ABSTRACT

PORTER, GLORIA L. Multiobjective Optimal Control Methodology for the Analysis of Certain Sociodynamic Problems. (Under the direction of Professor N.G. Medhin.)

Social networks involve studying how relations form between individuals in a group based on their shared preferences and attributes. This research addresses a very difficult question involving how social networks arise and evolve over time. Historically, some researchers have addressed this issue using loglinear modeling , continuous time Markov theory or rational choice theory. In this work, *social force theory* is used to model social interaction and long-term network dynamics while *multiobjective control theory* provides a basis for predicting network structural formation. Using computer simulations, we numerically analyze the evolution and long-term behavior of optimal network structures based on the demographics of a small data set. We pay special attention to the effect that *memory* has on relationship choices, especially clique formation.

After obtaining a snapshot of the network structure, we turn our attention to a very common task involving social networks: *missing link prediction*. The link prediction problem can be described as uncovering hidden or missing connections between nodes in an observed network. There are several link prediction methods in the literature that rely heavily on network topology to predict links and are primarily used on collaboration networks like email or coauthorship networks. The problem with these networks is that they offer no qualitative information on nodal attributes. In this work, we present a new model for link prediction that is centered on nodal attributes and uses social force theory to provide behavioral rules for nodal interaction.

The model for link prediction starts with the observed network structure and parameters derived from the multiobjective optimal control approach used above. We use the known relationship patterns in the network to construct a new multiobjective optimal control problem constrained in a way that reproduces those existing relationships in the network. We then remove a random subset of links from the network treating them as missing or hidden. Finally, we solve this new constrained multiobjective optimal control problem to reproduce existing links while uncovering the missing links in the process.

Multiobjective Optimal Control Methodology for the Analysis of Certain Sociodynamic Problems

by
Gloria L. Porter

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fullfillment of the
requirements for the Degree of
Doctor of Philosophy

Applied Mathematics

Raleigh, North Carolina

2009

APPROVED BY:

Dr. Jeffrey S. Scroggs	Dr. Robert E. White
Dr. Negash G. Medhin Chair of Advisory Committee	Dr. Stephen L. Campbell

DEDICATION

I dedicate this work to my parents for whom I have the utmost love and respect. Your exceptional wisdom has been my blanket of protection as well as a guiding light through life's toughest challenges. Please know that your incredible work ethic has set the standard for my life and your constant love and encouragement have enabled me to succeed.

BIOGRAPHY

Gloria Porter was born and educated in a small town in Alabama surrounded by the love of family and friends. After high school, she attended Alabama State University where she graduated with honors and was commissioned as a second lieutenant in the United States Air Force. Throughout her Air Force career, she has traveled the world and has had many challenging assignments to include managing complex, service-wide programs, analyzing vast personnel and budget issues, as well as the rewarding experience of teaching and mentoring cadets at the United States Air Force Academy. Upon graduation, she will continue serving in the United States Air Force.

ACKNOWLEDGMENTS

To my Lord and Savior, I give all the honor and glory to you for this achievement. Without your love, grace, and mercy, I could not have been victorious in this endeavor.

My sincerest thanks to my committee for taking time out of your busy schedules to assist me and review my work. To my advisor especially, who persevered with me throughout this entire process, I sincerely appreciate all of your hard work, your guidance and your encouragement...you have my undying gratitude!

To my friends who encouraged me during this difficult process, whether by praying for me, studying with me or listening to my practice presentations, thank you so much for your time, your support, your well wishes, and your friendship. I'm truly grateful!

To my mother and father whose early instruction, love, support and continued sacrifices have laid the foundation for all of my successes in life, thank you so much. To my mother-in-law, I really appreciate your many prayers, the words of wisdom, and your constant love and encouragement. To the rest of my family...my sisters, my brother, aunts, uncles, nieces, nephews, cousins, sisters-in-law and brothers-in-law...please know that your love and encouragement truly made the difference in this process and has meant the world to me. I love you all.

Finally, to my husband I say, thank you for everything...we did it together! Without your love, affection, constant support, and willingness to sacrifice, this goal could not have been achieved. Thank you so much for encouraging me daily, praying for me, and taking care of me. You are truly a special blessing from God and I love you with all of my heart.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

The views expressed in this article are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government.

Chapter 1

Overview

1.1 Introduction

Social networks have been studied extensively in particular in the social sciences arena. However, in recent years, the study of social networks has received great attention from many disciplines and entities outside the social sciences, in particular, the United States government. In fact, since September 11, 2001, the Department of Defense and Department of Homeland Security research initiatives have turned to social network analysis (SNA) to address complex issues that could impact the war on terror and national defense. These issues include 1.) identifying important actors, crucial links, subgroups, roles, and network characteristics and 2.) understanding and predicting human decisions and behavior. DOD researchers believe that understanding the fundamentals of SNA can lead to robust models capable of aiding in preventing wars and influencing adversarial nations, destabilizing/destroying terrorists and other criminal networks, and promoting cohesiveness in organizations as well as improving command and control structures [48], [18].

In the field of social sciences, the underlying principle of social networks has been based on the premise that people are more likely to associate with similar others in regard to personal qualities and traits. McPherson, Smith-Lovin and Cook (2001) [39] suggest that similarity breeds connections and "birds of a feather flock together". They suggest that individuals who are similar in age, ethnicity, educational level, and status are indeed

more inclined to interact with each other than with individuals with dissimilar traits; the tendency to behave in this manner is known as *homophily* and been studied extensively by researchers, [63], [14]. In fact, researchers have received widespread support and respect from different communities for their thorough investigation into how similarities in age, gender, race, education, and other individual characteristics impact the formation of ties within a network. Zeggelink, in particular, has extensively researched this notion that relations are formed based on similarity of shared traits and has summarized a detailed literature review regarding network evolution and modeling networks based on similarity traits in [51].

In contrast to social scientists, researchers [3] in the physical sciences have proposed that the behavior of large social groups can be explained by very basic rules of interaction, i.e, people, in essence, act like automata reacting to key stimuli in their immediate neighborhood. Clearly, this idea is both challenging and disturbing to contemporary social scientists who prefer to analyze human behavior from a more psychological perspective by exploring how people understand and react to their social environment. However, in recent years, more researchers have now proven by example that modeling individual and group behavior based on the principles of physics can also realistically capture many important characteristics of social systems. In fact, today, there exists an abundance of publications attempting to explain societal behavior by mathematically modeling the collective interaction of particles.

In particular, Helbing and Molnar [23], [22] have developed a social forces model for pedestrian dynamics which explains the behavior of pedestrians as though they were subject to acceleration forces to include attractive and repulsive forces which describe how pedestrians respond to their environment. The model [3] is "particle-based" and follows local rules, which are somewhat similar to flocking behavior. Specifically, each of the pedestrians possess an intended walking velocity and destination. All pedestrians maintain their desired velocity unless forced to deviate from their intended plan of action in order to avoid a collision with another pedestrian. The model is simplistic in that it enacts minimal assumptions concerning the psychology of the pedestrians; yet, "it yields group behavior

that looks remarkably life-like". For this reason, Helbing and Molnar's model serves as the motivation for our work involving mathematically modeling the long-term evolution and dynamics of social networks, in particular, friendship networks. This use of other disciplines like phsyics to mathematically explain behavioral changes in social interaction processes is often referred to as *sociodynamics* [62] and sometimes *quantitative sociodynamics* [21].

1.2 Thesis Goals and Contributions

Our goals for this work are as follows:

- **Goal 1**: A model for social networks is designed by adapting Helbing and Molnar's social forces theory to explain the underlying phenomena of social interaction and overall network dynamics while using multiobjective control theory to provide a basis for predicting network structural formation [40]. Unlike many models for social network evolution found in the literature, our method does not assume stationarity or require knowing the complete network structure in advance. Given only minimal data on a set of actors in a social group, our model can predict the long-term behavior and optimal network structures for the group. The model can also handle a larger number of actors than many other models since it uses an evolutionary algorithm (EA) such as Differential Evolution (DE) which is well-suited for parallel computation.

- **Goal 2**: The basic model in Goal 1 is extended here by adding a *memory* mechanism which enables the simulated social network to exhibit "life-like" collective group behavior [40]. When extending the model to include a large number of interacting agents which results in a many-person social network, we will show that the group can be broken down into subgroups or cliques. This addition explains the tendency of individuals to conform to group norms or societal pressures despite the desire to pursue their own interests. To the best of my knowledge, no other model in the literature accounts for *memory* when modeling social interaction among actors in a social group. The emergence of disjoint cliques within the network structure leads us to

explore network aggregation and destabilization techniques.

- **Goal 3**: Here the evolution of a social network is modeled as a state constrained multiobjective optimal control problem (MOCP). Using the necessary conditions for an optimum, we are able to structure a numerical algorithm that uses DE in a novel way to solve the control problem. One of the advantages of the approach presented here over other methods for solving control problems found in the literature is that this method can handle a large problem since DE is well-suited for parallel computing [42]. Only a few methods in the literature use DE to solve optimal control problems. As an advantage over them, the procedure here employs multiple applications of DE to handle the various constraints, and the process itself eliminates infeasible solutions [41] gradually improving the accuracy and speed of algorithm over previous researchers. Effectively, the state constrained problem has been converted into one without the constraints decreasing the time spent by the DE algorithm to handle constraints. To the best of my knowledge, DE has not been applied previously in the way that is used in this work to solve optimal control problems.

- **Goal 4**: Here a novel and effective method is presented for predicting missing links in a social network. We take the solution of the constrained MOCP to be an observed structure of the social network. Using the network along with the parameters and data that led to its formation, we construct and solve a new constrained MOCP in order to discover any missing or hidden links that may be present in the network [41]. There are some advantages to this approach over those in the literature that rely heavily on network topology for link prediction. Within the MOCP framework, nodal attributes and past history is considered for link formation. This approach does not depend on a given network structure and is capable of uncovering the qualitative, not just topological, reasons underlying link predictions unlike some other methods in the literature.

1.3 Organization of the Dissertation

In Chapter 2, we cover basic mathematical preliminaries to include notation, definitions, and theory needed to set the foundation for important concepts like *Pareto dominance, Pareto optimality, and Pareto front* which we explore in this dissertation. In Chapter 3, we characterize a solution to the constrained optimal control problem as well as the multiobjective optimal control problem (MOCP) by establishing necessary and sufficient conditions of Pareto optimality. We formulate a small MOCP to represent a small network and provide numerous methods and algorithms to generate Pareto-optimal solutions to the problem. In Chapter 4, we state the methodology for social network analysis defining key terminology in order to lay a solid foundation for discussion in the remainder of the work. In Chapter 5, we give a detailed description of our social forces model for network dynamics which can be represented by a MOCP; the model explains social interaction and the basis for friendship choice. Then, using an optimal solution to this MOCP, we form a social matrix to identify social ties within a social group. We show that our model exhibits actual characteristics indicative of real-life social networks such as self-organization, i.e., clustering or clique formation. In Chapter 6, we add *memory effect* to the social forces model to show the impact that long-term memory has on actors' friendship choices. Furthermore, we explore the model's capability to answer questions concerning clique connection and network destabilization. In Chapter 7, we uncover missing links via a constrained MOCP which is formed using an observed snapshot of the network. In addition, we explore clique expansion and forced clique infiltration. Finally, in Chapter 8, we discuss future research stemming from ideas presented in this work.

Chapter 2

Theoretical Basics of Multiobjective Optimization

2.1 Introduction

We use the term multiobjective optimization (sometimes referred to as *vector optimization*) to describe the process of minimizing or maximizing multiple objective functions simultaneously. Multiobjective problems typically have an infinite number of solutions and it is left to a decision maker to decide on which solution is "best". In this chapter, basic concepts of multiobjective optimization are introduced to the reader. Simple examples are used to explain important concepts like Pareto dominance, Pareto front, and Edgeworth-Pareto optimal points, which serve as solutions to multiobjective optimization problems (MOPs). We also introduce numerical methods for solving such problems. In this work, problems of minimization are the primary focus which is acceptable since if one desires to maximize a function, one simply needs to minimize the negative of the function [17].

Definition 2.1 *In general, a multiobjective optimization problem (MOP) has the form*

$$\min_{\boldsymbol{u} \in U} \ \boldsymbol{J}(\boldsymbol{u}) \tag{2.1}$$

where $\mathbf{u}$ is a vector of decision variables $[\mathbf{u}_1, ..., \mathbf{u}_m]^T$ and $\mathbf{J}(\mathbf{u})$ is a vector of objective functions $[J_1(\mathbf{u}), ..., J_s(\mathbf{u})]^T$. Here $J_1(\mathbf{u}), ..., J_s(\mathbf{u})$ are real-valued functions and U is nonempty

in $\mathbb{R}^m$. For now, we leave the constraint set and feasible region open to be described in more detail later.

2.2 Pareto Optimality

When attempting to solve MOPs, the concept of Pareto optimality is very important. It helps to clarify what it means to "minimize" a vector of objective functions. Problems having multiple objectives often have conflicting objectives and it is nearly impossible to satisfy all objectives at once. Since the ideal solution which we label as *utopia* in Figure 2.1 is unattainable, a realistic expectation is to find a set of Pareto optimal solutions that satisfy the objectives. Pareto optimality was first introduced by Edgeworth in 1881 and then further refined by Vilfredo Pareto, an engineer/economist back in 1866, which explains why sometimes we see the term Edgeworth-Pareto optimal. The concept of Pareto optimization is centered on Pareto dominance. In short, a solution dominates another if none of its objective function or fitness values are higher and at least one is less. A Pareto optimal solution (sometimes referred to as *strongly efficient*, *noninferior* or *nondominated*) is one that is not dominated by any other feasible solution.

The set of Pareto optimal solutions form a Pareto front, which is a hypersurface (sometimes called a *tradeoff surface*) in the objective function space. It separates the feasible region from the nonfeasible region. Therefore, the optimum for a multiobjective problem is not a single solution. Instead it is a set of Pareto optimal solutions, which are described as those solutions that are dominated by no other feasible solution. In essence, there exists no other solution, that yields a better value for at least one objective, and is equal or better in the other objective values [52]. The formal definitions for these Pareto optimality concepts follow.

Basic Concepts and Definitions

Coello [11] and Miettinen [43] use several important definitions in their explanation of multiobjective optimization. These concepts are presented graphically in Figure 2.1 [1].

Assumption 2.1 *Let U be a nonempty subset of $\mathbb{R}^m$ and let $\boldsymbol{J} : \mathcal{U} \rightarrow \mathbb{R}^s$ be a given vector function. Assume $\mathbb{R}^s$ is partially ordered in a natural way [27]: if $\boldsymbol{v}_1, \boldsymbol{v}_2 \in \mathbb{R}^s$ with $\boldsymbol{v}_1 = [v_{11}, ..., v_{1s}]^T$ and $\boldsymbol{v}_2 = [v_{21}, ..., v_{2s}]^T$, then $\boldsymbol{v}_1 \leq \boldsymbol{v}_2$ iff $v_{1i} \leq v_{2i}, i = 1, ..., s$.*

Definition 2.2 (Pareto Optimality) *Suppose Assumption 2.1 is satisfied. Then $\boldsymbol{u}^* \in U$ is said to be a Pareto optimal solution of the MOP in (2.1) if there is no $\boldsymbol{u} \in U$ such that $J_i(\boldsymbol{u}) \leq J_i(\boldsymbol{u}^*)\ \forall\ i \in \{1, ..., s\}$ and $J_i(\boldsymbol{u}^*) \neq J_i(\boldsymbol{u})$ for at least one $i \in \{1, ..., s\}$.*

Definition 2.3 (Pareto Dominance) *A vector $\boldsymbol{w} = (w_1, ..., w_s)$ is said to dominate (denoted by $\preceq$) $\boldsymbol{v} = (v_1, ..., v_s)$ if and only if $\boldsymbol{w}$ is partially less than $\boldsymbol{v}$, i.e., $\forall\ i \in \{1, ..., s\}, w_i \leq v_i$ and $\exists\ i \in \{1, ..., s\} : w_i < v_i$.*

Definition 2.4 (Pareto Optimal Set) *For a given MOP, the Pareto optimal set ($\mathcal{PS}(\Omega)$) is defined as:*

$$\mathcal{PS}(\Omega) \equiv \left\{ \boldsymbol{u} \in \Omega |\ \text{there does not exist}\ \ \boldsymbol{u}' \in\ \Omega\ \ \text{for which}\ \boldsymbol{J}(\boldsymbol{u}') \preceq \boldsymbol{J}(\boldsymbol{u}) \right\}.$$

Definition 2.5 (Locally Pareto Optimal) *A decision vector $\boldsymbol{u}^* \in U$ is locally Pareto optimal if there exist an $\epsilon > 0$ such that $\boldsymbol{u}^*$ is Pareto optimal in $U \cap B(\boldsymbol{u}^*, \epsilon)$ where $B(\boldsymbol{u}^*, \epsilon)$ is a ball of radius ϵ centered at $\boldsymbol{u}^*$.*

Definition 2.6 (Pareto Front) *For a given MOP and Pareto optimal set $\mathcal{PS}(\Omega)$, the Pareto front (PF^*) is defined as:*

$$PF^* \equiv \{ \boldsymbol{z} = \boldsymbol{J} = (J_1(\boldsymbol{u}), ..., J_s(\boldsymbol{u})) | \boldsymbol{u} \in \mathcal{PS}(\Omega) \}.$$

[1] This figure was modified from the original version found in "Multiobjective Optimization in Engineering Design" by Johan Andersson, (2001) [2]

Many of the problems in this work are too complex to offer an analytic solution; therefore, we must seek a solution using numerical techniques. Whenever numerical computing is involved, the generated solutions will be at best only locally optimal. Henceforth, in this work, when we refer to Pareto optimal points generated by numerical algorithms, we mean locally Pareto optimal points.

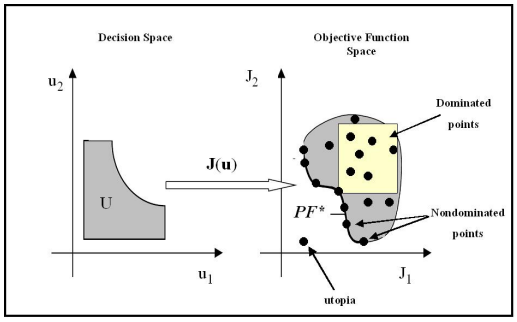


Figure 2.1: Evaluation Mapping of Multiobjective Problem

2.3 Weighted-Sum Method

The weighted-sum of objective functions approach [27] is the most common one used for solving MOPs. It transforms the problem into one with only a single objective function which can then be solved by existing classical methods. We simply take each objective function and assign them various coefficients or weights, $\alpha_i > 0$ and $\sum \alpha_i = 1$, then sum them to get a new function:

$$\min_{\mathbf{u} \in U} \sum_{i=1}^{s} \alpha_i J_i(\mathbf{u}) \tag{2.2}$$

Weights are usually assigned according to preferences; that is, they are assigned based on the level of importance placed on the particular objectives under consideration. Techniques used to assign the weights help to classify multiobjective optimization methods and they are as follows [46]:

1. *a priori* - they are assigned in advance of the optimization based on the expertise of a decision maker.

2. progressive - the weights are updated during the optimization process using feedback from the solutions as they evolve.

3. *a posteriori* - at the end of the optimization process, a decision maker selects a single solution from the generated set of Pareto optimal points, thus selecting its corresponding weights.

Theorem 2.1 *Suppose Assumption 2.1 holds and let* $\alpha_1 > 0, ..., \alpha_s > 0$ *be given real numbers. If* $\boldsymbol{u}^* \in U$ *is a solution of the scalar optimization problem*

$$\min_{\boldsymbol{u} \in U} \sum_{i=1}^{s} \alpha_i J_i(\boldsymbol{u})$$

then $\boldsymbol{u}^*$ *is a Pareto optimal solution for the multiobjective problem (2.1) [27].*

Proof Let $\mathbf{u}^* \in U$ be a solution to the single-objective problem (2.2) with positive weights. However, suppose that it is not Pareto optimal. If this is the case, then there exists some $\mathbf{u}$ such that $J_i(\mathbf{u}) \leq J_i(\mathbf{u}^*) \, \forall \, i = 1, ..., s$ and $J_i(\mathbf{u}) < J_i(\mathbf{u}^*)$ for at least one $i \in \{1, ..., s\}$. Recall $\alpha_i > 0 \, \forall \, i = \{1, ..., s\}$, so $\sum_{i=1}^{s} \alpha_i J_i(\mathbf{u}) < \sum_{i=1}^{s} \alpha_i J_i(\mathbf{u}^*)$. This contradicts the assumption that $\mathbf{u}^*$ is a solution to the posed problem; and therefore, it must be Pareto optimal [43].

Example 2.1 *We use the following test problem*[2] *to illustrate the weighted-sum method and some basic Pareto optimality concepts [12]. In this example, the MOP takes the specific form:*

$$\min_{u} \boldsymbol{J} = [J_1, J_2]^T$$

with

$$J_1(u) = u^2 \tag{2.3a}$$

$$J_2(u) = (u - 2)^2 \tag{2.3b}$$

and $u \in [0, 2]$.

[2]This problem is often referred to as "Schaffer's F2" and can be found in [12].

To solve the problem, we take the two objective functions and make them into the single objective function:

$$\begin{aligned} J(u) &= \alpha_1 \cdot J_1(u) + \alpha_2 \cdot J_2(u) \\ &= \alpha_1 \cdot (u^2) + \alpha_2 \cdot ((u-2)^2) \end{aligned} \tag{2.4}$$

Minimizing this new function using several different pairs of weights produces the results listed in Table 2.1[3]. Each set of weights corresponds to a specific nondominated point on the Pareto front.

Table 2.1: Objective Function Values for Various Sets of Weights, $\boldsymbol{\alpha}$

α_1	α_2	$\mathbf{u}^*$	$J_1(\mathbf{u}^*)$	$J_2(\mathbf{u}^*)$
0.2	0.8	1.6	2.56	0.16
0.4	0.6	1.2	1.44	0.64
0.6	0.4	0.8	0.64	1.44
0.8	0.2	0.4	0.16	2.56

In Figure 2.2, we plot the values for J_1 and J_2 from Table 2.1 to show the tradeoff surface or Pareto front. Notice that solutions from each pair of weights coincides with a single point on the Pareto front. To get these points, four different pairs of weights were selected *a priori* and then the weighted-sum approach was applied four times. Obviously, if there comes a time when a larger set of Pareto optimal points is needed, the weighted-sum approach may prove too tedious and computationally expensive. Therefore, a numerical procedure that generates a whole set of nondominated points at once with less computational effort is highly desired.

2.4 Numerical Methods

In this section, we cover numerical procedures to solve both the scalar-valued and vector-valued cost problems. We explain the steps of each method in detail including the various advantages and limitations of each method. While there are many numerical

[3]Table 2.1 was reproduced from results found in [12]

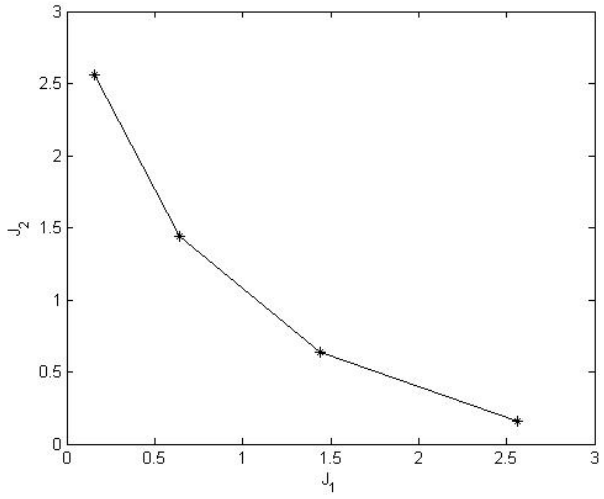


Figure 2.2: Pareto Optimal Points on the Pareto Front

techniques in the literature that could be used to solve these problems, the methods chosen are fairly simple to implement and are believed to give good results for the problems under consideration in this work.

2.4.1 Steepest Descent Method

The Steepest Descent Method is one such classical method that can be used to minimize the scalar-valued cost used in the weighted-sum problem in (2.2). It is based on the fact that the direction of steepest descent is in the direction of the negative gradient of the function. To minimize $J(\mathbf{u})$, the Steepest Descent Method as described by Gopal [17] employs the following steps:

Algorithm 2.1

- **Step 1:** Given $J(\mathbf{u})$, find $\frac{\partial J}{\partial \mathbf{u}}$ analytically.
- **Step 2:** Guess $\mathbf{u}^{(0)}$ and evaluate the search direction $\frac{\partial J}{\partial \mathbf{u}}$ at $\mathbf{u}^{(0)}$. If some norm of $\frac{\partial J}{\partial \mathbf{u}}$

is less than some small ϵ, then stop and output the optimal $\mathbf{u}$. Otherwise, proceed to the next step.

- **Step 3:** Move iteratively toward the minimum, by updating $\mathbf{u}$ according to the rule

$$\mathbf{u}^{(i+1)} = \mathbf{u}^{(i)} + \upsilon_i \frac{\partial J}{\partial \mathbf{u}}(\mathbf{u}^{(i)})$$

where the step size $\upsilon_i > 0$ is chosen in a manner that decreases J. Afterwards, Step 2 is repeated until the stopping criteria is met.

Convergence

The steepest descent method has been shown to be globally convergent. This implies the norm of $\frac{\partial J}{\partial \mathbf{u}}(\mathbf{u}^{(i)})$ converges to zero given any initial point $\mathbf{u}^{(0)}$. However, this result does not imply that convergence to a minimum is absolute. See [28] for a proof of this result.

Limitations

While advantages of the method are its ease of use and requirement for only first derivative information, this method does have its disadvantages. It works well for functions that are well-behaved; however, when dealing with more complex functions, the method exhibits some limitations. For instance, if $J(\mathbf{u})$ mimics an elongated valley, the method results in a zigzag and may be very slow to converge to the minimum or it may simply zigzag from side to side and not reach the minimum at all [17].

2.4.2 Differential Evolution Algorithm

In the past, researchers have used a variety of evolutionary and genetic algorithms to solve vector or multiobjective optimization problems [11]. Evolutionary algorithms (EA) are especially suitable for solving such problems because they can simultaneously handle a set of possible points which form the Pareto optimal set in a single run of the algorithm. Differential Evolution (DE) created by Price and Storm [52] is one such evolutionary algo-

rithm used to solve multiobjective optimization problems chosen in particular for its ease of use and fast convergence.

DE has proven very successful on a myriad of optimization problems in various applications to include neural networks and aerodynamic shapes [26] to name a few. It is a stochastic population-based direct search method and improves the existing solution through *mutation*, *crossover*, and *selection*. The algorithm includes the following steps.

Algorithm 2.2

- **Step 1: Initialization of Population**

  We start the algorithm by initializing the population, but first, upper and lower boundaries must be set for each vector coordinate. For the initial generation, g, each coordinate, i, of every vector, $\mathbf{u}_j^g \in \mathbb{R}^D$, is then randomly initialized within these specified boundaries. For instance, in generation g, the i-th coordinate of the j-th vector is initialized as follows:

  $$u_{j,i}^g = u_{j,i_{min}}^g + rand() * (u_{j,i_{max}}^g - u_{j,i_{min}}^g), \tag{2.5}$$

  where $rand()$ is a uniformly distributed random number $\in [0, 1)$ and $u_{j,i_{min}}^g$ and $u_{j,i_{max}}^g$ are lower and upper bounds respectively on the i-th component of the j-th vector, $j = 1, 2, ..., NP$.

- **Step 2: Mutation**

  After population initialization, the population undergoes mutation. For each population vector, $\mathbf{u}_j^g$, $j = 1, ..., NP$, DE generates NP mutated vectors, $\hat{\mathbf{z}}_j^g$:

  $$\hat{\mathbf{z}}_j^g = \mathbf{u}_{j_1}^g + W * (\mathbf{u}_{j_2}^g - \mathbf{u}_{j_3}^g) \tag{2.6}$$

  where j_1, j_2, j_3 are random mutually different vectors belonging to $\{1, 2, ..., NP\}$ and not equal to vector j. The parameter $W > 0$ is a real and constant scaling factor that usually belongs to $(0, 1)$ and controls the population's evolution rate. While there is no upper bound on W, values of W greater than 1.0 are rarely effective [52].

- **Step 3: Crossover**

  After mutation, DE performs crossover, sometimes referred to as discrete recombination, to increase the diversity of the coordinate variables. In essence, DE crossover develops the trial vectors, $\mathbf{z}_j^g$, from the coordinates of the three different vectors, $\mathbf{u}_{j1}^g, \mathbf{u}_{j2}^g, \mathbf{u}_{j3}^g$ involved in mutation or from the corresponding parent vector, $\mathbf{u}_j^g$ [52]. Each of the vectors is an element of $\mathbb{R}^D$. The crossover rate CR belongs to $[0, 1]$ and decides whether the trial vector gets its coordinates from the mutated vector or the parent vector using the formula,

$$z_{j,i}^g = \begin{cases} u_{j1,i}^g + W * (u_{j2,i}^g - u_{j3,i}^g) & \text{if } rand() < CR \text{ or } i = \hat{i}, \\ u_{j,i}^g & \text{otherwise.} \end{cases} \tag{2.7}$$

  where $rand()$ is a random number in $[0, 1]$ and $\hat{i}$ is a randomly selected index from $\{1, 2, ..., D\}$ [9].

- **Step 4: Selection**

  If the trial vector, $\mathbf{z}_j^g$, yields an objective function value that is less than or equal to that of the target vector, $\mathbf{u}_j^g$, then $\mathbf{z}_j^g$ is selected for the next generation; otherwise, the target vector moves forward to the next generation. Recombination and selection are accomplished to determine which vectors move forward to the next generation, $g + 1$. Each trial vector is compared against the target vector from which it gets its coordinate values:

$$\mathbf{u}_j^{g+1} = \begin{cases} \mathbf{z}_j^g & \text{if } J(\mathbf{z}_j^g) \leq J(\mathbf{u}_j^g), \\ \mathbf{u}_j^g & \text{otherwise} \end{cases} \tag{2.8}$$

  This process of mutation, recombination and selection are repeated until an optimal solution is found or some termination criteria is satisfied.

- **Step 5: Termination**

  Finally, we must terminate the algorithm. A very common termination criteria used in the literature is g_{max}, which is a total number of generations not to exceed. However, the operator must ensure that g_{max} is set high enough to achieve convergence based

on his desired level of accuracy. Often when there is only a single objective function, J, to be minimized, the termination criteria used is $|J_{best} - J_{worst}| < Tol$, where J_{best} and J_{worst} are respectively the best and worst objective function values obtained in a single generation and Tol is some small value representing the desired accuracy.

Implementation

It is interesting to note that DE's control variables, NP, W and CR, are not difficult to choose in order to obtain good results. According to recommendations from the literature [52], [33], [32], [31] and our experimentation, a reasonable choice for NP is between $2D$ and $10D$ but NP must be at least 4 to ensure that DE will have enough mutually different vectors with which to work. As for W, $W = 0.5$ is usually a good initial choice. If the population converges prematurely, then W and/or NP should be increased. Values of W smaller than 0.4 and greater than 1, are only occasionally effective according to the literature. A good first choice for CR is 0.1, but since a large CR often speeds convergence, to first try CR equal to 0.9 or 1.0 is appropriate to see if a quick solution is possible [52].

Convergence

Under certain conditions such as the ability to apply mutation and recombination to a decision vector along with monotonicity of the generated population sequence, Veldhuizen successfully proved that the probability of an EA converging to the global optimum of an MOP is equal to one. The reader is referred to [56] for the proof of this result.

Limitations

While DE is very easy to implement and has proved effective on a vast array of problems, DE can sometimes suffer similar issues as with other evolutionary and genetic algorithms used to solve multiobjective problems. These problems include:

- The estimated Pareto front may not be close to the true Pareto front.
- DE may not find enough Pareto optimal solutions.

- The Pareto optimal set does not cover the entire Pareto front.
- The set of Pareto optimal points is not evenly distributed along the Pareto front.
- Unlike single objective optimization, it's difficult to know when to end the search for the optimal set for the multiobjective problem; termination criteria may be hard to establish especially given that the population is not expected to converge.

Next, we give two examples to illustrate the effectiveness of the Differential Evolution algorithm on multiobjective problems with vector-valued cost.

Example 2.2 *Again, consider the single dimension problem used previously:*

$$\min_u \boldsymbol{J} = [J_1, J_2]$$

with

$$J_1(u) = u^2$$

$$J_2(u) = (u-2)^2$$

and $u \in [0, 2]$.

Generated from a single run of DE, Figure 2.3 plots 200 nondominated points that form the Pareto front for Example 2.2. Clearly, this method surpasses the weighted-sum approach which can be both time-consuming and computationally expensive.

