas et al. 1960). The best conclusion the authors could make was that subjects viewed the game as a competition and received more utility from "the psychological need to outdo the other person" than they did from the money they could have earned (Minas et al. 1960, p. 107). In other words, the pay-off matrix and utility matrix were not linear transformations of each other and the maximum payoff did not deliver the maximum utility.

The potential ineffectiveness of restructuring a dilemma's payoffs has also been demonstrated in actual IEAs. The Montreal protocol is often heralded as an example of the ability of multilateral IEAs to induce cooperative outcomes between nations (Barrett 2003b), with former United Nations General Secretary, Kofi Anan proclaiming it "the single most successful international agreement to date" (UNEP 2007). Taking effect in 1987, the treaty called for eventual phase out of the production of ozone-destroying chlorofluorocarbons (CFCs). Claims of treaty's success are based largely on the fact that countries substantially fulfilled their reduction obligations under the agreement. However, analysis of national behavior before and after the treaty went into effect reveals that the treaty had no true effect on national strategies – nations had intended to phase out CFCs regardless of whether they ratified the protocol or not (Murdoch and Sandler 1997b). The same phenomenon was described among nations ratifying the Helsinki Protocol calling for reductions in sulfur emissions – participating nations did nothing more than what they had intended to do regardless of the treaty (Murdoch and Sandler 1997a). The possible ineffectiveness of restructured dilemmas is not restricted to laboratory experiments but appears to be present in even the most acclaimed multilateral IEAs.

Enhancing the Ability of IEAs to Restructure Dilemmas

If we assume that national strategic behavior is truly affected by considerations of relative payoffs, and that such considerations can decrease the efficacy of IEAs, then we should explore parameters that will increase the probability of IEAs achieving their intended outcomes. Returning to [7.7] and Figure 7.2, we may determine changes that increase i, the value at which a player is indifferent between cooperation and defection. In other words enhancing the efficacy of IEAs requires increasing the range of $[0, i)$ and may be accomplished through the following mechanisms:

1) Increasing the absolute component of utility ($R - T$),
2) Decreasing the relative component of utility ($T - S$), or
3) Decreasing a nation's value of ω.

Both methods 1 and 2 may be accomplished through a decrease in T, or in other words, by decreasing the payoff earned from cheating on a cooperative partner. Likewise both may be accomplished through supplementing the payoffs of a cooperative player through rewards. In either case mechanisms for sanctions and rewards have been widely investigated, used in practice, and advocated among policy makers (Oliver 1980, Wagner 2001, Sefton et al. 2002, Andreoni et al. 2003, Fehr and Fischbacher 2004, Barrett 2005, Gürerk et al. 2006).

However, mechanisms that would influence the degree to which a nation's behavior is influenced by relative payoffs ω are far less obvious, as it is likely an intrinsic component of national culture. An extension of the simplistic model in [7.7] to incorporate the fact that nations are heterogeneous with respect to their relative standing or status (Sacks et al. 2001) would likely help elucidate such mechanisms and should enhance the ability to predict behavior under IEAs. Nations ranked near the bottom in some measure of comparative status will likely be influenced by even the most trivial relative payoff so that the incentive to cheat is great. Likewise, nations that already enjoy a high relative standing will likely be concerned primarily with absolute payoffs and so will be more amenable to any restructuring where $R > T$.

Conclusion

The phenomenon of interdependent preferences is well documented among humans in laboratory settings. This chapter demonstrates that if nations too are motivated by considerations of relative standing then much of the prevailing wisdom regarding solutions to global environmental problems may be inadequate. Use of diplomacy to restructure dilemmas may give the appearance of so-called win-win situations but in reality may have little effect on national strategies. This effect may be mitigated to the extent that nations are motivated by absolute payoffs instead of relative payoffs, but it is also complicated by a nation's pre-existing standing in the global community. If multilateral IEAs are to truly address global dilemmas such concerns for relative position must be better understood and incorporated into diplomatic theory and remedies, and weights that nations place on relative and absolute payoffs should be empirically determined to the extent possible.

Table 7.1

Expected strategies before and after restructuring a prisoners dilemma.

Game	Ordinal payoffs	$\omega = 0$	$0 < \omega < 1$	$\omega = 1$
Prisoners Dilemma	$T > R > P > S$	D	D	D
Restructured Game	$R > T > P > S$	C	?	D

Note: ω = the relative importance of absolute and relative payoffs to a player's utility. When $\omega = 0$ only absolute payoff affects utility. When $\omega = 1$ only relative payoff affects utility.

Player A	Player B: C	Player B: D
C	*R*	*S*
D	*T*	*P*

(a)

Player A	Player B: C	Player B: D
C	5, 5	0, 8
D	8, 0	1, 1

(b)

Player A	Player B: C	Player B: D
C	8, 8	0, 5
D	5, 0	1, 1

(c)

Player A	Player B: C	Player B: D
C	1 ,1	0, 2
D	2, 0	1, 1

(d)

Figure 7.1. Payoff matrices for the standard prisoners dilemma (PD). C = cooperate, D = defect, the first number in each payoff square equals the payoff to Player A, and the second number is the payoff to Player B. (a) Customary shorthand variables used to label all possible PD outcomes; (b) Typical payoff values of a PD; (c) the PD restructured by side payments or similar diplomatic tools so that joint cooperation is the equilibrium outcome; (d) the utility matrix of the restructured game in 7.1c if player satisfaction is derived from maximization of relative payoffs [7.4].

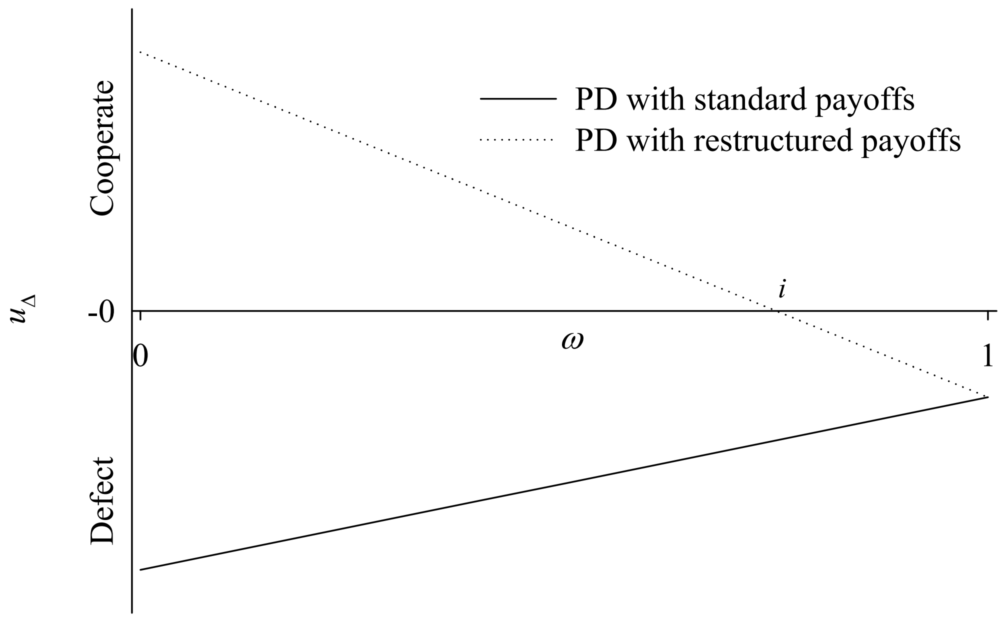


Figure 7.2. Weighted effect of relative vs. absolute payoffs in a restructured prisoners dilemma. When $\omega = 0$ only absolute payoffs affect decision making and when $\omega = 1$ only relative payoffs matter. Plotting [7.5] using the values in Figures 7.1b and 7.1c, the y-axis u_Δ represents the difference between the utility of the cooperative outcome $u(R)$ and the defection outcome $u(T)$. When $\omega \in [0, i)$ both players are expected to cooperate, but when $\omega \in (i, 1]$ the expectation is joint defection. When $\omega = i$, a player is indifferent between cooperating and defecting. Cooperation is never predicted in the standard PD.

CHAPTER 8

SUMMARY AND FUTURE DIRECTIONS

In producing this dissertation I have attempted to contribute to a new paradigm in social research - one that embraces the contributions of many disciplines while exploiting the massive computational power of today's computers (Harrison 2006). One realization accepted by those that embrace this paradigm is that the world is much more complex than can be represented in elegant, yet limited, mathematical models (North 2005). If scientists are to contribute to solving the array of complex problems facing the world today they must move beyond those methods with which they have traditionally been most comfortable.

This study has used the relatively new methodology of computational social simulation, or agent-based modelling, to first address some of the fundamental questions facing evolutionary biologists, and secondly, to explore corollaries to those biological questions in the global policy realm.

Summary of Findings

1) Systems dynamics and other atomistic models are inadequate to describe and predict the behavior of structured societies comprised of heterogeneous agents.

2) When structure, especially a complex network, is incorporated into social models, punishment becomes a viable mechanism for the evolution and maintenance of cooperation.

3) The ability of punishment to induce cooperation in a population diminishes as the population becomes more densely connected.

4) Though punishment may induce cooperation in computer simulations, its byproducts, such as retaliation, may limit its efficacy in real world situations.

5) Unlike cooperation, fairness did not evolve in simulations with social structure and punishment, indicating that it likely evolved and persists through cultural factors.

6) Agents evolving in structured populations necessarily evolve strategies attuned to their local environment. This leads to an awareness and assessment of surrounding neighbors and may provide a basis for understanding origins of the phenomenon of interdependent preferences.

7) Suggested remedies for international environmental dilemmas based on standard assumptions of economic game theory may be inadequate without considering network structure, other-regarding preferences, and cultural influences on national decision-making behavior.

Future Directions

Dynamic networks

Throughout this dissertation, populations are embedded in static networks. Relationship links between population members are fixed, making the network essentially exogenous to the system under study. However, it is widely recognized that social networks, institutions, and other forms of social capital are not static, but coevolve with the populations with which they are associated (Takács et al. 2008). Therefore, it is prudent to extend these types of simulations to dynamic networks. Social network scientists are just beginning to develop the tools necessary to address this need and results have thus far shown that simulations similar to those of this dissertation may have qualitatively different outcomes when networks themselves are allowed to evolve (Skyrms and Pemantle 2000, Santos et al. 2006a, Shutters and Cutts 2008).

Mixed models

In this dissertation I have asserted a benefit of agent-based computer simulations is the ability they give researchers to move beyond limited systems dynamics models. However, it is likely that a combination of models will allow an even broader set of conditions to be explored. Simulations in this dissertation began by placing a single agent at each node of a complex network. A simple implementation of mixed systems/agent-based models would be to expand the simulations used herein so that at each node resides an entire well-mixed populations instead of a single agent. This would allow phenomenon specific to well-mixed subpopulations to evolve and then further evolve through structured interactions with other well-mixed subpopulations. This type of model has a clear analogy in human and biological metapopulation models.

Getting the game right

The primary game theoretical framework used in this study is the prisoners dilemma, arguably the most widely used framework for the study of social dilemmas and cooperation in particular. However, the PD best describes dilemmas involving pure public goods and may have limited applicability to real world situations (Sandler 1999, Sandler 2004). Recently much interest has turned to the snowdrift game (SDG, also known as the chicken game or hawk-dove game), which differs formally from the PD with respect to [7.1b] while retaining the same preference ranking of outcomes (Hauert 2001, Kummerli et al. 2007). Specifically the inequality is reversed so that

[8.1] $2R < (T+S)$.

In more qualitative terms the snowdrift game acknowledges the gains made from cooperation may not benefit everyone in society equally but may preferentially benefit the agent performing a cooperative act. It is therefore important to reassess numerous studies utilizing the prisoners dilemma with this alternate model to determine if general conclusions regarding social dilemmas are consistent. Results have already begun to demonstrate significant differences in outcomes when dilemmas are modelled as SDGs instead of PDs (Hauert and Doebeli 2004, Hui et al. 2007, Kummerli et al. 2007, Lee et al. 2008). In addition, any number of other games, such as the stag hunt game (Skyrms 2004), may better describe specific instances of social conflict and should not be ignored if simulations studies such as this are to continue making substantial contributions to the understanding of cooperation and social dilemmas.

BIBLIOGRAPHY

Abbink, K, A Sadrieh, and S Zamir. 2004. Fairness, public good, and emotional aspects of punishment behavior. *Theory and Decision* **57**(1):25-57.

Abbot, P, JH Withgott, and NA Moran. 2001. Genetic conflict and conditional altruism in social aphid colonies. *Proceedings of the National Academy of Sciences of the United States of America* **98**(21):12068-12071.

Abbott, KW, RO Keohane, A Moravcsik, A-M Slaughter, and D Snidal. 2000. The Concept of Legalization. *International Organization* **54**(3):401-419.

Alpizar, F, F Carlsson, and O Johansson-Stenman. 2005. How much do we care about absolute versus relative income and consumption? *Journal of Economic Behavior & Organization* **56**(3):405-421.

Amaral, LAN, A Scala, M Barthelemy, and HE Stanley. 2000. Classes of small-world networks. *Proceedings of the National Academy of Sciences of the United States of America* **97**(21):11149-11152.

Anderies, JM. 2002. The transition from local to global dynamics: a proposed framework for agent-based thinking in social-ecological systems. Pages 13-34 In *Complexity and Ecosystem Management*, edited by MA Janssen. Edward Elgar Publishing, Cheltenham, UK.

Anderies, JM, MA Janssen, and E Ostrom. 2004. A framework to analyze the robustness of social-ecological systems from an institutional perspective. *Ecology and Society* **9**(1):18-34.

Andreoni, J, W Harbaugh, and L Vesterlund. 2003. The carrot or the stick: Rewards, punishments, and cooperation. *American Economic Review* **93**(3):893-902.

Axelrod, R and WD Hamilton. 1981. The Evolution of Cooperation. *Science* **211**(4489):1390-1396.

Barabási, AL and R Albert. 1999. Emergence of scaling in random networks. *Science* **286**(5439):509-512.

Barrett, S. 1999. A theory of full international cooperation. *Journal of Theoretical Politics* **11**(4):519-541.

Barrett, S. 2003a. Creating Incentives for Cooperation: Strategic Choices. Pages 308-328 In *Providing Global Public Goods*, edited by I Kaul, P Conceição, K Le Goulven, and RU Mendoza. Oxford University Press, New York.

Barrett, S. 2003b. *Environment and statecraft: the strategy of environmental treaty-making*. Oxford University Press, Oxford.

Barrett, S. 2005. The Theory of International Environmental Agreements. Pages 1457-1516 In *Handbook of Environmental Economics Volume 3: Economywide and International Environmental Issues*, edited by K-G Maeler and JR Vincent. Elsevier, Amsterdam.