Example 2.3 *Consider the following two-dimensional MOP with two conflicting objectives:*

$$\min_{\boldsymbol{u}} \boldsymbol{J} = [J_1, J_2]$$

with

$$J_1(\boldsymbol{u}) = u_0^2 + u_1$$

$$J_2(\boldsymbol{u}) = u_0 + u_1^2$$

subject to

$$-10 \leq u_0, u_1 \leq 10.$$

Figure 2.4 plots 200 nondominated solutions for Example 2.3. The distribution of these solutions in the objective function space approximates the Pareto front as shown in Figure 2.4. Notice that in this particular example, solutions also form a front in the decision space (see Figure 2.5) but in general this is not the case. The Pareto front is characteristic of the objective function space not the decision space.

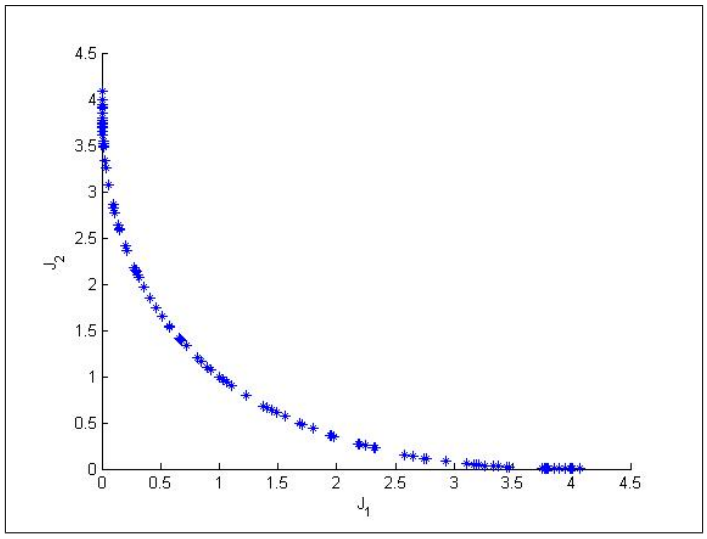


Figure 2.3: Pareto Front with 200 Pareto Optimal Points

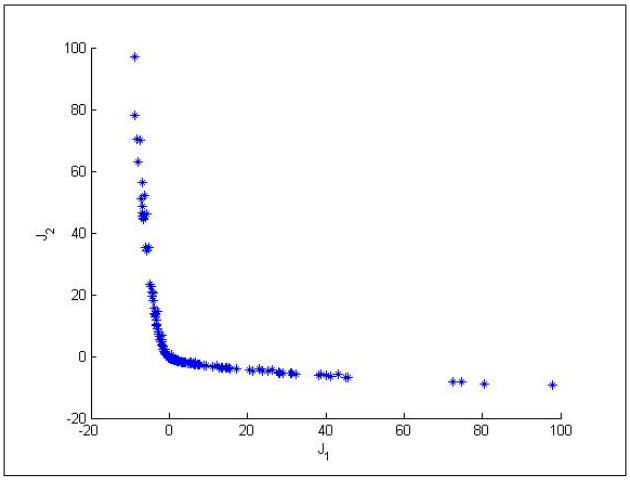


Figure 2.4: Pareto Front in Objective Function Space

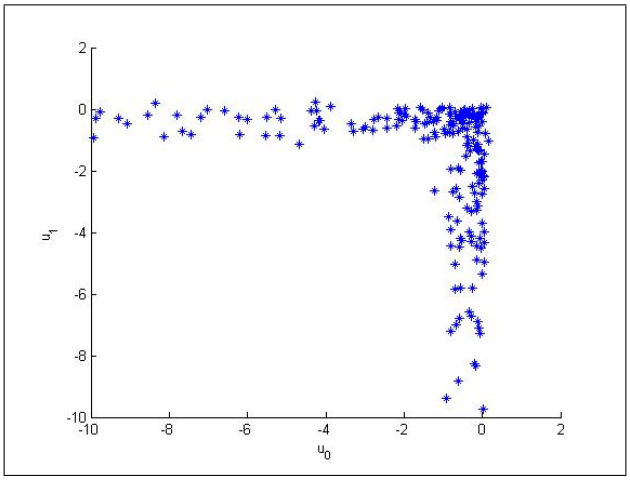


Figure 2.5: Set of Nondominated Points in Decision Parameter Space

Chapter 3

Multiobjective Optimal Control

3.1 Introduction

We turn our attention to more complex and dynamic problems which will be the main focus of this study. From hereafter, we will deal strictly with MOPs whose constraints are given by a system of differential equations along with some simple bounds on the state and control vectors. Such problems are referred to as constrained **multiobjective optimal control problems (MOCP)**.

3.2 Formulation of the Optimal Control Problem

Suppose the scalar cost or performance index (also known as objective function) to be minimized takes the form [17], [29]:

$$J(\mathbf{u}) = \Phi(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t))dt,$$

where L is a continuous and differential real-valued function in $\mathbf{x}$ and $\mathbf{u}$.

Consider the system described by

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \qquad \mathbf{x}(0) = \mathbf{x}^0$$

where each component of $\mathbf{f}$ is C^2 on $\mathbb{R}^n \times \mathbb{R}^m$ with initial conditions prescribed at $\mathbf{x}(0) \in \mathbb{R}^n$.

The state vector $\mathbf{x}(t)$ is an $(n \times 1)$ vector and its inequality constraints are of the form:

$$\boldsymbol{\eta}(\mathbf{x}(t)) \leq 0$$

where $\boldsymbol{\eta}$ is a $(p \times 1)$ vector function with $p \leq n$ and each component continuously differentiable in $\mathbf{x}$. The control vector $\mathbf{u}$ is an $(m \times 1)$ vector and its inequality constraints are of the form:

$$\mathbf{u}(t) \in U$$

where U is a closed and bounded interval in $\mathbb{R}^m$. The class $U \in \mathbb{R}^m$ is the set of all control functions which satisfy the problem constraints, termed *admissible controls*.

Definition 3.1 *A point $\boldsymbol{u}^*$ is a global minimizer if $J(\boldsymbol{u}^*) \leq J(\boldsymbol{u})$ for all $\boldsymbol{u} \in U$ [44].*

Definition 3.2 *A point $\boldsymbol{u}^*$ is a local minimizer if there is a neighborhood B of $\boldsymbol{u}^*$ such that $J(\boldsymbol{u}^*) \leq J(\boldsymbol{u})$ for all $\boldsymbol{u} \in B$.*

Now, we can formulate the optimal control problem:

$$\min_{\mathbf{u} \in U} J(\mathbf{u}) = \Phi(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t))dt \tag{3.1a}$$

s.t.

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \qquad \mathbf{x}(0) = \mathbf{x}^0 \tag{3.1b}$$

$$\mathbf{u}(t) \in U \tag{3.1c}$$

$$\boldsymbol{\eta}(\mathbf{x}(t)) \leq 0 \tag{3.1d}$$

In essence, we want to find $\mathbf{u}^*$ such that $J(\mathbf{u}^*) \leq J(\mathbf{u}) \ \forall \ \mathbf{u} \in U$. These controls are the *optimal controls* and we assume that at least one optimal solution, $(\mathbf{u}^*(t), \mathbf{x}^*(t))$ exists. The reader should consult [4], [19] for a discussion on the existence of optimal controls.

3.3 Formulation of the Multiobjective Optimal Control Problem

Whereas the previous optimal control problem had a scalar-valued cost function, the multiobjective optimal control problem has a vector-valued cost function and can be

loosely posed as follows:

$$\min_{\mathbf{u}\in U} \mathbf{J}(\mathbf{u}) = [J_1(\mathbf{u}), ..., J_s(\mathbf{u})]^T, \quad s \geq 2 \tag{3.2a}$$

s.t.

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \qquad \mathbf{x}(0) = \mathbf{x}^0 \tag{3.2b}$$

$$\boldsymbol{\eta}(\mathbf{x}(t)) \leq 0 \tag{3.2c}$$

$$\mathbf{u}(t) \in U \tag{3.2d}$$

with

$$J_i(\mathbf{x}^0, \mathbf{u}(t), \mathbf{x}(t)) = \Phi_i(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} L_i(\mathbf{x}(t), \mathbf{u}(t))dt \tag{3.2e}$$

Here $\mathbf{J}$ is a $(s \times 1)$ vector of objective functions to be minimized where L_i are continuous and differential real-valued functions in $\mathbf{x}$ and $\mathbf{u}$ and f_i are C^2 on $\mathbb{R}^n \times \mathbb{R}^m$ with initial conditions prescribed at $\mathbf{x}(0) \in \mathbb{R}^n$. The state constraint vector function, $\boldsymbol{\eta}$, is $(p \times 1)$ with each component continuously differentiable in $\mathbf{x}$.

3.4 Pareto Optimality

Most likely the objective functions in (3.2a) will be competing objectives which will make it difficult to minimize them all at once; yet, if it happens that a single solution is found for the MOCP, then the objectives are really not competing after all. That said, since no single minimum is likely to be found, the concept of *optimality* for multiobjective optimal control problems with vector-valued cost must be defined. Once again, our definition of optimality in the multiobjective framework is **Pareto optimality**.

A solution $\mathbf{u}^*$ dominates $\mathbf{u}$ if and only if $J_i(\mathbf{u}^*) \leq J_i(\mathbf{u})\ \forall\ i \in \{1, 2, ..., s\}$ and $J_i(\mathbf{u}^*) < J_i(\mathbf{u})$ for at least one $i \in \{1, 2, ..., s\}$. The set of nondominated points from the search space form Pareto front or Pareto optimal set.

Definition 3.3 *For a given vector of objective or cost functions* $\boldsymbol{J}(\boldsymbol{u}) = [J_1(\boldsymbol{u}), J_2(\boldsymbol{u}), ..., J_s(\boldsymbol{u})]^T$, *the control* $\boldsymbol{u}^*$ *is* ***Pareto optimal*** *if there does not exist*

$\boldsymbol{u}$ such that

$$J_i(\boldsymbol{u}) \leq J_i(\boldsymbol{u}^*)$$

and for at least one i, $i \in \{1, 2, ..., s\}$, we get

$$J_i(\boldsymbol{u}) < J_i(\boldsymbol{u}^*).$$

3.4.1 Necessary Conditions for Pareto Optimality

In order for $\mathbf{u}^*$ to be Pareto optimal at $\mathbf{x}^0$, there are some conditions which must be satisfied.

Theorem 3.1 *If the control $\boldsymbol{u}^*(t) : [t_0, t_f] \to \mathbb{R}^m$, generating the solution $\boldsymbol{x}^*(t) : [t_0, t_f] \to \mathbb{R}^n$, $\boldsymbol{x}^*(t_0) = \boldsymbol{x}^0$, is Pareto optimal at $\boldsymbol{x}^0$, then it is optimal at $\boldsymbol{x}^0$ for the system with scalar-valued cost*

$$J_i(\boldsymbol{x}^0, \boldsymbol{u}(t), \boldsymbol{x}(t)), \quad i \in \{1, 2, ..., s\}$$

and subject to isoperimetric constraints

$$J_j(\boldsymbol{x}^0, \boldsymbol{u}^*(t), \boldsymbol{x}^*(t)) \leq J_j(\boldsymbol{x}^0, \boldsymbol{u}(t), \boldsymbol{x}(t)),$$

$j = 1, 2, ..., s$ and $j \neq i$.

Proof Assume the theorem is not true. This implies there exists a $\mathbf{u} \in U$ and a corresponding $\mathbf{x}$ and some $i \in \{1, 2, ..., s\}$ so that

$$J_i(\mathbf{x}^0, \mathbf{u}(t), \mathbf{x}(t)) < J_i(\mathbf{x}^0, \mathbf{u}^*(t), \mathbf{x}^*(t))$$

and

$$J_j(\mathbf{x}^0, \mathbf{u}(t), \mathbf{x}(t)) \leq J_j(\mathbf{x}^0, \mathbf{u}^*(t), \mathbf{x}^*(t))$$

$j = 1, 2, ..., s$ and $j \neq i$. However, this contradicts the fact that $\mathbf{u}^*(t)$ is Pareto optimal at $\mathbf{x}^0$.

3.4.2 Sufficient Conditions for Pareto Optimality

In this section we introduce two lemmas and a theorem from Leitmann [35], [57] which imply the sufficiency for Pareto optimality. Note that this characterization of sufficient conditions is for the unconstrained case. However, adding state constraints, as we do later, does not result in an essential difference.

Lemma 3.1 *The solution $\boldsymbol{u}^*(t)$ producing the trajectory $\boldsymbol{x}^*(t)$ is Pareto optimal at $\boldsymbol{x}^0$ if a constant $\alpha \in \mathbb{R}^s$ exists with $\alpha_i > 0$ for $i = 1, 2, ..., s$ and $\sum_{i=1}^{s} \alpha_i = 1$, such that*

$$\sum_{i=1}^{s} \alpha_i J_i(\boldsymbol{x}^0, \boldsymbol{u}^*(t), \boldsymbol{x}^*(t)) \leq \sum_{i=1}^{s} \alpha_i J_i(\boldsymbol{x}^0, \boldsymbol{u}(t), \boldsymbol{x}(t)) \tag{3.3}$$

for every $\boldsymbol{u}(t) \in U$ producing the solution $\boldsymbol{x}(t)$.

Proof Let's consider a control $\mathbf{u}(t) \in U$. If the equality in (3.3) holds, then it must be true that either

$$J_i(\mathbf{x}^0, \mathbf{u}^*(t), \mathbf{x}^*(t)) = J_i(\mathbf{x}^0, \mathbf{u}(t), \mathbf{x}(t)) \ \ \forall \ \ i \in \{1, 2, ..., s\}$$

or there is an i and $j \in \{1, 2, ..., s\}$, $i \neq j$ for which

$$J_i(\mathbf{x}^0, \mathbf{u}^*(t), \mathbf{x}^*(t)) < J_i(\mathbf{x}^0, \mathbf{u}(t), \mathbf{x}(t))$$

and

$$J_j(\mathbf{x}^0, \mathbf{u}(t), \mathbf{x}(t)) < J_j(\mathbf{x}^0, \mathbf{u}^*(t), \mathbf{x}^*(t))$$

If the inequality in 3.3 holds, then there exists an $i \in \{1, 2, ..., s\}$ for which

$$J_i(\mathbf{x}^0, \mathbf{u}^*(t), \mathbf{x}^*(t)) < J_i(\mathbf{x}^0, \mathbf{u}(t), \mathbf{x}(t))$$

and we satisfy the the conditions for a control to be Pareto-optimal at $\mathbf{x}^0$.

Lemma 3.2 *The solution $\boldsymbol{u}^*(t)$ producing the trajectory $\boldsymbol{x}^*(t)$ is Pareto optimal at $\boldsymbol{x}^0$ if a constant $\boldsymbol{\alpha} \in \mathbb{R}^s$ exists with $\alpha_i \geq 0$ for $i = 1, 2, ..., s$ and $\sum_{i=1}^{s} \alpha_i = 1$, such that*

$$\sum_{i=1}^{s} \alpha_i J_i(\boldsymbol{x}^0, \boldsymbol{u}^*(t), \boldsymbol{x}^*(t)) < \sum_{i=1}^{s} \alpha_i J_i(\boldsymbol{x}^0, \boldsymbol{u}(t), \boldsymbol{x}(t)) \ \forall \ \boldsymbol{u}(t) \in U, \boldsymbol{u}(t) \neq \boldsymbol{u}^*(t). \tag{3.4}$$

Given these lemmas, the sufficiency conditions for Pareto optimality have been successfully reduced to sufficient conditions for an optimum of the optimal control problem (see section 3.2) with scalar-valued cost:

$$\sum_{i=1}^{s} \alpha_i J_i(\mathbf{x}^0, \mathbf{u}(t), \mathbf{x}(t)). \tag{3.5}$$

If there exists $\boldsymbol{\alpha} \in \mathbb{R}^s$ and if the scalar-valued objective function (3.5) has an optimum $\mathbf{u}^*$ at $\mathbf{x}^0$, then for the problem with a vector-valued objective function in (3.2a), $\mathbf{u}^*$ is also Pareto optimal at $\mathbf{x}^0$.

In our previous discussion, we took a set of weights $\alpha_1 > 0, ..., \alpha_k > 0$, $\sum_{i=1}^{k} \alpha_i = 1$ and used them to sum the multiple objectives into a new scalar cost which we minimized to find a Pareto optimal solution. Now we address the question of whether or not all Pareto optimal solutions can be found in such a manner. That is, if we have a given Pareto optimal solution, can we say with assurance that it was generated using a particular set of weights, $\alpha_1 > 0, ..., \alpha_k > 0,\ \sum_{i=1}^{k} \alpha_i = 1$.

Let

$$J_i(\phi, u) = \Phi_i(\phi(t_f), t_f) + \int_{t_0}^{t_f} L_i(\phi(t), u(t), t)dt \tag{3.6a}$$

s.t.

$$\dot{\phi}(t) = f(\phi(t), u(t), t) \ a.e., \qquad \phi(0) = \phi^0 \tag{3.6b}$$

$$u(t) \in U \ a.e. \tag{3.6c}$$

$$\eta(\phi(t)) \leq 0 \tag{3.6d}$$

Assumption 3.1

1. The function $f : X \times \mathcal{U} \times [t_0, t_f] \to \mathbb{R}^n$ (X is an open interval in $\mathbb{R}^n$), U an open interval in $\mathbb{R}^{mk}$) is differentiable in x and continuous in x and u, for each fixed t. It is measurable in t, for each fixed $(x, u) \in X \times \mathcal{U}$.

2. For each compact set $\Gamma \subset X \times \mathcal{U}$, $\exists$ a function $\Lambda \in L_2([t_0, t_f])$ such that $\forall\ (x, u, t) \in$

$\Gamma \times [t_0, t_f]$,

$$|f(x,u,t)| + |f_1(x,u,t)| \leq \Lambda(t),$$

where $f_1(x,u,t)$ denotes the matrix of the partial derivatives of f with respect to x. Here $|\cdot|$ denotes the Euclidean norm of the vector or matrix in question.

3. The function $\eta : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable.

4. To generate all admissible trajectories that lie in a fixed compact set $\acute{X} \subset X$, we require that, $\forall\ (x,u) \in \acute{X} \times \mathcal{U}$, we have

$$|x^T f(x,u,t)| \leq W(t)(1+|x|^2), W \in L_1([t_0,t_f])$$

5. We assume that $u(t) \in U$ *a.e.*, where $U = U_1 \times \cdots \times U_k$ and $U_i, i = 1, ..., k$ is fixed compact subset of $\mathbb{R}^m$ such that $U \subset \mathcal{U}$.

6. The functions h_1, h_2 are C^1 functions on $\mathbb{R}^n$.

Definition 3.4 *A relaxed admissible pair (ϕ, μ) (see section 3.7) is an absolutely continuous function ϕ and a relaxed control μ defined on $[t_0, t_f]$ such that*

$$\dot{\phi}(t) = f(\phi, \mu_t, t)$$
$$h_1(\phi(t_0)) = 0 = h_2(\phi(t_f)) = 0$$
$$\eta(\phi(t) \leq 0).$$

We denote the class of admissible pairs by *Ad*.

Lemma 3.3 *Suppose $Ad \neq \emptyset$. Let (ϕ_n, μ_n) be a minimizing sequence. Then, $\exists\ n_1 < n_2 < \cdots$; ϕ^* and μ^* s.t. $\phi_n \to \phi^*$ uniformly, $\phi'_{n_j} \to \phi^{*\prime}$ weakly in $(L_2([t_0,t_f]))^n$ and $\mu_{n_j} \to \mu^*$ weak-star and $(\phi^*, \mu^*) \in Ad$.*

Proof The proof is straightforward and is ommitted.

For $\alpha_1 > 0, ..., \alpha_k > 0,\ \alpha_1 + \cdots + \alpha_k = 1$ consider the problem

$$\min\{\alpha_1 J_1(\phi,\mu) + \cdots + \alpha_k J_k(\phi,\mu)\} \tag{3.7a}$$

subject to

$$\dot{\phi}(t) = f(\phi, \mu_t, t) \tag{3.7b}$$

$$\eta(\phi(t)) \leq 0 \tag{3.7c}$$

$$\text{support}(\mu_t) \subset U(t) \tag{3.7d}$$

$$h_1(\phi(t_0)) = 0 = h_2(\phi(t_f)) = 0 \tag{3.7e}$$

Lemma 3.4 *Suppose* (ϕ^*, μ^*) *is optimal for (3.7). Then, it must be a Pareto optimal solution.*

Proof Suppose it were not. Then, $\exists\ (\tilde{\phi}, \tilde{\mu})$ and $1 \leq i_1 \leq k$ such that

$$J_i(\tilde{\phi}, \tilde{\mu}) \leq J_i(\phi^*, \mu^*),$$

and

$$J_{i_1}(\tilde{\phi}, \tilde{\mu}) < J_{i_1}(\phi^*, \mu^*).$$

Then $(\tilde{\phi}, \tilde{\mu})$ would solve (3.7) forming a lower payoff.

Note that the vector $(\alpha_1, ..., \alpha_k)^T$ separates the sets

$$\mathcal{Q} = \{(J_1(\phi,\mu), \cdots, J_k(\phi,\mu))^T |\ (\phi,\mu) \in Ad\}$$

from the convex set

$$\mathbb{R}^k_- + (J_1(\phi^*,\mu^*), \cdots, J_k(\phi^*,\mu^*))^T,$$

where $\mathbb{R}^k_- = \{(x_1, ..., x_k)^T |\ x_i < 0, i = 1, ..., k\}$. Denote by $\mathcal{PS}$ the set of all $(\phi,\mu) \in Ad$ which are Pareto optimal solutions. Let $(\phi^*,\mu^*) \in \mathcal{PS}$. We note that

$$\mathcal{Q} \cap [\mathbb{R}^k_- + (J_1(\phi^*,\mu^*), \cdots, J_k(\phi^*,\mu^*))^T] = \emptyset.$$

If

$$[\mathbb{R}^k_- + (J_1(\phi^*,\mu^*),\cdots,J_k(\phi^*,\mu^*))^T]$$
$$\cap\,[cl\ co\{(J_1(\phi,\mu),\cdots,J_k(\phi,\mu))^T|(\phi,\mu)\in Ad\}]=\emptyset, \tag{3.8}$$

then $\exists\ \lambda_1,...,\lambda_k\geq 0,\ \sum_{i=1}^k\lambda_i=1$ such that

$$\sum_{i=1}^k\lambda_i J_i(\phi,\mu)\geq\sum_{i=1}^k\lambda_i J_i(\phi^*,\mu^*). \tag{3.9}$$

However, (3.8) is too strong. We now proceed to prepare for a similar but local result. Given two relaxed controls $\nu,\tilde{\nu}$, we define the norm

$$\|\nu-\tilde{\nu}\|_L=\sup\{|\nu_t-\tilde{\nu}_t|(U):\ t_0\leq t\leq t_f\}.$$

We denote the set of relaxed controls by $\mathcal{S}$ with the norn $\|\cdot\|_L$ and by H^1 the set of absolutely continuous functions with square integrable derivative on $[t_0,t_f]$.

Lemma 3.5 *Let $(\phi^*,\mu^*)\in\mathcal{PS}$ and $N(\phi^*,\mu^*)$ a neighborhood of (ϕ^*,μ^*) in $H^1\times\mathcal{S}$. Suppose that*

$$cl\ co\{(J_1(\phi,\mu),\cdots,J_k(\phi,\mu))^T|(\phi,\mu)\in Ad\cap N(\phi^*,\mu^*)\}$$
$$\cap\,[\mathbb{R}^k_- + (J_1(\phi^*,\mu^*),\cdots,J_k(\phi^*,\mu^*))^T]=\emptyset.$$

Then, $\exists\ \lambda=(\lambda_1,...,\lambda_k),\ \lambda_k\geq 0,\ \lambda_1+\cdots\lambda_k=1$ such that $\forall\ (\phi,\mu)\in Ad$ we have

$$\sum_{i=1}^k\lambda_i J_i(\phi,\mu)\geq\sum_{i=1}^k\lambda_i J_i(\phi^*,\mu^*).$$

Proof Using separation theorem for convex sets we can verify $\exists\ (\xi_1,...,\xi_k),\ \xi_i\geq 0, i=1,...,k;\ \xi_1+\cdots+\xi_k\neq 0$ such that

$$\sum_{i=1}^k\xi_i J_i(\phi,\mu)\geq\sum_{i=1}^k\xi_i J_i(\phi^*,\mu^*).$$

Dividing the two sides of this inequality by $\xi_1+\cdots+\xi_k$ and setting $\lambda_i=\xi_i/\sum_{i=1}^k\xi_i$ we get the desired result.

3.5 Numerical Experiment

The aim of this section is to illustrate the solution methods and assess the performance of the proposed algorithms on a small social network model consisting of just three actors ($N = 3$) and two attributes ($m = 2$). The objective here is not to go into great detail regarding the model's development since we give a thorough explanation of the model in Chapter 5. We simply state the model and assess the performance of the proposed solution methods and algorithms.

In this experiment, a set of nonlinear differential equations of motion is used to describe the social interaction of actors in a social group as though they were subject to physical forces. The objective of each actor is to minimize the distance between himself and others while not compromising his beliefs too much over the fixed time interval from $t_0 = 0$ to $t_f = 1$. Basically, this setup amounts to solving a multiobjective optimal control problem with three conflicting objectives by applying the necessary and sufficient conditions for Pareto optimality discussed earlier. The model variables and parameters are:

Data:

$\mathbf{r}_i(t)$ – position vector describing actor i's preference for each attribute, 1,...,m

$\mathbf{v}_i^0$ – vector of actor i's initial rate of change of attribute preferences at $t = 0$

$\mathbf{v}_i(t)$ – vector of actor i's rate of change of attribute preferences at time t

$\mathbf{u}_i(t)$ – vector of controls for actor i's attribute preferences

Parameters:

l_{ij} – constant value set to ensure that actor j respects the private space of actor i

τ_i – relaxation time of actor i (a measure of how fast he returns to his $\mathbf{v}_i^0$)

$\mathcal{N}_i$ – reflects an actor's desire to stick to his belief system

3.5.1 Problem Statement

Consider a social group with $N = 3$ actors ($i = 1, 2, 3$, $j = 1, 2, 3$, and $j \neq i$) and $m = 2$ attributes. Since there are $m = 2$ attributes, in the following equations, $\mathbf{r}_i$, $\mathbf{v}_i$, and $\mathbf{u}_i$ are all 2-component vectors. The dynamical system which models social interaction is given by

$$\begin{aligned} \dot{\mathbf{r}}_i &= \mathbf{v}_i && \text{(3.10a)} \\ \dot{\mathbf{v}}_i &= \frac{1}{\tau_i}(\mathbf{v}_i^0 - \mathbf{v}_i) - \nabla_{\mathbf{r}_i}\Big[\sum_{j\neq i} \|\mathbf{u}_i - \mathbf{u}_j\|^2 && \text{(3.10b)} \\ &\quad \cdot(1 + ((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\|)^2 - \|\mathbf{v}_j\Delta t\|^2)) \\ &\quad \cdot \exp\{-l_{ij}((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\|)^2 - \|\mathbf{v}_j\Delta t\|^2)\}\Big] \end{aligned}$$

$$\mathbf{r}_i(0) = \mathbf{r}_i^0, \quad \mathbf{v}_i(0) = \mathbf{v}_i^0$$

Constraints on the state and control variables are simple bounds, i.e,

$$\mathbf{r}_i(0) - \boldsymbol{\delta}_{i_{min}} \leq \mathbf{r}_i(t) \leq \mathbf{r}_i(0) + \boldsymbol{\delta}_{i_{max}} \tag{3.10c}$$

$$-\boldsymbol{\delta}_{i_{min}} \leq \mathbf{u}_i(t) \leq \boldsymbol{\delta}_{i_{max}} \tag{3.10d}$$

where l_{ij}, τ_i, and Δt are given parameter values.

Each actor i in the social group wishes to minimize his objective or cost function:

$$J_i = \sum_{j\neq i} \|\mathbf{r}_i(t_f) - \mathbf{r}_j(t_f)\|^2 + \mathcal{N}_i \int_{t_0}^{t_f} \|\mathbf{u}_i(t)\|^2 \, dt \tag{3.10e}$$

where $\mathcal{N}_i$ is a given parameter.

So the goal of this experiment is to find the optimal control, $\mathbf{u}^* = (\mathbf{u}_1^*, \mathbf{u}_2^*, \mathbf{u}_3^*)$, that minimizes $\mathbf{J} = [J_1, J_2, J_3]^T$ subject to the constraints imposed on the system. Let's recall that "minimizing" a vector of objective functions means finding a Pareto optimal set.

It is helpful to assume we know some data in advance. Suppose we have the following parameter choices (see Table 3.1) and initial data for actors ($i = 1, 2, 3$) (see Tables 3.2 and 3.3) with vector component ($k = 1, 2$).

Table 3.1: Parameters for each actor: $i = 1, 2, 3$

i	l_{ij}	τ_i	$\mathcal{N}_i$
1	0.3	1/20	1
2	0.1	1/10	1
3	0.2	1/15	1

Table 3.2: Initial position vector, $\mathbf{r}_i$ for each actor $i = 1, 2, 3$ at $t = 0$

i	r_{i1}	r_{i2}
1	0.2646	0.6325
2	0.6518	0.6491
3	0.1295	0.6518

Table 3.3: Initial rate of change of preferences, $\mathbf{v}_i^0$, for each actor $i = 1, 2, 3$

i	v_{i1}^0	v_{i2}^0
1	0.3	0.3
2	0.3	0.3
3	0.3	0.3

3.5.2 Implementation

In this experiment, we'll use three different techniques to obtain a solution. In the first two methods, we minimize a weighted-sum cost function using a gradient algorithm first and then Differential Evolution. In these methods, we only generate a single solution on the Pareto front. In the third approach, we minimize all three objective functions simultaneously, that is, we seek a Pareto optimal set of solutions using Differential Evolution. The reader is directed to [13], [60], [64] for applications on the use of evolutionary algorithms to solve optimal control problems.

To formulate the problem with weighted-sum cost, we use equal weights

$$\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \alpha_3]^T = \left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right]^T$$

chosen *a priori* as described in Chapter 2. The new single objective optimal control problem is

$$\begin{aligned}\min_{\mathbf{u}} J = &\frac{1}{3}\left\{\|\mathbf{r}_1(t_f)-\mathbf{r}_2(t_f)\|^2+\|\mathbf{r}_1(t_f)-\mathbf{r}_3(t_f)\|^2+\mathcal{N}_1\int_{t_0}^{t_f}\|\mathbf{u}_1(t)\|^2\,dt\right\}\\ &+\frac{1}{3}\left\{\|\mathbf{r}_2(t_f)-\mathbf{r}_1(t_f)\|^2+\|\mathbf{r}_2(t_f)-\mathbf{r}_3(t_f)\|^2+\mathcal{N}_2\int_{t_0}^{t_f}\|\mathbf{u}_2(t)\|^2\,dt\right\}\\ &+\frac{1}{3}\left\{\|\mathbf{r}_3(t_f)-\mathbf{r}_1(t_f)\|^2+\|\mathbf{r}_3(t_f)-\mathbf{r}_2(t_f)\|^2+\mathcal{N}_3\int_{t_0}^{t_f}\|\mathbf{u}_3(t)\|^2\,dt\right\}\end{aligned} \tag{3.11a}$$

s.t.

$$\dot{\mathbf{r}}_i = \mathbf{v}_i \tag{3.11b}$$

$$\begin{aligned}\dot{\mathbf{v}}_i = &\frac{1}{\tau_i}(\mathbf{v}_i^0-\mathbf{v}_i)-\nabla_{\mathbf{r}_i}\Big[\sum_{j\neq i}\|\mathbf{u}_i-\mathbf{u}_j\|^2\\ &\cdot(1+((\|\mathbf{r}_i-\mathbf{r}_j\|+\|\mathbf{r}_i-\mathbf{r}_j-\mathbf{v}_j\Delta t\|)^2-\|\mathbf{v}_j\Delta t\|^2)\\ &\cdot\exp\{-l_{ij}((\|\mathbf{r}_i-\mathbf{r}_j\|+\|\mathbf{r}_i-\mathbf{r}_j-\mathbf{v}_j\Delta t\|)^2-\|\mathbf{v}_j\Delta t\|^2)\}\Big],\end{aligned} \tag{3.11c}$$

$$\mathbf{r}(0)=\mathbf{r}_i^0, \qquad \mathbf{v}(0)=\mathbf{v}_i^0$$

and

$$\mathbf{r}_i(0)-\boldsymbol{\delta}_{i_{min}} \leq \mathbf{r}_i(t) \leq \mathbf{r}_i(0)+\boldsymbol{\delta}_{i_{max}} \tag{3.11d}$$

$$-\boldsymbol{\delta}_{i_{min}} \leq \mathbf{u}_i(t) \leq \boldsymbol{\delta}_{i_{max}}. \tag{3.11e}$$

3.5.3 Solution Method 1: Weighted-Sum Method Using Iterative Method

In this first solution method, since our constraints on the state and control are of simple form, we begin by initializing the controls within their upper and lower bounds and searching the space for only those controls that lead to state variables that do not violate the state constraints. Now we can state the following necessary conditions that solutions of the optimal control problem must satisfy as though there were no restrictions on the

state. For a derivation of these conditions, the reader should consult [17], [36] . We start by forming the Hamiltonian

$$H = L(\mathbf{x}, \mathbf{u}) + \mathbf{p}^T f(\mathbf{x}, \mathbf{u}). \tag{3.12}$$

In the scenario we have described, the necessary conditions for a optimal control solution are

$$\begin{aligned}
\dot{\mathbf{x}} &= H_{\mathbf{p}} && \text{(3.13a)}\\
\mathbf{x}(t_0) &= \mathbf{x}^0 && \text{(3.13b)}\\
\dot{\mathbf{p}} &= -H_{\mathbf{x}} && \text{(3.13c)}\\
H_{\mathbf{u}} &= 0 && \text{(3.13d)}\\
\mathbf{p}(t_f) &= \Phi_{\mathbf{x}}(t_f) && \text{(3.13e)}
\end{aligned}$$

where $t_0 = 0$ and $t_f = 1$.

Now, we form the Hamiltonian for the problem in (3.11). Since we have two sets of state equations, we need two sets of adjoint multipliers, i.e., $\mathbf{p}_1$ and $\mathbf{p}_2$. The Hamiltonian is

$$\begin{aligned}
H &= L(\mathbf{x}, \mathbf{u}) + \mathbf{p}^T f(\mathbf{x}, \mathbf{u})\\
&= \frac{1}{3}\sum_{i=1}^{3} \mathcal{N}_i \left\|\mathbf{u}_i(t)\right\|^2 + \sum_{i=1}^{3} \mathbf{p}_{1,i}^T \mathbf{v}_i\\
&\quad + \sum_{i=1}^{3} \mathbf{p}_{2,i}^T \Bigg[\frac{1}{\tau_i}(\mathbf{v}_i^0 - \mathbf{v}_i) - \nabla_{\mathbf{r}_i}\Big[\sum_{j\neq i} \left\|\mathbf{u}_i - \mathbf{u}_j\right\|^2\\
&\quad \cdot(1 + ((\left\|\mathbf{r}_i - \mathbf{r}_j\right\| + \left\|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\right\|)^2 - \left\|\mathbf{v}_j\Delta t\right\|^2)\\
&\quad \cdot \exp\{-l_{ij}((\left\|\mathbf{r}_i - \mathbf{r}_j\right\| + \left\|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\right\|)^2 - \left\|\mathbf{v}_j\Delta t\right\|^2)\}\Big]\Bigg]
\end{aligned}$$

where the vectors $\mathbf{p}_{1,i}$ and $\mathbf{p}_{2,i}$ have the form

$$\mathbf{p}_{1,i} = [p_{1,i1}, p_{1,i2}]^T \quad \textit{and} \quad \mathbf{p}_{2,i} = [p_{2,i1}, p_{2,i2}]^T.$$

The state equations are

$$\dot{\mathbf{r}}_i = \mathbf{v}_i \tag{3.14}$$

$$\begin{aligned}\dot{\mathbf{v}}_i = {} & \frac{1}{\tau_i}(\mathbf{v}_i^0 - \mathbf{v}_i) - \nabla_{\mathbf{r}_i}\Big[\sum_{j\neq i}\|\mathbf{u}_i - \mathbf{u}_j\|^2 \\ & \cdot(1 + ((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\|)^2 - \|\mathbf{v}_j\Delta t\|^2) \\ & \cdot\exp\{-l_{ij}((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\|)^2 - \|\mathbf{v}_j\Delta t\|^2)\}\Big]\end{aligned} \tag{3.15}$$

$$\mathbf{r}(0) = \mathbf{r}_i^0, \qquad \mathbf{v}(0) = \mathbf{v}_i^0$$

State and control bounds are

$$\mathbf{r}_i(0) - \boldsymbol{\delta}_{i_{min}} \leq \mathbf{r}_i(t) \leq \mathbf{r}_i(0) + \boldsymbol{\delta}_{i_{max}} \tag{3.16}$$

$$-\boldsymbol{\delta}_{i_{min}} \leq \mathbf{u}_i(t) \leq \boldsymbol{\delta}_{i_{max}} \tag{3.17}$$

The costate equations associated with the state equations in (3.11b) - (3.11c) are as follows

$$\begin{aligned}\dot{\mathbf{p}}_{1,i} = {} & -H_{\mathbf{r}_i} \\ = {} & -\mathbf{p}_{2,i}\Big[\nabla^2_{\mathbf{r}_i}\Big[\sum_{j\neq i}\|\mathbf{u}_i - \mathbf{u}_j\|^2 \\ & \cdot(1 + ((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\|)^2 - \|\mathbf{v}_j\Delta t\|^2) \\ & \cdot\exp\{-l_{ij}((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\|)^2 - \|\mathbf{v}_j\Delta t\|^2\}\Big]\Big]\end{aligned} \tag{3.18}$$

$$\begin{aligned}\dot{\mathbf{p}}_{2,i} = {} & -H_{\mathbf{v}_i} \\ = {} & -\mathbf{p}_{1,i} + \frac{1}{\tau_i}\mathbf{p}_{2,i}\end{aligned} \tag{3.19}$$

We also have the boundary conditions

$$\begin{aligned}\mathbf{p}_{1,i}(t_f) &= \nabla_{\mathbf{r}_i}\Big[\sum_{j\neq i}\|\mathbf{r}_i(t_f)-\mathbf{r}_j(t_f)\|^2\Big] \\ &= \nabla_{\mathbf{r}_i}\Big[\frac{1}{3}\{\|\mathbf{r}_1(t_f)-\mathbf{r}_2(t_f)\|^2+\|\mathbf{r}_1(t_f)-\mathbf{r}_3(t_f)\|^2 \\ &\quad +\|\mathbf{r}_2(t_f)-\mathbf{r}_1(t_f)\|^2+\|\mathbf{r}_2(t_f)-\mathbf{r}_3(t_f)\|^2 \\ &\quad +\|\mathbf{r}_3(t_f)-\mathbf{r}_1(t_f)\|^2+\|\mathbf{r}_3(t_f)-\mathbf{r}_2(t_f)\|^2\Big]\end{aligned} \tag{3.20}$$

$$\begin{aligned}\mathbf{p}_{2,i}(t_f) &= \nabla_{\mathbf{v}_i}\Big[\sum_{i=1,\ j\neq i}^{3}\|\mathbf{r}_i(t_f)-\mathbf{r}_j(t_f)\|^2\Big] \\ &= 0\end{aligned} \tag{3.21}$$

Clearly, to find the optimal strategy, we must solve the state and costate equations simultaneously subject to the boundary conditions. This gives rise to a two-point boundary value problem (TPBVP). Unfortunately, it is nearly impossible to find analytical solutions to such problems which have added difficulty due to these state and control constraints. Therefore, we will need to employ some numerical methods to solve the problem. The reader should consult [5] for advice on numerically solving the TPBVP.

Iterative Method for Solving the Two-Point Boundary Value Problem (TPBVP)

To solve the TPBVP, we use a rather straight-forward iterative approach based on gradient methods.

Algorithm 3.1

- **Step 1:** First, we divide the time interval $[t_0,\ t_f]$ into N equal intervals. Then, guess the initial values of the controls at time intervals $[t_0,\ t_1)$, $[t_1,\ t_2)$,..., $[t_{N-1},\ t_f)$ to get $\mathbf{u}^{(i)}(t_k)$, $i=0,\ k=0,1,...,N-1$. Since $\mathbf{u}$ is a $(m\times 1)$ vector, we'll need enough digital memory to store mN values. Note: The initial control is chosen randomly to satisfy the control constraints.

- **Step 2:** Next, use the control history $\mathbf{u}^{(i)}(t_k)$ along with the $\mathbf{x}(t_0) = \mathbf{x}^0$ to numerically integrate the state equations forward on $[t_0, t_f]$ to get and store nN values for $\mathbf{x}^{(i)}(t_k)$, $k = 0, 1, ..., N-1$. Note: If the state constraints are not satisfied, we return to the previous step and continue searching the control space until an appropriate $\mathbf{u}$ is found that ensures all the state constraints are satisfied.

- **Step 3:** Now that $\mathbf{x}^{(i)}(t_k)$ and $\mathbf{u}^{(i)}(t_k)$ have been successfully obtained, use them along with the condition $\mathbf{p}^{(i)}(t_f) = \Phi_{\mathbf{x}}(\mathbf{x}^{(i)})|_{t_f}$ to solve the costate equations by integrating backwards from t_f to t_0.

- **Step 4:** Of course, we need to ensure $H_{\mathbf{u}}^{(i)}(\mathbf{x}^{(i)}(t_k), \mathbf{u}^{(i)}(t_k), \mathbf{p}^{(i)}(t_k), t_k) = 0$ is satisfied. Establish some small ϵ_1 and ϵ_2 for the desired accuracy to be achieved. Then if some norm of $H_{\mathbf{u}}^{(i)}$ is less than the predetermined ϵ_1 and if the norm of the difference of objective function value in two consecutive iterations is less than some predetermined amount, i.e., $\left|J^{(i+1)} - J^{(i)}\right| < \epsilon_2$, we can stop the iterative procedure and output the optimal control strategy and state trajectory. Otherwise, proceed to the next step.

- **Step 5:** Update the controls using $\mathbf{u}^{(i+1)}(t_k) = \mathbf{u}^{(i)}(t_k) - \upsilon_i H_{\mathbf{u}}^{(i)}(t_k)$ where $\upsilon_i > 0$ is chosen to decrease H and return to the second step and repeat the rest of the procedure until the conditions in Step 4 are met. Note: If the updated control does not satisfy the control and state constraints, we keep updating it until we find an appropriate control.

In this approach, the fourth-order Runge Kutta method was used to integrate the state equations forward and costate equations backwards. The stopping criteria [17], [58] for Step 4 was $\left\|\frac{\partial H}{\partial \mathbf{u}}\right\| \leq 10^{-5}$ and $\left|J^{(i+1)} - J^{(i)}\right| \leq 10^{-8}$. The parameter Δt is a small time interval which is taken to be the same size as the time step in the numerical integration. With this stopping criteria, the Algorithm 3.1 was implemented in C++ and took approximately 16 hours to get the optimal value $J^* = 0.2251$. Table 3.1 holds the parameters values that were used in solving the problem, and Figure 3.1 plots the optimal trajectories of the attribute preferences.

If we calculate the distance between actor preferences using Euclidean distance, $d_{ij} = \|\mathbf{r}_i - \mathbf{r}_j\| = \sqrt{\sum_{k=1}^{m}(r_{ik} - r_{jk})^2}$ and use the average distance between actor preferences as a benchmark for closeness, we have the following results shown in Figure 3.2. Actors 1 and 3 are mutually close; actors 1 and 2 are not considered close, and actors 2 and 3 are not considered close.

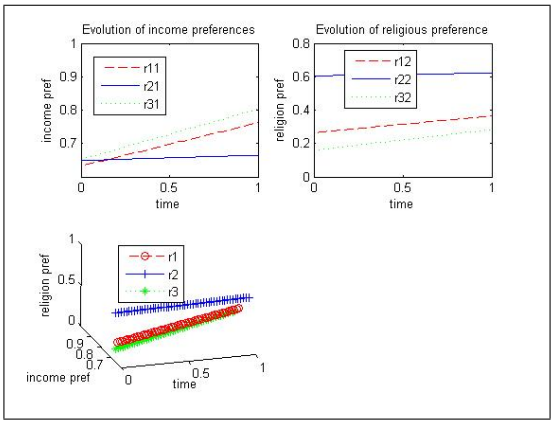


Figure 3.1: Optimal Trajectory - Actor Preferences

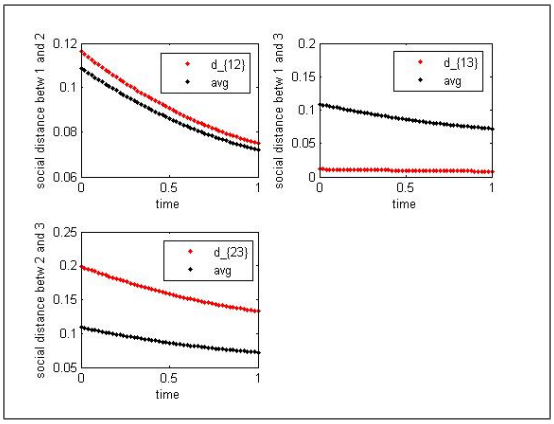


Figure 3.2: Distance between actor preferences from Solution Method 1

3.5.4 Solution Method 2: Weighted-Sum Method Using Differential Evolution

The same weighted-sum optimal control problem is solved again but this time using Differential Evolution as outlined in Algorithm 2.2. The parameters and stopping criteria are carefully selected to ensure good results. In selecting them, we used recommendations from the literature as well as our own observations of how to get convergence throughout hundreds of algorithmic runs.

1. **DE parameters:** $NP = 360,\ W = 0.5,\ \text{and}\ CR = 0.5$
2. **Termination Criteria:** $|J^g_{best} - J^g_{worst}| < 10^{-5}$ or $g_{max} = 100,000$

To initialize the population, we search the decision space for appropriate controls that satisfy both the control and state constraints. During selection, we do not allow trial vectors to move to next generation if they do not satisfy the control and state constraints. The algorithm was run in C++ and took approximately 4.5 hours to generate the optimal value $J^* = 0.00075$. Obviously, this method takes much less time than the gradient method based on Steepest Descent and it achieves a much lower cost function value.

Figure 3.3 plots the evolution of the the objective function value over generations. Notice that at first there is a very steep decrease in the objective function value, then the decline becomes more gradual. From the graph, it looks as though little improvement is made after generation number $70,000$; perhaps if the algorithm had been stopped there, time could have been saved by not having to run another $30,000$ generations. Of course that is a judgement call that must be made based on the desired level of accuracy.

Figure 3.4 plots the optimal trajectory. Again if we use the average distance between actors as a benchmark for closeness, Differential Evolution produces the results as shown in Figure 3.5. Actors 1 and 2 are mutually close; actors 1 and 3 are mutually close, and actors 2 and 3 are not considered close.

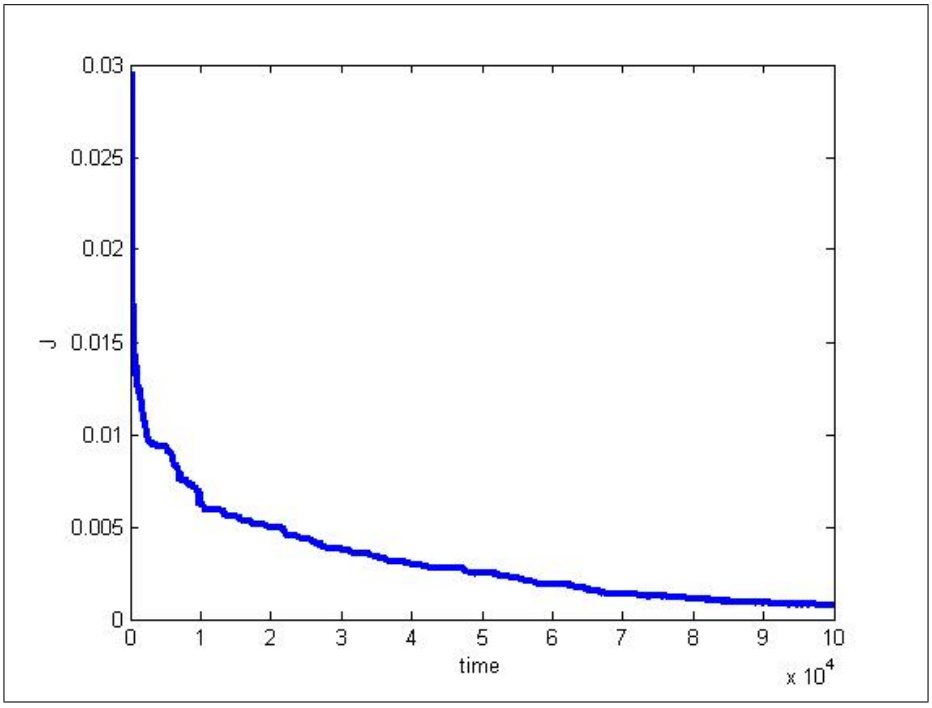


Figure 3.3: Evolution of the Best Objective Function Value

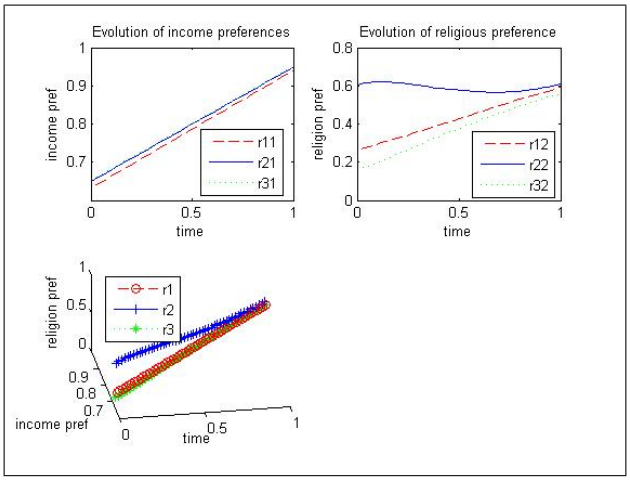


Figure 3.4: Optimal Trajectory - Actor Preferences

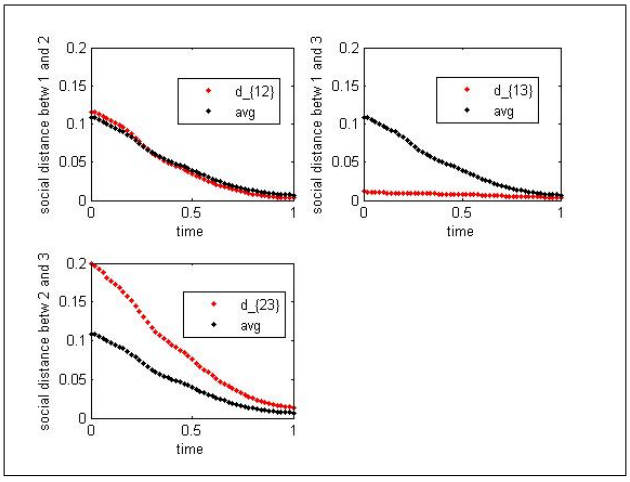


Figure 3.5: Distance between actor preferences from Solution Method 2

3.5.5 Solution Method 3: Pareto Optimization Using Differential Evolution

In this last solution approach, we use Differential Evolution to generate a whole set of Pareto optimal points to solve the multiobjective optimal control problem with vector-valued cost, $\mathbf{J} = [J_1, J_2, J_3]^T$:

$$\min_{\mathbf{u}} \mathbf{J} = [J_1, J_2, J_3]^T$$

where

$$\begin{aligned} J_1 &= \|\mathbf{r}_1(t_f) - \mathbf{r}_2(t_f)\|^2 + \|\mathbf{r}_1(t_f) - \mathbf{r}_3(t_f)\|^2 \\ &\quad + \mathcal{N}_1 \int_{t_0}^{t_f} \|\mathbf{u}_1(t)\|^2 \, dt \\ J_2 &= \|\mathbf{r}_2(t_f) - \mathbf{r}_1(t_f)\|^2 + \|\mathbf{r}_2(t_f) - \mathbf{r}_3(t_f)\|^2 \\ &\quad + \mathcal{N}_2 \int_{t_0}^{t_f} \|\mathbf{u}_2(t)\|^2 \, dt \\ J_3 &= \|\mathbf{r}_3(t_f) - \mathbf{r}_1(t_f)\|^2 + \|\mathbf{r}_3(t_f) - \mathbf{r}_2(t_f)\|^2 \\ &\quad + \mathcal{N}_3 \int_{t_0}^{t_f} \|\mathbf{u}_3(t)\|^2 \, dt \end{aligned}$$

with $t_0 = 0$ and $t_f = 1$ and the aforementioned control and state constraints must be satisfied.

Specifically, we seek a whole set of Pareto optimal points, where $\mathbf{u}^* = (\mathbf{u}_1^*, \mathbf{u}_2^*, \mathbf{u}_3^*)$ is Pareto optimal if there does not exist a feasible $\mathbf{u} = (\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$ such that

$$\begin{aligned} J_1(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3) &\leq J_1(\mathbf{u}_1^*, \mathbf{u}_2^*, \mathbf{u}_3^*), \\ J_2(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3) &\leq J_2(\mathbf{u}_1^*, \mathbf{u}_2^*, \mathbf{u}_3^*), \\ J_3(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3) &\leq J_3(\mathbf{u}_1^*, \mathbf{u}_2^*, \mathbf{u}_3^*), \end{aligned}$$

and for at least one $j \in \{1, 2, 3\}$, we get

$$J_j(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3) < J_j(\mathbf{u}_1^*, \mathbf{u}_2^*, \mathbf{u}_3^*).$$

We used Algorithm 2.2 to solve this problem with the following parameters and stopping criteria:

1. **DE Parameters:** $NP = 3000$, $W = 0.5$, and $CR = 0.5$

2. **Termination Criteria:**

   If the difference in the average of the objective function values between generations is less than some predetermined amount or if we reach a maximum number of generations not to exceed, we stop the algorithm:

$$\sum_{i=1}^{3} \left| \frac{J_i^{(k)}(\mathbf{u}^{(1)}) + \cdots + J_i^{(k)}(\mathbf{u}^{(NP)})}{NP} - \frac{J_i^{(k-1)}(\mathbf{u}^{(1)}) + \cdots + J_i^{(k-1)}(\mathbf{u}^{(NP)})}{NP} \right| < 10^{-5}$$

   or $g_{max} = 10,000$.

We created this stopping criteria to drive the problem toward convergence and ensure a good solution. The control and state constraints were handled by initializing the population only with controls that satisfy the constraints; we did not allow trial vectors that violated the constraints to move forward during selection. We used the fourth-order Runge Kutta method to integrate the state equations forward given some initial point. We implemented the algorithm in C++ and it took approximately 10.5 hours to solve the problem. The algorithm was attempted with larger values for NP, more time steps and values necessary to achieve more accuracy; however, limited computing resources (time and memory) prevent a solution under these stricter requirements.

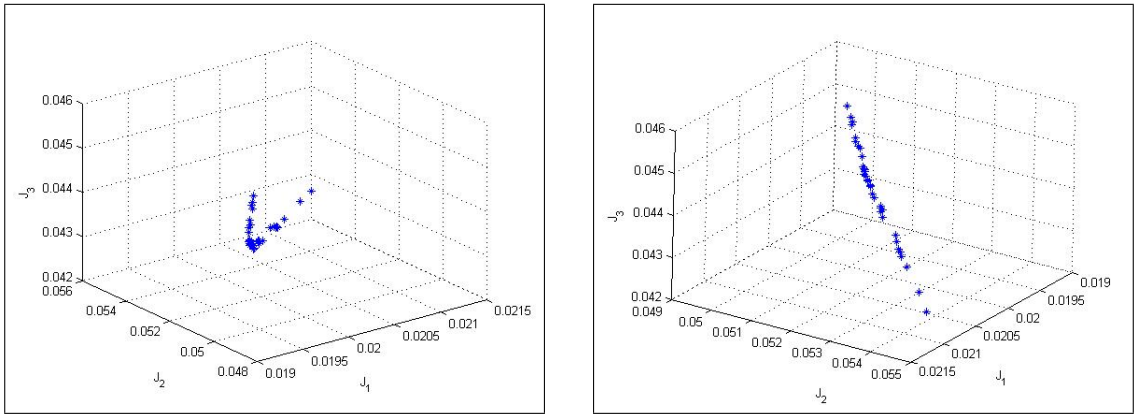
Using these DE settings, we were able to obtain numerous points on the Pareto front. Table 3.4 shows a set of objective function values generated by the nondominated set of points. When moving from one point in the table to another, we can clearly see that while some may improve, at least one fitness value is always degraded. Clearly, each of these points meets the definition of nondominated or Pareto optimal since all the objectives cannot be improved at once without worsening at least one objective.

Figure 3.6 below shows different views of the the Pareto front. As expected, we see that the Pareto front is a hypersurface (or tradeoff surface) formed in the objective function

space. We choose an arbitrary Pareto optimal point, $\mathbf{u}^*$, from this Pareto front and use it to plot the optimal trajectory in Figure 3.7 and the distance between actor preferences in Figure 3.8.

Table 3.4: Objective Function Values for a Subset of Nondominated Points

Objective	$\mathbf{u}^{(1)}$	$\mathbf{u}^{(2)}$	$\mathbf{u}^{(3)}$	$\mathbf{u}^{(4)}$	$\mathbf{u}^{(5)}$
J_1	0.0195	0.0204	0195	0.0199	0.0210
J_2	0.0501	0.0530	0.0506	0.0516	0.0544
J_3	0.0449	0.0431	0.0438	0.0434	0.0431



(a) View 1: Pareto Front

(b) View 2: Pareto Front

Figure 3.6: Multiple Views of the Pareto Front

Using the average distance between actors as a benchmark for closeness, we have: actors 1 and 3 are mutually close, actors 1 and 2 are mutually close, and actors 2 and 3 are not considered close. In fact, actor 1 makes the most friends!

In Figure 3.9, the solution produced by the Weighted-Sum Method is compared with a Pareto optimal solution generated by DE. In comparing the two graphs, it appears that actors would be "happier" with the choice of equal α_i in Figure 3.9(a) than with the unknown α_i in Figure 3.9(b).

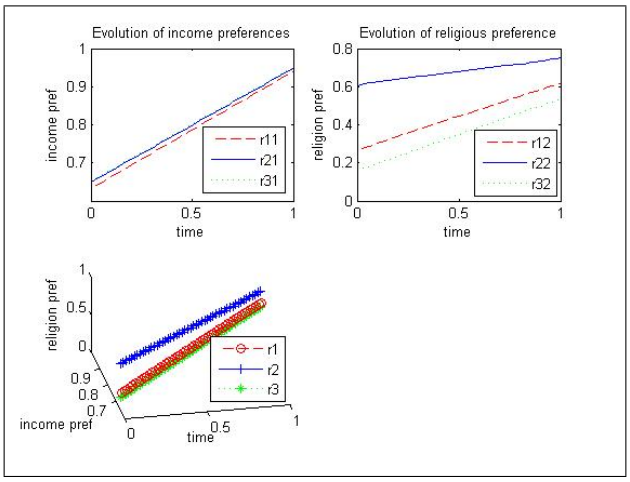


Figure 3.7: Optimal Trajectory - Actor Preferences

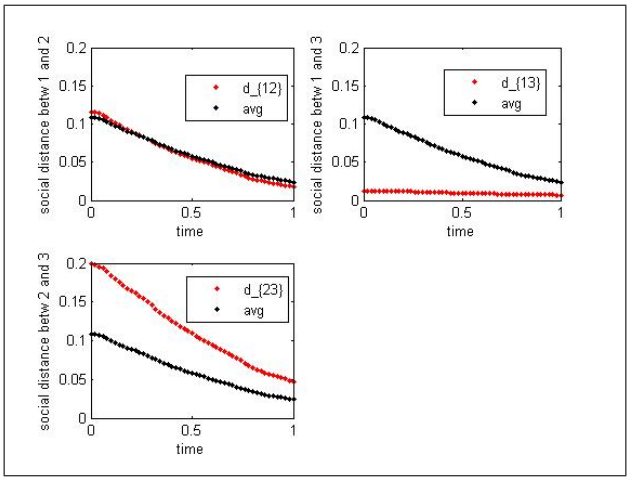
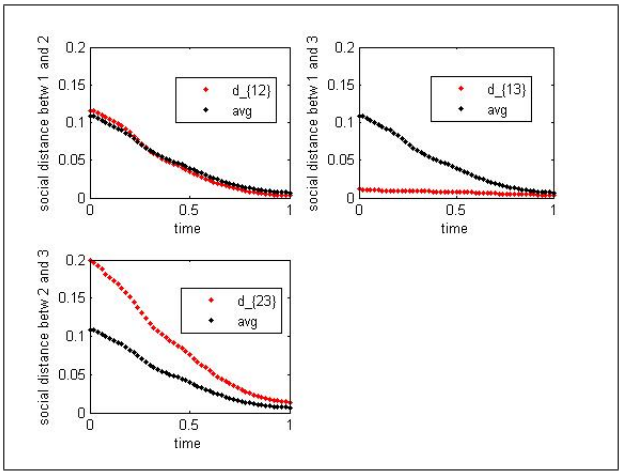
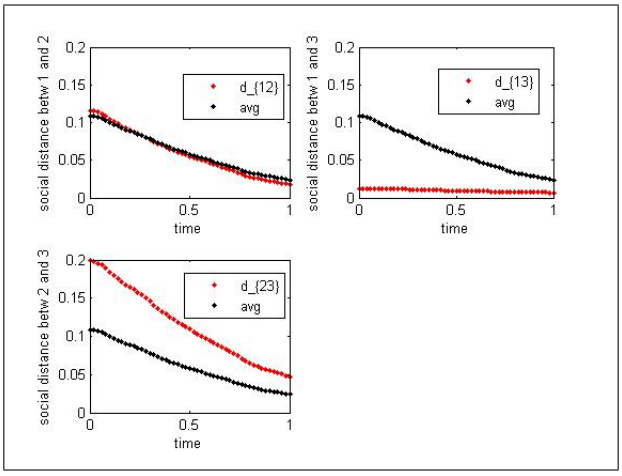


Figure 3.8: Distance between actor preferences from Solution Method 3



(a) Weighted-Sum Solution with $\boldsymbol{\alpha} = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})^T$ using DE



(b) Pareto Front Solution with unknown $\boldsymbol{\alpha}$ using DE

Figure 3.9: Comparison of solutions generated by DE

3.6 Solving Constrained Optimal Control Problems

Until now, we have taken advantage of the fact that our state constraints have been of simple form. However, we need to consider what might happen with more complex constraints. In this section, using the necessary conditions for an optimum, we are able to structure a numerical algorithm that uses DE in a novel way to solve the state constrained multiobjective optimal control problem. One of the advantages of the approach presented here is that we can handle a large problem since DE is well-suited for parallel computing [42]. Further, the procedure employs multiple applications of DE to handle the various constraints, and the process itself eliminates infeasible solutions [41] gradually improving the accuracy and speed of DE as applied by previous researchers [13], [60]. Effectively, the state constrained problem has been converted into one without the constraints decreasing the time spent by the DE algorithm to handle constraints. Also, this method improves accuracy. To see this, one can start with this method followed by a direct method.

3.7 Relaxed Controls

In general, an optimal control problem may not have a solution. Thus, one may have to consider a relaxed version of the control problem. That is, we must use relaxed controls as shown below. For a comprehensive theory of existence, we refer to Berkovitz and Medhin [4]. We use their text to offer the following important definitions and concepts as it relates to the relaxed controls and trajectories.

The function μ which is a *probability measure* on K, a compact set, "is a positive regular measure on the Borel sets of K such that $\mu(K) = 1$".

Definition 3.5 *A relaxed control on $[t_0, t_f]$ is a function*

$$\boldsymbol{\mu} : t \to \boldsymbol{\mu}_t \quad a.e.$$

where $\boldsymbol{\mu}_t$ is a probability measure on $U(t)$ such that for every continuous function g defined

on $[t_0, t_f] \times U$*, the function* h *defined by*

$$h(t) = \int_{U(t)} g(t, \boldsymbol{z}) d\boldsymbol{\mu}_t$$

is Lebesgue measurable.

For our purposes U is independent of t and a compact set.

An ordinary control $\mathbf{u} : [t_0, t_f] \to \mathbb{R}^m$ is a measurable function and it corresponds to the relaxed control $\delta_{\mathbf{u}(t)}$ which is the Dirac measure concentrated at $\mathbf{u}(t) \in U$. Thus,

$$g(t, \mathbf{u}(t)) = \int_U g(t, \mathbf{z}) d\delta_{\mathbf{u}(t)}(\mathbf{z})$$

is a measurable function of t. "Thus the mapping $t \to \delta_{u(t)}$ is a relaxed control." Therefore, we consider ordinary controls to be special types of relaxed controls.

Now we present relaxed controls which are not ordinary controls. Take the functions $\pi_1, ..., \pi_{n+1}$ to be nonnegative and measurable on $[t_0, t_f]$ such that $\sum_{i=1}^{n+1} \pi_i = 1$ and take the functions $\mathbf{u}_1, ..., \mathbf{u}_{n+1}$ to be measurable on $[t_0, t_f]$ such that $\mathbf{u}_i(t) \in U(t)$, where $U(t)$ is a Borel set in $\mathbb{R}^m$. Let

$$\boldsymbol{\mu}_t = \sum_{i=1}^{n+1} \pi_i(t) \delta_{\mathbf{u}_i(t)}.$$

Then μ_t is a probability measure, and $\boldsymbol{\mu} : t \to \boldsymbol{\mu}_t$ is a relaxed control since

$$\int_U g(t, \mathbf{z}) d\boldsymbol{\mu}_t(\mathbf{z}) = \sum_{i=1}^{n+1} \pi_i(t) g(t, \mathbf{u}_i(t))$$

is Lebesgue measurable. Instead of

$$\int_U g(t, \mathbf{z}) d\boldsymbol{\mu}_t(\mathbf{z}),$$

we simply write $g(t, \boldsymbol{\mu}_t)$. So if

$$\mu_t = \sum_{i=1}^{n+1} \pi_i(t) \delta_{\mathbf{u}_i(t)},$$

$$f(t, \boldsymbol{\phi}(t), \boldsymbol{\mu}_t) = \sum_{i=1}^{n+1} \pi_i(t) f(t, \boldsymbol{\phi}(t), \mathbf{u}_i(t)).$$

To μ_t we may associate the vector $(\pi_1(t), ..., \pi_{n+1}(t), \mathbf{u}_1(t), ..., \mathbf{u}_{n+1}(t))$. In fact, every relaxed control $\boldsymbol{\mu}_t$ corresponds to $(\pi_1(t), ..., \pi_{n+1}(t), \mathbf{u}_1(t), ..., \mathbf{u}_{n+1}(t))$ where $\pi_i \geq 0$ *a.e.*, $\sum_{i=1}^{n+1} \pi_i = 1$ *a.e.*, $\mathbf{u}_i \in U$, $i = 1, ..., n+1$.