Barrett, S and R Stavins. 2003. Increasing Participation and Compliance in International Climate Change Agreements. *International Environmental Agreements: Politics, Law and Economics* **3**(4):349-276.

Bastani, S. 2007. Towards a Theory of Relative Preferences. University of Uppsala, Uppsala, Sweden.

Bedau, MA. 1999. Can Unrealistic Computer Models Illuminate Theoretical Biology? Pages 20-23 In *Proceedings of the 1999 Genetic and Evolutionary Computation Conference*, Orlando, Florida.

Beddoe, R, R Costanza, J Farley, E Garza, J Kent, I Kubiszewski, L Martinez, T McCowen, K Murphy, N Myers, Z Ogden, K Stapleton, and J Woodward. 2009. Overcoming systemic roadblocks to sustainability: The evolutionary redesign of worldviews, institutions, and technologies. *Proceedings of the National Academy of Sciences of the United States of America* **106**(8):2483-2489.

Bernauer, T, A Kalbhenn, G Ruoff, and V Koubi. 2008. Globalization, Democracy, and Network Effects in International Cooperation: Insights from Studying the Evolution of Global Environmental Commitments. Midwest Political Science Association Annual National Conference, Chicago.

Binmore, K. 1992. *Fun and Games: A Text on Game Theory*. D. C. Heath and Company, Lexington, Massachusetts.

Bowles, S and H Gintis. 2004. The evolution of strong reciprocity: cooperation in heterogeneous populations. *Theoretical Population Biology* **65**(1):17-28.

Boyd, R, H Gintis, S Bowles, and PJ Richerson. 2003. The evolution of altruistic punishment. *Proceedings of the National Academy of Sciences of the United States of America* **100**(6):3531-3535.

Boyd, R and PJ Richerson. 1985. *Culture and the evolutionary process*. University of Chicago Press, Chicago.

Boyle, AE. 1991. Saving the World? - Implementation and Enforcement of International Environmental-Law through International Institutions. *Journal of Environmental Law* **3**(2):229-245.

Brandt, H, C Hauert, and K Sigmund. 2003. Punishment and reputation in spatial public goods games. *Proceedings of the Royal Society of London. Series B: Biological Sciences* **270**(1519):1099-1104.

Brandt, H, C Hauert, and K Sigmund. 2006. Punishing and abstaining for public goods. *Proceedings of the National Academy of Sciences of the United States of America* **103**(2):495-497.

Brosnan, SF and FBM de Waal. 2003. Monkeys reject unequal pay. *Nature* **425**(6955):297-299.

Buchanan, M. 2002. *Nexus: Small Worlds and the Groundbreaking Theory of Networks*. Norton Publishing, New York.

Burkart, JM, E Fehr, C Efferson, and CP van Schaik. 2007. Other-regarding preferences in a non-human primate: Common marmosets provision food altruistically. *Proceedings of the National Academy of Sciences of the United States of America* **104**(50):19762-19766.

Camerer, CF and E Fehr. 2006. When does "economic man" dominate social behavior? *Science* **311**(5757):47-52.

Casari, M. 2005. On the design of peer punishment experiments. *Experimental Economics* **8**(2):107-115.

Castillo, C and R Baeza-Yates. 2003. Visualizing the European Trade Graph. Page 4. Univerita di Roma, Rome.

Cederman, L-E. 2001. Agent-Based Modeling in Political Science. *The Political Methodologist* **10**(1):16-22.

Chander, P and H Tulkens. 2006. Cooperation, Stability and Self-enforcement in International Environmental Agreements: A Conceptual Discussion. Core Discussion Papers. National University of Singapore, Singapore.

Chen, XJ, F Fu, and L Wang. 2007. Prisoner's Dilemma on community networks. *Physica A: Statistical Mechanics and Its Applications* **378**(2):512-518.

Chwe, MSY. 1999. Structure and strategy in collective action. *American Journal of Sociology* **105**(1):128-156.

Chwe, MSY. 2000. Communication and coordination in social networks. *Review of Economic Studies* **67**(1):1-16.

Clutton-Brock, TH and GA Parker. 1995. Punishment in Animal Societies. *Nature* **373**(6511):209-216.

Cordes, C. 2007. Turning economics into an evolutionary science: Veblen, the selection metaphor, and analogical thinking. *Journal of Economic Issues* **41**(1):135-154.

Darwin, C. [1859] 1996. *On the origin of species by means of natural selection, or The preservation of favoured races in the struggle for life*. Oxford University Press, Oxford, UK.

de Quervain, DJF, U Fischbacher, V Treyer, M Schelthammer, U Schnyder, A Buck, and E Fehr. 2004. The neural basis of altruistic punishment. *Science* **305**(5688):1254-1258.

Denant-Boemont, L, D Masclet, and CN Noussair. 2007. Punishment, counterpunishment and sanction enforcement in a social dilemma experiment. *Economic Theory* **33**(1):145-167.

Di Paolo, EA. 1999. A little more than kind and less than kin: The unwarranted use of kin selection in spatial models of communication. *Advances in Artificial Life, Proceedings* **1674**:504-513.

Dietz, T, E Ostrom, and PC Stern. 2003. The struggle to govern the commons. *Science* **302**(5652):1907-1912.

Dorogtsev, SN and JF Mendes. 2003. *Evolution of networks: from biological nets to the Internet and WWW*. Oxford University Press, Oxford.

Dreber, A, DG Rand, D Fudenberg, and MA Nowak. 2008. Winners don't punish. *Nature* **452**(7185):348-351.

Drogoul, A and J Ferber. 1994. Multi-agent simulation as a tool for studying emergent processes in societies. Pages 127-142 In *Simulating societies: The computer simulation of social phenomena*, edited by N Gilbert and J Doran. UCL Press Limited, London.

Duesenberry, JS. 1949. *Income, saving, and the theory of consumer behavior*. Harvard University Press, Cambridge, Massachusetts.

Dugatkin, LA. 1997a. *Cooperation among animals: an evolutionary perspective*. Oxford University Press, New York.

Dugatkin, LA. 1997b. Winner and loser effects and the structure of dominance hierarchies. *Behavioral Ecology* **8**(6):583-587.

Easterlin, RA. 2001. Income and happiness: Towards a unified theory. *Economic Journal* **111**(473):465-484.

Edmonds, B and D Hales. 2003. Replication, Replication and Replication: Some Hard Lessons from Model Alignment. *Journal of Artificial Societies and Social Simulation* **6**(4):http://jasss.soc.surrey.ac.uk/6/4/11.html.

Fehr, E and U Fischbacher. 2003. The nature of human altruism. *Nature* **425**(6960):785-791.

Fehr, E and U Fischbacher. 2004. Third-party punishment and social norms. *Evolution and Human Behavior* **25**(2):63-87.

Fehr, E, U Fischbacher, and S Gächter. 2002. Strong reciprocity, human cooperation, and the enforcement of social norms. *Human Nature-an Interdisciplinary Biosocial Perspective* **13**(1):1-25.

Fehr, E and S Gächter. 2000. Cooperation and punishment in public goods experiments. *American Economic Review* **90**(4):980-994.

Fehr, E and S Gächter. 2002. Altruistic punishment in humans. *Nature* **415**(6868):137-140.

Fehr, E and B Rockenbach. 2003. Detrimental effects of sanctions on human altruism. *Nature* **422**(6928):137-140.

Fewell, JH. 2003. Social insect networks. *Science* **301**(5641):1867-1870.

Flache, A. 2002. The rational weakness of strong ties: Failure of group solidarity in a highly cohesive group of rational agents. *Journal of Mathematical Sociology* **26**(3):189-216.

Flache, A and MW Macy. 1996. The weakness of strong ties: Collective action failure in a highly cohesive group. *Journal of Mathematical Sociology* **21**(1-2):3-28.

Flack, JC, M Girvan, FBM de Waal, and DC Krakauer. 2006. Policing stabilizes construction of social niches in primates. *Nature* **439**(7075):426-429.

Fon, V and F Parisi. 2005. Revenge and Retaliation. Pages 141-168 In *The Law and Economics of Irrational Behavior*, edited by F Parisi and VL Smith. Stanford University Press, Stanford, California.

Foster, KR and FLW Ratnieks. 2001. Convergent evolution of worker policing by egg eating in the honeybee and common wasp. *Proceedings of the Royal Society of London. Series B: Biological Sciences* **268**(1463):169-174.

Foster, KR, G Shaulsky, JE Strassmann, DC Queller, and CRL Thompson. 2004. Pleiotropy as a mechanism to stabilize cooperation. *Nature* **431**(7009):693-696.

Fowler, JH. 2005. Altruistic punishment and the origin of cooperation. *Proceedings of the National Academy of Sciences of the United States of America* **102**(19):7047-7049.

Fowler, JH, T Johnson, and O Smirnov. 2005. Egalitarian motive and altruistic punishment. *Nature* **433**(7021):E1.

Frank, RH. 2005a. Departures from Rational Choice: With and Without Regret. Pages 13-36 In *The Law and Economics of Irrational Behavior*, edited by F Parisi and VL Smith. Stanford University Press, Stanford, California.

Frank, RH. 2005b. Positional externalities cause large and preventable welfare losses. *American Economic Review* **95**(2):137-141.

Frank, SA. 1995. Mutual Policing and Repression of Competition in the Evolution of Cooperative Groups. *Nature* **377**(6549):520-522.

Freyberg-Inan, A. 2006. Just how small is this world really? An application of small world theory to the study of globalization. *Global Networks-a Journal of Transnational Affairs* **6**(3):221-244.

Fryxell, JM, A Mosser, ARE Sinclair, and C Packer. 2007. Group formation stabilizes predator-prey dynamics. *Nature* **449**(7165):1041-U1044.

Gardner, A and SA West. 2004. Cooperation and punishment, especially in humans. *American Naturalist* **164**(6):753-764.

Gintis, H. 2000. Strong reciprocity and human sociality. *Journal of Theoretical Biology* **206**(2):169-179.

Goodnight, CJ. 2005. Multilevel selection: the evolution of cooperation in non-kin groups. *Population Ecology* **47**(1):3-12.

Gould, RV. 1993. Collective Action and Network Structure. *American Sociological Review* **58**(2):182-196.

Granovetter, M. 1985. Economic-Action and Social-Structure: the Problem of Embeddedness. *American Journal of Sociology* **91**(3):481-510.

Griffin, WA. 2006. Agent-Based Modeling for the Theoretical Biologist. *Biological Theory* **1**(14):404-409.

Gürerk, Ö, B Irlenbusch, and B Rockenbach. 2006. The competitive advantage of sanctioning institutions. *Science* **312**(5770):108-111.

Hadjimichalis, C and R Hudson. 2006. Networks, regional development and democratic control. *International Journal of Urban and Regional Research* **30**(4):858-872.

Hagen, EH and P Hammerstein. 2006. Game theory and human evolution: A critique of some recent interpretations of experimental games. *Theoretical Population Biology* **69**(3):339-348.

Hagen, JB. 1992. *An Entangled Bank: The Origin of Ecosystem Ecology*. Rutgers University Press, New Brunswick, New Jersey.

Hamilton, WD. 1964. Genetical Evolution of Social Behaviour 1 & 2. *Journal of Theoretical Biology* **7**(1):1-52.

Harbaugh, R. 1996. Falling behind the Joneses: Relative consumption and the growth-savings paradox. *Economics Letters* **53**(3):297-304.

Hardin, G. 1968. Tragedy of Commons. *Science* **162**(3859):1243-1248.

Harrison, N and JD Singer. 2006. Complexity is more than systems theory. Pages 25-41 In *Complexity in World Politics: Concepts and Methods of a New Paradigm*, edited by N Harrison. State University of New York Press, Albany, New York.

Harrison, NE. 2006. *Complexity in world politics: concepts and methods of a new paradigm*. State University of New York Press, Albany, New York.

Hauert, C. 2001. Fundamental clusters in spatial 2x2 games. *Proceedings of the Royal Society of London. Series B: Biological Sciences* **268**(1468):761-769.

Hauert, C and M Doebeli. 2004. Spatial structure often inhibits the evolution of cooperation in the snowdrift game. *Nature* **428**(6983):643-646.

Heckathorn, DD. 1998. Collective action, social dilemmas and ideology. *Rationality and Society* **10**(4):451-479.

Hedrick, PW. 2005. *Genetics of Populations*. 3rd edition. Jones and Bartlett Publishers, Sudbury, Massachusetts.

Henrich, J and R Boyd. 2001. Why people punish defectors: Weak conformist transmission can stabilize costly enforcement of norms in cooperative dilemmas. *Journal of Theoretical Biology* **208**(1):79-89.

Henrich, J, R Boyd, S Bowles, C Camerer, E Fehr, H Gintis, and R McElreath. 2001. In search of Homo economicus: Behavioral experiments in 15 small-scale societies. *American Economic Review* **91**(2):73-78.

Herrmann, B, C Thöni, and S Gächter. 2008. Antisocial punishment across societies. *Science* **319**(5868):1362-1367.

Hirshleifer, J and JCM Coll. 1988. What Strategies Can Support the Evolutionary Emergence of Cooperation. *Journal of Conflict Resolution* **32**(2):367-398.

Hobbes, T. [1651] 1946. *Leviathan*. Basil Blackwell, Oxford.

Hodgson, GM. 2009. On the Institutional Foundations of Law. *Journal of Economic Issues* **43**(1):143-166.

Hölldobler, B and EO Wilson. 2009. *The superorganism : the beauty, elegance, and strangeness of insect societies*. 1st edition. W.W. Norton, New York.

Hopfensitz, A and E Reuben. 2005. The Importance of Emotions for the Effectiveness of Social Punishment. Discussion Paper No. 05-075/1. Tinbergen Institute, University of Amsterdam, Amsterdam.

Hopkins, E and T Kornienko. 2004. Running to keep in the same place: Consumer choice as a game of status. *American Economic Review* **94**(4):1085-1107.

Hui, PM, C Xu, and DF Zheng. 2007. Cooperation in evolutionary snowdrift game: Networking effects. *International Journal of Modern Physics B* **21**(23-24):4035-4040.

Ifti, M, T Killingback, and M Doebelic. 2004. Effects of neighbourhood size and connectivity on the spatial Continuous Prisoner's Dilemma. *Journal of Theoretical Biology* **231**(1):97-106.

Jackson, MO and A Watts. 2002. The evolution of social and economic networks. *Journal of Economic Theory* **106**(2):265-295.

Jaffe, K. 2004. Altruism, altruistic punishment and social investment. *Acta Biotheoretica* **52**(3):155-172.

James, J. 1987. Positional Goods, Conspicuous Consumption and the International Demonstration Effect Reconsidered. *World Development* **15**(4):449-462.

James, J. 2000. Do consumers in developing countries gain or lose from globalization? *Journal of Economic Issues* **34**(3):537-551.