Definition 3.6 (Relaxed trajectory) *An absolutely continuous function* $\boldsymbol{\phi} = (\phi_1, ..., \phi_n)$ *defined on* $[t_0, t_f]$ *is a relaxed trajectory [4] corresponding to a relaxed control* $\boldsymbol{\mu}$ *if*

1. $(t, \boldsymbol{\phi}(t)) \in \mathbb{R}^m$ *for all* $t \in [t_0, t_f]$,

2. $\boldsymbol{\phi}$ *is a solution to the relaxed differential equation*

$$\dot{\boldsymbol{x}}(t) = \sum_{i=1}^{n+1} \pi_i(t) \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}_i(t))$$

We also write

$$\dot{\mathbf{x}}(t) = \mathbf{g}(\mathbf{x}(t), \boldsymbol{\mu}_t) \quad \text{where} \quad \boldsymbol{\mu}_t = \sum_{i=1}^{n+1} \pi_i(t) \delta_{\boldsymbol{u}_i(t)}$$

as mentioned above.

Definition 3.7 (Admissible pair) *The pair [4]* $(\boldsymbol{\phi}, \boldsymbol{\mu})$ *with a relaxed trajectory* $\boldsymbol{\phi}$ *corresponding to a relaxed control* $\boldsymbol{\mu}$ *is said to be admissible if* $\boldsymbol{\phi}(0) = \boldsymbol{\phi}^0$ *and the function*

$$t \rightarrow \sum_{i=1}^{n+1} \pi_i(t) f^0(\boldsymbol{\phi}(t), \boldsymbol{u}_i(t))$$

is integrable.

We can now state the *relaxed* version [4], [7] of the control problem in (3.1a) - (3.1d):

$$\min_{\boldsymbol{u_i} \in U} \mathbf{J}_r(\mathbf{z}) = \Phi(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} f^0(\mathbf{x}(t), \boldsymbol{\mu_t}) dt \tag{3.22a}$$

s.t.

$$\dot{\mathbf{x}}(t) = \mathbf{g}(\mathbf{x}(t), \boldsymbol{\mu_t}), \qquad \mathbf{x}(0) = \mathbf{x}^0 \tag{3.22b}$$

$$\mathbf{u}_i \in U \tag{3.22c}$$

$$\boldsymbol{\eta}(\mathbf{x}(t)) \leq 0 \tag{3.22d}$$

$$T(\mathbf{x}(t_0), \mathbf{x}(t_f)) = 0 \tag{3.22e}$$

where

$$\boldsymbol{\mu}_t = \sum_{i=1}^{n+1} \pi_i(t)\delta_{\mathbf{u}_i(t)},$$

$$\boldsymbol{u}_i(t) \in U(t) \;\; a.e.\; t, \;\; \pi_i \geq 0 \;\; a.e., \; \sum_{i=1}^{n+1} \pi_i = 1 \quad a.e.$$

Then

$$f^0(\mathbf{x}(t), \boldsymbol{\mu_t}) = \sum_{i=1}^{n+1} \pi_i(t) f^0(\mathbf{x}, \boldsymbol{u}_i(t)), \quad \mathbf{g}(\mathbf{x}(t), \boldsymbol{\mu_t}) = \sum_{i=1}^{n+1} \pi_i(t)\mathbf{g}(\mathbf{x}, \boldsymbol{u}_i(t)).$$

3.7.1 Necessary Conditions

While optimal control problems with inequality constraints on the state variables are common not just in the engineering sciences but occur often in management, economics and even social sciences. Such problems are often difficult to solve and the abundance of various formulations of the necessary and sufficient conditions in the literature adds to the ambiguity and makes it hard to solve practical problems [19]. Fortunately, Berkovitz and Medhin [4] offer the following theorem concerning the necessary conditions for a relaxed pair to be an optimal solution of the control problem described in (3.22a) - (3.22e). We use these conditions to numerically solve the social network problem previously mentioned.

Assumption 3.2 *We make the assumption that there exist $\delta > 0$ such that for $t \in (0, \delta) \cup (t_f - \delta, t_f), 0 < \delta < t_f$, we have $\eta(\phi(t)) < 0$.*

Our control set U is a fixed compact set. Then, with appropriate assumptions on T, η, f^0, f that are easily met in our current problem we have the following theorem [4].

Theorem 3.2 *Suppose Assumption 3.2 holds. In addition, assume that $\nabla_{\boldsymbol{x}}[\eta_l(\phi_0(t))] \neq 0, t_0 \leq t \leq t_f$. At any relaxed pair $(\boldsymbol{\phi}_0, \boldsymbol{z}_0)$ optimal for (3.22a) - (3.22e) the following conditions are met* [1]*: There exists an absolutely continuous function p, a bounded nonincreasing function $\boldsymbol{\lambda} \in \mathbb{R}^p$, $\boldsymbol{\lambda} \geq 0$, $\boldsymbol{\lambda}(t_f^-) = 0$, scalars $\beta, \beta_l, \boldsymbol{\gamma}, \lambda^0, (\lambda^0 \geq 0)$, such that*

1. $|p(t_f)| + \sum_l |\lambda_l(0^+)| + \sum_l |\beta_l| + \lambda^0 \neq 0$,

[1] This theorem has been modified from its original version [4] in a manner conducive to solving the problem under consideration.

2.

$$\begin{aligned}\dot{\boldsymbol{p}}(t) &= \lambda^0 f_x^{0T}(\boldsymbol{\phi}_0(t), \boldsymbol{z}_{0t}) - \boldsymbol{p}(t) \cdot \boldsymbol{g}_{\boldsymbol{x}}(\boldsymbol{\phi}_0(t), \boldsymbol{z}_{0t}) \\ &+ \sum_l \lambda_l(t)[\nabla_{\boldsymbol{x}}[\eta_l(\phi_0(t))]]^T \cdot \boldsymbol{g}_{\boldsymbol{x}}(\boldsymbol{\phi}_0(t), \boldsymbol{z}_{0t}) + \sum \lambda_l(t) d[\nabla_{\boldsymbol{x}}[\eta_l(\phi_0(t))]]^T/dt\end{aligned}$$

3.

$$\begin{aligned}\boldsymbol{p}^T(t_0) &= \sum_l \boldsymbol{\gamma}_l \nabla_{\boldsymbol{x}}[\eta_l(\boldsymbol{\phi}_0(t_f))] \\ &\quad + \sum_l \beta_l[\nabla_{\boldsymbol{x}} T(\boldsymbol{\phi}_0(t_0)) - (\nabla_{\boldsymbol{x}} T(\boldsymbol{\phi}_0(t_0)) \cdot \boldsymbol{u}_{0l})\boldsymbol{u}_{0l}], \\ \boldsymbol{u}_{0l} &= \nabla_{\boldsymbol{x}}[\eta_l(\boldsymbol{\phi}_0(t_0))]/|\nabla_{\boldsymbol{x}}[\eta_l(\boldsymbol{\phi}_0(t_0))]|.\end{aligned}$$

4. $\boldsymbol{p}^T(t_f) = \beta \Phi_x(\boldsymbol{\phi}_0(t_f)) + \sum_l \boldsymbol{\gamma}_l \nabla_{\boldsymbol{x}}[\eta_l(\boldsymbol{\phi}_0(t_f))]$

5.

$$\begin{aligned}&- \left[\boldsymbol{p} - \sum_{l=1}^{p} \lambda_l(t)[\nabla_{\boldsymbol{x}}[\eta_l(\boldsymbol{\phi}_0(t))]]^T\right] \boldsymbol{g}(\boldsymbol{\phi}_0(t), \boldsymbol{z}_{0t}) + \lambda^0 f^0(\boldsymbol{\phi}_0(t), \boldsymbol{z}_{0t}) \\ &\leq - \left[\boldsymbol{p} - \sum_{l=1}^{p} \lambda_l(t)[\nabla_{\boldsymbol{x}}[\eta_l(\boldsymbol{\phi}_0(t))]]^T\right] \boldsymbol{g}(\boldsymbol{\phi}_0(t), \boldsymbol{z}_t) + \boldsymbol{\lambda}^0 f^0(\boldsymbol{\phi}_0(t), \boldsymbol{z}_t) \quad a.e. \quad t\end{aligned}$$

Even after establishing necessary conditions for optimal control problems, the actual determination of the optimal control and trajectory is extremely challenging at best. Therefore, a numerical method is needed to get the solution. Traditionally, optimal control problems have been solved numerically by using methods that rely heavily on gradient information. However, in recent years, researchers have begun to explore evolutionary algorithms, like Differential Evolution (DE), since they eliminate the need for such information. I.L. Cruz (2003) [13] used DE to solve multimodal optimal control problems with great success and Feng-Sheng Wang and Ji-Pyng Chiou (1997) [60] have applied Differential Evolution to solve constrained problems in robotics. We will use this particular evolutionary algorithm to generate a solution for our problem since we believe it has a better chance of reaching a global solution [52] by initially randomly sampling the decision space at multiple points. For an overview of alternative direct and indirect solution methods for optimal control problems as well as explanations of various numerical techniques for solving optimal control problems with state constraints, the reader is directed to [38], [47].

3.8 Numerical Method

In the numerical procedure that follows, notice that solving the optimal control problem for a relaxed pair amounts to replacing the ordinary control $\mathbf{u}^{(i)}$ by the relaxed control,

$$\boldsymbol{\mu}_t = \sum_{l=1}^{n+1} \pi_l \delta_{\mathbf{u}_l(t)}, \quad \sum_{l=1}^{n+1} \pi_l = 1, \quad \pi_l \geq 0.$$

From the necessary conditions above, there exists boundary conditions on the state at t_0 and the costate at t_0 and t_f which gives rise to a multipoint BVP which we must solve subject to the aforementioned constraints on the control and state variables in (3.22c) and (3.22d) respectively. From the necessary conditions, we determine that there are two objectives that we wish to minimize in this problem:

$$\min_{\mathbf{u},\boldsymbol{\lambda}} H$$

and

$$\min_{\boldsymbol{\gamma}} \|\mathbf{p}^T(t_f) - (\beta \Phi_{\mathbf{x}}(\mathbf{x}(t_f)) + \sum_l \boldsymbol{\gamma}_l \nabla_{\mathbf{x}}[\eta_l(\boldsymbol{x}(t_f))])\| < \epsilon.$$

To accomplish this multiobjective minimization, we construct a numerical algorithm which uses Differential Evolution (DE). The necessary conditions will be used to guide our numerical procedure toward an optimum and we describe the procedure as follows. To begin, we determine a population size NP to use with DE. Then, in the initial generation, we initialize each of the NP controls within the predetermined bounds. We also initialize two sets of multipliers, $\boldsymbol{\lambda}$ and $\boldsymbol{\gamma}$, corresponding to each of the NP controls. For every control in the population, we compute the associated state variables. Then using the controls, the associated state variables and multipliers, we get the costate variables. We then check whether the relation in condition four of Theorem 3.2 is satisfied to some desired level of accuracy for each member of the population. If so, we take the associated multipliers to be optimal; otherwise, we update each set of multipliers using the *mutation*, *crossover*, and *selection* steps of Differential Evolution. We then record the values of the Hamiltonian on each time interval as well as the objective function value generated by each member of the

population. Once the stopping criteria for DE is met, we take the control from the Pareto optimal set that provides the lowest objective function value and call it the "best" control. We fix the multipliers associated with this "best" control and then by perturbing it slightly, we hope to see whether or not we can further decrease the values of the Hamiltonian on each time interval using some other feasible control. To do this, we perform DE again initializing the population with controls that are slightly perturbed versions of the "best" control and using the fixed multipliers. However, during DE *selection*, we reject any control that does not lead to the desired satisfaction of condition four of Theorem 3.2 using these fixed multipliers. When some desired stopping criteria is achieved, we stop and output the optimal control.

3.8.1 Numerical Algorithm for Solving the Constrained Optimal Control Problem

To solve the constrained control problem, we design the following algorithm which uses the necessary conditions in Theorem 3.2 and Differential Evolution. Note that in the particular problem we are considering we can use just ordinary controls because the effect of a discretized relaxed control can be approximated to a desired degree of accuracy by a discretized ordinary control.

Algorithm 3.2

- **Step 1:** First, in the initial generation, we initialize a population of NP controls $\mathbf{u}^g = [\mathbf{u}^{(1)}, \mathbf{u}^{(2)}, ..., \mathbf{u}^{(NP)}]$. For the i-th member of the population, $i \in \{1, 2, ..., NP\}$, we divide the time interval $[t_0,\ t_f]$ into N equal intervals and guess the initial values of the i-th control at time intervals $[t_0,\ t_1)$, $[t_1,\ t_2)$,..., $[t_{N-1},\ t_f)$ to get $\mathbf{u}^{(i)}(t_k)$, $k = 0, 1, ..., N-1$. Note: We randomly initialize controls to meet the control constraints in (3.22c) and we only choose controls that produce state variables that do not violate the state constraints in (3.22d). We only force the controls to satisfy the state constraints in the initial generation. In successive generations, the problem is formulated via the necessary conditions to address the state constraints.

- **Step 2:** Next, we use the control history $\mathbf{u}^g(t_k)$, along with the initial condition $\mathbf{x}(t_0) = \mathbf{x}^0$, to numerically integrate the state equations forward on $[t_0, t_f]$ to get $\mathbf{x}^g(t_k) = [\mathbf{x}^{(1)}(t_k), \mathbf{x}^{(2)}(t_k), ..., \mathbf{x}^{(NP)}(t_k)]$, $k = 0, 1, ..., N-1$.

- **Step 3:** In the initial generation, we randomly guess NP sets of $\boldsymbol{\gamma}$ and $\boldsymbol{\lambda}$ values: $\boldsymbol{\gamma}^g = [\boldsymbol{\gamma}^{(1)}, \boldsymbol{\gamma}^{(2)}, ..., \boldsymbol{\gamma}^{(NP)}]$, $k = 0, 1, ..., N-1$ and $\boldsymbol{\lambda}^g(t_k) = [\boldsymbol{\lambda}^{(1)}(t_k), \boldsymbol{\lambda}^{(2)}(t_k), ..., \boldsymbol{\lambda}^{(NP)}(t_k)]$, $k = 0, 1, ..., N-1$. For each set of $\boldsymbol{\lambda}$ multipliers, we satisfy the conditions that $\boldsymbol{\lambda}^{(i)}(t_k) \geq 0$, $i = 1, ..., NP$ and nonincreasing over time and $\boldsymbol{\lambda}^{(i)}(t_f) = 0$. We place no restrictions on $\boldsymbol{\gamma}$ multipliers.

- **Step 4:** Now that $\mathbf{x}^g(t_k)$, $\mathbf{u}^g(t_k)$ have been successfully obtained and the starting values for $\boldsymbol{\gamma}^g$ and $\boldsymbol{\lambda}^g(t_k)$ have been initialized, use them along with the condition

$$\mathbf{p}^T(t_0) = \sum_l \boldsymbol{\gamma}_l \nabla_{\mathbf{x}}[\eta_l(\boldsymbol{x}(t_f))] + \sum_l \beta_l[\nabla_{\mathbf{x}} T(\boldsymbol{x}(t_0)) - (\nabla_{\mathbf{x}} T(\boldsymbol{x}(t_0)) \cdot \mathbf{u}_{0l})\mathbf{u}_{0l}],$$

$$\mathbf{u}_{0l} = \nabla_{\mathbf{x}}[\eta_l(\boldsymbol{x}(t_0))]/|\nabla_{\mathbf{x}}[\eta_l(\boldsymbol{x}(t_0))]|.$$

  to solve the costate equations by integrating forwards from t_0 to t_f to get $\mathbf{p}^g(t_k) = [\mathbf{p}^{(1)}(t_k), \mathbf{p}^{(2)}(t_k), ..., \mathbf{p}^{(NP)}(t_k)]$, $k = 1, ..., N-1$. Use $\mathbf{p}^{(i)}(t_f)$, $i = 1, ..., NP$, to minimize

$$\|\mathbf{p}^T(t_f) - (\beta \Phi_x(\mathbf{x}(t_f)) + \sum_l \boldsymbol{\gamma}_l \nabla_{\mathbf{x}}[\eta_l(\boldsymbol{x}(t_f))])\| < \epsilon,$$

  where ϵ is some small parameter set to achieve a desired level of accuracy. As we proceed, we fix the $\boldsymbol{\lambda}$ multipliers and satisfy this condition by updating the multipliers, $\boldsymbol{\gamma}^{(i)}$, using the *mutation, crossover* and *selection* steps of Differential Evolution.

- **Step 5:** With each generation, we also want to find controls that decrease the value of the Hamiltonian on each time interval . So for each member of the initial population, we record the value of the Hamiltonian, $H(\mathbf{x}^{(i)}(t_k), \mathbf{u}^{(i)}(t_k), \mathbf{p}^{(i)}(t_k), \lambda^{(i)}(t_k))$. Simultaneously, we track the value of the weighted-sum of objective functions, $J(\mathbf{x}^{(i)}(t_k), \mathbf{u}^{(i)}(t_k))$, $i = 1, ..., NP$. To update the controls, $\mathbf{u}^{(g+1)}(t_k)$, and continue decreasing the Hamiltonian, we use the *mutation, crossover,* and *selection* steps of Differential Evolution.

- **Step 6:** Once some desired stopping criteria for DE is met, output $\mathbf{u}^{g*}(t_k)$, the Pareto optimal controls from the final generation and the corresponding multipliers, $\boldsymbol{\gamma}^{g*}$.

- **Step 7:** Of course, we need to ensure the condition

$$H(\mathbf{x}^g(t_k), \mathbf{u}^g(t_k), \mathbf{p}^g(t_k)) \geq H(\mathbf{x}^g(t_k), \mathbf{u}^{g*}(t_k), \mathbf{p}^g(t_k))$$

  is satisfied. To start, select $\tilde{\mathbf{u}}$, $\tilde{\boldsymbol{\lambda}}$ and $\tilde{\boldsymbol{\gamma}}$ to be the control and its corresponding sets of multipliers from the Pareto optimal set in Step 6 that provided the lowest objective function value. Next, repeat Steps 1 - 6 above with fixed $\boldsymbol{\gamma} = \tilde{\boldsymbol{\gamma}}$ and perturbing $\tilde{\mathbf{u}}$ to initialize the population. Use DE to minimize the Hamiltonian but during selection, reject any trial vector that does not meet the condition mentioned in Step 4 with fixed $\tilde{\boldsymbol{\gamma}}$. When some desired stopping criteria for DE is met, output the optimal control.

3.8.2 Numerical Experiment

In the previous sections, we satisfied the state constraints in the social network problem by choosing the controls appropriately. It was not a difficult task since the state and control constraints were of simple form. We are now ready to tackle the same weighted-sum problem in (3.11) but this time, we relax the problem and use the necessary conditions for a relaxed optimal pair along with the above numerical procedure to handle the state constraints.

Since we have two sets of state equations, we need two sets of adjoint multipliers, $\mathbf{p}_1$ and $\mathbf{p}_2$. Again if $i = 1, 2, 3$ and there are $m = 2$ attributes then there are two components in vectors $\mathbf{p}_{1,i}$ and $\mathbf{p}_{2,i}$ which have the form

$$\mathbf{p}_{1,i} = [p_{1,i1}, p_{1,i2}] \quad \text{and} \quad \mathbf{p}_{2,i} = [p_{2,i1}, p_{2,i2}].$$

By combining these two, we get

$$\mathbf{p} = [p_{1,11} \;\; p_{1,12} \;\; p_{1,21} \;\; p_{1,22} \;\; p_{1,31} \;\; p_{1,32} \;\; p_{2,11} \;\; p_{2,12} \;\; p_{2,21} \;\; p_{2,22} \;\; p_{2,31} \;\; p_{2,32}]^T.$$

We can convert the state inequality constraints into the form $\boldsymbol{\eta}_{1,i} \leq 0$ and $\boldsymbol{\eta}_{2,i} \leq 0$

$$\boldsymbol{\eta}_{1,i} = \mathbf{r}_i(t) - (\mathbf{r}_i(0) + \boldsymbol{\delta}_{i_{max}}) \leq 0$$

and

$$\boldsymbol{\eta}_{2,i} = -\mathbf{r}_i(t) + (\mathbf{r}_i(0) - \boldsymbol{\delta}_{i_{min}}) \leq 0$$

.

Since there are two constraints, $\boldsymbol{\eta}_{1,i}$ and $\boldsymbol{\eta}_{2,i}$, associated with each actor, the constraint vectors have the form

$$\boldsymbol{\eta}_{1,i} = [\eta_{1,i1}, \eta_{1,i2}]^T \quad \text{and} \quad \boldsymbol{\eta}_{2,i} = [\eta_{2,i1}, \eta_{2,i2}]^T.$$

We also need two sets of Lagrange multipliers, $\boldsymbol{\lambda}_{1,i}$ and $\boldsymbol{\lambda}_{2,i}$ corresponding to the constraints $\boldsymbol{\eta}_1 \leq 0$, and $\boldsymbol{\eta}_2 \leq 0$ to the Hamiltonian. These multiplier vectors, $\boldsymbol{\lambda}_{1,i}$ and $\boldsymbol{\lambda}_{2,i}$, have the form

$$\boldsymbol{\lambda}_{1,i} = [\lambda_{1,i1}, \lambda_{1,i2}]^T \quad \text{and} \quad \boldsymbol{\lambda}_{2,i} = [\lambda_{2,i1}, \lambda_{2,i2}]^T.$$

Similarly, we also need the multiplier vectors, $\boldsymbol{\gamma}_{1,i}$ and $\boldsymbol{\gamma}_{2,i}$, that have the form

$$\boldsymbol{\gamma}_{1,i} = [\gamma_{1,i1}, \gamma_{1,i2}]^T \quad \text{and} \quad \boldsymbol{\gamma}_{2,i} = [\gamma_{2,i1}, \gamma_{2,i2}]^T.$$

By combining the two vectors for each multiplier set, we get

$$\boldsymbol{\lambda} = [\lambda_{1,11} \;\; \lambda_{1,12} \;\; \lambda_{1,21} \;\; \lambda_{1,22} \;\; \lambda_{1,31} \;\; \lambda_{1,32} \;\; \lambda_{2,11} \;\; \lambda_{2,12} \;\; \lambda_{2,21} \;\; \lambda_{2,22} \;\; \lambda_{2,31} \;\; \lambda_{2,32}]^T$$

and

$$\boldsymbol{\gamma} = [\gamma_{1,11} \;\; \gamma_{1,12} \;\; \gamma_{1,21} \;\; \gamma_{1,22} \;\; \gamma_{1,31} \;\; \gamma_{1,32} \;\; \gamma_{2,11} \;\; \gamma_{2,12} \;\; \gamma_{2,21} \;\; \gamma_{2,22} \;\; \gamma_{2,31} \;\; \gamma_{2,32}]^T.$$

Now we can form the Hamiltonian and present the necessary conditions as presented in Theorem 3.2. In the remainder of this section, we will use $\mathbf{x} = [\mathbf{r}, \mathbf{v}]^T$ to simplify the notation.

$$\begin{aligned} H = -\Big[\mathbf{p} - \Big(&\lambda_{1,11}[\nabla\eta_{1,11}]^T + \lambda_{1,12}[\nabla\eta_{1,12}]^T + \lambda_{1,21}[\nabla\eta_{1,21}]^T + \lambda_{1,22}[\nabla\eta_{1,22}]^T \\ &+ \lambda_{1,31}[\nabla\eta_{1,31}]^T + \lambda_{1,32}[\nabla\eta_{1,32}]^T + \lambda_{2,11}[\nabla\eta_{2,11}]^T + \lambda_{2,12}[\nabla\eta_{2,12}]^T \\ &+ \lambda_{2,21}[\nabla\eta_{2,21}]^T + \lambda_{2,22}[\nabla\eta_{2,22}]^T + \lambda_{2,31}[\nabla\eta_{2,31}]^T + \lambda_{2,32}[\nabla\eta_{2,32}]^T\Big)\Big]\mathbf{g}(\mathbf{x}, \boldsymbol{\mu}_t), \end{aligned}$$

$$\mathbf{g}(\mathbf{x}, \boldsymbol{\mu}_t) = \begin{bmatrix} v_{11} \\ v_{12} \\ v_{21} \\ v_{22} \\ v_{31} \\ v_{32} \\ v_{32} \\ \frac{1}{\tau_1}(v_{11}^0 - v_{11}) - \nabla_{r_{11}}[F_{12} + F_{13}] \\ \frac{1}{\tau_1}(v_{12}^0 - v_{12}) - \nabla_{r_{12}}[F_{12} + F_{13}] \\ \frac{1}{\tau_2}(v_{21}^0 - v_{21}) - \nabla_{r_{21}}[F_{21} + F_{23}] \\ \frac{1}{\tau_2}(v_{22}^0 - v_{22}) - \nabla_{r_{22}}[F_{21} + F_{23}] \\ \frac{1}{\tau_3}(v_{31}^0 - v_{31}) - \nabla_{r_{31}}[F_{31} + F_{32}] \\ \frac{1}{\tau_3}(v_{32}^0 - v_{32}) - \nabla_{r_{32}}[F_{31} + F_{32}] \end{bmatrix}$$

where

$$\begin{aligned} F_{ij} = \quad & \textstyle\sum_{l=1}^{13} \pi_l(t) \left\| \mathbf{u}_i^{(l)} - \mathbf{u}_j^{(l)} \right\|^2 (1 + ((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j \Delta t\|)^2 - \|\mathbf{v}_j \Delta t\|^2) \\ & \cdot \exp\{-l_{ij}((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j \Delta t\|)^2 - \|\mathbf{v}_j \Delta t\|^2)\}\Big] \end{aligned}$$

The costate equations associated with the state equations in (3.11b) - (3.11c) are as follows

$$\begin{aligned} \dot{\mathbf{p}} =& f_{\mathbf{x}}^{0T} - \mathbf{p}\mathbf{g}_{\mathbf{x}} \\ & + \Big[\lambda_{1,11}[\nabla\eta_{1,11}]^T + \lambda_{1,12}[\nabla\eta_{1,12}]^T + \lambda_{1,21}[\nabla\eta_{1,21}]^T + \lambda_{1,22}[\nabla\eta_{1,22}]^T \\ & + \lambda_{1,31}[\nabla\eta_{1,31}]^T + \lambda_{1,32}[\nabla\eta_{1,32}]^T + \lambda_{2,11}[\nabla\eta_{2,11}]^T + \lambda_{2,12}[\nabla\eta_{2,12}]^T \\ & + \lambda_{2,21}[\nabla\eta_{2,21}]^T + \lambda_{2,22}[\nabla\eta_{2,22}]^T + \lambda_{2,31}[\nabla\eta_{2,31}]^T + \lambda_{2,32}[\nabla\eta_{2,32}]^T\Big]\mathbf{g}_{\mathbf{x}} \end{aligned}$$

where $\mathbf{g}_{\mathbf{x}}$ is a $n \times n$ Jacobian matrix.

We also have the costate boundary conditions

$$\begin{aligned}\mathbf{p}(t_0) =&\gamma_{1,11}[\nabla\eta_{1,11}(\mathbf{x}(t_f))]^T + \gamma_{1,12}[\nabla\eta_{1,12}(\mathbf{x}(t_f))]^T \\ &+ \gamma_{1,21}[\nabla\eta_{1,21}(\mathbf{x}(t_f))]^T + \gamma_{1,22}[\nabla\eta_{1,22}(\mathbf{x}(t_f))]^T \\ &+ \gamma_{1,31}[\nabla\eta_{1,31}(\mathbf{x}(t_f))]^T + \gamma_{1,32}[\nabla\eta_{1,32}(\mathbf{x}(t_f))]^T \\ &+ \gamma_{2,11}[\nabla\eta_{2,11}(\mathbf{x}(t_f))]^T + \gamma_{2,12}[\nabla\eta_{2,12}(\mathbf{x}(t_f))]^T \\ &+ \gamma_{2,21}[\nabla\eta_{2,21}(\mathbf{x}(t_f))]^T + \gamma_{2,22}[\nabla\eta_{2,22}(\mathbf{x}(t_f))]^T \\ &+ \gamma_{2,31}[\nabla\eta_{2,31}(\mathbf{x}(t_f))]^T + \gamma_{2,32}[\nabla\eta_{2,32}(\mathbf{x}(t_f))]^T\end{aligned}$$

$$\begin{aligned}\mathbf{p}(t_f) =&\beta\nabla_{\mathbf{x}}\Big[\sum_{j\neq i}\|\mathbf{r}_i(t_f) - \mathbf{r}_j(t_f)\|^2\Big]^T \\ &\gamma_{1,11}[\nabla\eta_{1,11}(\mathbf{x}(t_f))]^T + \gamma_{1,12}[\nabla\eta_{1,12}(\mathbf{x}(t_f))]^T \\ &+ \gamma_{1,21}[\nabla\eta_{1,21}(\mathbf{x}(t_f))]^T + \gamma_{1,22}[\nabla\eta_{1,22}(\mathbf{x}(t_f))]^T \\ &+ \gamma_{1,31}[\nabla\eta_{1,31}(\mathbf{x}(t_f))]^T + \gamma_{1,32}[\nabla\eta_{1,32}(\mathbf{x}(t_f))]^T \\ &+ \gamma_{2,11}[\nabla\eta_{2,11}(\mathbf{x}(t_f))]^T + \gamma_{2,12}[\nabla\eta_{2,12}(\mathbf{x}(t_f))]^T \\ &+ \gamma_{2,21}[\nabla\eta_{2,21}(\mathbf{x}(t_f))]^T + \gamma_{2,22}[\nabla\eta_{2,22}(\mathbf{x}(t_f))]^T \\ &+ \gamma_{2,31}[\nabla\eta_{2,31}(\mathbf{x}(t_f))]^T + \gamma_{2,32}[\nabla\eta_{2,32}(\mathbf{x}(t_f))]^T\end{aligned}$$

with $\beta \geq 0$.

The optimal solution must satisfy the condition

$$H(\mathbf{x}, \boldsymbol{\mu}_t, \mathbf{p}) \geq H(\mathbf{x}, \boldsymbol{\mu}_t^*, \mathbf{p})$$

for any admissible pair $(\mathbf{x}, \boldsymbol{\mu}_t)$.

Algorithm 3.2 was used to solve the problem with the following parameters and stopping criteria:

1. **DE Parameters:** $NP = 3000$, $W = 0.5$, and $CR = 0.5$

2. **Termination Criteria:**

   We decided to run the algorithm until J^* reached a value equal to or better than that achieved by the gradient-based algorithm used in the first solution method.

Using this method, we obtained $J^* = 0.0720$ in the first generation, which is better than 0.2251 produced by the gradient algorithm. Figure 3.10 plots the optimal trajectory for the evolution of actor preferences over time and Figure 3.11 plots the social distance between actor preferences. Once again if we use the average distance between actors as a benchmark for closeness, this algorithm produces the same results as the previous methods: actors 1 and 2 are mutually close; actors 1 and 3 are mutually close; and actors 2 and 3 are not considered close.

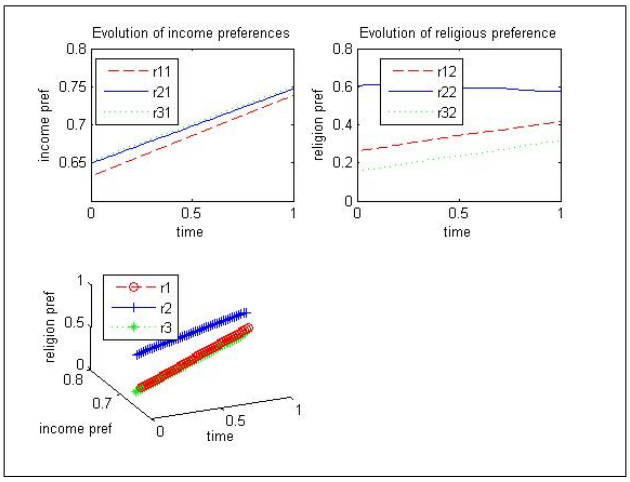


Figure 3.10: Optimal Trajectory - Actor Preferences

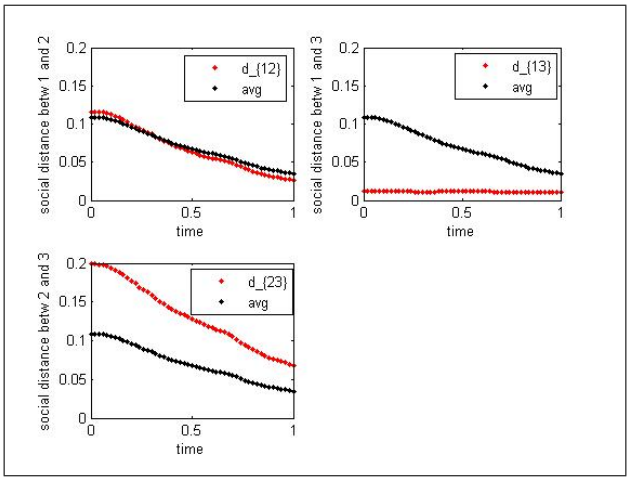


Figure 3.11: Distance between actor preferences using Algorithm 3.2

3.9 Conclusion

The Weighted-Sum Approach is a method that is fairly simple to implement but is computationally expensive producing only one point on the Pareto front per set of weights. Additionally, preassigning the weights can be difficult to determine for more complex problems or when a larger number of objectives are being minimized. We employed three numerical techniques to solve the weighted-sum problem. The first approach using iterative algorithm based on gradient methods was simple to implement but was very slow to converge. It was the most computationally expensive procedure used and gave the worst function value. However, the second method, which employed Differential Evolution to directly minimize the weighted-sum objective function, provided the best answer and was very easy to implement.

As for the third method, Pareto optimality using Differential Evolution to minimize each cost function separately, the method was simple and efficient with reasonable convergence. It provided a better solution to the MOCP than the gradient-based method and generated a whole set of Pareto optimal points within a single run.

Finally, by employing necessary conditions of optimality we could structure a numerical method in which we used Differential Evolution in a novel way to minimize the Hamiltonian and ultimately the objective function value which was lower than that obtained using a gradient method used in an earlier paper [40]. To the best of my knowledge, DE has not been used in this manner before to solve the optimal control problem with state constraints. Furthermore, using the necessary conditions in the manner shown here considerably enhances the effectiveness of the DE method both in speed and accuracy.

Chapter 4

Methodology for Social Networks

4.1 Introduction

Social network analysis focuses on relationships among social entities and the implications from such relationships. In recent decades, social network analysis has received a lot of attention from social and behavioral scientists, who leverage it to answer standard research questions about politics, economics and society at large [61]. Apparently, being able to organize separate entities into networks and groups is very important in determining social and economic outcomes. For instance, personal contacts provide valuable job information; networks are important to "trade and exchange of goods in non-centralized markets"; and finally, organizing societies into groups allows the proper distribution of public good and services. The applications do not stop there; the literature on social networks is vast and covers a variety of applications, some even extending to national defense in recent years.

4.2 Network Concepts

Several key concepts [15], [61] form the basis of social network analysis and are fundamental to our study of social networks. Wasserman and Faust [61] give an excellent broad introduction on social network analysis and their applications. As such, we use their

text to establish the foundation for our study of social networks. Below, we define and discuss in detail those concepts relevant to our work.

4.2.1 Key Terminology for Social Networks

A *social network* is a finite set of actors and relations defined among them. Friendships between residents in a neighborhood is an example of a social network. In this work, we consider networks with at least three actors since they are more interesting than networks with only two actors.

Nodes form the basis of social networks and are often referred to as actors, agents or points depending on the context of discussion. Nodes in a social network can be social entities such as people, businesses, organizations, cities, nations, etc. An *edge* is a line connecting nodes. Edges are also referred to as links, ties, lines or arcs, representing a relationship or connection between a pair of nodes. In network analysis, there are many types of ties to include behavioral interaction ties (i.e., conversing or emailing), physical movement ties (i.e., migration) and individual evaluation ties (i.e., friendship among actors which is the focus of this paper). Network ties are often made based on some type of individual or entity attributes. *Attributes* describe characteristics of actors in a group. For example, for a friendship network, such attribute variables might include income potential, gender, race, sex, education level, political tendency, religious affiliation, marital status, etc. In fact, measurements on actors' attributes often constitute the make-up of social data and social networks [61].

Many studies on social networks focus on relationships between pairs of actors. The term *dyad* refers to just two actors and the possible relations between them. Dyadic analysis is primarily concerned with relationships between pairs of actors and whether or not ties are reciprocated. In our work, ties which are based on attribute preferences alone are reciprocated since we use the Euclidean distance as our measure of relationship between two actors. However, when we factor in categorical data, ties may not be reciprocated since we use a derived categorical distance measure which depends on similarity weighting factors specified by individual actors [61].

Community structure is an important property in the study of social networks. Networks can often be separated into groups based on various characteristics like religion, political affiliation, education, etc. [50]. At the aggregate level, we define a *group* to be the set of all actors on which we measure the links or relationships. We require the group to be a bounded set consisting of a finite number of N actors on whom we measure network links. It follows that a *subgroup* then is any subset of the group of actors and all links between them [61]. This subgroup is sometimes referred to as a cluster or clique. *Clustering*, another important property of social networks, is a method of placing actors into subgroups such that actors belonging to a particular subgroup are more like each other versus actors residing in different clusters [1]. Here, it is assumed that our data set is conducive to distinguishing features of one subgroup from another.

4.2.2 Tools for Representing Social Networks

Social networks can be represented as *graphs* or *matrices*. In this work, we will use both in illustrative examples of friendship networks. In addition, we explain the concept *social distance* which involves using Euclidean distance and weighted categorical distance to decide and measure relationships between actors.

Sociomatrix

A *sociomatrix* is the primary matrix used in social network analysis and is denoted by $\mathbf{X}$. If there are N actors in a social group, then the sociomatrix for the group would be an $N \times N$ matrix of binary entries representing the relations between the actors. Each actors in the sociomatrix has a row and column both indexed $1, 2, ..., N$. The entries in the sociomatrix, x_{ij}, represent which nodes are linked. For our friendship model, relations in the sociomatrix may be directional and nondirectional which will lead to both symmetric and nonsymmetric sociomatrices. For symmetric sociomatrices, if two actors are friends, there will be a 1 in the ij-th and ji-th cells and a 0 if they're not friends. The ii-th cells will contain a value of 0 since actors do not befriend themselves. For nonsymmetric sociomatrices, while the ij-th cells may contain a 1, this may not be the case for the ji-th

cell if the relation is not reciprocated [61].

Graphs

A *graph* (often referred to as *digraph*) has a set of nodes representing the actors in the network and a set of lines to represent the existence of ties or links between pairs of actors. The graph can be drawn directly from the sociomatrix. Since relations in our model may or may not be symmetric, lines are both directional and nondirectional. In essence, if a directional line exists from actor i to j, it may not exist from j to i. We exclude any loops, which are lines between actors and themselves since actors do not befriend themselves.

Distance

The following distance formula [61] will be used to calculate the distance between attribute preferences only:

$$d_{ij} = \sqrt{\sum_{k=1}^{m} (y_i^k - y_j^k)^2}. \tag{4.1}$$

We define another distance measure for categorical data in subsubsection 5.2.3. Primarily, we should use the measure presented here only when determining whether or not a pair of actors are linked based solely on their attribute preferences. However, note that this measure alone may be inadequate when determining relationship ties when considering categorical attributes. While actors may agree on a certain level of preferences for a particular attribute, they may disagree on the category of the attribute. For this reason, to get overall distance between actors, we compare each actor's preferences for certain attributes and categories to decide how close they are socially and whether or not they can be friends. If the distance between two actors' based on their attribute preferences and categories is less than the average distance between all actors' attribute preferences and categories, then we conclude that two actors are friends; otherwise, they are not friends. We use this measure to construct our sociomatrix and graphs for friendships between actors.

4.2.3 Social Network Data

Methods of Collection

Questionnaires, interviews, observations, archival records and experiments are just a few ways in which social network data is collected. In the literature, many social network studies have collected data from classrooms, offices, social clubs, and occasionally, data sets have been artificially created [61]. For this work, we collect archival data from Delaware County, Indiana respondents who participated in a 2004 telephone survey, called the Middleton Area Study, 2004, conducted by the Association of Religious Data Archives (ARDA) [6]. The purpose of the study "was to assess views and lifestyles of citizens on a diverse range of subjects." The study included 600 observations on 70 different variables which covered major topics such as "life satisfaction, education, income, family, religion, and politics". From this survey, we created a specially constructed data set by selecting survey questions relevant to our needs on a small sample of the population. Whereas the statistician may wish to study the correlation of the survey variables or make inferences about the the population of Middleton at large, we do not. Our goal is simply to gather some realistic, initial known facts on a sample of actors to use in our model to predict the evolution of a social network. This newly formed data set in Table 4.2 is taken from responses to the survey questions in Table 4.1 and suits our purpose.

Transforming Ordinal Data to Interval Data

Responses to the survey questions were used in our simulation to draw inferences about the level of importance actors place on certain attributes when forming a social network, in particular a friendship network. The type of data gathered from these surveys can be classified as ordinal scale data and is categorical in nature. To be effective in our research, it was necessary to transform the raw data in Table 4.2 from an ordinal scale to a continuous interval scale to reflect each actor's level of preference for a particular attribute. Therefore, we employed a seemingly standard practice used by social scientists, which involved using equal distance intervals. To get our interval data, we simply took the

range of the desired interval responses and divided it by the number of ordinal categories for each question. This allowed us to map each ordinal data response to a range of interval responses. We should note that the statistician will most likely disagree with this simplified approach because it lacks the rigor needed to perform complex analysis on the variable set and population. However, the reader should remember that our goal is not to perform such rigorous analysis on the population at large; our goal is simply to create a realistic data set of some initial known facts on a set of actors to simulate our friendship model over time.

Deriving Data from Survey Questions

In our model, we use both quantitative and qualitative data and must determine how to treat both types. Quantitative data includes preferences or numerical values that each actor gives to express the level of importance he places on particular attributes when making friends. Qualitative data stem from attribute variables that can be divided into various categories. In the model that we present, every actor will provide provide both quantitative and qualitative data for each attribute. For example, take religion as an attribute. In our model, an actor may rate the importance of religion to making friends as only a 0.2 on a 0 to 1 scale. This is an example of quantitative data. For this same attribute, the actor may be a Catholic, which is just one of many religious categories (Catholic, Protestant, etc.); so this categorical designation represents qualitative data.

The questions in the survey involve several attributes (race, sex, age, income, marital status, religion, politics, etc.) from which to choose our data. When working with income and education levels, we will take the response to these questions to be the actor's actual levels of preference for income and education. The rest of the questions easily provide categorical data but level of preference will have to be inferred. For instance, questions three, four, six and eight provide various categories for race, age, political and religious affiliation. However, questions seven and nine will be used to infer the level of importance (or preference) an actor places on religion and politics. We believe that, in general, an individual's preference for religion can be inferred based on how often he attends church. Likewise, an actor's level of preference for politics can reasonably be inferred from whether

or not on he has intentions to vote in national elections. Unfortunately, there were no questions present in the survey from which we can infer preference levels for race, gender, or marital status. Therefore, it may be necessary to supplement the source data with synthetic data at times to meet the needs of our model.

Handling Quantitative and Qualitative Data

Since we have a mixed data set which includes both numerical and categorical data, we have to determine how to compare the two types of data. In essence, we need a mechanism to measure social distance between individuals who are ultimately described by both numerical and categorical features. Handling numerical data is pretty straightforward. We simply use the Euclidean norm to measure distance between two vectors containing attribute preferences. However, for categorical data, we slightly modify how we calculate the distance between two actors using what we call *weighted categorical distance* which is more formally defined later. Essentially, it amounts to each actor assigning a weight (or *preference value*) to identify how willing he is to befriend dissimilar actors. Once calculated separately, we simply add the numerical and weighted categorical distances together to get the total or social distance between two actors.

Table 4.1: Survey Questions

1. (I-MARITAL) Are you currently married, widowed, separated, divorced, or have you never been married? 1) Married 2) Widowed 3) Separated 4) Divorced 5) Never been married
2. (I-EDUC) What was the last grade you completed in school? 0) 1) High School 2) Some College 3) College Grad 4) Graduate School
3. (I-RACE) To which racial group do you belong? 0) White 1) Black 2) Other
4. (I-AGE) What is your age, please? 0) 18-24 1) 25-36 2) 37-49 3) 50-64 4) 65+
5. (I-HHINCOME) I'm going to read you a series of income categories. Please stop me when I get to the category that includes your total household income the last tax year, from all sources, before taxes? 0) 0-20K 1) 20-40K 2) 50-70K 3) 70K+
6. (I-POLITICAL) Have you usually thought of yourself: 0) Repub 1) Indep 2) Democ 3) Changes 4) OtherParty
7. (I-VOTE) How likely are you to vote in the upcoming presidential election on Tuesday, November 2nd? Would you say you are: 1) Not Likely to Vote 2) Somewhat Likely 3) Very Likely
8. (I-RELIGION) First, what is your religious preference? Would you say you are: 0) Catholic 1) Protestant 3) Other 4) None
9. (I-ATTEND) How often do you attend church? 0) Never 1) Few times per year 2) Few times per month 3) One time per week 4) More than once a week

Table 4.2: Raw Data

i	I-EDUC	I-AGE	I-HHICOME	I-POLITICAL	I-RELIGION	I-ATTEND	I-VOTE
1	2	1	0	3	4	0	3
2	2	3	2	3	1	1	3
3	4	2	1	3	1	4	3
4	4	1	1	3	4	3	3
5	4	1	0	2	3	3	3
6	4	1	1	2	3	1	3
7	4	2	1	0	1	3	3
8	3	4	0	3	0	4	3
9	3	0	0	3	1	1	3
10	3	1	1	2	1	1	3
11	3	0	0	2	1	4	3
12	3	1	2	2	3	3	3
13	3	0	0	1	4	0	3
14	3	0	0	0	1	1	3
15	2	0	0	4	1	2	3
16	2	0	0	3	1	3	3
17	2	0	0	3	0	2	3
18	4	2	2	0	1	0	3
19	2	0	0	2	0	2	3
20	2	1	1	2	1	1	3
21	2	1	2	2	1	2	3
22	2	0	0	0	1	1	3
23	2	0	2	3	1	3	3
24	2	1	1	2	1	3	3
25	2	0	0	0	1	2	3

Chapter 5

A Multiobjective Optimal Control Approach to Social Networks

5.1 Introduction

Different modeling approaches have been used to model social interaction between individuals within a group. When it comes to modeling social networks, a common criticism found in the literature is that researchers often treat networks as though they were static, i.e., not changing over time. In this work, we present a novel method for predicting how social networks develop and evolve over time. In our method, the evolution of the social network is governed by a multiobjective optimal control problem (MOCP) where the dynamics is formulated based on social force theory inspired by Helbing's social forces model for pedestrian walking behavior [23]. We adapt the social forces model to describe the long-term dynamics of social interaction and ultimately, formulate a friendship network mathematically. Using a social forces framework means that actors interact as though they were subject to acceleration, repulsive and attractive forces when making their friendship choices. In addition, formulating the social network model as a multiobjective optimal control problem means that individuals behave according to a set of rules in a manner that promotes their utility minimization, i.e, they choose courses of action with the most benefit and least cost.

Social forces theory assumes that each actor possesses a specific attitude toward making friends, a desire to befriend those who share their preferences for attributes as well as categories and that they respect the private space of others. Consequently, following Helbing and Molnar's theory [24], these rules describing social interaction can be placed into a set of equations of motion to describe individual behavior.

5.2 The Model

5.2.1 Assumptions

We start with a fixed set of actors, denoted Λ, consisting of N actors, who begin as mutual strangers and enter into social relationships with other actors as time evolves. We make the following assumptions [24] in our model of network dynamics:

1. All actors consider the same attributes when attempting to make friends.
2. Actors do not change categories within a particular attribute.
3. Relationships between actors depend on shared preferences for attributes and categories.
4. Reciprocity for numerical preference levels is automatic by virtue of using the Euclidean distance as a measurement of closeness but this is not so for categorical preferences.
5. Each actor attempts to maximize his status in the social group, i.e, he wishes to form as many relationships as possible.
6. Finally, the objective functional of each actor decreases with an increase in shared attribute preferences and categories.

5.2.2 Data

The following data is required to run our model of network dynamics:

Data:

N – total number of actors in a social environment

m – total number of attributes under consideration

l – total number of categorical attributes under consideration

k – number of categories in a particular categorical attribute

$\mathbf{r}_i(t)$ – position vector describing actor i's preference for each attribute, 1,...,m

$\mathbf{y}_i$ – vector identifying various attribute categories to which actor i belongs

$\mathbf{w}_i$ – vector containing actor i's preferences for similar attribute categories

$\mathbf{v}_i^0$ – vector of actor i's initial rate of change of attribute preferences at time $t = 0$

$\mathbf{v}_i(t)$ – vector of actor i's rate of change of attribute preferences at time t

$\mathbf{u}_i(t)$ – vector of controls for actor i's attribute preferences

Parameters:

l_{ij} – constant value set to ensure that actor j respects the private space of actor i

τ_i – relaxation time of actor i (a measure of how fast he returns to his $\mathbf{v}_i^0$)

$\mathcal{N}_i$ – reflects an actor's desire to stick to his belief system

Now that we have formally stated what each data variable represents, we can describe a few variables in more detail. For instance, $\mathbf{v}_i^0$ is meant to reflect how quickly a person intends to change their preference on a certain attribute in order to make friends; it is represented by a "velocity" vector in the social forces model described in the next subsection and hereafter, we will call it intended *social velocity*. Therefore, if a person intends to change their attribute preference levels rapidly, we'd expect to see a larger $\mathbf{v}_i^0$ compared to those who intend to change less rapidly. Similarly, $\mathbf{u}_i(t)$ controls how much actors vary their attribute preferences within a given set of bounds in order to make friends.

The control variables of people who desire to make many friends will fluctuate greatly when compared to those actors who desire fewer relationships, reflected by control variables which are greatly restricted. Similarly, since l_{ij} controls how close actors allow others to get to them, those actors who desire to make many friends will have a larger value for l_{ij} than those who desire to keep others at a distance. Further, a large $\mathcal{N}_i$ is meant to penalize an actor for deviating from his belief system and thus results in an increase in an actor's performance index. Finally, when τ_i is small, individuals will return to their v_i^0 quicker.

5.2.3 Model Components

It is well documented that individuals tend to behave in ways that maximize a utility function of interest. Thus, we can formulate a multiobjective optimal control problem (MOCP) involving a set actors who wish to make as many individual relations as possible while minimizing the associated costs and maintaining their core beliefs. We use the optimal solution of the MOCP to form the social matrix of the social group.

Problem Formation

The social forces model for social networks is represented by a multiobjective optimal control problem subject to state and control constraints. The problem can be stated as follows:

$$\min_{\mathbf{u}} \sum_{j\neq i} \|\mathbf{r}_i(t_f) - \mathbf{r}_j(t_f)\|^2 + \sum_{j\neq i} \left\|\mathbf{w}_i(t_f)\cdot(\mathbf{y}_i(t_f) - \mathbf{y}_j(t_f))\right\|^2 + \mathcal{N}_i \int_{t_0}^{t_f} \|\mathbf{u}_i(t)\|^2\, dt \tag{5.1a}$$

such that

$$\dot{\mathbf{r}}_i = \mathbf{v}_i \tag{5.1b}$$

$$\begin{aligned}\dot{\mathbf{v}}_i = {} & \frac{1}{\tau_i}(\mathbf{v}_i^0 - \mathbf{v}_i) - \nabla_{\mathbf{r}_i} \sum_{j\neq i} \|\mathbf{u}_i - \mathbf{u}_j\|^2 \\ & \cdot(1 + ((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\|)^2 - \|\mathbf{v}_j\Delta t\|^2) \\ & \cdot \exp\{-(l_{ij}((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\|)^2 - \|\mathbf{v}_j\Delta t\|^2))\}\end{aligned} \tag{5.1c}$$

with

$$\mathbf{r}_i(0) = \mathbf{r}_i^0$$

$$\mathbf{v}_i(0) = \mathbf{v}_i^0$$

and

$$(\mathbf{r}_i(0) - \boldsymbol{\delta}_{i_{min}}) \leq \mathbf{r}_i(t) \leq (\mathbf{r}_i(0) + \boldsymbol{\delta}_{i_{max}}) \tag{5.1d}$$

$$-\boldsymbol{\delta}_{i_{min}} \leq \mathbf{u}_i \leq \boldsymbol{\delta}_{i_{max}} \tag{5.1e}$$

Before stating exactly how to form the social matrix using the Pareto optimal solution, we describe the MOCP in detail.

The Performance Index

When forming friendships, the goal of each actor is to form as many ties as possible by minimizing the *social distance* between himself and others but not at the expense of his belief system. This desire is described in the following performance index (also referred to as the objective, cost or payoff function):

$$\mathbf{J}_i = \sum_{j \neq i} \|\mathbf{r}_i(t_f) - \mathbf{r}_j(t_f)\|^2 + \sum_{j \neq i} \left\|\mathbf{w}_i(t_f) \cdot (\mathbf{y}_i(t_f) - \mathbf{y}_j(t_f))\right\|^2 + \mathcal{N}_i \int_{t_0}^{t_f} \|\mathbf{u}_i(t)\|^2 \, dt.$$

We examine each term of the objective function separately. The first term

$$\sum_{j \neq i} \|\mathbf{r}_i(t_f) - \mathbf{r}_j(t_f)\|^2$$

is a component of social distance which represents the vector distance between actors i and j on their levels of preference for the various attributes under consideration.

The second term

$$\sum_{j \neq i} \left\|\mathbf{w}_i(t_f) \cdot (\mathbf{y}_i(t_f) - \mathbf{y}_j(t_f))\right\|^2$$

is the final component of social distance; it represents the weighted categorical distance between actors i and j. The categorical distance, y_{ij}^k, between the two actors on attribute

k is calculated as follows:

$$y_{ij}^k = \left| y_i^k - y_j^k \right| = \begin{cases} 1 & \text{if } y_i^k \neq y_j^k, \\ 0 & \text{if } y_i^k = y_j^k. \end{cases}$$

The vector $\mathbf{w}_i$ holds actor i's weighting factors for each categorical attribute. This weighting factor, $0 \leq w_i^k \leq 1$, describes the actor's attitude toward *similarity* on attributes when making friends. A larger w_i^k reflects that making friends with actors from similar attribute categories is of utmost importance to actor i while not as important for smaller w_i^k. Essentially, the weighting factor describes an actor's tolerance for diversity.

The third term

$$\mathcal{N}_i \int_{t_0}^{t_f} \|\mathbf{u}_i(t)\|^2 \, dt$$

represents the desire of each actor to stay as true to his beliefs as possible over time. $\mathcal{N}_i$ is a weight that actor i uses to express how strongly he desires to stick to his belief system. A large $\mathcal{N}_i$ reflects that actor i is less willing to deviate from his core values while a small $\mathcal{N}_i$ means that he does not care as much to stick to his beliefs.

Social Network Dynamics

Social force theory is used to govern the dynamics of social interaction in our model. Actors behave as though they were subject to acceleration, repulsive and attractive forces; however, these force are not actually exerted on the actors but instead represent a *motivation to act* in a certain way due to the influence of the social group [49].

Furthermore, people are very likely *familiar with* the situations they normally encounter. When reacting to issues that arise in their immediate environment, they usually choose the best decision based on past experience. Therefore, we can say that their reactions are somewhat *automatic* and predictable. This allows us to describe how they react or behave using the below set of equation of motions [24]. Specifically, these equations form a system of nonlinear ordinary differential equations which describe the state dynamics and constraints for our MOCP.

Consider a set of actors $\Lambda = \{1, 2, ..., N\}$ in a social group with k=1, . . . , m attributes under consideration. Each actor, $i \in \Lambda$, is described by a position vector, de-

noted by $\mathbf{r}_i(t) = [r_i^1(t), r_i^2(t), ..., r_i^m(t)]^T$ and an associated *social velocity* vector, denoted by $\mathbf{v}_i(t) = [v_i^1(t), v_i^2(t), ..., v_i^m(t)]^T$. Each quantity, r_i^k, in the position vector describes the actor's preference for a particular attribute while v_i^k describes the motivation an actor has regarding making friends based on a particular attribute preference.

First, an actor changes his position or preferences according to the following differential equation:

$$\begin{aligned} \dot{\mathbf{r}}_i &= \mathbf{v}_i, \\ \mathbf{r}_i(0) &= \mathbf{r}_i^0 \end{aligned}$$

Each actor's level of preference for the various attributes is allowed to fluctuate up or down by some fixed amount so that we have the following constraint on the position vector:

$$(\mathbf{r}_i(0) - \boldsymbol{\delta}_{i_{min}}) \leq \mathbf{r}_i(t) \leq (\mathbf{r}_i(0) + \boldsymbol{\delta}_{i_{max}})$$

Each actor has a vector of controls, $\mathbf{u}_i = [u_i^1, u_i^2, \ldots, u_i^m]^T$ used to vary his attribute preferences within the fixed attribute limits. The constraints on the control vector are represented as:

$$-\boldsymbol{\delta}_{i_{min}} \leq \mathbf{u}_i \leq \boldsymbol{\delta}_{i_{max}}.$$

Next, to describe the actor's change in *social velocity* over time, we use

$$\begin{aligned} \dot{\mathbf{v}}_i &= \frac{1}{\tau_i}(\mathbf{v}_i^0 - \mathbf{v}_i) - \nabla_{\mathbf{r}_i} \sum_{j \neq i} \|\mathbf{u}_i - \mathbf{u}_j\|^2 \\ &\quad \cdot(1 + ((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j \Delta t\|)^2 - \|\mathbf{v}_j \Delta t\|^2)) \\ &\quad \cdot \exp\{-(l_{ij}((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j \Delta t\|)^2 - \|\mathbf{v}_j \Delta t\|^2))\} \\ \mathbf{v}_i(0) &= \mathbf{v}_i^0. \end{aligned}$$

This equation is of utmost importance to our model since it explains how each actor *moves* through time. Therefore, we explain each component of this term in careful detail.

The first term in the acceleration equation reflects an actor's desire to move as efficiently as possible and in a desired direction, $\mathbf{e}_i$, toward his next destination of preferred attributes with a certain speed or enthusiasm, $\mathbf{v}_i^0$. A smaller relaxation time, τ_i, indicates

that an actor returns to his intended *social velocity* quicker. This effect is modeled by a *social velocity* term, $\mathbf{g}_i^0$:

$$\mathbf{g}_i^0(\mathbf{v}_i, \mathbf{v}_i^0) := \frac{1}{\tau_i}(\mathbf{v}_i^0 - \mathbf{v}_i).$$

The second term in the acceleration equation stems from the fact that the behavior of an actor, i, can be influenced by other actors, j, in his social group. While interacting with others, each actor generally respects the *private space* of other actors and tries not to get too close too fast. We can model these *territorial effects*, $\mathbf{g}_{ij}$, using a repulsive potential, $V_j(q)$:

$$\mathbf{g}_{ij}(\mathbf{r}_i - \mathbf{r}_j) = -\nabla_{\mathbf{r}_i} V_j[q(\mathbf{r}_i - \mathbf{r}_j)].$$

The interaction potential which is affected by each actor's behavior is defined by the sum of the repulsive potentials, V_j:

$$V_{int}(\mathbf{r}, t) := \sum_j V_j[q(\mathbf{r}_i - \mathbf{r}_j)]$$

where

$$q = (\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j \Delta t\|)^2 - \|\mathbf{v}_j \Delta t\|^2$$

As previously mentioned, actors require space to make their next move, which is respected by other actors. Therefore, for $V_j(q)$ we use a monotonic decreasing function in β as shown in Figure 5.1.

Let's use Figure 5.1 to explain the effect of the repulsive potential, $V_j(q)$. Start by using the vertical axis as a reference point for actor i and suppose that q reflects the distance between actors i and j. When looking at the figure, notice that as q gets large, i.e., as actor j moves further away from actor i, the repulsive potential $V_j(q)$ gets smaller reflecting that actor j is repelled less. However, as q gets smaller, that is, as actor j mover closer to actor i thereby decreasing the distance between them, the repulsive potential $V_j(q)$ gets very large because more repulsion is needed when actor j gets too close to actor i.

Since the previously mentioned effects or forces all influence an actor's behavior simultaneously, their total effect equals the sum of all these forces. Therefore, the total

motivation to act or the social force, $\mathbf{g}_i$, is:

$$\begin{aligned}\mathbf{g}_i(t) &:= \mathbf{g}_i^0(\mathbf{v}_i, \mathbf{v}_i^0) + \sum_{j \neq i} \mathbf{g}_{ij}(\mathbf{r}_i - \mathbf{r}_j) \\ &= \frac{1}{\tau_i}(\mathbf{v}_i^0 - \mathbf{v}_i) - \nabla_{\mathbf{r}_i} V_{int}(\mathbf{r}_i, t)\end{aligned}$$

where

$$\begin{aligned}V_{int}(\mathbf{r}_i, t) &= \sum_{j \neq i} \|\mathbf{u}_i - \mathbf{u}_j\|^2 \left(1 + ((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j \Delta t\|)^2 - \|\mathbf{v}_j \Delta t\|^2)\right) \\ &\quad \cdot \exp\{-(l_{ij}((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j \Delta t\|)^2 - (\|\mathbf{v}_j \Delta t\|^2))\}.\end{aligned}$$

Hence, we have successfully described how an actor's attributes or preferences evolve through time when attempting to make friends and now we are able to state definitively that the (5.1) make up our social forces model for friendship dynamics [24].

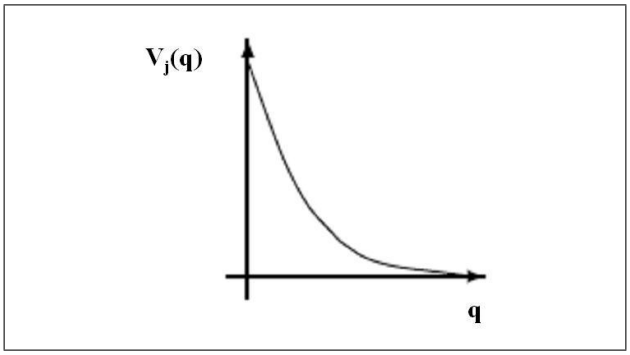


Figure 5.1: Repulsive Potential

5.3 The Social Matrix

We use the optimal solution from the MOCP described in the previous section to identify the group structure of a social network. First, we calculate the total social distance

between two actors i and j by using

$$\begin{aligned} \text{social distance} &= \text{numerical distance} + \text{weighted categorical distance} \\ d_{ij} &= \sum_{j \neq i} \|\mathbf{r}_i(t) - \mathbf{r}_j(t)\|^2 + \sum_{j \neq i} \left\|\mathbf{w}_i(t) \cdot (\mathbf{y}_i(t) - \mathbf{y}_j(t))\right\|^2 \end{aligned} \tag{5.2}$$

Before determining whether or not relationship ties exist using our model, we need to calculate the average social distance between the actors:

$$d_{avg} = \frac{\sum_{i=1}^{N} \sum_{j \neq i} d_{ij}}{N^2 - N},\ j = 1, 2, ..., N. \tag{5.3}$$

Now, we are ready to determine whether or not two actors are connected. We decide the entries of the social matrix $\mathbf{X}$ using the rule

$$x_{ij} = \begin{cases} 1 & \text{if } d_{ij} \leq .8 d_{avg}, \\ 0 & \text{otherwise.} \end{cases}$$

So, if the social distance between two actors is less than a fraction of the average social distance between actors, then we say that two actors are friends.

5.4 Computer Simulation of a Social Network

Using the aforementioned performance index along with the nonlinear time varying dynamical system described by the social forces model from the previous section, we next provide an application to illustrate the features of the model more clearly as well as the usefulness of applying a multiobjective optimal control approach to study social networks.

In this example, we use our model to illustrate solving a multiobjective problem with numerous conflicting objectives. We'll show by computer simulation of interacting actors that our model is capable of producing realistic effects of friendship formation in a social network.