Janssen, MA, editor. 2002. *Complexity and ecosystem management: the theory and practice of multi-agent systems*. Edward Elgar Publishing, Northampton, Massachusetts.

Jensen, K, J Call, and M Tomasello. 2007. Chimpanzees are rational maximizers in an ultimatum game. *Science* **318**(5847):107-109.

Johansson-Stenman, O and P Martinsson. 2006. Honestly, why are you driving a BMW? *Journal of Economic Behavior & Organization* **60**(2):129-146.

Jun, T and R Sethi. 2007. Neighborhood structure and the evolution of cooperation. *Journal of Evolutionary Economics* **17**(5):623-646.

Kaul, I and RU Mendoza. 2003. Advancing the Concept of Public Goods. Pages 78-111 In *Providing Global Public Goods: Managing Globalization*, edited by I Kaul, P Conceição, KL Goulven, and RU Mendoza. Oxford University Press, New York.

Kazemi, A and D Eek. 2008. Promoting Cooperation in Social Dilemmas via Fairness Norms and Group Goals. Pages 72-92 In *New Issues and Paradigms in Research on Social Dilemmas*, edited by A Biel, D Eek, T Gärling, and M Gustafsson. Springer, New York.

Killingback, T and M Doebeli. 1996. Spatial evolutionary game theory: Hawks and Doves revisited. *Proceedings of the Royal Society of London. Series B: Biological Sciences* **263**(1374):1135-1144.

Killingback, T and M Doebeli. 2002. The continuous prisoner's dilemma and the evolution of cooperation through reciprocal altruism with variable investment. *American Naturalist* **160**(4):421-438.

Killingback, T and E Studer. 2001. Spatial Ultimatum Games, collaborations and the evolution of fairness. *Proceedings of the Royal Society of London. Series B: Biological Sciences* **268**(1478):1797-1801.

Knell, M. 1999. Social comparisons, inequality, and growth. *Journal of Institutional and Theoretical Economics* **155**(4):664-695.

Ko, K, KJ Lee, and C Park. 2008. Rethinking Preferential Attachment Scheme: Degree centrality versus closeness centrality. *Connections* **28**(1):4-15.

Kummerli, R, C Colliard, N Fiechter, B Petitpierre, F Russier, and L Keller. 2007. Human cooperation in social dilemmas: comparing the Snowdrift game with the Prisoner's Dilemma. *Proceedings of the Royal Society of London. Series B: Biological Sciences* **274**(1628):2965-2970.

Lee, KH, CH Chan, PM Hui, and DF Zheng. 2008. Cooperation in *N*-person evolutionary snowdrift game in scale-free Barabási-Albert networks. *Physica A: Statistical Mechanics and Its Applications* **387**(22):5602-5608.

Leimar, O and P Hammerstein. 2001. Evolution of cooperation through indirect reciprocity. *Proceedings of the Royal Society of London. Series B: Biological Sciences* **268**(1468):745-753.

Liu, PT and GZ Sun. 2005. The international demonstration effect and the domestic division of labour: A simple model. *Pacific Economic Review* **10**(4):515-528.

Lusseau, D and MEJ Newman. 2004. Identifying the Role That Animals Play in Their Social Networks. *Proceedings of the Royal Society of London. Series B: Biological Sciences* **271**:S477-S481.

Luttmer, EFP. 2005. Neighbors as negatives: Relative earnings and well-being. *Quarterly Journal of Economics* **120**(3):963-1002.

Macy, MW and J Skvoretz. 1998. The evolution of trust and cooperation between strangers: A computational model. *American Sociological Review* **63**(5):638-660.

Marlowe, FW, JC Berbesque, A Barr, C Barrett, A Bolyanatz, JC Cardenas, J Ensminger, M Gurven, E Gwako, J Henrich, N Henrich, C Lesorogol, R McElreath, and D Tracer. 2008. More 'altruistic' punishment in larger societies. *Proceedings of the Royal Society of London. Series B: Biological Sciences* **275**(1634):587-590.

Marwell, G and PE Oliver. 1993. *Critical Mass in Collective Action*. Cambridge Publishing, Cambridge.

Marwell, G, PE Oliver, and R Prahl. 1988. Social Networks and Collective Action: a Theory of the Critical Mass III. *American Journal of Sociology* **94**(3):502-534.

Mas-Colell, A, MD Whinston, and JR Green. 1995. *Microeconomic Theory*. Oxford University Press, New York.

Masclet, D, C Noussair, S Tucker, and MC Villeval. 2003. Monetary and nonmonetary punishment in the voluntary contributions mechanism. *American Economic Review* **93**(1):366-380.

Maynard-Smith, J. 1982. *Evolution and the Theory of Games*. Cambridge University Press, Cambridge.

Maynard Smith, J and E Szathmáry. 1997. *The major transitions in evolution*. Oxford University Press, New York.

McAdams, RH. 1992. Relative Preferences. *Yale Law Journal* **102**(1):1-104.

McCormick, K. 1983. Duesenberry and Veblen: the Demonstration Effect Revisited. *Journal of Economic Issues* **17**(4):1125-1129.

Michod, RE. 1996. Cooperation and conflict in the evolution of individuality II: Conflict mediation. *Proceedings of the Royal Society of London. Series B: Biological Sciences* **263**:813-822.

Michod, RE. 1997. Cooperation and Conflict in the Evolution of Individuality I: Multilevel Selection of the Organism. *American Naturalist* **149**(4):607-645.

Minas, JS, A Scodel, D Marlowe, and H Rawson. 1960. Some descriptive aspects of two-person non-zero-sum games II. *Journal of Conflict Resolution* **4**(2):193-197.

Mitteldorf, J and DS Wilson. 2000. Population viscosity and the evolution of altruism. *Journal of Theoretical Biology* **204**(4):481-496.

Molm, LD. 1989a. An Experimental-Analysis of Imbalance in Punishment Power. *Social Forces* **68**(1):178-203.

Molm, LD. 1989b. Punishment Power - a Balancing Process in Power-Dependence Relations. *American Journal of Sociology* **94**(6):1392-1418.

Molm, LD. 1994. Is Punishment Effective? Coercive Strategies in Social-Exchange. *Social Psychology Quarterly* **57**(2):75-94.

Moon, IC and KM Carley. 2007. Modeling and simulating terrorist networks in social and geospatial. *IEEE Intelligent Systems* **22**(5):40-49.

Morrow, JD. 1994. *Game Theory for Political Scientists*. Princeton University Press, Princeton, New Jersey.

Mulder, MB. 1998. The demographic transition: are we any closer to an evolutionary explanation? *Trends in Ecology & Evolution* **13**(7):266-270.

Murdoch, JC and T Sandler. 1997a. Voluntary cutbacks and pretreaty behavior: The Helsinki protocol and sulfur emissions. *Public Finance Review* **25**(2):139-162.

Murdoch, JC and T Sandler. 1997b. The voluntary provision of a pure public good: The case of reduced CFC emissions and the Montreal Protocol. *Journal of Public Economics* **63**(3):331-349.

Nikiforakis, N. 2008. Punishment and counter-punishment in public good games: Can we really govern ourselves? *Journal of Public Economics* **92**(1-2):91-112.

North, DC. 2005. *Understanding the Process of Economic Change*. Princeton University Press, Princeton, New Jersey.

Nowak, M. 1990. Stochastic Strategies in the Prisoners-Dilemma. *Theoretical Population Biology* **38**(1):93-112.

Nowak, MA, S Bonhoeffer, and RM May. 1994. Spatial games and the maintenance of cooperation. *Proceedings of the National Academy of Sciences of the United States of America* **91**:4877-4881.

Nowak, MA and RM May. 1992. Evolutionary games and spatial chaos. *Nature* **359**(6398):826-829.

Nowak, MA, KM Page, and K Sigmund. 2000. Fairness versus reason in the Ultimatum Game. *Science* **289**(5485):1773-1775.

Nowak, MA and K Sigmund. 1998a. The dynamics of indirect reciprocity. *Journal of Theoretical Biology* **194**(4):561-574.

Nowak, MA and K Sigmund. 1998b. Evolution of indirect reciprocity by image scoring. *Nature* **393**(6685):573-577.

Nowak, MA and K Sigmund. 2005. Evolution of indirect reciprocity. *Nature* **437**(7063):1291-1298.

Ohtsuki, H, C Hauert, E Lieberman, and MA Nowak. 2006. A simple rule for the evolution of cooperation on graphs and social networks. *Nature* **441**(7092):502-505.

Okada, A. 2008. The second-order dilemma of public goods and capital accumulation. *Public Choice* **135**(3-4):165-182.

Oldero, KK. 2002. Collective Action, Inaction and the Global Commons. Pages 1-8. University of Zimbabwe, Harare.

Olfati-Saber, R, JA Fax, and RM Murray. 2007. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE* **95**(1):215-233.

Oliver, P. 1980. Rewards and Punishments as Selective Incentives for Collective Action: Theoretical Investigations. *American Journal of Sociology* **85**(6):1356-1375.

Oliver, P. 1984. If You Don't Do It, Nobody Else Will: Active and Token Contributors to Local Collective Action. *American Sociological Review* **49**(5):601-610.

Olson, M. 1965. *The Logic of Collective Action: Public Goods and the Theory of Groups*. Schocken Books, New York.

Opp, KD and C Gern. 1993. Dissident Groups, Personal Networks, and Spontaneous Cooperation - the East-German Revolution of 1989. *American Sociological Review* **58**(5):659-680.

Osborne, MJ. 2004. *An Introduction to Game Theory*. Oxford University Press, New York.

Ostrom, E. 1990. *Governing The Commons: The Evolution of Institutions for Collective Action*. Cambridge University Press, New York.

Ostrom, E. 1998. A behavioral approach to the rational choice theory of collective action. *American Political Science Review* **92**(1):1-22.

Ostrom, E, R Gardner, and J Walker. 1994. *Rules, Games, and Common-Pool Resources*. University of Michigan Press, Ann Arbor, Michigan.

Ostrom, E, J Walker, and R Gardner. 1992. Covenants with and without a Sword: Self-Governance Is Possible. *American Political Science Review* **86**(2):404-417.

Oswald, AJ. 1997. Happiness and Economic Performance. *Economic Journal* **66**(3):309-320.

Pacheco, JM and FC Santos. 2005. Network Dependence of the Dilemmas of Cooperation. Pages 90-100 In *Science of Complex Networks: From Biology to the Internet and WWW; CNET 2004*, edited by JF Mendes, SN Dorogtsev, A Povolotsky, FV Abreu, and JG Oliveira. American Institute of Physics, Melville, New York.

Page, KM, MA Nowak, and K Sigmund. 2000. The spatial ultimatum game. *Proceedings of the Royal Society of London. Series B: Biological Sciences* **267**(1458):2177-2182.

Panchanathan, K and R Boyd. 2004. Indirect reciprocity can stabilize cooperation without the second-order free rider problem. *Nature* **432**(7016):499-502.

Poirot, C. 2007. How can institutional economics be an evolutionary science? *Journal of Economic Issues* **41**(1):155-179.

Pollak, RA. 1976. Interdependent Preferences. *American Economic Review* **66**(3):309-320.

Price, GR. 1970. Selection and Covariance. *Nature* **227**(5257):520-521.

Queller, DC. 1992. Does Population Viscosity Promote Kin Selection. *Trends in Ecology & Evolution* **7**(10):322-324.

Rauscher, M. 1992. Keeping up with the Joneses: Chaotic Patterns in a Status Game. *Economics Letters* **40**(3):287-290.

Rayo, L and GS Becker. 2007. Evolutionary efficiency and happiness. *Journal of Political Economy* **115**(2):302-337.

Rees, WE. 2002. An ecological economics perspective on sustainability and prospects for ending poverty. *Population and Environment* **24**(1):15-46.

Reeve, HK and B Hölldobler. 2007. The emergence of a superorganism through intergroup competition. *Proceedings of the National Academy of Sciences of the United States of America* **104**(23):9736-9740.

Ricardo, D. 2001 [1817]. *On the principles of political economy and taxation*. Electric Book Co., London.

Richerson, PJ and R Boyd. 2005. *Not by genes alone: how culture transformed human evolution*. University of Chicago Press, Chicago.

Riolo, RL, MD Cohen, and R Axelrod. 2001. Evolution of cooperation without reciprocity. *Nature* **414**(6862):441-443.

Roth, AE, V Prasnikar, M Okunofujiwara, and S Zamir. 1991. Bargaining and Market Behavior in Jerusalem, Ljubljana, Pittsburgh, and Tokyo - an Experimental Study. *American Economic Review* **81**(5):1068-1095.

Rothstein, SI and R Pierotti. 1988. Distinctions among Reciprocal Altruism, Kin Selection, and Cooperation and a Model for the Initial Evolution of Beneficent Behavior. *Ethology and Sociobiology* **9**(2-4):189-209.

Sacks, MA, MJ Ventresca, and B Uzzi. 2001. Global institutions and networks - Contingent change in the structure of world trade advantage, 1965-1980. *American Behavioral Scientist* **44**(10):1579-1601.

Saijo, T and H Nakamura. 1995. The Spite Dilemma in Voluntary Contribution Mechanism Experiments. *Journal of Conflict Resolution* **39**(3):535-560.

Sandler, T. 1992. *Collective action: theory and applications*. The University of Michigan Press, Ann Arbor, Michigan.

Sandler, T. 1999. International Cooperation and the International Commons. *Duke Environmental Law & Policy Forum* **10**:131-145.

Sandler, T. 2004. *Global Collective Action*. Cambridge University Press, Cambridge.

Sanfey, AG, JK Rilling, JA Aronson, LE Nystrom, and JD Cohen. 2003. The neural basis of economic decision-making in the ultimatum game. *Science* **300**(5626):1755-1758.

Santos, FC and JM Pacheco. 2005. Scale-free networks provide a unifying framework for the emergence of cooperation. *Physical Review Letters* **95**(9/098104):1-4.

Santos, FC and JM Pacheco. 2006. A new route to the evolution of cooperation. *Journal of Evolutionary Biology* **19**(3):726-733.

Santos, FC, JM Pacheco, and T Lenaerts. 2006a. Cooperation prevails when individuals adjust their social ties. *PLoS Computational Biology* **2**(10):1284-1291.

Santos, FC, JF Rodrigues, and JM Pacheco. 2006b. Graph topology plays a determinant role in the evolution of cooperation. *Proceedings of the Royal Society of London. Series B: Biological Sciences* **273**(1582):51-55.

Sawyer, RK. 2005. *Social Emergence: Societies as Complex Systems*. Cambridge University Press, Cambridge, UK.

Schelling, TC. 1971. Dynamic Models of Segregation. *Journal of Mathematical Sociology* **1**(2):143-186.

Scodel, A, JS Minas, P Ratoosh, and M Lipetz. 1959. Some descriptive aspects of two-person non-zero-sum games I. *Journal of Conflict Resolution* **3**(2):114-119.

Sefton, M, R Shupp, and J Walker. 2002. The Effect of Rewards and Sanctions in Provision of Public Goods. Working paper W00-16. University of Nottingham, Nottingham, UK.

Selten, R. 2001. What is Bounded Rationality. Pages 13-36 In *Bounded rationality: the adaptive toolbox*, edited by G Gigerenzer and R Selten. MIT Press, Cambridge, Mass.

Serrano, MA and M Boguna. 2003. Topology of the world trade web. *Physical Review E* **68**(1).

Sharpless, NE and RA DePinho. 2005. Cancer: Crime and punishment. *Nature* **436**(7051):636-637.

Shinada, M, T Yamagishi, and Y Ohmura. 2004. False friends are worse than bitter enemies: "Altruistic" punishment of in-group members. *Evolution and Human Behavior* **25**(6):379-393.

Shutters, ST. 2008. Strong reciprocity, social structure, and the evolution of fair allocations in a simulated ultimatum game. *Computational and Mathematical Organization Theory* **15**(2):64-77.

Shutters, ST. 2009. Punishment, Rational Expectations, and Relative Payoffs in a Networked Prisoners Dilemma. In *Social Computing and Behavioral Modeling*, edited by H Liu, J Salerno, and MJ Young. Springer, Phoenix, Arizona.

Shutters, ST and BB Cutts. 2008. A Simulation Model of Cultural Consensus and Persistent Conflict. Pages 71-78 In *Proceedings of the Second International Conference on Computational Cultural Dynamics*, edited by VS Subrahamanian and A Kruglanski. AAAI Press, Menlo Park, California.

Silk, JB, SF Brosnan, J Vonk, J Henrich, DJ Povinelli, AS Richardson, SP Lambeth, J Mascaro, and SJ Schapiro. 2005. Chimpanzees are indifferent to the welfare of unrelated group members. *Nature* **437**(7063):1357-1359.

Skyrms, B. 2001. The Stag Hunt. Pacific Division of the American Philosophical Association: Presidential Address.

Skyrms, B. 2004. *The Stag Hunt and the Evolution of Social Structure*. Cambridge University Press, Cambridge.

Skyrms, B and R Pemantle. 2000. A dynamic model of social network formation. *Proceedings of the National Academy of Sciences of the United States of America* **97**(16):9340-9346.

Solnick, SJ and D Hemenway. 1998. Is more always better?: A survey on positional concerns. *Journal of Economic Behavior & Organization* **37**(3):373-383.

Spitzer, M, U Fischbacher, B Herrnberger, G Gron, and E Fehr. 2007. The neural signature of social norm compliance. *Neuron* **56**(1):185-196.

Sulloway, FJ. 1998. Darwinian Virtues. *The New York Review of Books* **45**(6).

Suzuki, Y and Y Toquenaga. 2005. Effects of information and group structure on evolution of altruism: analysis of two-score model by covariance and contextual analyses. *Journal of Theoretical Biology* **232**(2):191-201.

Takács, K, B Janky, and A Flache. 2008. Collective action and network change. *Social Networks* **30**(3):177-189.

Todd, PM. 2001. Fast and Frugal Heuristics for Environmentally Bounded. Pages 51-70 In *Bounded rationality: the adaptive toolbox*, edited by G Gigerenzer and R Selten. MIT Press, Cambridge, Mass.

Tomassini, M, E Pestelacci, and L Luthi. 2007. Social dilemmas and cooperation in complex networks. *International Journal of Modern Physics C* **18**(7):1173-1185.

Travisano, M and GJ Velicer. 2004. Strategies of microbial cheater control. *Trends in Microbiology* **12**(2):72-78.

Trivers, RL. 1971. Evolution of Reciprocal Altruism. *Quarterly Review of Biology* **46**(1):35-57.

UNEP. 2007. The Montreal Protocol on Substances That Deplete the Ozone Layer 2007: A Success in the Making. United Nations Environment Programme, Ozone Secretariat, Nairobi, Kenya.

USDL. *Whistleblower Protections*. http://www.dol.gov/compliance/laws/comp-whistleblower.htm United States Department of Labor, Office of Compliance Assistance Policy. Accessed 06-Mar-2009.

Varian, HR. 1974. Equity, Envy, and Efficiency. *Journal of Economic Theory* **9**:63-91.

Veblen, T. 1898. Why is economics not an evolutionary science? *Quarterly Journal of Economics* **12**(4):373-397.

Vogel, G. 2004. Behavioral evolution: The evolution of the golden rule. *Science* **303**(5661):1128-1131.

Wagner, UJ. 2001. The design of stable international environmental agreements: Economic theory and political economy. *Journal of Economic Surveys* **15**(3):377-411.

Ward, H. 2006. International linkages and environmental sustainability: The effectiveness of the regime network. *Journal of Peace Research* **43**(2):149-166.

Wasserman, S and K Faust. 1994. *Social Network Analysis: Methods and Applications*. Cambridge University Press, New York.

Watts, DJ and SH Strogatz. 1998. Collective dynamics of 'small-world' networks. *Nature* **393**(6684):440-442.

Weirich, P. 1998. *Equilibrium and Rationality*. Cambridge University Press, Cambridge, UK.

West, SA, AS Griffin, and A Gardner. 2007. Social semantics: altruism, cooperation, mutualism, strong reciprocity and group selection. *Journal of Evolutionary Biology* **20**(2):415-432.

West, SA, AS Griffin, and A Gardner. 2008. Social semantics: how useful has group selection been? *Journal of Evolutionary Biology* **21**(1):374-385.

Williams, GC. 1966. *Adaptation and Natural Selection: A Critique of Some Current Evolutionary Thought*. Princeton University Press, Princeton, New Jersey.

Wilson, DS. 1975. Theory of Group Selection. *Proceedings of the National Academy of Sciences of the United States of America* **72**(1):143-146.

Wilson, DS. 2007. Social semantics: towards a genuine pluralism in the study of behaviour. *Journal of Evolutionary Biology* **21**(1):368-373.

Wilson, DS and E Sober. 1994. Re-introducing Group Selection to the Human Behavioral Sciences. *Behavioral and Brain Sciences* **17**:585-654.

Wilson, DS and EO Wilson. 2007. Rethinking the theoretical foundation of sociobiology. *Quarterly Review of Biology* **82**(4):327-348.

Wilson, EO. 2000. *Sociobiology: the new synthesis*. Belknap Press of Harvard University Press, Cambridge, Massachusetts.

Wilson, EO. 2005. Kin selection as the key to altruism: Its rise and fall. *Social Research* **72**(1):159-166.

Wilson, EO and B Hölldobler. 2005. Eusociality: Origin and consequences. *Proceedings of the National Academy of Sciences of the United States of America* **102**(38):13367-13371.

Zehavi, A and A Zahavi. 1997. *The handicap principle : a missing piece of Darwin's puzzle*. Oxford University Press, New York.

APPENDIX A

CALCULATION OF APPROXIMATE TRANSITION VALUES

IN TABLES 3.5 AND 5.7

Let $\bar{x}$ = a population's mean ending contribution (or offer when using the Ultimatum Game) at the end of a simulation. Approximate transition values in Figures 3.2 and 5.2 (Tables 3.5 and 5.7) were calculated using simple linear interpolation between the last *M* value at which $\bar{x} \approx 0$ and the first *M* value at which $\bar{x} \approx 1$, and assuming that the transition takes place at the midpoint between full cooperation and full defection ($\bar{x} = 0.5$). Accordingly, the point-slope formula for a line is used and solved for $y = 0.5$

[A.1] $$(y - y_1) = m(x - x_1), \text{ where } m = \frac{(y_2 - y_1)}{(x_2 - x_1)}.$$

Using simulation notation of independent variable *M* (punishment multiplier), response variable $\bar{x}$ (mean ending contribution), and letting *t* = the transition value of *M*, [A.1] is rewritten as

[A.2] $$(0.5 - \bar{x}_1) = \frac{(\bar{x}_2 - \bar{x}_1)}{(M_2 - M_1)}(t - M_1)$$

and solving for *t* yields

[A.3] $$t = \frac{(M_2 - M_1)}{(\bar{x}_2 - \bar{x}_1)}(0.5 - \bar{x}_1) + M_1.$$

APPENDIX B

SIMULATION PROGRAM CODE

The program for simulations used for this dissertation was developed over several years using Java within the Eclipse software development kit (SDK). Though written in an object-oriented language, anyone experienced in object-oriented programming will likely be appalled at the code, which is written almost entirely in procedural style. However, while it does not take full advantage of Java's object-oriented capabilities, the program has been completely adequate for the purposes of this research. The program, including its many subroutines, is reproduced here in its entirety so that others my scrutinize its validity or reproduce its results.

The program is presented in three parts or packages: code specific to the simulated CPD used in chapters 3, 4, and 6, code specific to the simulated ultimatum game used in chapter 5, and a large set of utilities used in both simulations (and designed for use in future simulations). A table of contents is provided below to help navigate the lengthy program. For the sake of clarity and brevity the program is presented in landscape format and with smaller font than the rest of the dissertation.