5.4.1 Problem Statement

Consider our model in subsection 5.2.3 with $N = 25$ and $m = 5$ attributes. Specifically, we have the following objective functions to be minimized:

$$\begin{aligned} J_1 = &\|\mathbf{r}_1(t_f) - \mathbf{r}_2(t_f)\|^2 + \|\mathbf{r}_1(t_f) - \mathbf{r}_3(t_f)\|^2 + \cdots + \|\mathbf{r}_1(t_f) - \mathbf{r}_{25}(t_f)\|^2 \\ &+ \|\mathbf{w}_1 \cdot (\mathbf{y}_1(t_f) - \mathbf{y}_2(t_f))\|^2 + \|\mathbf{w}_1 \cdot (\mathbf{y}_1(t_f) - \mathbf{y}_3(t_f))\|^2 \\ &+ \cdots + \|\mathbf{w}_1 \cdot (\mathbf{y}_1(t_f) - \mathbf{y}_{25}(t_f))\|^2 + \mathcal{N}_1 \int_{t_0}^{t_f} \|\mathbf{u}_1(t)\|^2 \, dt \end{aligned}$$

$$\begin{aligned} J_2 = &\|\mathbf{r}_2(t_f) - \mathbf{r}_1(t_f)\|^2 + \|\mathbf{r}_2(t_f) - \mathbf{r}_3(t_f)\|^2 + \cdots + \|\mathbf{r}_2(t_f) - \mathbf{r}_{25}(t_f)\|^2 \\ &+ \|\mathbf{w}_2 \cdot (\mathbf{y}_2(t_f) - \mathbf{y}_1(t_f))\|^2 + \|\mathbf{w}_2 \cdot (\mathbf{y}_2(t_f) - \mathbf{y}_3(t_f))\|^2 \\ &+ \cdots + \|\mathbf{w}_2 \cdot (\mathbf{y}_2(t_f) - \mathbf{y}_{25}(t_f))\|^2 + \mathcal{N}_2 \int_{t_0}^{t_f} \|\mathbf{u}_2(t)\|^2 \, dt \end{aligned}$$

$$\vdots$$

$$\begin{aligned} J_{25} = &\|\mathbf{r}_{25}(t_f) - \mathbf{r}_1(t_f)\|^2 + \|\mathbf{r}_{25}(t_f) - \mathbf{r}_2(t_f)\|^2 + \cdots + \|\mathbf{r}_{25}(t_f) - \mathbf{r}_{24}(t_f)\|^2 \\ &+ \|\mathbf{w}_{25} \cdot (\mathbf{y}_{25}(t_f) - \mathbf{y}_1(t_f))\|^2 + \|\mathbf{w}_{25} \cdot (\mathbf{y}_{25}(t_f) - \mathbf{y}_2(t_f))\|^2 \\ &+ \cdots + \|\mathbf{w}_{25} \cdot (\mathbf{y}_{25}(t_f) - \mathbf{y}_{24}(t_f))\|^2 + \mathcal{N}_{25} \int_{t_0}^{t_f} \|\mathbf{u}_{25}(t)\|^2 \, dt \end{aligned}$$

where $t_0 = 0$ and $t_f = 1$ which we try to minimize subject to the social forces model and constraints defined in subsection 5.2.3.

5.4.2 Data Sets Defined

In order to simulate our model, the specific attributes under consideration are education level, age, income, political tendency, and religious affiliation. All of our attributes will serve as both quantitative and qualitative data in the sense that for each particular attribute, an actor must first state his level of preference for the attribute then describe the attribute category to which he belongs. Again, determining whether two actors are friends will be based on social distance, which is comprised of distance between attribute preferences and weighted categorical distance. At first glance, one might think that the relationships would be symmetric and that reciprocity is automatic since we are using Euclidean distance. However, this is not the case since actor's are allowed to vary their weights (or tolerances)

for categorical attributes. Therefore, the links between actors will be directed instead of symmetric. For some of the data like political tendency and religious affiliation we can make inferences about level of preference using other indicators from our chosen data set (see Table 4.1). However, for the other variables, education level, age, and income, we will simply take the individuals level of preference to coincide with his given attribute category.

Data Initialization and Scaling

It's important to have comparable numbers so we transform attributes like income preferences to $[0, 1]$ so that we can perform the necessary mathematical operations needed to compare them with the other attributes. We use the formula, $n = (s - s_{min})/(s_{max} - s_{min})$, to make such transformations where s is the number to be transformed, s_{min} is the minimum number of the data range and s_{max} is the maximum number of the data range [55]. For instance, suppose we need to transform $\$18,000$ to $[0, 1]$, we simply use the formula to get the new number which is 0.18 normalized to fit on our preferred scale, $[0, 1]$. This newly scaled number can now be compared to numbers like political preferences, which already exist on the $[0, 1]$ scale.

Data Ranges

Once all the numbers for the attribute preferences have been properly scaled, attention is then turned to deciding how these attribute preferences will be allowed to fluctuate over time. Special care must be given to choosing the bounds that are appropriate for each attribute. At $t = 0$, each actor's attribute preferences, $r_i^k,\ k = 1, 2, ..., m$, are initialized within the range $[0, 1]$ and allowed to fluctuate up or down within the range $[\delta_{i_{min}}^k, \delta_{i_{max}}^k]$, while remaining within the overall boundaries, 0 and 1 at all times. If the initial level of preference is greater than 0.5, $\delta_{i_{max}}^k = 1.0 - r_i^k$. However, if the initial level of preference is less than 0.5, $\delta_{i_{min}}^k = r_i^k$. We use this procedure to set the constraints for all attributes. Overall, $[\boldsymbol{\delta}_{i_{min}}, \boldsymbol{\delta}_{i_{max}}]$ accounts for how each actor attempts to control his behavior or acceleration through time. The idea is that if an actor meets someone that he would like to be friends with, he can simply raise or lower his preferences and change his

categorical weights on certain attributes accordingly in order to make friends and accomplish his objective.

Similarity Weights for Categorical Data

As for the categorical data weights, $\mathbf{w}_i$, we use synthetic data to assign these values measuring the importance of similarity. In particular, we use somewhat of a Likert scale to assign weights between certain bounds on the interval $[0, 1]$ for the given attribute under consideration. For example, we might assign weights for an attribute with five categories in the following way: 0 - does not care, 0.25 - cares a little, 0.5 - cares, 0.75 - cares very much, 1.0 - cares the most. Obviously, the lower the weight the more opportunity there is to make friends unless, of course, there are others in the model who share similar restrictive views and enforce those views by selecting weights that indicate their intolerances. Take the attribute, income, for instance; in general, we believe that people with very high incomes may not be as open to befriend people with very low incomes. Similarly, younger people may be hesitant to associate with much older people and vice versa. For political tendency, we expect that conservatives will be less tolerant toward the political views of others than say independents. For these reasons, we restrict the weights to certain limits for the various attributes. Finally, we assume that actors do not switch their categories during the time period $[t_0, t_f]$.

Other Data Variables

Other input data that must be determined includes initial (or desired) *social velocity*, $\mathbf{v}_i^0$, time step, Δt, and the various model parameters previously mentioned. We decide on synthetic data for these inputs initially by borrowing from pedestrian behavior theory and making modifications via experimentation [23]. We took Δt to be a small time interval the same size as the time step in the numerical integration, and through our experimentation, we discovered that a suitable range for $\mathbf{v}_i^0$ is $(0, 0.3]$. The parameter, $\mathbf{v}_i^0$, is assigned based on how motivated an actor is to make friends. In fact, the similarity weights an actor chooses for certain attributes serve as good indicators of his desire to make friends regarding

those attributes. For instance, if an actor chooses a larger weight for a particular attribute, then his $\mathbf{v}_i^0$ value for that attribute needs to be at the lower end of the allowable range to reflect his lack of motivation to make friends. If he has a strong desire for friendship given a particular attribute, then he will need a larger $\mathbf{v}_i^0$ value.

Data Demographics

We construct a data set to illustrate the capabilities of our model using a random sample, that is, we take 25 observations from a larger demographic data set and derive numerical data needed for our model variables. Specifically, the collection method involved poll responses to phone survey questions for the 2004 presidential election [6]. Questions covered demographics on education, age, income, political and religious preferences, etc. The overall demographics of data set are presented in Table 5.1.

Challenges with data included missing data so to handle this, we chose only observations with complete data. Scale also presented a problem so we initialized data within the interval [0, 1] for comparison purposes. To run our computer simulation, assume we have some initial known data as described in Tables 5.2, 5.3, 5.4, and 5.5.

Table 5.1: Data Demographics

Attributes	**Categories**
Education Level	48% - Some College
	28% - Grad School
	24% - Grad School
Age Group	44% - 18 to 24
	36% - 25 to 36
	12% - 37 to 49
	8% - 50+
Income Level	52% - $< \$20K$
	28% - \$20K to \$40K
	20% - \$50K to \$70K
Political Affiliation	36% - Changes
	36% - Democrat
	20% - Republican
	8% - Other
Religious Affiliation	12% - None
	64% - Protestant
	12% - Other
	12% - Catholic

Table 5.2: Model parameters for each actor: $i = 1, ..., 25$

Actor	l_{ij}	τ_i	N_i
1	0.05	1/15.0	1.0
2	0.05	1/15.0	1.0
3	0.15	1/10.0	1.0
4	0.25	1/10.0	1.0
5	0.25	1/5.0	1.0
6	0.25	1/5.0	1.0
7	0.25	1/5.0	1.0
8	0.05	1/15.0	1.0
9	0.05	1/15.0	1.0
10	0.05	1/15.0	1.0
11	0.15	1/15.0	1.0
12	0.25	1/5.0	1.0
13	0.15	1/10.0	1.0
14	0.25	1/5.0	1.0
15	0.25	1/5.0	1.0
16	0.05	1/15.0	1.0
17	0.05	1/15.0	1.0
18	0.25	1/5.0	1.0
19	0.15	1/10.0	1.0
20	0.05	1/15.0	1.0
21	0.25	1/5.0	1.0
22	0.15	1/10.0	1.0
23	0.15	1/10.0	1.0
24	0.15	1/10.0	1.0
25	0.25	1/5.0	1.0

Table 5.3: Level of Preference, $\mathbf{r}_i$, for each attribute by actor: $i = 1, ..., 25$

Actor	Education	Age	Income	Politics	Religion
1	0.3915	0.5731	0.7367	0.7782	0.212
2	0.2664	0.4418	0.2583	0.3758	0.6072
3	0.7879	0.4397	0.6925	0.279	0.7069
4	0.3396	0.2403	0.3989	0.5451	0.2821
5	0.7207	0.3976	0.4928	0.4589	0.3563
6	0.7949	0.7931	0.5	0.7846	0.6355
7	0.5995	0.5	0.6016	0.5498	0.7485
8	0.693	0.2967	0.5079	0.6066	0.7232
9	0.5896	0.5544	0.5	0.6246	0.7602
10	0.6875	0.5166	0.6212	0.5906	0.5457
11	0.1618	0.5007	0.2875	0.3353	0.1646
12	0.3512	0.2325	0.1104	0.105	0.3839
13	0.1884	0.509	0.1929	0.1027	0.1104
14	0.1869	0.1797	0.6236	0.14	0.3878
15	0.5904	0.0353	0.6234	0.3037	0.1729
16	0.732	0.7485	0.5955	0.6788	0.7345
17	0.5101	0.7571	0.7048	0.7406	0.7753
18	0.6164	0.6416	0.5264	0.7775	0.7035
19	0.6923	0.7988	0.6499	0.629	0.7488
20	0.535	0.6796	0.5664	0.6196	0.5903
21	0.2634	0.2945	0.5018	0.745	0.7073
22	0.2981	0.5209	0.5135	0.274	0.2819
23	0.5783	0.7112	0.2296	0.2082	0.7626
24	0.5232	0.7223	0.2125	0.6331	0.2592
25	0.6909	0.2533	0.5626	0.7021	0.5726

Table 5.4: Measures for Similarity $\mathbf{w}_i$ for each attribute by actor: $i = 1, ..., 25$

Actor	Education	Age	Income	Politics	Religion
1	0.25	0.5	0.0	0.0	0.0
2	0.25	0.5	0.5	0.0	0.25
3	0.85	0.5	0.25	0.0	0.25
4	0.85	0.5	0.25	0.0	0.0
5	0.85	0.5	0.0	0.5	1.0
6	0.85	0.5	0.25	0.5	1.0
7	0.85	0.5	0.25	0.85	0.25
8	0.5	0.85	0.0	0.0	0.5
9	0.5	0.85	0.0	0.0	0.25
10	0.5	0.5	0.25	0.5	0.25
11	0.5	0.85	0.0	0.5	0.25
12	0.5	0.5	0.5	0.5	1.0
13	0.5	0.85	0.0	0.25	0.0
14	0.5	0.85	0.0	0.85	0.25
15	0.25	0.85	0.0	1.0	0.25
16	0.25	0.85	0.0	0.0	0.25
17	0.25	0.85	0.0	0.0	0.5
18	0.85	0.5	0.5	0.85	0.25
19	0.25	0.85	0.0	0.5	0.5
20	0.25	0.5	0.25	0.5	0.25
21	0.25	0.5	0.5	0.5	0.25
22	0.25	0.85	0.0	0.85	0.25
23	0.25	0.85	0.5	0.0	0.25
24	0.25	0.5	0.25	0.5	0.25
25	0.25	0.85	0.0	0.85	0.25

Table 5.5: Initial rate of change of preferences, $\mathbf{v}_i^0$, for each actor: $i = 1, ..., 25$

Actor	Education	Age	Income	Politics	Religion
1	0.1216	0.1158	0.1144	0.1116	0.1049
2	0.112	0.1276	0.129	0.129	0.1162
3	0.0394	0.1114	0.1235	0.102	0.1277
4	0.0412	0.1192	0.1123	0.1095	0.1131
5	0.015	0.1052	0.1287	0.1038	0.0054
6	0.0878	0.1236	0.1059	0.1035	0.007
7	0.0022	0.111	0.1159	0.0164	0.1001
8	0.1277	0.0777	0.1151	0.1028	0.1251
9	0.119	0.0342	0.1145	0.116	0.1267
10	0.102	0.1232	0.1113	0.1229	0.1006
11	0.119	0.0148	0.1156	0.1125	0.103
12	0.1226	0.001	0.1245	0.1251	0.0037
13	0.1231	0.0639	0.1063	0.1241	0.129
14	0.1242	0.001	0.1117	0.0009	0.001
15	0.1163	0.0929	0.1014	0.0064	0.1201
16	0.1146	0.0485	0.001	0.1034	0.1137
17	0.001	0.0163	0.1244	0.1065	0.1251
18	0.0626	0.1083	0.1205	0.0909	0.1247
19	0.1023	0.0131	0.1265	0.1043	0.1163
20	0.1163	0.1007	0.1025	0.1062	0.1028
21	0.1063	0.1069	0.1148	0.1255	0.1089
22	0.1081	0.0845	0.1129	0.0713	0.1181
23	0.1088	0.0726	0.1104	0.1024	0.1055
24	0.1247	0.1208	0.1274	0.1246	0.122
25	0.1123	0.0383	0.1048	0.0349	0.1165

5.4.3 Implementation

We use Differential Evolution (DE) to solve the multiobjective optimization problem:

$$\min_{\mathbf{u}} \mathbf{J} = [J_1, J_2, ..., J_{25}]^T \tag{5.4}$$

subject to the constraints described in subsection 5.2.3. Using this approach,we successfully generate a whole set of Pareto-optimal solutions, which are all equally good.

There were several computational challenges which have to be overcome to solve this problem. A system of 250 nonlinear ODEs must be solved and 25 nonlinear cost functions must be minimized. Using parameter recommendations by the developers of DE means that the system has approximately $12,500$ parameters and the suggested NP for DE is at least $25,000$. The problem is computationally expensive to solve given very limited computing resources (time and memory). Therefore, the problem requires a modified algorithm to generate a solution.

5.4.4 Parallel Differential Evolution

There are several variations of Parallel Differential Evolution found in the literature and here we have modified and merged the different ones into one suitable for our problem [54]. We implement our version as follows:

Algorithm 5.1

- **Step 1:** Request K nodes (or processors) taking one node to be the master node.
- **Step 2:** At the master node, create K-1 populations and send one to each of the remaining K-1 nodes.
- **Step 3:** At each of the K-1 nodes, each population evolves toward a nondominated set using DE.

- **Step 4:** As the termination criteria is met, each node sends its nondominated set to the master node.
- **Step 5:** At the master node, compare the K-1 nondominated sets to get the final Pareto optimal set.

5.4.5 Numerical Results and Analysis

To solve our problem, we used Parallel DE as outlined in Algorithm 5.1 with the following criteria:

1. **Requested number of nodes**: 61
2. **DE parameters:** $NP = 50$ per node, $W = 0.5$, and $CR = 0.5$
3. **Termination Criteria:**
   If the difference in the average of the objective values between generations is less than some predetermined amount, we stop the algorithm:
   $$\sum_{i=1}^{25} \left| \frac{J_i^{(k)}(\mathbf{u}^{(1)}) + \cdots + J_i^{(k)}(\mathbf{u}^{(NP)})}{NP} - \frac{J_i^{(k-1)}(\mathbf{u}^{(1)}) + \cdots + J_i^{(k-1)}(\mathbf{u}^{(NP)})}{NP} \right| < 10^{-5}$$

We used the same procedures to integrate the state equations as well as handle the state and control constraints that were employed when using DE in Chapter 3. Parallel DE was implemented in C++ and took approximately 4.5 hours to generate a set of Pareto optimal points.

We use an arbitrary nondominated point of the MOCP and its corresponding trajectory to form the social matrix in Figure 5.2. By analyzing the relations between actors, we identify two disjoint cliques as shown in Figure 5.3: **Clique 1** $= \{5, 6, 12\}$ and **Clique 2** $= \{9, 11, 16, 22, 23, 25\}$. In Figure 5.4, we plot the social distance for Clique 1 members along with the average distance. In Figure 5.5, we see the evolution of the preferences for members in Clique 1 and it is clear that they share similar preferences. Using this preference information along with the model parameters, we can determine the basis for the cliques. For instance, if we look at Clique 1, we conclude that the basis for friendship

between actors actors 5, 6, and 12 stems from the fact that their chosen parameters are very repulsive. They have large weights, $\mathbf{w}_i$, for most categories and their $\mathbf{v}_i^0$ is small for several attribute preferences indicating a lack of motivation to make lots of friends. In fact, they even share similar attribute preferences and demographics. Actors 5 and 6 share four out of five attribute categories; actor 12 shares three out of five attribute categories with actors 5 and 6. All three members of the clique are college educated, democrats, and range in age from $25 - 36$ with religion "other".

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	1	0	1	1	1	1	1	1
3	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
4	1	0	1	0	1	1	1	0	0	1	0	1	0	0	0	0	0	1	0	1	1	0	0	1	0
5	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1	0	1	0	0	1	1	0	1
10	0	0	0	0	1	1	0	0	0	0	1	1	0	0	0	0	0	0	0	1	1	0	0	1	0
11	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	1	0	0	1	0	0	1	1	0	1
12	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	1	1	0	1	0	0	1	1	0	1
14	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	1
16	1	1	1	0	0	0	1	0	1	1	1	0	1	1	1	0	1	1	1	1	1	1	1	1	1
17	0	0	0	0	0	0	0	1	1	0	1	0	1	1	1	1	0	0	1	0	0	1	1	0	1
18	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	1	1	0	1
20	1	0	0	1	1	1	0	0	0	1	1	1	0	0	0	0	0	0	1	0	1	0	0	1	0
21	0	1	0	0	1	1	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	1	1	0
22	0	0	0	0	0	0	1	0	1	0	1	0	0	1	1	1	1	1	1	0	0	0	1	0	1
23	0	1	0	0	0	0	0	0	1	0	1	0	1	1	1	1	1	1	1	0	1	1	0	0	1
24	1	0	0	1	1	1	0	0	0	1	1	1	0	0	0	0	0	0	1	1	1	0	0	0	0
25	0	0	0	0	0	0	1	0	1	0	1	0	0	1	1	1	1	1	1	0	0	1	1	0	0

Figure 5.2: Sociomatrix with Interaction Potential, V_{int} $(N = 25)$

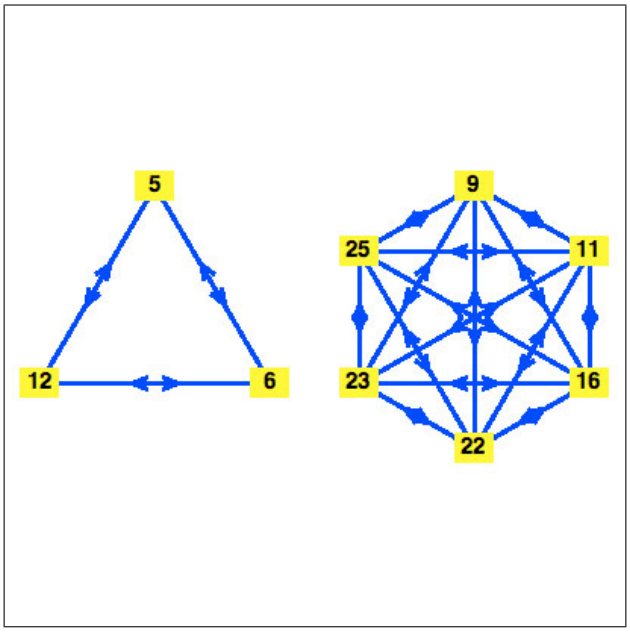


Figure 5.3: Cliques 1 and 2

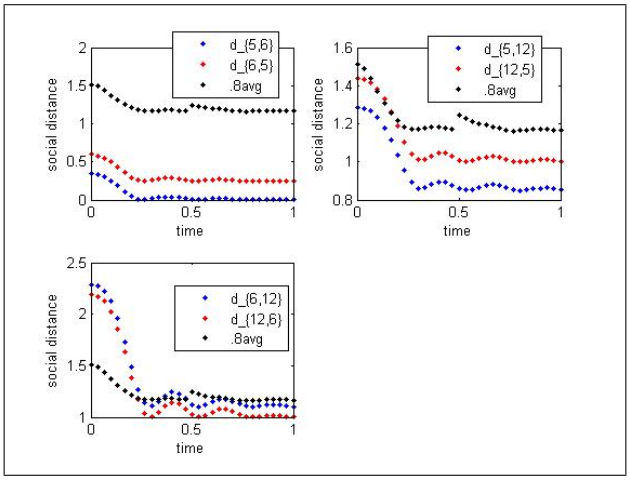


Figure 5.4: Clique 1: Social distance ($d_{ij} \leq .8avg$)

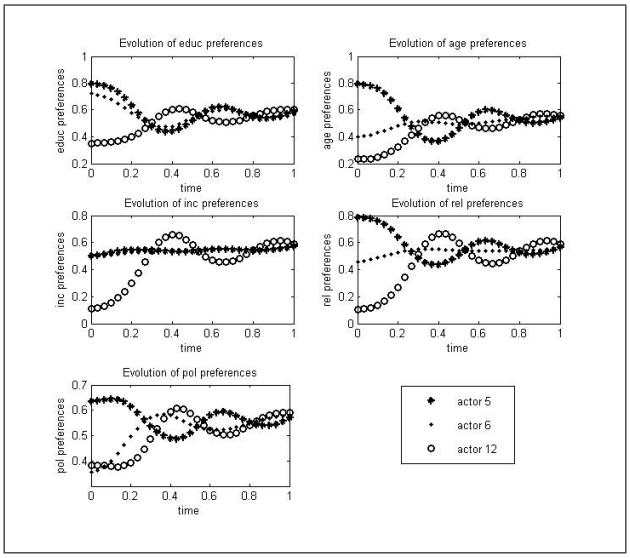


Figure 5.5: Clique 1: Evolution of Preferences

At first glance, determining the basis for friendship among members in Clique 2 is slightly more difficult since it has more members. Fortunately, we're able to use our model to take a *microscopic* look at Clique 2. Tables 5.6 - 5.10 break out the distances between members in the clique by attribute and therefore make it easier to distinguish reasons for their relations. Clearly, from analyzing the tables, the members of Clique 2 strongly relate to each other on attributes 2, 3, and 5 apparent from the smaller distance values in the tables. This is further confirmed by the evolution of actor preferences in Figure 5.6.

Table 5.6: Distance between Education Preferences for actors in Clique 2

Actor	j					
i	9	11	16	22	23	25
9	0	0.0001	0.5002	0.5012	0.5028	0.5189
11	0.0001	0	0.5003	0.5011	0.5028	0.5190
16	0.2502	0.2503	0	0.0014	0.0026	0.0187
22	0.2512	0.2511	0.0014	0	0.0040	0.0201
23	0.2528	0.2528	0.0026	0.0040	0	0.0162
25	0.2689	0.2690	0.0187	0.0201	0.0162	0

Table 5.7: Distance between Age Preferences for actors in Clique 2

Actor	j					
i	9	11	16	22	23	25
9	0	0.0006	0.0014	0.0050	0.0132	0.0188
11	0.0006	0	0.0020	0.0056	0.0138	0.0182
16	0.0014	0.0020	0	0.0036	0.0118	0.0202
22	0.0050	0.0056	0.0036	0	0.0082	0.0238
23	0.0132	0.0138	0.0118	0.0082	0	0.0320
25	0.0188	0.0182	0.0202	0.0238	0.0320	0

Table 5.8: Distance between Income Preferences for actors in Clique 2

Actor	j					
i	9	11	16	22	23	25
9	0	0.0001	0.0081	0.0004	0.0013	0.0089
11	0.0001	0	0.0081	0.0005	0.0012	0.0090
16	0.0081	0.0081	0	0.0085	0.0068	0.0171
22	0.0004	0.0005	0.0085	0	0.0017	0.0085
23	1.0013	1.0012	1.0068	1.0017	0	1.0102
25	0.0089	0.0090	0.0171	0.0085	0.0102	0

Table 5.9: Distance between Political Preferences for actors in Clique 2

Actor	j					
i	9	11	16	22	23	25
9	0	0.0005	0.0003	0.0027	0.0007	0.0214
11	0.5005	0	0.5002	1.0023	0.5012	1.0219
16	0.0003	0.0002	0	0.0024	0.0010	0.02182
22	2.5527	1.7023	2.5524	0	2.5534	0.0242
23	0.0007	0.0012	0.0010	0.0034	0	0.0207
25	2.5714	1.7219	2.5718	0.0242	2.5707	0

Table 5.10: Distance between Religious Preference for actors in Clique 2

Actor	j					
i	9	11	16	22	23	25
9	0	0.0017	0.0004	0.0014	0.0043	0.0044
11	0.0017	0	0.0013	0.0004	0.0060	0.0061
16	0.0004	0.0013	0	0.0010	0.0046	0.0048
22	0.0014	0.0004	0.0010	0	0.0056	0.0057
23	0.0043	0.0060	0.0046	0.0056	0	0.0001
25	0.0044	0.0061	0.0048	0.0057	0.0001	0

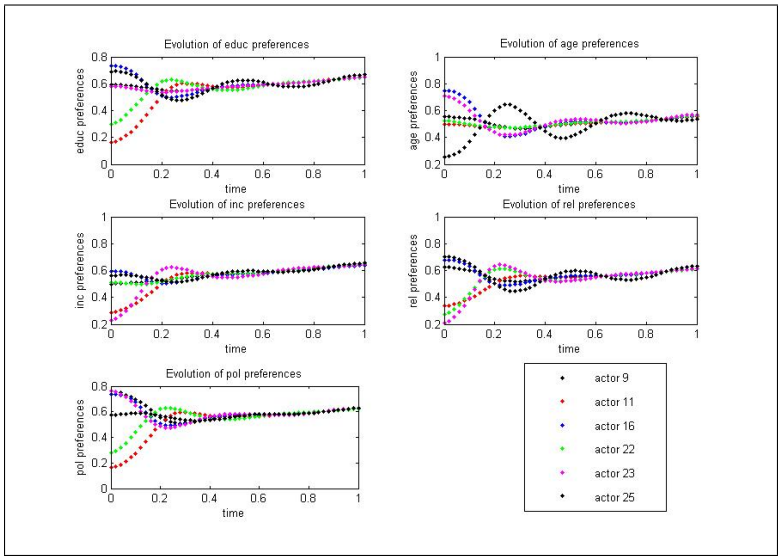


Figure 5.6: Clique 2: Evolution of Preferences

Now that we know the basis for Clique 1's formation, is it possible to break it? Table 5.11 suggests that the answer is 'yes'. By definition a clique requires three mutually friendly actors. From Table 5.11, we see that if the criteria for friendship becomes slightly stricter, actor 6 no longer perceives that he has two friends and thus the clique is broken. This fact is confirmed in Figure 5.7 where the social distance between actors 6 and 12 clearly exceeds the average distance between actors. We will explore the possibility of breaking cliques more in Section 6.3.1.

Table 5.11: Number of friends per actor

	d_{ij} < avg	d_{ij} < .8avg	d_{ij} < .75avg	d_{ij} < .5avg
1	24	24	24	16
2	19	10	10	2
3	8	5	5	2
4	11	11	8	4
5	2	2	2	1
6	2	2	1	1
7	2	2	2	1
8	8	0	0	0
9	20	13	11	8
10	10	7	7	3
11	15	11	10	2
12	2	2	2	0
13	13	13	10	3
14	10	5	4	2
15	9	4	4	0
16	24	19	14	10
17	16	11	10	6
18	2	1	1	1
19	13	7	7	1
20	18	10	10	4
21	12	8	8	3
22	12	11	5	2
23	16	13	12	4
24	18	10	10	4
25	12	11	5	2

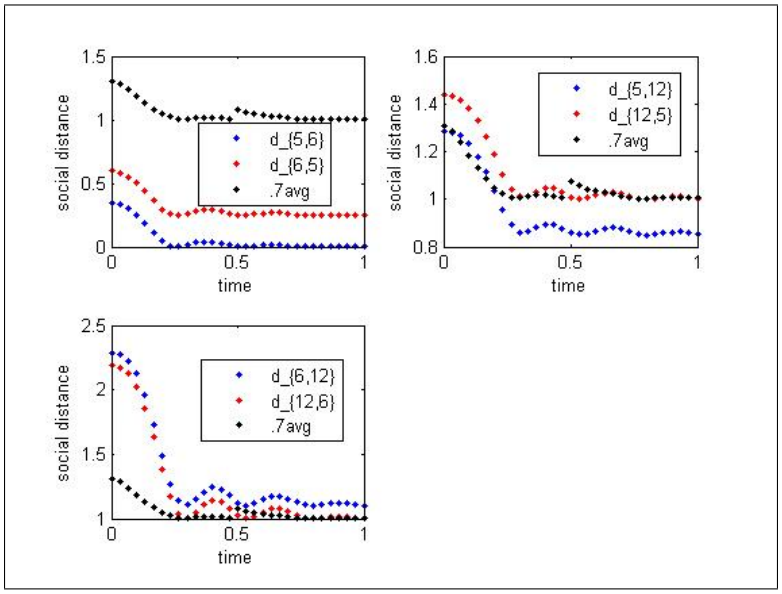


Figure 5.7: Clique 1: Social distance ($d_{ij} \leq .7avg$)

5.5 Reciprocity

In our model, reciprocity is defined as mutual affection between actors. While we have allowed directed relations in our model, it is a well-known fact that reciprocated relationships are often preferred by individuals [34]. In fact, reciprocity absolutely must be present for membership within cliques. In this section, we examine how close the relationships are between the actors within the identified cliques.

While plotting the social distance over time indicates whether or not relations are reciprocated, it does not indicate to what degree reciprocation occurs. Figures 5.8 and 5.9 convey the degree of reciprocity between actors in the two respective cliques. Take, for example, the graph labeled actor 5 in Figure 5.8. It was constructed from the values in Table 5.12. For plotting this graph in the plane, we took the x values to be those in the row labeled 5 and the y values to be those in the column labeled 5. When plotted, those points falling close to the line $y = x$, indicate equal reciprocity between actors. In general, those points that fall close and lower along this diagonal line reflect close reciprocated relations between actors. Points that are higher up and fall close to the diagonal line reflect mutual, less close relations between actors [25].

Tables 5.12 and 5.13 show the distances between actors in the respective cliques. While all the actors within clique 1 are considered to be friends, some actors may be closer than others. Perhaps, the distances in these tables can be viewed as indicators of how close actors really are. For instance, the point $(0.0006, 0.2506)$ is taken from the value, 0.0006 in the row labeled 5 and column labeled 6 and the value 0.2506 in the row labeled 6 and column labeled 5. Clearly, this point indicates that actor 5 feels much closer to actor 6 than actor 6 feels toward actor 5. Actually actor 5 feels closer to actor 6 than he does to actor 12. When we examine Table 5.12 further, we see that actor 6 actually shares this sentiment, since in looking at row 6, the values indicate that actor 6 feels closer to actor 5 than to actor 12. Finally, the values in actor 12's row indicates that actor 12 feels about the same about both actors 5 and 6. In contrast, examining the point value $(1.1019, 1.0019)$ for actors 6 and 12 shows almost equal reciprocation indicating their mutual affection for each other. In Clique 2, actor 9 feels closest to actor 11 and actor 11 returns that sentiment but

not at the same level as evidenced by the values in Table 5.13. It is interesting to note that actors 22 and 25 share the exact level of reciprocation for each other. In this manner, we can use social distance as a numerical measure of friendship.

Table 5.12: Distance between actor preferences in Clique 1

Actor	j		
i	5	6	12
5	0	0.0010	0.8536
6	0.2510	0	1.1060
12	1.0036	1.0060	0

Table 5.13: Distance between actors preferences in Clique 2

Actor	j					
i	9	11	16	22	23	25
9	0	0.0000	0.5002	0.5001	0.5004	0.5012
11	0.5000	0	1.0002	1.0001	1.0004	1.0012
16	0.2502	0.2502	0	0.0001	0.0001	0.0011
22	1.1001	1.1001	0.8501	0	0.8501	0.0012
23	0.7504	0.7504	0.5001	0.5001	0	0.5010
25	1.1012	1.1012	0.8511	0.0012	0.8510	0

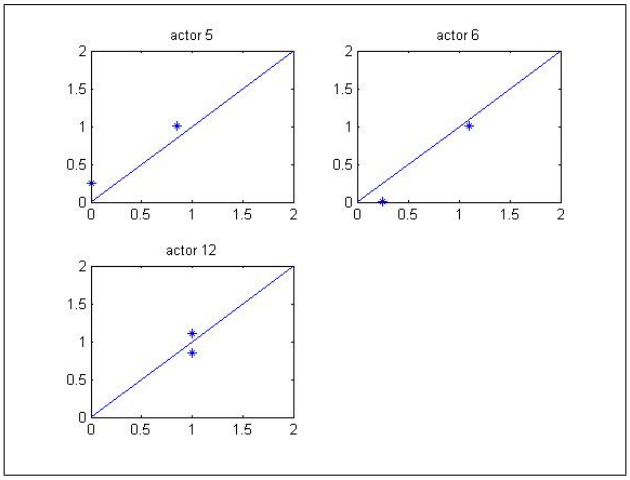


Figure 5.8: Degree of Reciprocity between actors in Clique 1

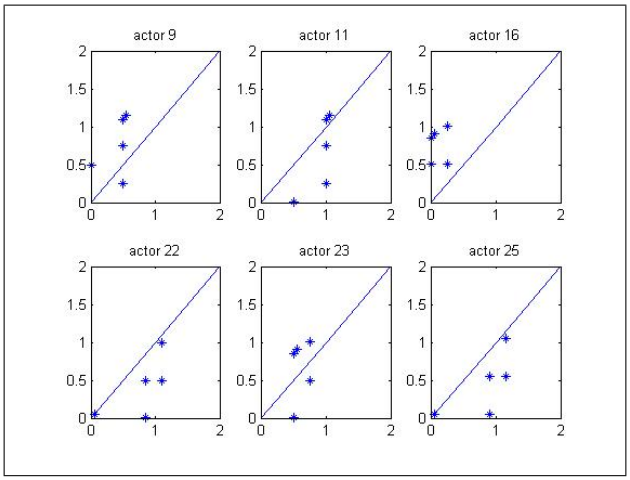


Figure 5.9: Degree of Reciprocity between actors in Clique 2

5.6 Conclusion

From our work, we conclude that multiobjective optimal control theory provides a suitable framework for the design and prediction of network evolution as well as utility maximization among actors in a social group. Social force theory adequately models social interaction between individuals in a social group using a repulsive potential. In addition, the model was able to exhibit clustering (or clique formation) which is a very important structural characteristic of networks. Finally, evolutionary algorithms like Differential Evolution can be successfully employed to solve the MOCP associated with the social network model and gives reasonable results to describe relationship patterns.

Chapter 6

Memory Effect

6.1 Introduction

The model of friendship dynamics described in the previous chapter assumed that every agent has specific intentions including desired velocity and distance from other agents via a repulsive potential. By employing these assumptions, we were able to realistically describe actor interactions but now we shift our focus to what happens when we add **memory effect** to the model. Whereas before the actors only made local decisions about the direction of their next prospect, now actors will rely on the *memory* of whom they have been connected with in the past to influence who they will connect with in the future.

In this chapter, an attractive potential is added to the model that has the same form as the repulsive potential but with opposite sign. With memory effect, actors will consider the entire history when deciding to make friends. We expect memory effect to bring actors who are close together closer making cliques stronger; it also maintains distance between actors who are far apart. Next, we illustrate the impact of the memory potential in the model using a similar multiobjective control approach.

6.2 The Model

Now, by assuming that agents will choose to follow similar paths as those individuals that share similar characteristics, we extend our current social forces model to one with memory effect. We make this change to the model in subsection 5.2.3 by adding the memory mechanism, V_{mem}, to the acceleration force. The social forces model for social networks with memory effect can then be represented by the following multiobjective optimal control problem (MOCP): Find the Pareto optimal set in order to

$$\min_{\mathbf{u}} \mathbf{J} = [J_1, ..., J_N]^T \tag{6.1a}$$

where (6.1b)

$$J_i = \sum_{j \neq i} \|\mathbf{r}_i(t_f) - \mathbf{r}_j(t_f)\|^2 + \sum_{j \neq i} \left\|\mathbf{w}_i(t_f) \cdot (\mathbf{y}_i - \mathbf{y}_j)\right\|^2 + \int_{t_0}^{t_f} \|\mathbf{u}_i(t)\|^2 \, dt \tag{6.1c}$$

s.t. (6.1d)

$$\dot{\mathbf{r}}_i = \mathbf{v}_i \tag{6.1e}$$

$$\dot{\mathbf{v}}_i = \frac{1}{\tau}(\mathbf{v}_i^0 - \mathbf{v}_i) - \nabla_{\mathbf{r}_i} V_{int} - \nabla_{\mathbf{r}_i} V_{mem} \tag{6.1f}$$

where V_{int} and V_{mem} are **repulsive** and **attractive** potentials respectively. Simple bounds on state and controls must also be satisfied as before.

The **repulsive potential** remains in the model and has the form:

$$\begin{aligned} V_{int} \quad = \quad & \sum_{j \neq i} \|\mathbf{u}_i - \mathbf{u}_j\|^2 \\ & \cdot(1 + ((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j \Delta t\|)^2 - \|\mathbf{v}_j \Delta t\|^2)) \\ & \cdot \exp\{-l_{ij}((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j \Delta t\|)^2 - \|\mathbf{v}_j \Delta t\|^2)\} \end{aligned}$$

The **attractive potential** for the model has the form:

$$V_{mem}(\mathbf{r}_i, t) = \int_0^t \sum_{j \neq i} G(\mathbf{r}, s) \exp\left\{\frac{t - s}{\mathcal{T}_m}\right\} ds$$

where

$$G(\mathbf{r},s) = -\gamma_m(1 + ((\|\mathbf{r}_i(t) - \mathbf{r}_j(s)\| + \|\mathbf{r}_i(t) - \mathbf{r}_j(s) - \mathbf{v}_j(s)\Delta t\|)^2 - \|\mathbf{v}_j(s)\Delta t\|^2)$$
$$\cdot \exp\Big\{ - l_{ij}((\|\mathbf{r}_i(t) - \mathbf{r}_j(s)\| + \|\mathbf{r}_i(t) - \mathbf{r}_j(s) - \mathbf{v}_j(s)\Delta t\|)^2 - \|\mathbf{v}_j(s)\Delta t\|^2)\Big\}$$

The memory effect term includes $G(\mathbf{r},t)$, which is a term of the same functional form as the repulsive potential but with a negative sign. In essence, $G(\mathbf{r},t)$ is an attractive potential that reflects the belief that individuals remember to stick together. Notice the addition of two more parameters with this new model: γ_m and $\mathcal{T}_m$. The parameter, γ_m, belongs to $[0,1]$ and reflects how much effect memory has on an actors friendship choice. Obviously, when $\gamma_m = 0$ we simply revert back to the social forces model for social networks *without memory* as described in the previous chapter. Large γ_m indicates that the **memory potential** has a large effect when making friends while smaller γ_m means that the **interaction potential** plays a greater role in the friendship choice. Memory effect decays at a rate of $1/\mathcal{T}_m$ in the model.

6.2.1 Problem Formation and Implementation

Consider this new model with memory as in section 6.1 with $N = 25$ actors and $m = 5$ attributes. We illustrate the effect of adding memory to the model using two different scenarios:

Scenario (1): We allow actor 25 from Clique 2 to change his attribute preferences drastically near the middle of the time interval and we observe how actors in the clique respond to him with and without memory effect in the model.

Scenario (2): We allow actor 12 from Clique 1 to change his attribute preferences drastically near the middle of the time interval and we observe how actors in the clique respond to him with and without memory effect in the model.

The problem can be stated as: Find a Pareto optimal set to

$$\min_{\mathbf{u}} \mathbf{J} = [J_1, ..., J_N]^T$$

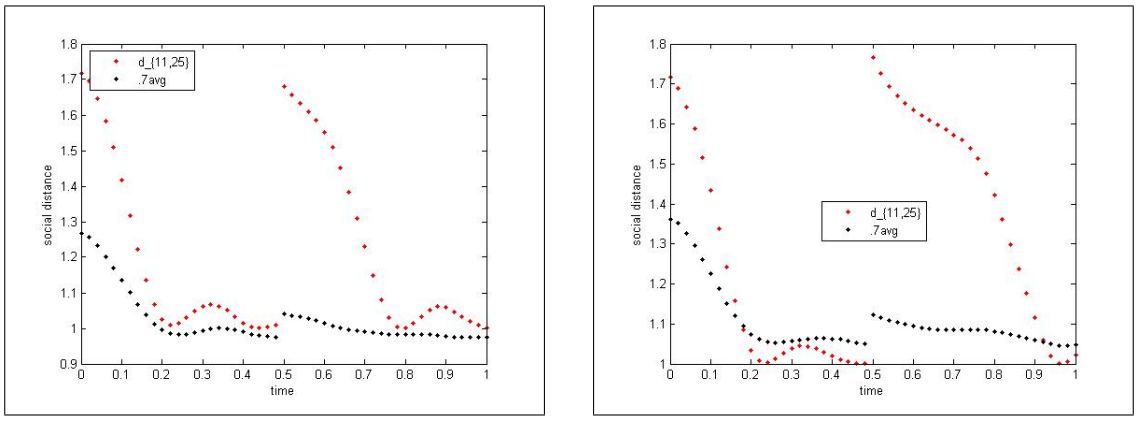
subject to the constraints described above with the added memory potential and encompassing the above two scenarios.

6.2.2 Numerical Results and Analysis

Once again, we used Parallel Differential Evolution to solve the MOCP; the same computational challenges we faced in the previous chapter still exist. Using this approach, we again successfully generate a set of Pareto optimal solutions from which one is selected to construct a sociomatrix that will reflect friendship links that were formed based on the entire history of the actors. With parameters $\gamma_m = 0.3$ and $\mathcal{T}_m = 0.7$, we used the same data and criteria as before:

- **Requested number of nodes**: 61
- **DE parameters:** $NP = 50$ per node, $W = 0.5$, and $CR = 0.5$
- **Termination Criteria:**
  $\sum_{i=1}^{25} \left| \frac{J_i^{(k)}(\mathbf{u}^{(1)}) + \cdots + J_i^{(k)}(\mathbf{u}^{(NP)})}{NP} - \frac{J_i^{(k-1)}(\mathbf{u}^{(1)}) + \cdots + J_i^{(k-1)}(\mathbf{u}^{(NP)})}{NP} \right| < 10^{-5}$

For the first scenario, we see that **without memory effect**, social distance between actors 25 and 9, 11, 16, 22 fluctuates as shown in Figure 6.2. In Figure 6.1, we present an enlarged view of the social distance between actors 11 and 25 in order to see clearly that actor 11 is no longer a friend to actor 25 near the end of the time period. Consequently, actor 25's preference changes actually put him out of the clique since they cause actor 11 to no loner relate to him. Then in Figure 6.3, we see that **with memory effect**, actors 9, 11, 16, and 22 get even closer to actor 25 after the change. Adding memory effect to the model enables actors to remember the long-term history of their friends. Clearly, from the side by side comparison in Figure 6.1, memory effect enables actor 11 to once again relate to actor 25 and thus, **25 is back in the clique**.



(a) $d_{11,25}$ without Memory Effect

(b) $d_{11,25}$ with Memory Effect

Figure 6.1: Distance, d_{ij}, between actors 11 and 25

It is important to note that memory effect can also have the opposite effect of that just discussed. In the second scenario where actor 12 changes his preferences drastically, we see that without memory effect, the social distance between actors 12, 5, and 6 fluctuates as in Figure 6.4. However, with memory effect, actor 5 remembers that he and actor 12 were friends while actor 6 remembers that they were not friends as indicated in Figure 6.5. In these scenarios, clearly memory effect brings those who are friends together while keeping those who are not friends apart.

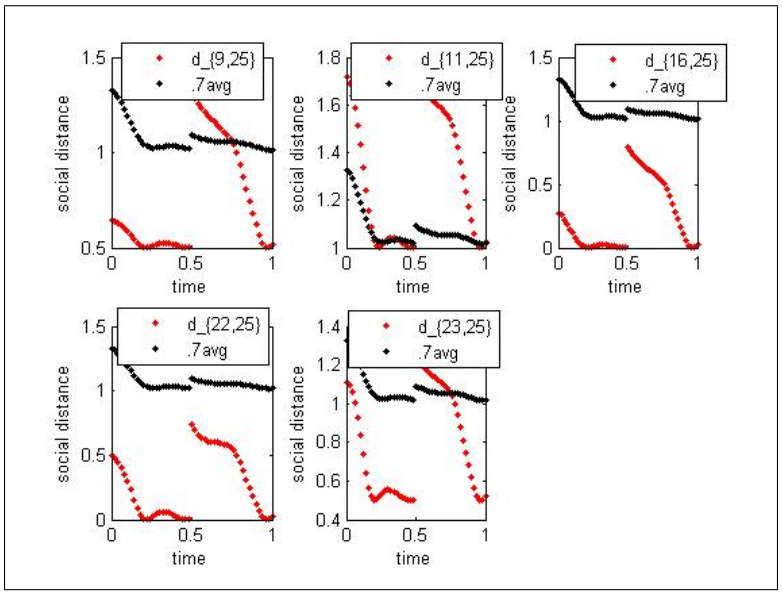


Figure 6.2: Social Distance for Clique 2 without Memory Effect

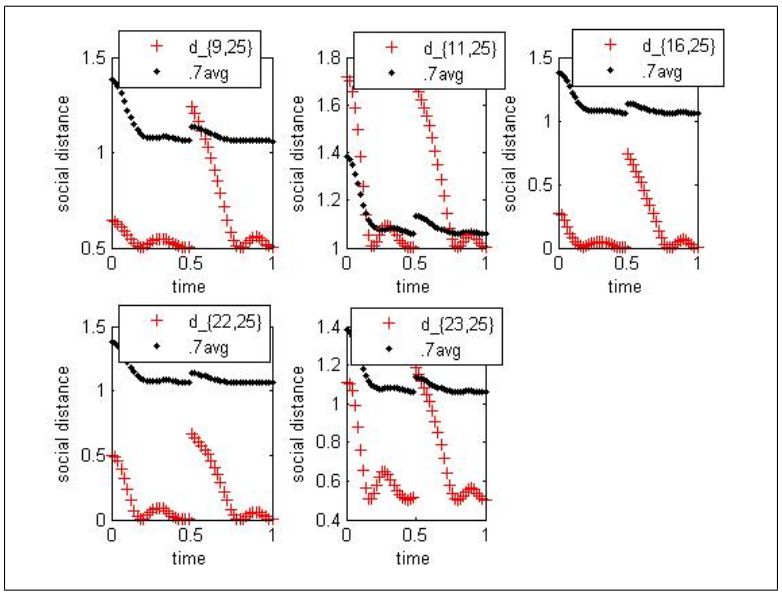


Figure 6.3: Social Distance for Clique 2 with Memory Effect

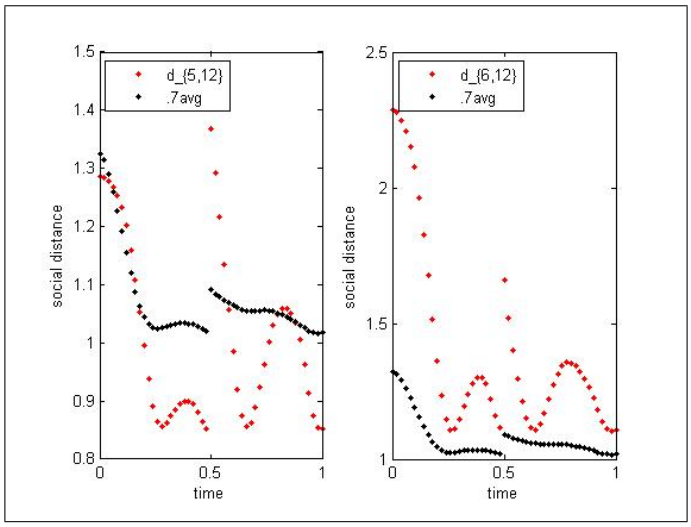


Figure 6.4: Social Distance for Clique 1 without Memory Effect

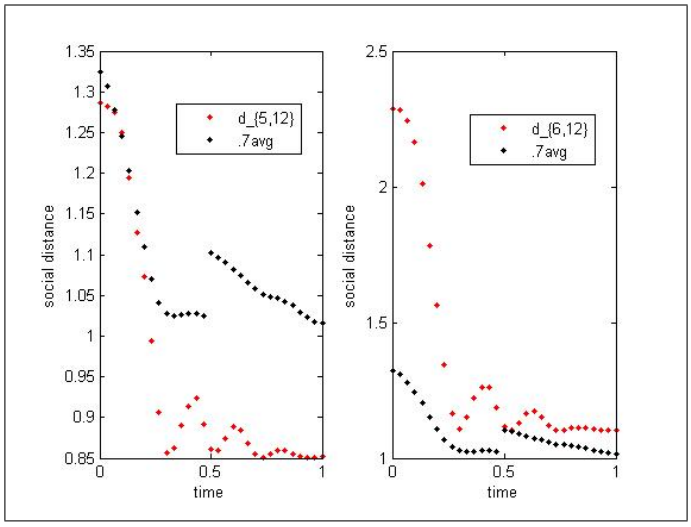


Figure 6.5: Social Distance for Clique 1 with Memory Effect

6.3 Network Destabilization

Since the events September 11, 2001, network destabilization has become a topic deemed critical to national security by U.S. leaders. In particular, the government now employs significant resources, human and financial, toward learning how to disrupt/destroy terrorist and other criminal networks [48], [18]. The literature continues to grow when it comes to destabilizing terrorist networks. In particular, Valdis Krebs (2002) [30] was the first researcher to form a mapping of the terrorist cell that carried out the 9/11 attacks. Farley (2003) [16] offers a discussion on breaking Al Qaeda cells and Carley, Lee, and Krackhardt (2001) [8] explore ways of destabilizing the overall network.

While smaller groups that make up friendship networks are referred to as cliques, the equivalent in a terrorist network is called a cell. In section 5.4.5 we were able to determine the basis for clique formation. If we could successfully use this information to determine how to connect cliques, we might also learn how to break ties between members of different cliques thus destabilizing the overall network. Such techniques would certainly prove useful when dealing with cells in a terrorist network.

6.3.1 "Breaking the ties that bind"

In this section we explore how our model can address questions of how to bring cliques together and how to break them apart. Suppose we start by addressing the question: is there a way to connect the two cliques? The answer is 'yes'. Since actors 9 and 11 from Clique 2 consider actor 12 from Clique 1 a friend, we should start there. Of course, the fact that actor 12 does not reciprocate their friendship keeps the cliques apart. However, if there was some way to get actor 12 to reciprocate, then the two cliques could connect totally.

We reviewed the model data and parameters used to construct the social matrix over time and determined that changing actor 12's preferences and parameters alone is not enough to connect the cliques. Actor 12 will have to change his preference for diversity, $\mathbf{w}_i$. If he relaxes his weights for categorical similarity on the various attributes, then he

will be able to reciprocate the friendship of actors 9 and 11 thus connecting the two cliques (see Figure 6.6). Figure 6.7 shows the social distance between these individuals after their preferences/parameters have been altered; clearly, this change enables mutually reciprocated friendship between the actors.

In recent years, the topic of breaking cliques has become even more intriguing than connecting cliques for various reasons. For instance, such a topic appeals to military leaders for its potential to aid in the war on terror. Knowing how to split terror cells would be crucial to our nation's defense. In the example above, we were able to model connecting cliques by changing model parameters and preferences. In the process, we discover valuable information on how to keep the cliques apart. Since the connection of the two cliques centered around actors 9,11, and 12, these actors are indeed potential **"targets"** for disrupting communication between the two cliques. So using our model, not only have we discovered ways to connect the cliques, but we have identified key nodes to focus on when trying to break ties between cliques within a social network. This techniques could prove useful when trying to destabilize a network.

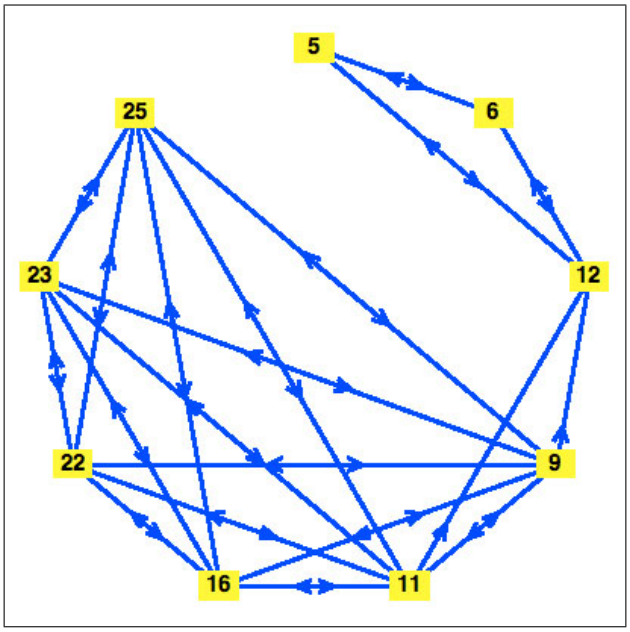


Figure 6.6: Connecting Cliques 1 and 2

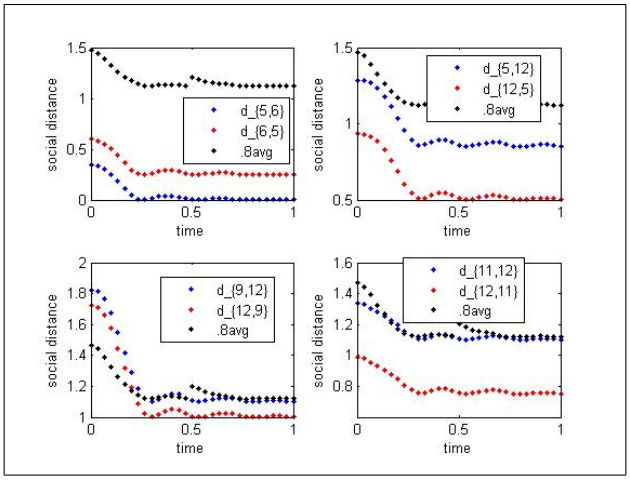


Figure 6.7: Altered Social Distance for Clique 2

6.4 Conclusion

In this chapter, we added memory to the model as an attractive social force. Various experiments were used to illustrate the effect of the memory potential and as expected, actors who were close based on shared preferences were drawn even closer while those who were initially far apart were kept apart by the memory potential. Furthermore, we were able to join the two cliques using information concerning the basis for their ties and in the process, we discovered how to break ties between cliques and possibly destabilize the network. Since our focus was primarily on cooperative networks like friendships, it will be interesting to research whether or not such models can be refined in the future to include noncooperative or criminal networks such as terrorist, drug, and other illegal networks.

Chapter 7

Prediction of Missing Links

7.1 Introduction

In this chapter, we consider how to predict missing links within a social network. Missing links are those that are unobserved but are actually present within the social networks [10]. The links may be unobserved for various reasons: it may be that the network itself is simply incomplete or it may be that the actors themselves attempt to hide their ties. Whatever the reason, predicting these unobserved links has garnered much attention from researchers in today's society. In particular, in the aftermath of the events of September 11, 2001, the capability to uncover missing links in terrorist and other criminal networks is believed to be crucial to national security. The ability to uncover hidden network relations provides a total picture of current and future interactions between entities within a social group.

In the literature, there are various methods used to predict missing networks links [37], [45], [53], [59], [20]. For instance, Liben-Nowell and Kleinberg (2004), [37], focus on network topology such as common neighbors, degree, and shortest paths to predict links for actors within collaborations. Clauset, Moore, and Newman (2008), [10], use hierarchical structure of networks for link prediction. In their approach, Markov chains are used to sample hierarchical random graphs to fit the observed network. Pairs that have high average connection probability but appear unconnected in the observed network are identified as

missing links. Our method for link prediction [41] also starts with an observed network link structure but differs from those in the literature by not relying so heavily on network topology to predict missing links. Instead, our model uses constrained multiobjective optimal control and social force theory to predict missing links.

7.2 The Model

Our model for predicting missing links employs the same components as the social forces model for network dynamics with memory but adds additional constraints on the state. Specifically, our approach starts with the solution of a multiobjective optimal control problem (MOCP) as presented in Chapter 5. It gives us an observed network structure as well as the initial conditions, parameters and constraints that led to its formation. From this observed network, we label some links as *known* and then identify a random subset of network connections and consider them as *unknown* or *missing*. Afterwards, a new MOCP is formed using the same network dynamics as before but now with constraints on the state to preserve the known links. In solving the new MOCP, we are able to reproduce existing links as well as predict or uncover the unknown or missing links.

There are some advantages to this approach over those in the literature that rely heavily on network topology for link prediction. Within the MOCP framework, nodal attributes and past history is considered for link formation. Unlike some of the previously mentioned methods from the literature, this approach works for any given network structure and is capable of uncovering the qualitative, not just topological, reasons underlying link predictions.

Model Components

The model components for predicting missing links are as follows. First, we modify the performance index by adding penalties to remove the inequality constraints [25], $h_q(\mathbf{u})$:

$$\min_{\mathbf{u}} [J_1, ..., J_N] \tag{7.1a}$$

$$J_i = \Big[\sum_{j\neq i} \|\mathbf{r}_i(t_f) - \mathbf{r}_j(t_f)\|^2 + \sum_{j\neq i} \big\|\mathbf{w}_i(t_f)\cdot(\mathbf{y}_i(t_f) - \mathbf{y}_j(t_f))\big\|^2 + \mathcal{N}_i \int_{t_0}^{t_f} \|\mathbf{u}_i(t)\|^2\, dt\Big] \prod_{q=1}^{p} \xi_q^{c_q} \tag{7.1b}$$

where

$$\xi_q = \begin{cases} 1 + h_q(\mathbf{u}), & \text{if } h_q(\mathbf{u}) > 0,\ q = 1, ..., p \\ 1, & \text{otherwise} \end{cases}$$

and the constant $c_q = 10$.

The network dynamics remain unchanged:

$$\dot{\mathbf{r}}_i = \mathbf{v}_i \tag{7.1c}$$

$$\dot{\mathbf{v}}_i = \frac{1}{\tau_i}(\mathbf{v}_i^0 - \mathbf{v}_i) - \nabla_{\mathbf{r}_i} V_{int} - \nabla_{\mathbf{r}_i} V_{mem} \tag{7.1d}$$

where

$$\begin{aligned} V_{int} = & \sum_{j\neq i} \|\mathbf{u}_i - \mathbf{u}_j\|^2 \\ & \cdot(1 + ((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\|)^2 - \|\mathbf{v}_j\Delta t\|^2)) \\ & \cdot \exp\{-l_{ij}((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\|)^2 - \|\mathbf{v}_j\Delta t\|^2)\} \end{aligned}$$

and

$$V_{mem}(\mathbf{r}_i, t) = \int_0^t \sum_{j\neq i} G(\mathbf{r}, s) \exp\Big\{\frac{t-s}{\mathcal{T}_m}\Big\} ds$$

where

$$\begin{aligned} G(\mathbf{r}, s) = & -\gamma_m(1 + ((\|\mathbf{r}_i(t) - \mathbf{r}_j(s)\| + \|\mathbf{r}_i(t) - \mathbf{r}_j(s) - \mathbf{v}_j(s)\Delta t\|)^2 - \|\mathbf{v}_j(s)\Delta t\|^2) \\ & \cdot \exp\Big\{ - l_{ij}((\|\mathbf{r}_i(t) - \mathbf{r}_j(s)\| + \|\mathbf{r}_i(t) - \mathbf{r}_j(s) - \mathbf{v}_j(s)\Delta t\|)^2 - \|\mathbf{v}_j(s)\Delta t\|^2)\Big\} \end{aligned}$$

The same state and control bounds must be satisfied:

$$(\mathbf{r}_i(0) - \boldsymbol{\delta}_{i_{min}}) \leq \mathbf{r}_i(t) \leq (\mathbf{r}_i(0) + \boldsymbol{\delta}_{i_{max}}) \tag{7.1e}$$

$$-\boldsymbol{\delta}_{i_{min}} \leq \mathbf{u}_i \leq \boldsymbol{\delta}_{i_{max}} \tag{7.1f}$$

The following state constraints are added to the model in order to preserve existing relations between actors which are known in advance:

$$d_{ij} \leq .8d_{avg}, \quad \text{if } i,j \text{ are linked} \tag{7.1g}$$

$$d_{ij} > .8d_{avg}, \quad \text{if } i,j \text{ are not linked} \tag{7.1h}$$

Recall that the social distance between actors, d_{ij}, and the average distance, d_{avg}, are calculated as follows:

$$d_{ij} = \sum_{j \neq i} \|\mathbf{r}_i(t) - \mathbf{r}_j(t)\|^2 + \sum_{j \neq i} \left\|\mathbf{w}_i(t) \cdot (\mathbf{y}_i(t) - \mathbf{y}_j(t))\right\|^2,$$

$$d_{avg} = \frac{\sum_i \sum_{j \neq i} d_{ij}}{N^2 - N}.$$

In the missing link model, obviously, the most significant difference from the original model are the added state constraints and the modified performance index to deal with these new constraints. Again, the constraints exist on the state variables to ensure that we reproduce known relations as well as uncover the missing links. In essence, we want there to be links between those people who were already friends and we don't want links between those who were not friends. By using known information on parameters and data for existing links, the new model uncovers missing link information.

7.3 Computer Simulation

7.3.1 Problem Formation

To demonstrate the capabilities of the missing link model, we solve the MOCP in (7.1a) - (7.1h) with $N = 25$ actors and five attributes. We use the sociomatrix in Figure 5.2 as our observed network. From this matrix, we discovered two disjoint cliques: Clique 1: {5,6,12} and Clique 2: {9,11,16,23,23,25}. Relations between members in these cliques will serves as the *known* links. We create an additional set of actors chosen randomly from the

sociomatrix and denote them by M: {2,4,7,10,14}. We pretend the internal links between actors in M as well external links between actors in M and actors in the cliques are *unknown*. The objective now is to try to reproduce known relations amongst members in the cliques while simultaneously uncovering the relations between the cliques and actors in the set M using the information we already have on existing links and parameters.

7.3.2 Implementation

To start, we must assume that we have some observable data on the nodes in set M for whom we do not know the links and wish to identify them. Therefore, we assume we know their initial attribute data to include preferences and categories. From this information, we infer their similarity weights, $\mathbf{w}_i$, and intended motivation toward making friends, $\mathbf{v}_i^0$.

For actors within Cliques 1 and 2, we use their known parameter values but we estimate the parameter values of those actors in set M by permitting them to fall within the allowable limits previously established. For instance, in determining the previous sociomatrix, certain maximum and minimum bounds were identified for parameters, l_{ij} and τ_i:

$$0.05 \leq l_{ij} \leq 0.25,$$
$$\frac{1}{15} \leq \tau_i \leq \frac{1}{5}.$$

There are several state constraints which must be satisfied in order to maintain the existing links between actors in Clique 1. Here is an example of such constraints using the smaller clique:

$$d_{5,6} \leq .8d_{avg}$$
$$d_{5,12} \leq .8d_{avg}$$
$$d_{6,5} \leq .8d_{avg}$$
$$d_{6,12} \leq .8d_{avg}$$
$$d_{12,5} \leq .8d_{avg}$$
$$d_{12,6} \leq .8d_{avg}$$

Similar constraints must be satisfied to maintain the existing links between actors belonging to Clique 2 as well.

In addition, the model must ensure that there are no links between members belonging to the different cliques. For example, here are the constraints that must be met for actor 5 from Clique 1 to maintain his "no link" status to actors in Clique 2:

$$d_{5,9} > .8d_{avg}$$

$$d_{5,11} > .8d_{avg}$$

$$d_{5,16} > .8d_{avg}$$

$$d_{5,22} > .8d_{avg}$$

$$d_{5,23} > .8d_{avg}$$

$$d_{5,25} > .8d_{avg}$$

Constraints similar to these must be established and satisfied for each actor belonging to the two cliques.

Once the existing links as shown in Table 7.1 have been established in the manner suggested, attention can then be focused on filling in the missing links between actors in the cliques and actors in set M. To do so, we again construct the multiobjective optimal control problem (MOCP) in similar fashion as before but subject to the additional state constraints used to ensure the existing relationships. Finally, a Pareto optimal solution of this newly formed MOCP as outlined in (7.1) is used to construct the sociomatrix and identify the existing links as well as the missing links.

7.3.3 Numerical Results and Analysis

To solve the MOCP, we used Parallel DE as outlined in Algorithm 5.1 with slight modifications to the basic DE Algorithm 2.2. This time, for those actors in set M, the parameters, l_{ij} and τ_i, are treated as control variables and allowed to evolve as a population using the random search method. Specifically, in the basic Differential Evolution scheme, the initialization and mutation steps were modified for parameters, l_{ij} and τ_i. For the sake

of clarity in the following modifications, we drop the j from the l_{ij} parameter and just use l_i; this notation does not change the parameter's definition.