APPENDIX B: TABLE OF CONTENTS

Page

```
package prisonersDilemma;

import java.util.*;

/*
By Shade T. Shutters, School of Life Sciences, Arizona State University.
Based on a previous model by the author in which a spatially explicit population played
the ultimatum game.

Trait array elements of an agent p (Trait[p][element]):
1 = contribution
2 = publicGoodMultiplier (in this dissertation this is constant for the population)
3 = punishLimit
4 = punishAmount
5 = rewardLimit
6 = rewardAmount / retaliationAmount when used with punishment

for stats, the following elements are included:
7 = payoff
8 = number of contributions in a generation that were punished by the 3rd party observer

Neighbors array:
the element neighbor[p][0] stores the total number of neighbors connected to agent p (degree of p)
*/

import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.text.DecimalFormat;
import java.util.Arrays;
import simulationTools.MatrixTools;
import simulationTools.NetworkTools;

public class PDGame implements PDGameParameters {

    public static void main(String[] args) {
        double[] simSum = new double[9];
```

```
            double[] simAvg = new double[9];
            double[] genSum = new double[9];
            double[] genAvg = new double[9];
            DecimalFormat dataOut = new DecimalFormat("#0.0000");
//          Random rnd = new Random();

            try {
                  BufferedWriter genData = new BufferedWriter(new FileWriter("output\\Game_data_gens.txt"));
                  genData.write("Network,RecFactor,PGMultiplier,Run,Generation,Offer,RewLimit,PunLimit,Payoff,PunishRate\r\n");
                  BufferedWriter runData = new BufferedWriter(new FileWriter("output\\Game_data_runs.txt"));
                  runData.write("Network,NetworkDegree,NetworkDegreeVariance,RecFactor,Run,Offer,RewLimit,PunLimit,Payoff,PunishRate\r\n");
                  BufferedWriter simData = new BufferedWriter(new FileWriter("output\\Game_data_sims.txt"));
                  simData.write("Network,RecFactor,PGMultiplier,Offer,RewLimit,PunLimit,Payoff,PunishRate\r\n");
                  BufferedWriter popData = new BufferedWriter(new FileWriter("output\\Population_data_runs.txt"));
                  popData.write("Network,RecFactor,Run,Generation,Agent,Neighbors,Offer,PGMultiplier,PunishLimit,PunishAmount,Payoff,TimesPunished\r\n");

// Increment parameter of interest=======
                  for (int networkType=BEGIN_NEIGHBORHOOD; networkType<=END_NEIGHBORHOOD; networkType++){
                        int population = NetworkTools.getPopulationSize(networkType, THREED_ROWS, THREED_COLS, DEPTH, ROWS, COLS, NODES);
                        double[][] trait = new double [population+1][9];
                        double[][] childTrait = new double [population+1][9];
// Begin a new simulation====
                        for (double multiplier=BEGIN_RM; multiplier<=END_RM; multiplier += RM_INCREMENT){
                              Arrays.fill(simSum,0);
//Generate a new network===========
                              for (int r=1; r<=RUNS; r++){
                                    int[][] adjMatrix = NetworkTools.getAdjacencyMatrix(population, networkType,
                                    THREED_ROWS, THREED_COLS, DEPTH, ROWS, COLS, NODES, RANGE, GRAPH_DENSITY,
                                                PROB_REWIRE, SEED_NODES, LINKS_PER_NODE, SMALL_WORLD_RADIUS, FIXED_DEGREE);
                                    int[][] neighbors = NetworkTools.getNeighbors(adjMatrix);
                                    double networkDegree = NetworkTools.getNetworkDegree(neighbors);
                                    double networkDegreeVariance = NetworkTools.getNetworkDegreeVariance(neighbors);

// Begin a new run=================
                                    trait = MatrixTools.fillRandom(trait);
                                    for (int a=1; a<=population; a++){
                                          trait[a][2]=PG_MULTIPLIER;                         // symmetric agents, all have same multiplier
```

```
		}

// Begin a new generation====
		for (int g=0; g<GENERATIONS; g++){
			Arrays.fill(genSum,0);
			Arrays.fill(genAvg,0);
			for (int a=1; a<=population; a++){
				trait[a][8] = 0;
				if (RESET_FITNESS){
					trait[a][7] = 0;
				}
			}
			childTrait = MatrixTools.fillConstant(childTrait, 0);
// Game-play Routine=========
			trait = PDGameRoutines.getGamePayoffs(population, trait, neighbors, multiplier);
// Reproduction Routine======
			childTrait = PDGameRoutines.getChildren(population, trait, neighbors);
// Generation Stats==========
			for (int a=1; a<=population; a++){
				for (int t=1; t<=8; t++){
					genSum[t] += trait[a][t];
				}
			}
			for (int t=1; t<=7; t++){
				genAvg[t] = genSum[t] / (population);
			}
			genAvg[8] = genSum[8] / (population * PAIRINGS);			// punishment rate
			if (DETAILTOSCREEN){
				System.out.println("Network: " + networkType + ", Generation: " + g + ", RM: " + multiplier + ", Pairings: " +
				dataOut.format(population * PAIRINGS) + ", Punishments: " + dataOut.format(genSum[8]) + ", Punish Pct: " +
				dataOut.format(genAvg[8]));
			}
			if (REPORT_LEVEL == 3){
				if (TEXT_FILE_OUT){
					genData.write(networkType + "," +		multiplier + "," + PG_MULTIPLIER + "," + r + "," + g + "," +
					dataOut.format(genAvg[1]) + "," +		dataOut.format(genAvg[5]) + "," + dataOut.format(genAvg[3]) + "," +
					dataOut.format(genAvg[7]) + "," +		dataOut.format(genAvg[8]) + "\r\n");
```

```
                if (AGENT_DATA_ALL_GENS){
                    for (int a=1; a<=population; a++){
                        popData.write(networkType + "," + multiplier + "," + r + "," + g + "," + a + "," + neighbors[a][0] + "," +
                        dataOut.format(trait[a][1]) + "," + dataOut.format(trait[a][2])  + "," + dataOut.format(trait[a][3])  + "," +
                        dataOut.format(trait[a][4])  + "," + dataOut.format(trait[a][7])  + "," + dataOut.format(trait[a][8]) + "\r\n");
                    }
                }
            }
        }
// Run Stats (Final Generation stats)
        if (g == GENERATIONS-1 && REPORT_LEVEL >1){
            runData.write(networkType + "," + networkDegree + "," + networkDegreeVariance + "," + multiplier + "," + r + "," +
            dataOut.format(genAvg[1]) + "," + dataOut.format(genAvg[5]) + "," + dataOut.format(genAvg[3]) + "," +
            dataOut.format(genAvg[7]) + "," + dataOut.format(genAvg[8]) + "\r\n");

            if (AGENT_DATA_ALL_GENS == false){
                for (int a=1; a<=population; a++){
                    popData.write(networkType + "," + multiplier + "," + r + "," + g + "," + a + "," + neighbors[a][0] + "," +
                    dataOut.format(trait[a][1]) + "," + dataOut.format(trait[a][2])  + "," + dataOut.format(trait[a][3])  + "," +
                    dataOut.format(trait[a][4])  + "," + dataOut.format(trait[a][7])  + "," + dataOut.format(trait[a][8]) + "\r\n");
                }
            }
        }
// Birth Routine=============
        for (int a=1; a<=population; a++){
            for (int t=1; t<=6; t++){
                trait[a][t] = childTrait[a][t];
            }
            for (int t=7; t<=8; t++){
                trait[a][t] = 0;
            }
        trait[a][2] = PG_MULTIPLIER;
        }
    }
// Simulation Stats=========
    for (int t=1; t<=8; t++){
        simSum[t] += genAvg[t];
```

```
                }
            }
            for (int t=1; t<=8; t++){
                simAvg[t] = simSum[t] / RUNS;
            }
            if (TEXT_FILE_OUT){
                simData.write(networkType + "," + multiplier + "," + PG_MULTIPLIER + "," + dataOut.format(simAvg[1]) + "," +
                dataOut.format(simAvg[5]) + "," + dataOut.format(simAvg[3]) + "," + dataOut.format(simAvg[7]) + dataOut.format(simAvg[8]) +
                "\r\n");
            }
        }
    }
// Shutdown=================
    genData.close();
    runData.close();
    simData.close();
    popData.close();
  } catch (IOException ioe){
    System.out.println("Houston, we have a read/write problem.\n");
  } catch (Exception e){
    System.out.println("Houston, we have a database problem.\n");
  }
 }
}
```

```
package prisonersDilemma;
import java.util.*;
public class PDGameRoutines implements PDGameParameters{

	public static double[][] getChildren(int population, double[][] trait, int[][] neighbors){
		double[][] childTrait = new double[population+1][7];
		switch (REPRODUCTION_METHOD){

			case 1:
				childTrait = compareRandomNeighbor(population, trait, neighbors);
				break;

			case 2:
				childTrait = compareBestNeighbor(population, trait, neighbors);
				break;
		}
		return childTrait;
	}

	public static double[][] compareRandomNeighbor(int population, double[][] trait, int[][] neighbors){

		double[][] childTrait = new double[population+1][7];
		Random rnd = new Random();
		for (int a=1; a<=population; a++){
			int mate = 0;
			if (LOCAL_MATING){
				mate = neighbors[a][rnd.nextInt(neighbors[a][0])+1];
			} else {
				while (mate == a){
					mate = rnd.nextInt(population+1);
				}
			}
			if (trait[a][7] < trait[mate][7]){
				for (int t=1; t<=6; t++){childTrait[a][t] = trait[mate][t];}
			} else if (trait[a][7] > trait[mate][7]){
				for (int t=1; t<=6; t++){childTrait[a][t] = trait[a][t];}
			} else if (rnd.nextBoolean() == true){
```

```
                for (int t=1; t<=6; t++){childTrait[a][t] = trait[mate][t];}
            } else {
                for (int t=1; t<=6; t++){childTrait[a][t] = trait[a][t];}
            }
            for (int t=1; t<=6; t++){
                if (rnd.nextDouble() <= MUTATION_RATE){
                    childTrait[a][t] = childTrait[a][t] + (rnd.nextGaussian() / 100.);
                    if (childTrait[a][t] < 0.){
                        childTrait[a][t] = 0.;
                    }
                    if (childTrait[a][t] > 1.){
                        childTrait[a][t] = 1.;
                    }
                }
            }
        }
        return childTrait;
}

public static double[][] compareBestNeighbor(int population, double[][] trait, int[][] neighbors){

        double[][] childTrait = new double[population+1][7];
        Random rnd = new Random();
        for (int a=1; a<=population; a++){
            int mate = 0;
            double maxPayoff = -10000;
            for (int n=1; n<= neighbors[a][0]; n++){
                if (trait[n][7] > maxPayoff){
                    maxPayoff = trait[n][7];
                    mate = n;
                }
            }
            if (trait[a][7] < trait[mate][7]){
                for (int t=1; t<=6; t++){childTrait[a][t] = trait[mate][t];}
            } else if (trait[a][7] > trait[mate][7]){
                for (int t=1; t<=6; t++){childTrait[a][t] = trait[a][t];}
            } else if (rnd.nextBoolean() == true){
```

```
                for (int t=1; t<=6; t++){childTrait[a][t] = trait[mate][t];}
            } else {
                for (int t=1; t<=6; t++){childTrait[a][t] = trait[a][t];}
            }
            for (int t=1; t<=6; t++){
                if (rnd.nextDouble() <= MUTATION_RATE){
                    childTrait[a][t] = childTrait[a][t] + (rnd.nextGaussian() / 100.);
                    if (childTrait[a][t] < 0.){childTrait[a][t] = 0.;}
                    if (childTrait[a][t] > 1.){childTrait[a][t] = 1.;}
                }
            }
        }
        return childTrait;
    }

    public static double[][] getGamePayoffs(int population, double[][] trait, int[][] neighbors, double multiplier){

        switch (INTERACTION_METHOD){

            case 1:
                trait = playRandomNeighbors(population, trait, neighbors, multiplier);
                break;

            case 2:
                trait = playAllNeighborsSequentially(population, trait, neighbors, multiplier);
                break;

            case 3:
                trait = playAllNeighborsSimultaneously(population, trait, neighbors, multiplier);
                break;
        }
        return trait;
    }

    public static double[][] playRandomNeighbors(int population, double[][]         trait, int[][] neighbors, double multiplier){

        Random rnd = new Random();
```

```
		for (int initiator=1; initiator<=population; initiator++){
			for (int p=1; p<=PAIRINGS; p++){
				int partner = neighbors[initiator][rnd.nextInt(neighbors[initiator][0])+1];
				int initiatorObserver = neighbors[initiator][rnd.nextInt(neighbors[initiator][0])+1];
				int partnerObserver = neighbors[partner][rnd.nextInt(neighbors[partner][0])+1];
				int observerObserver = neighbors[initiatorObserver][rnd.nextInt(neighbors[initiatorObserver][0])+1];

				// play CPD
				double pool = trait[initiator][1] + trait[partner][1];
				trait[initiator][7] += (1 - trait[initiator][1]) + (pool * trait[initiator][2]);
				trait[partner][7] += (1 - trait[partner][1]) + (pool * trait[partner][2]);

				// do reciprocity
				if (multiplier != 0){
					doReciprocity(trait, multiplier, initiator, initiatorObserver);
					if (OBSERVE_ALL){
						doReciprocity(trait, multiplier, partner, partnerObserver);
					}
					if (PUNISH_OBSERVER){
						doSecondOrderReciprocity(trait, multiplier, initiatorObserver, observerObserver);
					}
				}
			}
		}
		return trait;
	}

	public static double[][] playAllNeighborsSequentially(int population, double[][] trait, int[][] neighbors, double multiplier){

		for (int initiator=1; initiator<=population; initiator++){
			for (int p=1; p<=neighbors[initiator][0]; p++){
				int partner = neighbors[initiator][p];
//				int initiatorObserver = neighbors[initiator][rnd.nextInt(neighbors[initiator][0])+1];
//				int observerObserver = neighbors[initiatorObserver][rnd.nextInt(neighbors[initiatorObserver][0])+1];

				// play CPD
				double pool = trait[initiator][1] + trait[partner][1];
```

```
				trait[initiator][7] += (1 - trait[initiator][1]) + (pool * trait[initiator][2]);
				trait[partner][7] += (1 - trait[partner][1]) + (pool * trait[partner][2]);

				// do reciprocity
				if (multiplier != 0){
					doReciprocity(trait, multiplier, initiator, partner);
					if (OBSERVE_ALL){
						doReciprocity(trait, multiplier, partner, initiator);
					}
				}
			}
		}
		return trait;
	}

	public static double[][] playAllNeighborsSimultaneously(int population, double[][] trait, int[][] neighbors, double multiplier){

		for (int a=1; a<=population; a++){
			double pool = trait[a][1];
			for (int p=1; p<=neighbors[a][0]; p++){
				int partner = neighbors[a][p];
				pool += trait[partner][1];
			}
			trait[a][7] += (1 - trait[a][1]) + (pool * trait[a][2]);
			for (int p=1; p<=neighbors[a][0]; p++){
				int partner = neighbors[a][p];
				trait[partner][7] += (1 - trait[partner][1]) + (pool * trait[partner][2]);
			}

			// do reciprocity
			for (int p=1; p<=neighbors[a][0]; p++){
				int partner = neighbors[a][p];
				trait = doReciprocity(trait, multiplier, a, partner);
			}
			if (OBSERVE_ALL){
				for (int p=1; p<=neighbors[a][0]; p++){
					int partner = neighbors[a][p];
```

```
                for (int n=1; n<=neighbors[partner][0]; n++){
                    int partnersPartner = neighbors[partner][n];
                    trait = doReciprocity(trait, multiplier, partner, partnersPartner);
                }
            }
        }
    }
    return trait;
}

public static double[][] doReciprocity(double[][] trait, double multiplier, int player, int observer){

    if (INSTITUTION == 1 || INSTITUTION == 3){
        if (trait[player][1] < trait[observer][3]){
            trait[player][7] -= (multiplier * trait[observer][4]);
            trait[observer][7] -= trait[observer][4];
            trait[player][8]++;
            if (RETALIATE){
                trait = doRetaliation(trait, multiplier, player, observer);
            }
        }
    }

    if (INSTITUTION == 2 || INSTITUTION == 3){
        if (trait[player][1] > trait[observer][5]){
            trait[player][7] += (multiplier * trait[observer][6]);
            trait[observer][7] -= trait[observer][6];
        }
    }
    return trait;
}

public static double[][] doSecondOrderReciprocity(double[][] trait, double multiplier, int observer, int observersObserver){

    if (INSTITUTION == 1 || INSTITUTION == 3){
        if (trait[observer][3] < trait[observersObserver][3]){
            trait[observer][7] -= (multiplier *                         trait[observersObserver][4]);
```

```
                    trait[observersObserver][7] -= trait[observersObserver][4];
//                  trait[observer][8]++;
                    if (RETALIATE){
                        trait = doRetaliation(trait, multiplier, observer, observersObserver);
                    }
                }
            }

            if (INSTITUTION == 2 || INSTITUTION == 3){
                if (trait[observer][5] > trait[observersObserver][5]){
                    trait[observer][7] += (multiplier * trait[observersObserver][6]);
                    trait[observersObserver][7] -= trait[observersObserver][6];
                }
            }
            return trait;
        }

        public static double[][] doRetaliation(double[][] trait, double multiplier, int originalPunishee, int originalPunisher){

            switch (RETALIATE_METHOD){

                case 1:   // the amount an agent retaliates is taken from the agent's punish_amount trait
                    trait[originalPunisher][7] -= (multiplier * trait[originalPunishee][4]);
                    trait[originalPunishee][7] -= trait[originalPunishee][4];
                    break;

                case 2: // the amount an agent retaliates is taken from an independent retaliate_amount trait
                    trait[originalPunisher][7] -= (multiplier * trait[originalPunishee][6]);
                    trait[originalPunishee][7] -= trait[originalPunishee][6];

            }
            return trait;
        }
}
```

```
package prisonersDilemma;

interface PDGameParameters{

// general parameters
    static final double PG_MULTIPLIER = 0.75;           // must be 1/n < PGM < 1, where n = number of participants per game (2 in PD)
    static final int INSTITUTION = 1;                   // 1 = punish only, 2 = reward only, 3 = punish & reward, not (1, 2 or 3) = no institution
    static final int RUNS = 1;                          // number of replications per parameter set
    static final int GENERATIONS = 10000;               // number of generations of game play per run
    static final double MUTATION_RATE = 0.1;            // prob of adding gaussian noise with mean = 0 stdv = 0.01 to a trait during reproduction

// parameters covering general game play
    static final int PAIRINGS = 3;                      // number of interactions per generation (with interaction method 1)
    static final int INTERACTION_METHOD = 1;            // 1 = pair with randome neighbors sequentially
                                                        // 2 = pair with all neighbors sequentially
                                                        // 3 = play all neighbors simultaneously (public good game)

// parameters covering punishment behavior
    static final boolean OBSERVE_ALL = true;            // only the initiator subject to observation or all players contributing to the pool?
    static final boolean PUNISH_OBSERVER = false;       // when false, only player has an observer, A.
                                                        // when true, each observer, A, has its own observer, B. If player's contribution was below
                                                        // B's punish threshold and A did not punish player, then B punishes A for not punishing player

// parameters covering retaliation
    static final boolean RETALIATE = false;             // punishee can inflict punishment on the agent which initially sanctioned the punishee?
    static final int RETALIATE_METHOD = 2;              // 1 = the amount to retaliate is the same as punish amount
                                                        // 2 = the amount to retaliate is an independent, evolving amount

// parameters coving reproduction
    static final boolean RESET_FITNESS = true;          // set fitness to 0 after each generation?
    static final boolean LOCAL_MATING = true;           // get mate from neighbors (true) or from anywhere (false)
    static final int REPRODUCTION_METHOD = 1;           // 1 = compare to 1 random neighbor
                                                        // 2 = compare to best neighbor

// parameter sweeps
// NEIGHBORHOOD_TYPES:    1 = 3D Moore, 2 = 3D von Neumann, 3 =       2D Moore, 4 = 2D hexagonal,
//                        5 = 2D von Neumann, 6 = linear (1D), 7      = complete graph, 8 = random graph,
```

```
//                                          9 = small-world graph, 10 = scale-free graph, 11 = random regular graph
    static final int BEGIN_NEIGHBORHOOD  = 9;        // starting neighborhood for parameter
    static final int END_NEIGHBORHOOD = 9;           // ending neighborhood for simulation loop
    static final double BEGIN_RM = 1.5;              // starting value of reciprocity multiplier
    static final double END_RM = 1.5;                // ending value of reciprocity multiplier
    static final double RM_INCREMENT = 1;            // value to increment reciprocity multiplier during sweep

// parameters specific to 2D spatially explicit neighborhoods
    static final int ROWS = 20;                      // rows & cols must be even numbers for hexagonal lattice (neighborhood_type 4)
    static final int COLS = 20;

// parameters specific to rings
    static final int RANGE = 2;                      // number of links away from an agent to include as neighbors

// parameters specific to 3D spatially explicit neighborhoods
    static final int THREED_ROWS = 8;
    static final int THREED_COLS = 8;
    static final int DEPTH = 8;

// parameters specific to network structures
    static final int NODES = 400;                    // number of agents in non-spacially explict neighborhoods
    static final double GRAPH_DENSITY = 0.005;       // probability of being connected to a member of population in a random graph
    static final double PROB_REWIRE = 0.05;          // probability of rewiring an edge of a ring substrate to produce a Watts-Strogatz small-world network
    static final int SEED_NODES = 3;                 // number of initial nodes in scale free network
    static final int SMALL_WORLD_RADIUS = 2;         // neighbor radius of ring stubstrate for small world networks
    static final int LINKS_PER_NODE = 2;             // number of links per new node in a scale free network
    static final int FIXED_DEGREE = 4;               // degree of a random regular graph

// output parameters
    static final boolean DATABASE_OUT = false;       // export simulation results to database?
    static final boolean TEXT_FILE_OUT = true;       // export simulation results to text files?
    static final int REPORT_LEVEL = 3;               // 3 = generation level, 2 = run level, 1 = simulation level
    static final boolean AGENT_DATA_ALL_GENS = true; // true: output agent data after every generation, false: only after final generation
    static final boolean DETAILTOSCREEN = false;     // print generation detail on screen?
```

```
package ultimatumGame;

/*
By Shade T. Shutters, School of Life Sciences, Arizona State University.
Original Implementation: Feb-2005.
Included routine to allow altruistic punishment by neighbors: Mar-2005.
Modified routice to allow either punishment or reward or both: Mar-2006.
Modified to allow a selection of institutional arrangement: May-2006.
Modified to play either the ultimatum game or dictator game: June-2006.
Modified to allow a selection of neighborhood structure: June-2006.
Modified to do parameter sweeps of multiplier and neighborhoods. Also modified so that
the reciprocity routine is integrated with the game play routine. This will allow a further
modification in which an observer may punish the acceptor for receiving a large payoff: Nov-2007.

Trait elements:
1 = offer
2 = accept
3 = punishLimit against offerer
4 = punishAmount against offerer
5 = punishLimit against acceptor
6 = punishAmount against acceptor

for stats, a 7th element is included - Payoffs
 */

import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.text.DecimalFormat;
import java.util.Arrays;
import java.util.Random;

import simulationTools.*;

public class UltimatumGame implements UltimatumGameParameters {

	public static void main(String[] args) {
```

```
    double[] simSum = new double[8];
    double[] simAvg = new double[8];
    double[] genSum = new double[8];
    double[] genAvg = new double[8];
    Random rnd = new Random();
    DecimalFormat dataOut = new DecimalFormat("#0.0000");

    try {
        BufferedWriter genData = new BufferedWriter(new FileWriter("output\\Game_data_gens.txt"));
        genData.write("Game,Institution,RecFactor,Run,Generation,Offer,RewLimit,PunLimit,Payoff\r\n");
        BufferedWriter runData = new BufferedWriter(new FileWriter("output\\Game_data_runs.txt"));
        runData.write("Game,Institution,RecFactor,Run,Offer,RewLimit,PunLimit,Payoff\r\n");
        BufferedWriter simData = new BufferedWriter(new FileWriter("output\\Game_data_sims.txt"));
        simData.write("Game,Institution,RecFactor,Offer,RewLimit,PunLimit,Payoff\r\n");

// Increment parameter of interest=======
        for (int networkType=BEGIN_NEIGHBORHOOD; networkType<=END_NEIGHBORHOOD; networkType++){
            int population = NetworkTools.getPopulationSize(networkType, THREEDROWS, THREEDCOLS, DEPTH, ROWS, COLS, NODES);
            double[][] trait = new double [population+1][7];
            double[][] childTrait = new double [population+1][7];
            double[] payoff = new double[population+1];
            String networkName = NetworkTools.getNetworkName(networkType);

// Begin a new simulation====
            for (double multiplier=BEGIN_RM; multiplier<=END_RM; multiplier += RM_INCREMENT){
                Arrays.fill(simSum,0);

// Begin a new run===========
                int[][] adjMatrix = NetworkTools.getAdjacencyMatrix(population, networkType, THREEDROWS,
                        THREEDCOLS, DEPTH, ROWS, COLS, NODES, RANGE, GRAPH_DENSITY, PROB_REWIRE,
                        SEED_NODES, LINKS_PER_NODE, SMALL_WORLD_RADIUS, FIXED_DEGREE);
                int[][] neighbors = NetworkTools.getNeighbors(adjMatrix);
                double networkDegree = NetworkTools.getNetworkDegree(neighbors);
                for (int r=1; r<=RUNS; r++){
                    for (int a=1; a<=population; a++){
                        for (int t=1; t<=6; t++){
```

```
                    trait[a][t] = rnd.nextDouble();
                }
            }

// Begin a new generation====
            for (int g=1; g<=GENERATIONS; g++){
                Arrays.fill(genSum,0);
                Arrays.fill(genAvg,0);
                Arrays.fill(payoff,0);
                for (int a=1; a<=population; a++){
                    for (int t=1; t<=6; t++){
                        childTrait[a][t] = 0;
                    }
                }

// Game-play & Punishment Routine=========
                for (int a=1; a<=population; a++){
                    for (int p=1; p<=PAIRINGS; p++){
                        int partner = neighbors[a][rnd.nextInt(neighbors[a][0])+1];
                        double agentPayoff = 0;
                        double partnerPayoff = 0;
                        if ((trait[a][1] >= trait[partner][2]) || (GAMETYPE == 1)){
                            agentPayoff = (1 - trait[a][1]);
                            partnerPayoff = trait[a][1];
                            payoff[a] += agentPayoff;
                            payoff[partner] += partnerPayoff;
                        }
                        if (multiplier != 0){
                            int observer = neighbors[a][rnd.nextInt(neighbors[a][0])+1];
                            if (trait[a][1] < trait[observer][3]){
                                payoff[a] -= (multiplier * trait[observer][4]);
                                payoff[observer] -= trait[observer][4];
                            }
                        }
                    }
                }
```

```
// Reproduction Routine=====
			for (int a=1; a<=population; a++){
				int mate = neighbors[a][rnd.nextInt(neighbors[a][0])+1];
				if (payoff[a] < payoff[mate]){
					for (int t=1; t<=6; t++){childTrait[a][t] = trait[mate][t];}
				} else if (payoff[a] > payoff[mate]){
					for (int t=1; t<=6; t++){childTrait[a][t] = trait[a][t];}
				} else if (rnd.nextBoolean() == true){
					for (int t=1; t<=6; t++){childTrait[a][t] = trait[mate][t];}
				} else {
					for (int t=1; t<=6; t++){childTrait[a][t] = trait[a][t];}
				}
				for (int t=1; t<=6; t++){
					if (rnd.nextDouble() <= MUTATIONRATE){
						childTrait[a][t] = childTrait[a][t] + (rnd.nextGaussian() / 100.);
						if (childTrait[a][t] < 0.){childTrait[a][t] = 0.;}
						if (childTrait[a][t] > 1.){childTrait[a][t] = 1.;}
					}
				}
			}

// Generation Stats=========
			for (int a=1; a<=population; a++){
				for (int t=1; t<=6; t++){
					genSum[t] = genSum[t] + trait[a][t];
				}
				genSum[7] = genSum[7] + payoff[a];
			}
			for (int t=1; t<=6; t++){
				genAvg[t] = genSum[t] / (population);
			}
			genAvg[7] = genSum[7] / (population);
			if (REPORTLEVEL == 3){
				genData.write(GAMETYPE + "," + INSTITUTION + "," + multiplier + "," + r + "," + g + "," + genAvg[1] + "," + genAvg[5]
				+ "," + genAvg[3] + "," + genAvg[7] +		"\r\n");
			}
```

```
// Birth Routine============
                                    for (int a=1; a<=population; a++){
                                          for (int t=1; t<=6; t++){
                                                trait[a][t] = childTrait[a][t];
                                          }
                                    }
                              }

// Run Stats================
                              System.out.println("Run: " + r + ", factor: " + multiplier + ", Degree: " + networkDegree + ", Offer: " + dataOut.format(genAvg[1]));
                              if (REPORTLEVEL == 2 || REPORTLEVEL == 3){
                                    runData.write(GAMETYPE + "," + INSTITUTION + "," + multiplier + "," + r + "," + genAvg[1] + "," + genAvg[5] + "," + g
                                    genAvg[3] + "," + genAvg[7] + "\r\n");
                              }

// Simulation Stats=========
                              for (int t=1; t<=6; t++){
                                    simSum[t] = simSum[t] + genAvg[t];
                              }
                              simSum[7] = simSum[7] + genAvg[7];
                        }
                        for (int t=1; t<=6; t++){
                              simAvg[t] = simSum[t] / RUNS;
                        }
                        simAvg[7] = simSum[7] / RUNS;
                        System.out.println(networkName + ", degree: " + networkDegree + ", factor: " + multiplier + ", Offer: "+ dataOut.format(simAvg[1]));
                        simData.write(GAMETYPE + "," + INSTITUTION + "," + multiplier + "," + simAvg[1] + "," + simAvg[5] + "," + simAvg[3] + "," +
                        simAvg[7] + "\r\n");
                  }
            }
// Shutdown=================
            genData.close();
            runData.close();
            simData.close();

      } catch (IOException ioe){
            System.out.println("Houston, we have a read/write                    problem.\n");
```

```
        }
    }
}
```

```
package ultimatumGame;

public interface UltimatumGameParameters {

// general parameters
	static final int GAMETYPE = 1;				// 1 = dictator game, not 1 = ultimatum game
	static final int INSTITUTION = 1;			// 1 = punish only, 2 = reward only, 3 = punish & reward, not (1, 2 or 3) = no institution
	static final int RUNS = 1;
	static final int GENERATIONS = 10000;
	static final double MUTATIONRATE = 0.1;
	static final int PAIRINGS = 3;
	static final int REPORTLEVEL = 2;			// 3 = generation level, 2 = run level, not (2 or 3) = simulation level

// parameter sweeps
// NEIGHBORHOOD_TYPES:	1 = 3D Moore, 2 = 3D von Neumann, 3 = 2D Moore, 4 = 2D hexagonal,
//			5 = 2D von Neumann, 6 = linear (1D), 7 = complete graph, 8 = random graph,
//			9 = small-world graph, 10 = scale-free graph, 11 = random regular graph
	static final int BEGIN_NEIGHBORHOOD  = 3;	// starting neighborhood for parameter
	static final int END_NEIGHBORHOOD = 11;		// ending neighborhood for simulation loop
	static final double BEGIN_RM = 3.0;		// starting value of reciprocity multiplier
	static final double END_RM = 3.0;		// ending value of reciprocity multiplier
	static final double RM_INCREMENT = 1.0;		// value to increment reciprocity multiplier during sweep

// parameters specific to 2D spatially explicit neighborhoods
	static final int ROWS = 20;			// rows & cols must be even numbers for hexagonal lattice (neighborhood_type 4)
	static final int COLS = 20;

// parameters specific to 3D spatially explicit neighborhoods
	static final int THREEDROWS = 8;
	static final int THREEDCOLS = 8;
	static final int DEPTH = 8;

// parameters specific to network structures
	static final double GRAPH_DENSITY = 0.005;	// probability of being connected to a member of population in a random graph
	static final double PROB_REWIRE = 0.05;		// probability of 	rewiring an edge of a ring substrate to produce a Watts-Strogatz small-world
network
	static final int NODES = 500;			// number of agents 	in non-spacially explict neighborhoods
```

```
    static final int RANGE = 1;                    // number of links away from an agent to include as neighbors
    static final int SEED_NODES = 3;               // number of initial nodes in scale free network
    static final int LINKS_PER_NODE = 2;           // number of links per new node in a scale free network
    static final int SMALL_WORLD_RADIUS = 2;       // neighbor radius of ring stubstrate for small world networks