In the **initialization** step, we added the following equations:

$$l^g_{k,i} = l^g_{k,i_{min}} + rand() * (l^g_{k,i_{max}} - l^g_{k,i_{min}})$$

and

$$\tau^g_{k,i} = \tau^g_{k,i_{min}} + rand() * (\tau^g_{k,i_{max}} - \tau^g_{k,i_{min}})$$

where $rand()$ is a uniformly distributed random number $\in [0, 1)$ and $l^g_{k,i_{min}}$ and $l^g_{k,i_{max}}$ are lower and upper bounds respectively on the i-th component of the k-th vector, $k = 1, 2, ..., NP$.

In the **mutation** step, in addition to the existing control vectors, for each of the population vectors, $\boldsymbol{l}_k$ and $\boldsymbol{\tau}_k$, $k = 1, ..., NP$, Differential Evolution would generate competing trial vectors, $\hat{\boldsymbol{l}}_k$ and $\hat{\boldsymbol{\tau}}_k$:

$$\hat{\boldsymbol{l}}^g_k = \boldsymbol{l}^g_{j_1} + W * (\boldsymbol{l}^g_{j_2} - \boldsymbol{l}^g_{j_3})$$

and

$$\hat{\boldsymbol{\tau}}^g_k = \boldsymbol{\tau}^g_{j_1} + W * (\boldsymbol{\tau}^g_{j_2} - \boldsymbol{\tau}^g_{j_3,i})$$

where j_1, j_2, and j_3 are random mutually different vectors belonging to $[0,\ NP]$ and not equal to vector k.

To solve the problem, we used Parallel DE as outlined in Algorithm 5.1 with the following criteria:

1. **Requested number of nodes**: 61
2. **DE parameters:** $NP = 30$ per node, $W = 0.5$, and $CR = 0.5$
3. **Termination Criteria:**

$$\sum_{i=1}^{25} \left| \frac{J_i^{(k)}(\mathbf{u}^{(1)}) + \cdots + J_i^{(k)}(\mathbf{u}^{(NP)})}{NP} - \frac{J_i^{(k-1)}(\mathbf{u}^{(1)}) + \cdots + J_i^{(k-1)}(\mathbf{u}^{(NP)})}{NP} \right| < 10^{-5}$$

We used the same procedures as in Chapter 3 to integrate the state equations and handle the bounds on the state and control vectors. In order to avoid problems in distinguishing the added state constraints for link prediction, we had to make modifications in order to solve the problem:

$$d_{ij} + \epsilon \leq .8d_{avg}, \quad \text{if } i, j \text{ are linked,} \tag{7.2}$$

and

$$d_{ij} + \epsilon > .8d_{avg}, \quad \text{if } i, j \text{ are not linked.} \tag{7.3}$$

Essentially, we simply adjust these constraints by some small amount, $\epsilon = .1 * \min(d_{ij})$, in order to solve the problem. We believe this slight adjustment does not impact the accuracy of link prediction. Parallel DE was implemented in C++ and took approximately 8.3 hours to generate a set of Pareto optimal points.

In Table 7.2, the sociomatrix for members in the cliques and set M is shown. The model was able to reproduce the known relations as well as fill in the blanks for the unknown relations. When compared to the observed network in the sociomatrix from Figure 5.2, which shows the actual links between actors, our link prediction model is 100% accurate in predicting missing links. Figure 7.1 shows the associated digraph for the predicted links.

Table 7.1: Sociomatrix with Missing Links

Actor	2	4	5	6	7	9	10	11	12	14	16	22	23	25
2	0													
4		0												
5			0	1		0		0	1		0	0	0	0
6			1	0		0		0	1		0	0	0	0
7					0									
9			0	0		0		1	0		1	1	1	1
10							0							
11			0	0		1		0	0		1	1	1	1
12			1	1		0		0	0		0	0	0	0
14										0				
16			0	0		1		1	0		0	1	1	1
22			0	0		1		1	0		1	0	1	1
23			0	0		1		1	0		1	1	0	1
25			0	0		1		1	0		1	1	1	0

Table 7.2: Sociomatrix with Predicted Links

Actor	2	4	5	6	7	9	10	11	12	14	16	22	23	25
2	0	0	0	0	0	0	0	0	1	0	1	1	1	1
4	0	0	1	1	1	0	1	0	1	0	0	0	0	0
5	0	0	0	1	0	0	0	0	1	0	0	0	0	0
6	0	0	1	0	0	0	0	0	1	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	1	1	0	1	1	1	1	1
10	0	0	1	1	0	0	0	1	1	0	0	0	0	0
11	0	0	0	0	0	1	1	0	0	1	1	1	1	1
12	0	0	1	1	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	1	0	1	0	0	0	1	0	1
16	1	0	0	0	1	1	1	1	0	1	0	1	1	1
22	0	0	0	0	1	1	0	1	0	1	1	0	1	1
23	1	0	0	0	0	1	0	1	0	1	1	1	0	1
25	0	0	0	0	1	1	0	1	0	1	1	1	1	0

7.4 Clique Expansion and Infiltration

Introduction

In this section, we leverage what was learned concerning clique formation in Chapter 5 to explore clique infiltration. Previously, it was discovered clique formation requires

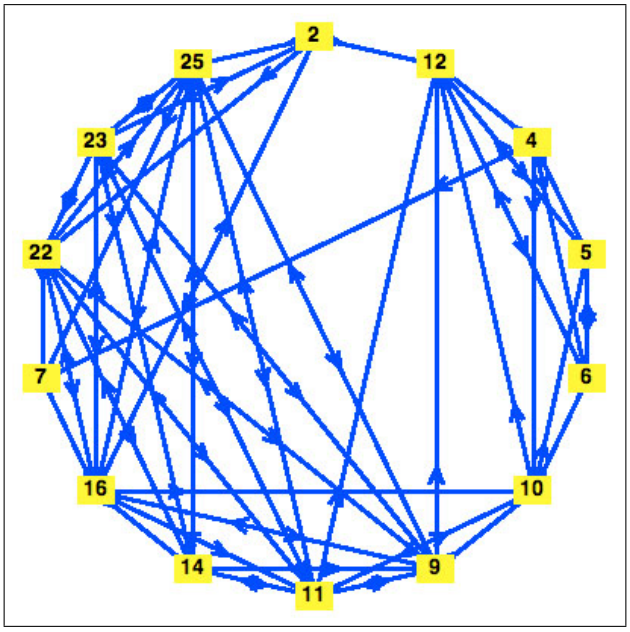


Figure 7.1: Digraph for Predicted Links

mutual affection amongst actors which is based on shared attribute preferences and categories as well as similar choices for the various model parameters. To build on that knowledge, the goal of this section is to try to determine under what circumstances existing cliques would allow other actors to join them. Alternatively, the goal can be restated as how to forcibly insert certain actors into cliques.

When looking across the row labeled 10 in Table 7.2, we see clearly that actor 10 has directional relations or perceived closeness on his part toward members of Clique 1. Yet, this affection from actor 10 is not shared by Clique 1 members as shown by looking down the column labeled 10 in the table. This lack of reciprocity is further confirmed by the social distance plotted in Figure 7.2(a). It is interesting to see what choice of parameters on the part of actor 10 will allow Clique 1 members to show reciprocity thus allowing actor 10 to infiltrate the clique.

7.4.1 Infiltration of Clique 1

To start the process, we implement a slight change to the model by deleting the *memory* potential, V_{mem}, in (7.1d). This should allow individuals more freedom to interact since actors in Clique 1 will then base their friendship decision on current information instead of actor 10's entire history.

Suppose that in addition to the constraints in (7.1), we add the below constraints to the model in an attempt to try to force Clique 1 to accept actor 10. That is, we use

$$d_{10,5} \leq .8d_{avg}$$

$$d_{10,6} \leq .8d_{avg}$$

$$d_{10,12} \leq .8d_{avg}$$

$$d_{5,10} \leq .8d_{avg}$$

$$d_{6,10} \leq .8d_{avg}$$

$$d_{12,10} \leq .8d_{avg}$$

Afterwards, we initialize actor 10's parameters, l_{ij} and τ_i, to values ranging between the previously identified minimum and maximum values and allow them to evolve as a population along with the control parameters using Differential Evolution. In this manner, we hope to discover whether or not certain choices for the various parameters guarantee membership in a particular clique.

Once again, the same algorithmic criteria as before was used to solve the problem. The result was that actor 10 did not immediately become a member of Clique 1 given changes to his parameters values, l_{ij} and τ_i. After several more attempts and even modifying other parameters like actor 10's similarity weights, $\mathbf{w}_i$, and attitude toward making friends, $\mathbf{v}_i^0$, actor 10 was still unsuccessful in penetrating the clique. A thorough review of the raw data indicates that actor 10 actually has many things in common with members of the clique. In fact, actor 10 is in the same age group and shares the same political preference as members of the clique. While they all share similar education and income preferences as well as views on tolerating diversity, actor 10 belongs to a different religious category than clique

members. Significantly, it turns out that Clique 1 is very strongly aligned when it comes to religious preference evident by the fact that all of its members have the highest similarity weight possible, 1.0, for this particular attribute preference.

Tables 7.3 - 7.7 show the social distance between Clique 1 and actor 10 by attribute and allow us to take a microscopic look at their preferential differences. Analyzing these tables confirms what we learned from reviewing the raw data. Indeed, Clique 1 members are strongly aligned in all attribute preferences especially religious preference. We highlight the significant difference in religious preference between Clique 1 members and actor 10 in Table 7.7.

Undoubtedly, we have discovered the primary reason for actor 10's failure, thus far, to successfully penetrate the clique. Since it is so important to members of Clique 1 to only associate with actors of their same religious category, actor 10 may have to take some drastic measures to successfully penetrate the clique. Suppose we relax one of the basic assumptions of the model as discussed in subsection 5.2.1 and allow actor 10 to change his religious category. With this change, members of the clique finally find actor 10 appealing enough to reciprocate his friendship. This mutual affection is captured in Figure 7.2 which graphs the social distance between Clique 1 and actor 10 before and after the change in category. Actor 10 successfully enters Clique 1 with evolved parameter values $l_{ij} = 0.177365$ and $\tau_i = 1/8$ which are close to the range of values used by Clique 1 members.

Table 7.3: Distance between Education Preferences for actor 10 and Clique 1

Actor	j			
i	5	6	10	12
5	0	0.0014	0.8574	0.8638
6	0.0014	0	0.8560	0.8624
10	0.5074	0.5060	0	0.0064
12	0.5138	0.5124	0.0064	0

Table 7.4: Distance between Age Preferences for actor 10 and Clique 1

Actor	j			
i	5	6	10	12
5	0	0.0062	0.0043	0.0090
6	0.0062	0	0.0020	0.0152
10	0.0043	0.0020	0	0.0132
12	0.0090	0.0152	0.0132	0

Table 7.5: Distance between Income Preferences for actor 10 and Clique 1

Actor	j			
i	5	6	10	12
5	0	0.0025	0.0007	0.0216
6	0.2525	0	0.0018	0.2691
10	0.2507	0.0018	0	0.2709
12	1.0216	0.5191	0.5209	0

Table 7.6: Distance between Political Preferences for actor 10 and Clique 1

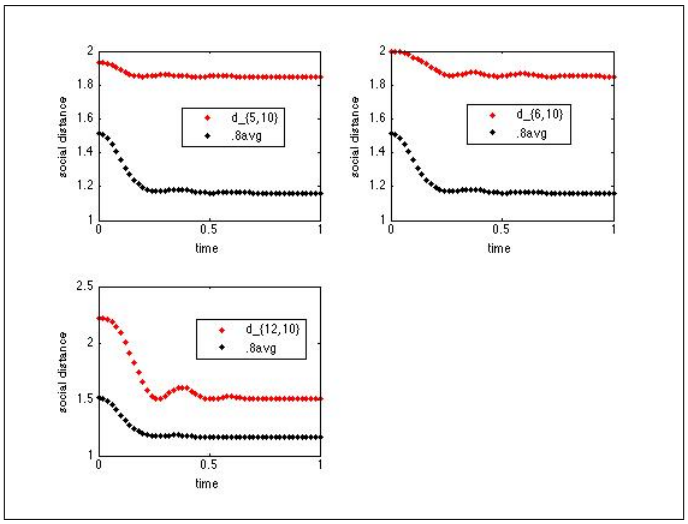
Actor	j			
i	5	6	10	12
5	0	0.0004	0.0013	0.0258
6	0.0004	0	0.0008	0.0262
10	0.0013	0.0008	0	0.0271
12	0.0258	0.0262	0.0271	0

Table 7.7: Distance between Religious Preferences for actor 10 and Clique 1

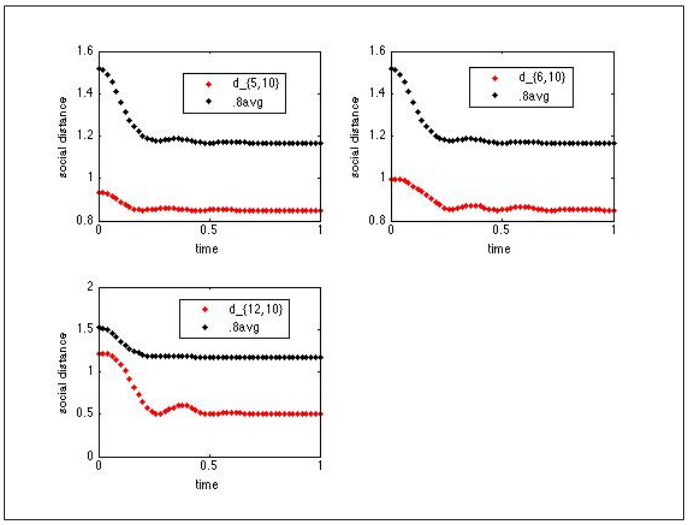
Actor	j			
i	5	6	10	12
5	0	0.0005	**2.0019**	0.0050
6	0.0005	0	**2.0013**	0.0045
10	0.5019	0.5013	0	0.5031
12	0.0050	0.0045	**2.0031**	0

7.4.2 Infiltration of Clique 2

As for Clique 2, we repeat the same infiltration experiment using actor 14 with much success and far greater ease than with Clique 1. The primary reason for this is that Clique 2 is very tolerant of others which is evident by their choice of parameters, in particular, their similarity weights. If we look down the column labeled 14 in Table 7.2, it is clear that every member in Clique 2 has directional ties toward actor 14, which means they already consider him their friend. However, the problem in this experiment is that actor 14 is not friendly toward all members in the clique, in particular, actors 16 and 23, from looking across the row labeled 14 in the same table. While actor 14 shares numerous parameters, preferences, and categories with Clique 2 members, his beliefs on diversity initially prevent him from entering the clique as indicated by Figure 7.3(a). Once again, breaking out the social distance between Clique 2 members and actor 14 by attribute preferences in Tables 7.8 - 7.12 allows us to take a microscopic look at their relations. When analyzing the tables, we focus on the distance between actors 14, 16, and 23. We discover that actor 14 may need to reduce his large similarity weights for age preference and political preference to reflect more tolerance for diversity. These changes allow him to easily infiltrate Clique 2 as supported by the before and after pictures in Figure 7.3. Actor 14 successfully enters Clique 2 with evolved parameter values $l_{ij} = 0.234454$ and $\tau_i = 1/12$ which are within the range of values used by the rest of the members in Clique 2.



(a) Before Infiltration



(b) After Infiltration

Figure 7.2: Social Distance between Clique 1 and Actor 10

Table 7.8: Distance between Education Preferences for actor 14 and Clique 2

Actor	j						
i	9	11	14	16	22	23	25
9	0	0.0016	0.0008	0.5007	0.5026	0.5013	0.5062
11	0.0016	0	0.0008	0.5009	0.5010	0.5002	0.5078
14	0.0008	0.0008	0	0.5001	0.5018	0.5005	0.5070
16	0.2507	0.2509	0.2501	0	0.0019	0.0007	0.0069
22	0.2526	0.2510	0.2518	0.0019	0	0.0012	0.0088
23	0.2513	0.2502	0.2505	0.0007	0.0012	0	0.0075
25	0.2562	0.2578	0.2570	0.0069	0.0088	0.0075	0

Table 7.9: Distance between Age Preferences for actor 14 and Clique 2

Actor	j						
i	9	11	14	16	22	23	25
9	0	0.0028	0.0062	0.0028	0.0055	0.0067	0.0054
11	0.0028	0	0.0034	0.0000	0.0027	0.0039	0.0082
14	0.0062	0.0034	0	0.0034	0.0007	0.0005	0.0116
16	0.0028	0.0000	0.0034	0	0.0027	0.0039	0.0082
22	0.0055	0.0027	0.0007	0.0027	0	0.0012	0.0109
23	0.0067	0.0039	0.0005	0.0039	0.0012	0	0.0120
25	0.0054	0.0082	0.0116	0.0082	0.0109	0.0120	0

Table 7.10: Distance between Income Preferences for actor 14 and Clique 2

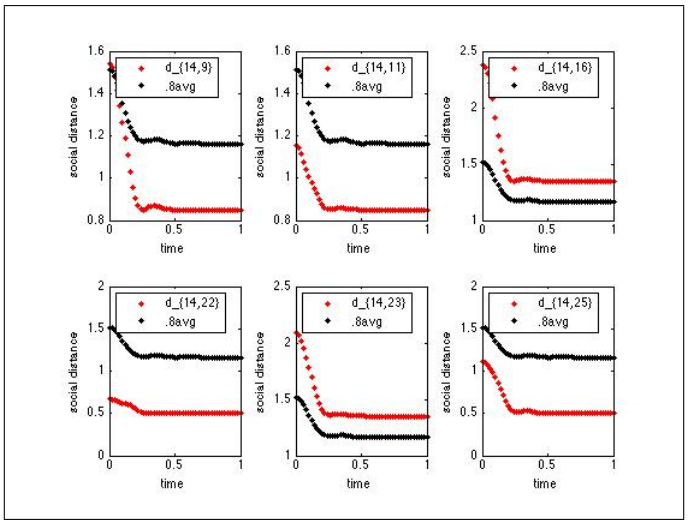
Actor	j						
i	9	11	14	16	22	23	25
9	0	0.0009	0.0013	0.0003	0.0006	0.0002	0.0055
11	0.0009	0	0.0004	0.0006	0.0002	0.0011	0.0047
14	0.0013	0.0004	0	0.0010	0.0007	0.0015	0.0042
16	0.0003	0.0006	0.0010	0	0.0003	0.0005	0.0052
22	0.0006	0.0002	0.0007	0.0003	0	0.0009	0.0049
23	1.0002	1.0011	1.0015	1.0005	1.0009	0	1.0058
25	0.0055	0.0047	0.0042	0.0052	0.0049	0.0058	0

Table 7.11: Distance between Political Preferences for actor 14 and Clique 2

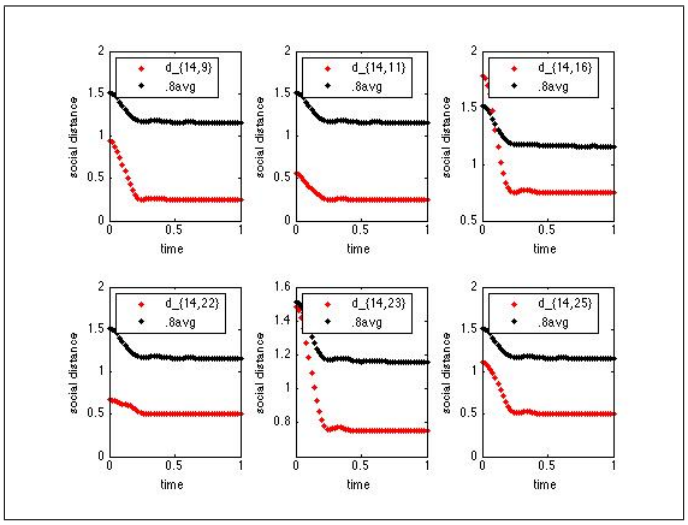
Actor	j						
i	9	11	14	16	22	23	25
9	0	0.0012	0.0010	0.0009	0.0035	0.0028	0.0064
11	0.5012	0	1.0002	0.5003	1.0022	0.5016	1.0076
14	2.5510	1.7002	0	2.5501	0.0024	2.5517	0.0074
16	0.0009	0.0003	0.0001	0	0.0025	0.0018	0.0073
22	2.5535	1.7022	0.0024	2.5525	0	2.5507	0.0098
23	0.0028	0.0016	0.0017	0.0018	0.0007	0	0.0091
25	2.5564	1.7076	0.0074	2.5573	0.0098	2.5591	0

Table 7.12: Distance between Religious Preferences for actor 14 and Clique 2

Actor	j						
i	9	11	14	16	22	23	25
9	0	0.0016	0.0000	0.0008	0.0013	0.0003	0.0025
11	0.0016	0	0.0017	0.0008	0.0003	0.0019	0.0041
14	0.0000	0.0017	0	0.0009	0.0014	0.0002	0.0025
16	0.0008	0.0008	0.0009	0	0.0005	0.0011	0.0033
22	0.0013	0.0003	0.0014	0.0005	0	0.0016	0.0039
22	0.0003	0.0019	0.0002	0.0011	0.0016	0	0.0022
25	0.0025	0.0041	0.0025	0.0033	0.0039	0.0022	0



(a) Before Infiltration



(b) After Infiltration

Figure 7.3: Social Distance between Actor 14 and Clique 2

7.5 Conclusion

In this chapter, we discovered that multiobjective optimal control coupled with social forces theory provides a suitable framework for uncovering missing links within social networks with reasonable accuracy. We were successful in our approach to use known information regarding existing network links to predict hidden links. We also gained insight as it relates to clique infiltration. In fact, the clique expansion experiments indicate that the model is performing as designed which is very reassuring. At times, actors were able to relate to each other on shared attribute preferences alone; yet, at other times, shared preferences for attributes did not seem to be enough to make connections. For instance, the numerous failures of actor 10 to penetrate Clique 1 reflect the impact of categorical differences on friendship choices. The model highlights the significant role that attitudes toward diversity can play in making connections via its similarity measures which account for categorical preferences thus adding an element of realism.

Chapter 8

Future Research

In this work, a novel and practical approach using social forces theory was used to study social networks. In addition, multiobjective optimal control theory was employed to model the long-term evolution of social networks. Clearly, the methodologies proposed may be applied to formulate government policy in a variety of areas.

While our study was primarily conducted using a cooperative network such as a friendship network, it will be interesting to see how the proposed models perform on noncooperative networks involving criminal activity like drug use and terrorism. For instance, one can clearly see how social forces theory could easily be applied to model the social interaction that occurs between drug users. Using multiobjective optimal control theory to predict the resulting structure of the drug users' network as well as identifying the underlying qualitative reasons for individual drug use would certainly be beneficial to drug interdiction. Furthermore, within this work, we provided several illustrations regarding breaking cliques in an attempt to destabilize the overall friendship network. We also conducted "life-like" experiments which exposed the circumstances under which we could successfully expand or infiltrate cliques. By supposing that these same tasks could be accomplished on cells within a terrorist network, one can surely imagine the impact that such capabilities would have on the government's ability to wage the war on terror.

Finally, many computational challenges arose as it relates to numerically solving the various types of optimal control problems that arose from the proposed models in our

study. In particular, adding state constraints to the social network models really increased the difficulty in numerically solving the associated control problems. In addition, extending the number of actors and attributes in the model greatly increased the number of model parameters which further complicated things. Therefore, any future study involving social networks should certainly address the need for numerical algorithms which are computationally effective but less expensive in both time and memory.

References

[1] R. Albert and A-L Barabasi. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74:47–97, Jan 2002.

[2] J. Andersson. A survey of multiobjective optimization in engineering design. report, Linkping University, johan@ikp.liu.se, 2001.

[3] P. Ball. The physical modelling of human social systems. *ComPlexUs Review*, (1):190–206, August 2004.

[4] L. D. Berkovitz and N. G. Medhin. *Optimal Control Theory.* Unpublished manuscript.

[5] N. Biehn. Implicit runge-kutta methods for stiff and state constrained optimal control problems. Phd thesis, North Carolina State University, 2001.

[6] Jerry G. Bode, Ronald Burton, John G. Condran, Stephen D. Johnson, David C. Morris, William Morris, Harry Nelson, and Joseph B. Tamney. Middletown area study, 2004. The Association of Religious Data Archives, www.TheARDA.com, 2004.

[7] S. Butzek and W. H. Schmidt. Relaxation gaps in optimal control processes with state constraints. *International Series of Numerical Mathematics*, 124:21–29, 1998.

[8] K. M. Carley, J. S. Lee, and D. Krackhardt. Destabilizing networks. *Connections*, 24(3):79–92, 2001.

[9] S. Cheng and C. Hwang. Optimal approximation of linear systems by a differential evolution algorithm. *IEEE Transactions on Systems, Man, and Cybernetics–Part A: Systems and Humans*, 31(6):698–707, 2001.

[10] A. Clauset, C. Moore, and M. E. J. Newman. Hierarchical structure and the prediction of missing links in networks. report, University of Michigan, mejn@umich.edu, 2008.

[11] C. A. Coello, D. A. Van Veldhuizen, and G. B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishing, New York, 1st edition, 2002.

[12] Y. Collette and P. Siarry. *Multiobjective Optimization: Principles and Case Studies*. Springer, New York, 1st edition, 2003.

[13] I. L. Lopez Cruz, L.G. Van Willigenburg, and G.Van Straten. Efficient differential evolution algorithms for multimodal optimal control problems. *Applied Soft Computing*, 3:97–122, 2003.

[14] S. Currariniy, M. O. Jackson, and P. Pinx. An economic model of friendship: Homophily, minorities and segregation. Technical report, Universita di Venezia, 2007.

[15] B. Dutta and M. Jackson. *Networks and Groups: Models of Strategic Formation*. Springer-Verlag, New York, NY, 2003.

[16] J. D. Farley. Breaking al qaeda cells: A mathematical analysis of counterterrorism operations. *Studies in Conflict & Terrorism*, 26:399–411, 2003.

[17] M. Gopal. *Modern Control System Theory*, volume 2. John Wiley & Sons, New York, 1993.

[18] J. T. Hamill. Analysis of layered social networks. Phd thesis, Air Force Institute of Technology, Wright-Patterson AFB, OH, 2001.

[19] R. Hartl, S. Sethi, and R. Vickson. A survey of the maximum principles for optimal control problem with state constraints. *SIAM Review*, 37(2):181–218, 1985.

[20] M. A. Hasan, V. Chaoji, S. Salem, and M. Zaki. Link prediction using supervised learning. report, Rensselaer Polytechnic Institute, falhasan, chaojv, salems, zakig@cs.rpi.edu, 2005.

[21] D. Helbing. *Quantitative Sociodynamics: Stochastic Methods and Models of Social Interaction Processes*. Kluwer Academic Publishers, The Netherlands, 1995.

[22] D. Helbing. A mathematical model for the behavior of individuals in a social field. Technical report, University of Stuttgart, 1998.

[23] D. Helbing and P. Molnar. Social forces model for pedestrian dynamics. *Physics Review*, E(51):4282–4286, 1995.

[24] D. Helbing, P. Molnar, and F. Schweitzer. Computer simulation of pedestrian dynamics and trail formation. May 1998.

[25] C. Hong. Nonlinear programming and optimal control approach to the study of social networks. Phd thesis, North Carolina State Univ, 2007.

[26] A. W. Iorio and Xiaodong Li. Incorporating directional information within a differential evolution algorithm for multiobjective optimization. *GECCO*, July 06.

[27] J. Jahn. *Vector Optimization: Theory, Applications, and Extensions*. Springer-Verlag, Berlin, 2004.

[28] C. T. Kelley. *Iterative Methods for Optimization*. No. 18 in Frontiers in Applied Mathematics. SIAM, Philadelphia, 1999.

[29] G. Knowles. *An Introduction to Applied Optimal Control*. Academic Press, New York, 1981.

[30] V. Krebs. Mapping networks of terrorist cells. *Connections*, 24(3):43–52, 2002.

[31] S. Kukkonen and J. Lampinen. A differential evolution algorithm for constrained multi-objective optimization: Initial assessment. Technical report, Lappeenranta University Of Technology.

[32] S. Kukkonen and J. Lampinen. An extension of generalized differential evolution for multi-objective optimization with constraints. Technical report, Lappeenranta University Of Technology.

[33] J. Lampinen and I. Zelinka. Mixed variable non-linear optimization by differential evolution. *Proceedings of Nostradamus '99, 2nd International Prediction Conference*, pages 44–55, October 1999.

[34] R. Th. A.J. Leenders. Evolution of friendship and best friendship choices. *The Journal of Mathematical Sociology*, 21(1-2):133–148, 1996.

[35] G. Leitmann. *The Calculus of Variations and Optimal Control: An Introduction.* Plenum Press, New York, 1981.

[36] F. L. Lewis and V. L. Syrmos. *Optimal Control.* Wiley and Sons, Inc., 1995.

[37] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. report, Massachusetts Institute of Technology, dln@theory.lcs.mit.edu, 2004.

[38] K. C. P. Machielsen. *Numerical solution of optimal control problems with state constraints by sequential quadratic programming in function space.* Center for Mathematics and Computer Science, The Netherlands, 1988.

[39] M. McPherson, L. Smith-Lovin, and J. M. Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27:415–444, 2001.

[40] N. G. Medhin and G. L. Porter. Constrained multiobjective control problems: Application to social networks. *(Submitted: Nonlinear Analysis Series A: Theory, Method & Applications)*, 2008.

[41] N. G. Medhin and G. L. Porter. Control theory methodology for social networks: Prediction of missing links. *(To be Submitted)*, 2008.

[42] N. G. Medhin and G. L. Porter. Numerical solution of a constrained multiobjective control problem modeling the evolution of a social network. *(To be Submitted)*, 2008.

[43] K. Miettinen. *Nonlinear Multiobjective Optimization.* Kluwer Academic Publishing, Massachusettes, 1st edition, 1999.

[44] J. Nocedal and S. J. Wright. *Numerical Optimization Series in Operations Research.* Springer, New York, 1999.

[45] A. Popescul and L. H. Ungar. Statistical relational learning for link prediction. report, University of Pennsylvania, popescul,ungar@cis.upenn.edu, 2003.

[46] K. V. Price, R. M. Storn, and J. A. Lampinen. *Differential Evolution: A Practical Approach to Global Optimization.* Springer, Germany, 2005.

[47] R. Pytlak. *Numerical Methods for optimal control problems with state constraints.* Springer-Verlag, Berlin, 1999.

[48] R. S. Renfro. Modeling and analysis of social networks. Phd thesis, Air Force Institute of Technology, Wright-Patterson AFB, OH, 2001.

[49] F. Schweitzer. *Brownian Agents and Active Particles.* Springer, Santa Fe, NM, 2003.

[50] P. Sen. Complexities of social networks: A physicist's perspective. Technical report, 2006.

[51] F. N. Stockman and E. P.H. Zeggelink. 'self-organizing' friendship networks. Technical report, Universita of Groningen.

[52] R. Storn and K. V. Price. Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. *Computer-Mediated Communication*, 11((2006)):1062–1084, 2006.

[53] B. Taskar, Ming-FaiWong, P. Abbeel, and D. Koller. Link prediction in relational data. report, Stanford University, btaskar, mingfai.wong, abbeel, koller @cs.stanford.edu, 2005.

[54] D. K. Tasoulis, N. G. Pavlidis, V. P. Plagianakos, and M. N. Vrahatis. Parallel differential evolution. Technical report, Greece.

[55] K. Teknomo. http://people.revoledu.com/kardi/tutorial/Similarity/Normalization.html.

[56] D. A. Van Veldhuizen. Multiobjective evolutionary algorithms: Classifications, analyses, and new innovations. phd thesis. Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, May 1999.

[57] T. L. Vincent and G. Leitmann. Control-space properties of cooperative games. *Journal of Optimization Theory and Applications*, 6(2), Jun 1970.

[58] W. Wan. Dynamic game theoretic models in marketing and finance. Phd thesis, North Carolina State Univ, 2008.

[59] C. Wang, V. Satuluri, and S. Parthasarathy. Local probabilistic models for link prediction. Technical report, The Ohio State, 2007.

[60] F. Wang and J. Chiou. Optimal control and optimal time location problems of differential-algebraic systems by differential evolution. *Ind. Eng. Chem. Res.*, 36:5348–5357, 1997.

[61] S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications*. Cambridge University, New York, NY, 1994.

[62] W. Weidlich. *Sociodynamics: A Systematic Approach to Modelling in the Social Sciences*. Harwood Acedemic Publishers, The Netherlands, 2000.

[63] K. Yamaguchi. Homophily and social distance in the choice of multiple friends: An analysis based on conditionally symmetric log-bilinear association model. *Journal of the American Statistical Associations*, 85(410):356–366, Jun 1990.

[64] D. Zeidler, S. Frey, K. L. Kompa, and M. Motzkus. Evolutionary algorithms and their application to optimal control studies. *Physical Review A*, 64(023420):2–13, July 2001.