    static final int FIXED_DEGREE = 4;             // degree of a random regular graph
}
```

```
package simulationTools;

import java.io.FileWriter;
import java.io.IOException;
import java.util.Arrays;
import java.util.Calendar;
import java.util.Random;
import java.util.TimeZone;

public class NetworkTools {

    public static int[][] getNeighbors(int[][] adjMatrix){

        int populationSize = adjMatrix.length-1;
        exportPajek(adjMatrix);
        exportAdjMatrix(adjMatrix);
        int neighborMatrix[][] = new int[populationSize+1][populationSize+1];
        for (int i=1; i<=populationSize; i++){
            int neighbor_count = 0;
            for (int j=1; j<=populationSize; j++){
                if (adjMatrix[i][j] == 1){
                    neighbor_count++;
                    neighborMatrix[i][neighbor_count] = j;
                }
                neighborMatrix[i][0] = neighbor_count;     // stores the total neighbor count of each node
            }
        }
        return neighborMatrix;
    }

    public static int[][] getNonNeighbors(int[][] adjMatrix){

        int populationSize = adjMatrix.length-1;
        int nonNeighborMatrix[][] = new int[populationSize+1][populationSize+1];
        for (int i=1; i<=populationSize; i++){
            int nonNeighbor_count = 0;
            for (int j=1; j<=populationSize; j++){
```

```
            if (i != j){
                if (adjMatrix[i][j] == 0){
                    nonNeighbor_count++;
                    nonNeighborMatrix[i][nonNeighbor_count] = j;
                }
            }
            nonNeighborMatrix[i][0] = nonNeighbor_count;      // stores the total non-neighbor count of each node
        }
    }
    return nonNeighborMatrix;
}

public static int[][] getAdjacencyMatrix(int population, int networkType, int threeDRows,
                                         int threeDCols, int depth, int rows, int cols, int nodes,
                                         int range, double graphDensity, double probRewire,
                                         int seedNodes, int linksPerNode, int smallWorldRadius,
                                         int fixedDegree){

    int adjMatrix[][] = new int[population+1][population+1];

    switch(networkType){

        case 1: // 3D Moore neighborhood (neighbors N,S,E,W,up,down + diagonals)
            adjMatrix = make3DMooreNetwork(threeDRows,threeDCols,depth);
            break;

        case 2: // 3D von Neumann neighborhood (neighbors N,S,E,W,up & down)
            adjMatrix = make3DVonNeumannNetwork(threeDRows,threeDCols,depth);
            break;

        case 3: // 2D Moore neighborhood (neighbors N,S,E,W + diagonals)
            adjMatrix = make2DMooreNetwork(rows,cols);
            break;

        case 4: // 2D Hexagonal lattice
            adjMatrix = makeHexagonalNetwork(rows,cols);
            break;
```

```
            case 5: // 2D von Neumann neighborhood, (neighbors N,S,E & W)
                adjMatrix = make2DVonNeumannNetwork(rows,cols);
                break;

            case 6: // 1D Linear (neighbors E + W in a ring)
                adjMatrix = makeRingNetwork(nodes,range);
                break;

            case 7: // Homogenous graph (Fully connected - every agent connected to every other agent)
                adjMatrix = makeHomogenousNetwork(nodes);
                break;

            case 8: // Random Erdos-Renyi graph (random links)
                adjMatrix = makeRandomNetwork(nodes,graphDensity);
                break;

             case 9: // Random Watts-Strogratz small-world graph
                adjMatrix = makeWattsStrogatzNetwork(nodes,smallWorldRadius,probRewire);
                break;

            case 10: // Random Barabasi-Albert scale-free graph
                adjMatrix = makeBarabasiAlbertNetwork(nodes,seedNodes,linksPerNode);
                break;

            case 11: // Random regular graph (every node has same degree but random links)
                adjMatrix = makeRandomRegularNetwork(nodes,fixedDegree);
                break;
        }
        return adjMatrix;
}

public static int[][] make2DMooreNetwork(int rows, int cols){

        int[][] adjMatrix = new int[rows*cols+1][rows*cols+1];
        int[][] nameMatrix = assign2DSpaceToNodes(rows,cols);
        for (int x=1; x<=rows; x++){
```

```
            for (int y=1; y<=cols; y++){
                for (int nr=-1; nr<=1; nr++){
                    for (int nc=-1; nc<=1; nc++){
                        if(Math.abs(nr) + Math.abs(nc) !=0){
                            int row = x + nr;
                            if (row < 1){
                                row += rows;
                            } else if (row > rows){
                                row -= rows;
                            }
                            int col = y + nc;
                            if (col < 1){
                                col += cols;
                            } else if (col > cols){
                                col -= cols;
                            }
                            adjMatrix[nameMatrix[x][y]][nameMatrix[row][col]] = 1;
                        }
                    }
                }
            }
        }
        return adjMatrix;
    }

    public static int[][] make3DMooreNetwork(int rows, int cols, int depth){

        int[][] adjMatrix = new int[rows*cols*depth+1][rows*cols*depth+1];
        int[][][] nameMatrix = assign3DSpaceToNodes(rows,cols,depth);
        for (int x=1; x<=rows; x++){
            for (int y=1; y<=cols; y++){
                for (int z=1; z<=depth; z++){
                    for (int nr=-1; nr<=1; nr++){
                        for (int nc=-1; nc<=1; nc++){
                            for (int nd=-1; nd<=1; nd++){
                                if(Math.abs(nr) + Math.abs(nc) +                Math.abs(nd) !=0){
                                    int row = x + nr;
```

```
                                        if (row < 1){
                                            row += rows;
                                        } else if  (row > rows){
                                            row -= rows;
                                        }
                                        int col = y + nc;
                                        if (col < 1){
                                            col += cols;
                                        } else if  (col > cols){
                                            col -= cols;
                                        }
                                        int dep = z + nd;
                                        if (dep < 1){
                                            dep += depth;
                                        }
                                        else if (dep > depth){
                                            dep -= depth;
                                        }
                                        adjMatrix[nameMatrix[x][y][z]][nameMatrix[row][col][dep]] = 1;
                                    }
                                }
                            }
                        }
                    }
                }
            }
            return adjMatrix;
        }

        public static int[][] make2DVonNeumannNetwork(int rows, int cols){

            int[][] adjMatrix = new int[rows*cols+1][rows*cols+1];
            int[][] nameMatrix = assign2DSpaceToNodes(rows,cols);
            for (int x=1; x<=rows; x++){
                for (int y=1; y<=cols; y++){
                    for (int nr=-1; nr<=1; nr++){
                        for (int nc=-1; nc<=1; nc++){
```

```
                    if(Math.abs(nr) + Math.abs(nc) == 1){
                        int row = x + nr;
                        if (row < 1){
                            row += rows;
                        }
                        if (row > rows){
                            row -= rows;
                        }
                        int col = y + nc;
                        if (col < 1){
                            col += cols;
                        }
                        if (col > cols){
                            col -= cols;
                        }
                        adjMatrix[nameMatrix[x][y]][nameMatrix[row][col]] = 1;
                    }
                }
            }
        }
    }
    return adjMatrix;
}

public static int[][] make3DVonNeumannNetwork(int rows, int cols, int depth){

    int[][] adjMatrix = new int[rows*cols*depth+1][rows*cols*depth+1];
    int[][][] nameMatrix = assign3DSpaceToNodes(rows,cols,depth);
    for (int x=1; x<=rows; x++){
        for (int y=1; y<=cols; y++){
            for (int z=1; z<=depth; z++){
                for (int nr=-1; nr<=1; nr++){
                    for (int nc=-1; nc<=1; nc++){
                        for (int nd=-1; nd<=1;nd++){
                            if(Math.abs(nr) + Math.abs(nc) +             Math.abs(nd) == 1){
                                int row = x + nr;
                                if (row < 1){
```

```
                                    row += rows;
                                }
                                if (row > rows){
                                    row -= rows;
                                }
                                int col = y + nc;
                                if (col < 1){
                                    col += cols;
                                }
                                if (col > cols){
                                    col -= cols;
                                }
                                int dep = z + nd;
                                if (dep < 1){
                                    dep += depth;
                                }
                                if (dep > depth){
                                    dep -= depth;
                                }
                                adjMatrix[nameMatrix[x][y][z]][nameMatrix[row][col][dep]] = 1;
                            }
                        }
                    }
                }
            }
        }
    }
    return adjMatrix;
}

public static int[][] makeHexagonalNetwork(int rows, int cols){ // requires even number of rows/columns

    int[][] adjMatrix = new int[rows*cols+1][rows*cols+1];
    int[][] nameMatrix = assign2DSpaceToNodes(rows,cols);
    int rowType = 1;
    for (int x=1; x<=rows; x++){
        for (int y=1; y<=cols; y++){
```

```
                for (int nr=-1; nr<=1; nr++){
                    for (int nc=-1; nc<=1; nc++){
                        if(Math.abs(nr) + Math.abs(nc) !=0){
                            int row = x + nr;
                            if (row < 1){
                                row += rows;
                            }
                            if (row > rows){
                                row -= rows;
                            }
                            int col = y + nc;
                            if (col < 1){
                                col += cols;
                            }
                            if (col > cols){
                                col -= cols;
                            }
                            adjMatrix[nameMatrix[x][y]][nameMatrix[row][col]] = 1;
                            if (nr !=0 && nc == rowType){
                                adjMatrix[nameMatrix[x][y]][nameMatrix[row][col]] = 0;
                            }
                        }
                    }
                }
            }
            rowType = rowType * -1;
        }
        return adjMatrix;
    }

    public static int[][] makeRingNetwork(int nodes, int neighborhoodRadius){

        int[][] ringMatrix = new int[nodes+1][nodes+1];
        for (int i=1; i<=nodes; i++){
            for (int j=1; j<=nodes; j++){
                if (((j > i) && (j <= i+neighborhoodRadius)) ||
                        ((j < i) && (j >= i-neighborhoodRadius)) ||
```

```
                        ((j > i) && (j >= nodes - neighborhoodRadius + i)) ||
                        ((j < i) && (i >= nodes - neighborhoodRadius + j))){
                    ringMatrix[i][j] = 1;
                } else {
                    ringMatrix[i][j] = 0;
                }
            }
        }
        return ringMatrix;
    }

    public static int[][] makeWattsStrogatzNetwork(int nodes, int smallWorldRadius, double probRewire){

        int[][] adjMatrix = new int[nodes+1][nodes+1];
        Random rnd = new Random();
        int totalUnlinked;
        int[][] tempAdjMatrix = new int[nodes+1][nodes+1];
        int[] unlinkedNodes = new int[nodes+1];

        adjMatrix = makeRingNetwork(nodes, smallWorldRadius);

        for (int i=1; i<=nodes; i++){
            for (int j=1; j<=nodes; j++){
                tempAdjMatrix[i][j] = adjMatrix[i][j];
            }
        }

        for (int i=2; i<=nodes; i++){
            for (int j=1; j<i; j++){
                if (adjMatrix[i][j] == 1){
                    totalUnlinked = 0;
                    Arrays.fill(unlinkedNodes,0);
                    for (int k=1; k<=nodes; k++){
                        if ((tempAdjMatrix[i][k] == 0) && (i != k)){
                            unlinkedNodes[totalUnlinked] = k;
                            totalUnlinked++;
                        }
```

```
                }
                if (rnd.nextDouble() < probRewire){
                    int newNode = rnd.nextInt(totalUnlinked)+1;
                    tempAdjMatrix[i][j] = 0;
                    tempAdjMatrix[j][i] = 0;
                    tempAdjMatrix[i][unlinkedNodes[newNode]] = 1;
                    tempAdjMatrix[unlinkedNodes[newNode]][i] = 1;
                }
            }
        }
    }

    // make sure every node is linked, else add a random link
    for (int i=1; i<=nodes; i++) {
        totalUnlinked = 0;
        for (int j=1; j<=nodes; j++) {
            if (tempAdjMatrix[i][j] == 0){
                totalUnlinked++;
            }
        }
        if (totalUnlinked==nodes){
            int newNode = rnd.nextInt(nodes)+1;
            tempAdjMatrix[i][newNode]=1;
            tempAdjMatrix[newNode][i]=1;
        }
    }

    for (int i=1; i<=nodes; i++) {
        for (int j=1; j<=nodes; j++) {
            adjMatrix[i][j] = tempAdjMatrix[i][j];
        }
    }

return adjMatrix;
}

public static int[][] makeRandomRegularNetwork(int nodes, int          linksPerNode){
```

```
Random rnd = new Random();
int[][] adjMatrix = new int[nodes+1][nodes+1];
int[] nodeOrder = new int[nodes+1];
int[] potentialLinks = new int[nodes+1];
int[] currentLinks = new int[nodes+1];
nodeOrder = ArrayTools.getRandPermutation(nodes);
for (int a=1; a<=nodes; a++){
    int currentNode = nodeOrder[a];
    potentialLinks = ArrayTools.getRandPermutation(nodes);
    int n=1;
    while (currentLinks[currentNode] < linksPerNode){
        if (n == nodes){
            int j = currentNode;
            int i = currentNode;
            boolean needLink = true;
            while (needLink){
                boolean needJ = true;
                while (needJ){
                    j = rnd.nextInt(nodes) + 1;
                    if (j != currentNode){
                        needJ = false;
                    }
                }
                boolean needI = true;
                while (needI){
                    i = rnd.nextInt(nodes) + 1;
                    if (i != currentNode){
                        needI = false;
                    }
                }
                if (adjMatrix[i][j]==1){
                    needLink = false;
                }
            }
            adjMatrix[i][j] = 0;
            adjMatrix[j][i] = 0;
            adjMatrix[currentNode][i] = 1;
```

```
                        adjMatrix[i][currentNode] = 1;
                        currentLinks[currentNode]++;
                        adjMatrix[currentNode][j] = 1;
                        adjMatrix[j][currentNode] = 1;
                        currentLinks[currentNode]++;
                    } else {
                        if (currentNode == potentialLinks[n]){
                            n++;
                        } else {
                            if (adjMatrix[currentNode][potentialLinks[n]]==1){
                                n++;
                            } else {
                                if (currentLinks[potentialLinks[n]]==linksPerNode){
                                    n++;
                                } else {
                                    adjMatrix[currentNode][potentialLinks[n]] = 1;
                                    adjMatrix[potentialLinks[n]][currentNode] = 1;
                                    currentLinks[currentNode]++;
                                    currentLinks[potentialLinks[n]]++;
                                    n++;
                                }
                            }
                        }
                    }
                }
            }
            return adjMatrix;
        }

public static int[][] makeBarabasiAlbertNetwork(int nodes, int seedNodes, int linksPerNewNode){
        // generate a random scale-free network by preferential growth

        int[][] adjMatrix = new int[nodes+1][nodes+1];
        int[] linkNumber = new int[nodes*nodes];
        Random rnd = new Random();

        // fully connect the seed nodes
```

```
		for (int i=1; i<=seedNodes-1; i++){
			for (int j=i+1; j<=seedNodes; j++){
				adjMatrix[i][j]=1;
				adjMatrix[j][i]=1;
			}
		}
		for (int i=seedNodes+1; i<=nodes; i++){
			int totalLinks = 0;
			int newConnections = 0;

			// weight existing nodes by number of links each has
			for (int t=1; t<i; t++){
				for (int j=1; j<i; j++){
					if (adjMatrix[t][j]==1){
						totalLinks++;
						linkNumber[totalLinks]=t;
					}
				}
			}
			while (newConnections < linksPerNewNode){
				int newLink = rnd.nextInt(totalLinks)+1;
				if (adjMatrix[i][linkNumber[newLink]] == 0){
					adjMatrix[i][linkNumber[newLink]]=1;
					adjMatrix[linkNumber[newLink]][i]=1;
					newConnections++;
				}
			}
		}
		return adjMatrix;
	}

	public static int[][] makeRandomNetwork(int nodes, double linkProbability){

		int[][] adjMatrix = new int[nodes+1][nodes+1];
		int[][] tempAdjMatrix = new int[nodes+1][nodes+1];
		Random rnd = new Random();
		for (int i=1; i<=nodes; i++){
```

```
            for (int j=1; j<=nodes; j++){
                if (i != j){
                    if (rnd.nextDouble() <= linkProbability){
                        tempAdjMatrix[i][j] = 1;
                        tempAdjMatrix[j][i] = 1;
                    }
                }
            }
        }
        // make sure every node is linked, else add a random link
        for (int i=1; i<=nodes; i++) {
            int totalUnlinked = 0;
            for (int j=1; j<=nodes; j++) {
                if (tempAdjMatrix[i][j] == 0){
                    totalUnlinked++;
                }
            }
            if (totalUnlinked==nodes){
                int newNode = rnd.nextInt(nodes)+1;
                tempAdjMatrix[i][newNode]=1;
                tempAdjMatrix[newNode][i]=1;
            }
        }

        for (int i=1; i<=nodes; i++) {
            for (int j=1; j<=nodes; j++) {
                adjMatrix[i][j] = tempAdjMatrix[i][j];
            }
        }
    return adjMatrix;
    }

    public static int[][] makeHomogenousNetwork(int nodes){

        int adjMatrix[][] = new int[nodes+1][nodes+1];
        for (int i=1; i<=nodes; i++){
            for (int j=1; j<=nodes; j++){
```

```
                if (i != j){
                    adjMatrix[i][j] = 1;
                } else {
                    adjMatrix[i][j] = 0;
                }
            }
        }
        return adjMatrix;
    }

    public static int getPopulationSize(int networkType, int threeDRows, int threeDCols, int depth,
                                        int rows, int cols, int nodes){
        int populationSize = 0;
        switch(networkType){

            case 1: populationSize = threeDRows * threeDCols * depth; break;
            case 2: populationSize = threeDRows * threeDCols * depth; break;
            case 3: populationSize = rows * cols; break;
            case 4: populationSize = rows * cols; break;
            case 5: populationSize = rows * cols; break;
            case 6: populationSize = nodes; break;
            case 7: populationSize = nodes; break;
            case 8: populationSize = nodes; break;
            case 9: populationSize = nodes; break;
            case 10: populationSize = nodes; break;
            case 11: populationSize = nodes; break;
        }
        return populationSize;
    }

    public static double getNetworkDegree(int[][] neighbors){

        int populationSize = neighbors.length-1;
        double totalLinks = 0;
        for (int i=1; i<=populationSize; i++){
            totalLinks += neighbors[i][0];
        }
```

```
        double networkDegree = totalLinks / populationSize;
        return networkDegree;
}

public static double getNetworkDegreeVariance(int[][] neighbors){
        int populationSize = neighbors.length-1;
        double sumOfSquares = 0;
        double networkDegreeMean = getNetworkDegree(neighbors);
        for (int i=1; i<=populationSize; i++){
                sumOfSquares += Math.pow((networkDegreeMean - neighbors[i][0]), 2);
        }
        double networkDegreeVariance = sumOfSquares / populationSize;
        return networkDegreeVariance;
}

public static void exportPajek(int[][] adjMatrix){

        int nodes = adjMatrix.length-1;
        try{
                FileWriter pajekOut = new FileWriter("output\\PajekOut.net");
                pajekOut.write("*Vertices " + nodes +"\r\n");
                pajekOut.write("*Edges\r\n");
                for(int i=1; i<=nodes; i++){
                        for(int j=i; j<=nodes; j++){
                                if(adjMatrix[i][j]==1){
                                        pajekOut.write(i + " " + j + "\r\n");
                                }
                        }
                }
                pajekOut.close();
        } catch (IOException ioe){
                System.out.println("Houston, we have a Pajek output problem.\n");
        }
}

public static void exportAdjMatrix(int[][] adjMatrix){
        int nodes = adjMatrix.length-1;
```

```
        try{
            FileWriter outputMatrix = new FileWriter("output\\adjMatrix.txt");
            for(int i=1; i<=nodes; i++){
                for(int j=1; j<=nodes; j++){
                    outputMatrix.write(adjMatrix[i][j] + " ");
                }
                outputMatrix.write("\r\n");
            }
            outputMatrix.close();
        } catch (IOException ioe){
            System.out.println("Houston, we have a problem outputing the adjacency matrix.\n");
        }
    }

    public static int[][] assign2DSpaceToNodes(int rows, int cols){

        int nameMatrix[][] = new int[rows+1][cols+1];
        int agentNumber = 1;
        for (int x=1; x<=rows; x++){
            for (int y=1; y<=cols; y++){
                nameMatrix[x][y] = agentNumber;
                agentNumber++;
            }
        }
        return nameMatrix;
    }

    public static int[][][] assign3DSpaceToNodes(int rows, int cols, int depth){

        int nameMatrix[][][] = new int[rows+1][cols+1][depth+1];
        int agentNumber = 1;
        for (int x=1; x<=rows; x++){
            for (int y=1; y<=cols; y++){
                for (int z=1; z<=depth; z++){
                    nameMatrix[x][y][z] = agentNumber;
                    agentNumber++;
                }
```

```
            }
        }
        return nameMatrix;
    }

    public static String getTimeStamp(){

        Calendar cal = Calendar.getInstance(TimeZone.getDefault());
        java.text.SimpleDateFormat sdf = new java.text.SimpleDateFormat("yyyyMMddHHmmssSSS");
        String timestamp = sdf.format(cal.getTime());
        return timestamp;
    }

    public static String getNetworkName(int networkType){

        String networkName = "not defined";
        switch(networkType){
            case 1: networkName = "3D Moore"; break;
            case 2: networkName = "3D von Neumann"; break;
            case 3: networkName = "2D Moore"; break;
            case 4: networkName = "Hexagonal"; break;
            case 5: networkName = "2D von Neumann"; break;
            case 6: networkName = "Linear"; break;
            case 7: networkName = "Complete"; break;
            case 8: networkName = "Random"; break;
            case 9: networkName = "Small-world"; break;
            case 10: networkName = "Scale-free"; break;
            case 11: networkName = "Random regular"; break;
        }
        return networkName;
    }
}
```

```
package simulationTools;

import java.util.*;

public class MatrixTools {

	public static double getMean(int[][] matrixIn, int indexStart){
		int elements = 0;
		int sum = 0;
		for (int i=indexStart; i<matrixIn.length; i++){
			for (int j=indexStart; j<matrixIn[i].length; j++){
				sum += matrixIn[i][j];
				elements++;
			}
		}
		double matrixMean = sum/elements;
		return matrixMean;
	}

	public static double getMean(double[][] matrixIn, int indexStart){
		int elements = 0;
		double sum = 0;
		for (int i=indexStart; i<matrixIn.length; i++){
			for (int j=indexStart; j<matrixIn[i].length; j++){
				sum += matrixIn[i][j];
				elements++;
			}
		}
		double matrixMean = sum/elements;
		return matrixMean;
	}

	public static double getStdDev(int[][] matrixIn, int indexStart){
		int elements = 0;
		double sumOfSquares = 0;
		double mean = getMean(matrixIn, indexStart);
		for (int i=indexStart; i<matrixIn.length; i++){
```

```
        for (int j=indexStart; j<matrixIn[i].length;j++){
            sumOfSquares += ((matrixIn[i][j] - mean)*(matrixIn[i][j] - mean));
            elements++;
        }
    }
    System.out.println(sumOfSquares + " , " + elements);
    double matrixStdDev = Math.sqrt(sumOfSquares / (elements - 1));
    return matrixStdDev;
}

public static double getStdDev(double[][] matrixIn, int indexStart){
    int elements = 0;
    double sumOfSquares = 0;
    double mean = getMean(matrixIn, indexStart);
    for (int i=indexStart; i<matrixIn.length; i++){
        for (int j=indexStart; j<matrixIn[i].length;j++){
            sumOfSquares += ((matrixIn[i][j] - mean)*(matrixIn[i][j] - mean));
            elements++;
        }
    }
    System.out.println(sumOfSquares + " , " + elements);
    double matrixStdDev = Math.sqrt(sumOfSquares / (elements - 1));
    return matrixStdDev;
}

public static double[][] fillRandom (double matrixIn[][]){
    double rndMatrix[][] = copyMatrix(matrixIn);
    Random rnd = new Random();
    for (int i=0; i<rndMatrix.length; i++){
        for (int j=0; j<rndMatrix[i].length;j++){
            rndMatrix[i][j] = rnd.nextDouble();
        }
    }
    return rndMatrix;
}

public static int[][] fillRandom (int matrixIn[][], int minValue, int        maxValue){
```

```
        int rndMatrix[][] = copyMatrix(matrixIn);;
        int range = maxValue - minValue;
        Random rnd = new Random();
        for (int i=0; i<rndMatrix.length; i++){
            for (int j=0; j<rndMatrix[i].length;j++){
                rndMatrix[i][j] = rnd.nextInt(range + 1) + minValue;
            }
        }
        return rndMatrix;
    }

    public static double[][] fillRandom (double matrixIn[][], double minValue, double maxValue){
        double rndMatrix[][] = copyMatrix(matrixIn);;
        double range = maxValue - minValue;
        Random rnd = new Random();
        for (int i=0; i<rndMatrix.length; i++){
            for (int j=0; j<rndMatrix[i].length;j++){
                rndMatrix[i][j] = (rnd.nextDouble() * range) + minValue;
            }
        }
        return rndMatrix;
    }

    public static int[][] fillConstant (int matrixIn[][], int constant){
        int filledMatrix[][] = copyMatrix(matrixIn);
        for (int i=0; i<matrixIn.length; i++){
            Arrays.fill(filledMatrix[i], constant);
        }
        return filledMatrix;
    }

    public static double[][] fillConstant (double matrixIn[][], double constant){
        double filledMatrix[][] = copyMatrix(matrixIn);
        for (int i=0; i<matrixIn.length; i++){
            Arrays.fill(filledMatrix[i],constant);
        }
        return filledMatrix;
```

```
	}

	public static int[][] copyMatrix(int matrixIn[][]){
		int matrixCopy[][] = new int [matrixIn.length][];
		for (int i=0; i<matrixIn.length; i++){
			matrixCopy[i] = (int[]) matrixIn[i].clone();
		}
		return matrixCopy;
	}

	public static double[][] copyMatrix(double matrixIn[][]){
		double matrixCopy[][] = new double [matrixIn.length][];
		for (int i=0; i<matrixIn.length; i++){
			matrixCopy[i] = (double[]) matrixIn[i].clone();
		}
		return matrixCopy;
	}
}
```

```
package simulationTools;

import java.util.Random;

public class ArrayTools {

	public static double getMean(int[] arrayIn, int indexStart){
		double sum = 0;
		for (int i=indexStart; i<arrayIn.length; i++){
			sum += arrayIn[i];
		}
		double arrayMean = sum/(arrayIn.length-indexStart);
		return arrayMean;
	}

	public static double getMean(double[] arrayIn, int indexStart){
		double sum = 0;
		for (int i=1; i<arrayIn.length; i++){
			sum += arrayIn[i];
		}
		double arrayMean = sum/(arrayIn.length-indexStart);
		return arrayMean;
	}

	public static double getHarmonicMean(int[] arrayIn, int indexStart){
		double sum = 0;
		for (int i=indexStart; i<arrayIn.length; i++){
			sum += (1. / arrayIn[i]);
		}
		double arrayHarmonicMean = 1/(sum/(arrayIn.length-indexStart));
		return arrayHarmonicMean;
	}

	public static double getHarmonicMean(double[] arrayIn, int indexStart){
		double sum = 0;
		for (int i=indexStart; i<arrayIn.length; i++){
			sum += (1. / arrayIn[i]);
```

```
        }
        double arrayHarmonicMean = 1/(sum/(arrayIn.length-indexStart));
        return arrayHarmonicMean;
}

public static double getGeoMean(int[] arrayIn, int indexStart){
        double product = 1;
        for (int i=indexStart; i<arrayIn.length; i++){
                product *= arrayIn[i];
        }
        double arrayGeoMean = Math.pow(product,(1./(arrayIn.length-indexStart)));
        return arrayGeoMean;
}

public static double getGeoMean(double[] arrayIn, int indexStart){
        double product = 1;
        for (int i=indexStart; i<arrayIn.length; i++){
                product *= arrayIn[i];
        }
        double arrayGeoMean = Math.pow(product,(1./(arrayIn.length-indexStart)));
        return arrayGeoMean;
}

public static double getStdDev(int[] arrayIn, int indexStart){
        double sumOfSquares = 0;
        double mean = getMean(arrayIn, indexStart);
        for (int i=1; i<arrayIn.length; i++){
                sumOfSquares += ((arrayIn[i] - mean)*(arrayIn[i] - mean));
        }
        double arrayStdDev = Math.sqrt(sumOfSquares / ((arrayIn.length-indexStart) - 1));
        return arrayStdDev;
}

public static double getStdDev(double[] arrayIn, int indexStart){
        double sumOfSquares = 0;
        double mean = getMean(arrayIn, indexStart);
        for (int i=1; i<arrayIn.length; i++){
```

```
            sumOfSquares += ((arrayIn[i] - mean)*(arrayIn[i] - mean));
        }
        double arrayStdDev = Math.sqrt(sumOfSquares / ((arrayIn.length-indexStart) - 1));
        return arrayStdDev;
    }

    public static double[] fillRandom (double arrayIn[]){
        double rndArray[] = new double [arrayIn.length];
        Random rnd = new Random();
        for (int i=0; i<arrayIn.length; i++){
            rndArray[i] = rnd.nextDouble();
        }
        return rndArray;
    }

    public static int[] fillRandom (int arrayIn[], int minValue, int maxValue){
        int rndArray[] = new int [arrayIn.length];
        int range = maxValue - minValue;
        Random rnd = new Random();
        for (int i=0; i<arrayIn.length; i++){
            rndArray[i] = rnd.nextInt(range + 1) + minValue;
        }
        return rndArray;
    }

    public static double[] fillRandom (double arrayIn[], double minValue, double maxValue){
        double rndArray[] = new double [arrayIn.length];
        double range = maxValue - minValue;
        Random rnd = new Random();
        for (int i=0; i<arrayIn.length; i++){
            rndArray[i] = (rnd.nextDouble() * range) + minValue;
        }
        return rndArray;
    }

    public static int[] getRandPermutation (int arraySize){
        int populationOrder[] = new int[arraySize+1];
```

```
        RandPermutation order = new RandPermutation();
        order.setPermutation(arraySize);
        for (int i=1; i<=arraySize; i++){
            int nextUp = (int)order.get(i-1);
            nextUp += 1;
            populationOrder[i]=nextUp;
        }
        return populationOrder;
    }
}
```

```
package simulationTools;

import java.util.NoSuchElementException;
import java.util.Random;

public class RandPermutation implements IndexIterator{

    private int[] buffer = null;
    private int len = 0;
    private int pointer = 0;
    private final Random r;

    public RandPermutation(){
        this.r= new Random();
    }

    public void setPermutation(int k){
        reset(k);
        for(int i=len; i>1; i--){
            int j = r.nextInt(i);
            int a = buffer[j];
            buffer[j] = buffer[i-1];
            buffer[i-1] = a;
        }
    }

    public int get(int i){
        if( i >= len ) throw new IndexOutOfBoundsException();
            return buffer[i];
        }

    public void reset(int k){
        pointer = k;
        if( len == k ) return;
        if( buffer == null || buffer.length < k ){
            buffer = new int[k];
        }
```

```
        len = k;
        for( int i=0; i<len; ++i ) buffer[i]=i;
    }

    public int next(){

        if( pointer < 1 ) throw new NoSuchElementException();
        int j = r.nextInt(pointer);
        int a = buffer[j];
        buffer[j] = buffer[pointer-1];
        buffer[pointer-1] = a;
        return buffer[--pointer];
    }

    public boolean hasNext(){
        return pointer > 0;
    }
}
```

```
package simulationTools;

public interface IndexIterator{

	public void reset(int k);

	public int next();

	public boolean hasNext();
}
```