Neural, Parallel, and Scientific Computations 19 (2011) 373-396

NUMERICAL SOLUTION OF A CONSTRAINED MULTIOBJECTIVE CONTROL PROBLEM MODELING THE EVOLUTION OF A SOCIAL NETWORK

N. G. MEDHIN AND G.L. PORTER

Department of Mathematics, North Carolina State University Raleigh, NC

ABSTRACT. In this paper, we explore how to numerically solve a constrained multiobjective optimal control problem (MOCP) modeling the evolution of a social network using an evolutionary algorithm such as Differential Evolution (DE). For the problems under consideration, the constraints are given by a system of ordinary differential equations along with some simple bounds on the state and control vectors. Using weights we form a single objective function as a linear combination of the multiple objective functions. The single objective function thus formed is then minimized. Then, we use the necessary conditions for the constrained optimal control problem adapting DE in a novel and effective way to find an optimal solution. A numerical algorithm is developed to solve the constrained MOCP and is illustrated using a social network model.

Key words: Multiobjective optimal control, weighted-sum method, relaxed control, Differential Evolution, social networks

1. INTRODUCTION

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) [4] used DE to solve multimodal optimal control problems with great success and Feng-Sheng WangJi-Pyng Chiou (1997) [21] has applied Differential Evolution to solve constrained problems in robotics. In this paper, 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 [15]. Further, the procedure employs multiple applications of DE to handle the various constraints, and the process itself eliminates infeasible solutions [14] gradually improving the accuracy and speed of DE as applied by previous researchers [4], [21]. Effectively, the state constrained problem has been

Received February 27, 2011

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.

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 [5], [10]:

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

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

Consider the system described by

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

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 $\mathbb{R}^m$.

Definition 2.1. A point $\mathbf{u}^*$ is a global minimizer if $J(\mathbf{u}^*) \leq J(\mathbf{u})$ for all $\mathbf{u} \in U$ [16].

Definition 2.2. A point $\mathbf{u}^*$ is a local minimizer if there is a neighborhood B of $\mathbf{u}^*$ such that $J(\mathbf{u}^*) \leq J(\mathbf{u})$ for all $\mathbf{u} \in B$.

Now, we can formulate the optimal control problem:

$$\min_{\mathbf{u} \in U} \mathbf{J}(\mathbf{u}) = \Phi(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t))dt \tag{1a}$$

s.t.

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)), \qquad \mathbf{x}(0) = \mathbf{x}^0 \tag{1b}$$

$$T(\mathbf{x}(t_0)) \tag{1c}$$

$$\mathbf{u}(t) \in U \tag{1d}$$

$$\boldsymbol{\eta}(\mathbf{x}(t)) \leq 0 \tag{1e}$$

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 [1], [7] for a discussion on the existence of optimal controls.

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:

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

s.t.

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)), \quad \mathbf{x}(0) = \mathbf{x}^0 \tag{2b}$$

$$T(\mathbf{x}(t_0)) \tag{2c}$$

$$\mathbf{u}(t) \in U \tag{2d}$$

$$\boldsymbol{\eta}(\mathbf{x}(t)) \leq 0 \tag{2e}$$

$$J_i(\mathbf{x}^0, \mathbf{u}(t), \mathbf{x}(t)) = \Phi_i(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} L_i[\mathbf{x}(t), \mathbf{u}(t)]dt \tag{2f}$$

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 $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{u} \in \mathbb{R}^m$ 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 $\mathbf{x}$.

4. PARETO OPTIMALITY

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

Definition 4.1. For a given vector of objective or cost functions $\mathbf{J}(u) = [J_1(u), J_2(u), \ldots, J_s(u)]$, the control u^* is **Pareto optimal** if there does not exist $\mathbf{u}$ such that

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

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

$$J_i(\mathbf{u}) < J_i(\mathbf{u}^*)$$

We have seen several definitions of Pareto optimality in literature and here is another one that is often used:

Definition 4.2. An admissible control u^* is Pareto optimal if, and only if, for every admissible control $\mathbf{u}$, $\Delta J_i = J_i(\mathbf{u}) - J_i(\mathbf{u}^*)$ for all $i \in \{1, 2, \ldots, N\}$ or there exist at least one $i \in \{1, 2, \ldots, N\}$ so that $\Delta J_i > 0$.

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 [11].

Theorem 4.1. *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*

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

and subject to isoperimetric constraints

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

$j = 1, 2, \ldots, s$ 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 $i \in \{1, 2, \ldots, s\}$ 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

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

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

4.2. **Sufficient Conditions for Pareto Optimality.** In this section we introduce two lemmas and a theorem from Lietmann [11], [20] which embody the sufficient conditions for Pareto optimality. He suggests that if a control meets the sufficient conditions to be an optimum for the related optimal control problem, then it also satisfies the conditions of Lemma 4.1 and Lemma 4.2 and is therefore Pareto optimal. 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 4.1. *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 $i = 1, 2, \ldots, s$ and $\sum_{i=1}^s \alpha_i = 1$, such that*

$$\sum_{i=1}^{s} \alpha_i J_i(\boldsymbol{x}^0, \boldsymbol{u}^*(t), \boldsymbol{x}^*(t)) \leq \sum_{i=1}^{s} \alpha_i J_i(\boldsymbol{x}^0, \boldsymbol{u}(t), \boldsymbol{x}(t)) \tag{3}$$

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

Proof of Lemma 4.1. Let's consider a control $\mathbf{u}(t) \in U$. If the equality in (3) holds, then it must be true that either

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

or there is an i and $j \in \{1, 2, \ldots, s\}$, $i \neq j$ 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 holds, then there exists an $i \in \{1, 2, \ldots, s\}$ 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 4.2. *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* $i = 1, 2, \ldots, s$ *and* $\sum_{i=1}^{s} \alpha_i = 1$, *such that*

$$\sum_{i=1}^{s} \alpha_i J_i(\boldsymbol{x}^0, \boldsymbol{u}^*(t), \boldsymbol{x}^*(t)) < \sum_{i=1}^{s} \alpha_i J_i(\boldsymbol{x}^0, \boldsymbol{u}(t), \boldsymbol{x}(t)) \ \forall \ \boldsymbol{u}(t) \in U, \boldsymbol{u}(t) \neq \boldsymbol{u}^*(t). \tag{4}$$

Given these lemmas, in order to find candidates for Pareto optimality, we simply need to review the requirements for an optimum in the optimal control problem from Section 2 and invoke the Minimum Principle [6]. However, the procedure used will not produce all Pareto-optimal candidates but only those that satisfy the conditions of the above two lemmas [11].

By adapting sufficient conditions of the associated optimal control problem in Section 2, we can state the following theorem.

Theorem 4.2. *The solution* $\boldsymbol{u}^* \in U$ *generating* $\boldsymbol{x}^*$ *is Pareto optimal if there exist*
1.) *an* $\boldsymbol{\alpha} \in \mathbb{R}^s$ *with* $\alpha_i > 0$, $i = 1, 2, \ldots, s$, *and* $\sum_{i=1}^{s} \alpha_i = 1$, *and*
2.) *an absolutely continuous* $\boldsymbol{p}$*:*$[t_0, t_f] \to \mathbb{R}^n$
such that

$$\begin{aligned} &\sum_{i=1}^{s} \alpha_i L_i(\boldsymbol{u}^*(t), \boldsymbol{x}^*(t)) - \boldsymbol{p}^T(t) \boldsymbol{f}(\boldsymbol{u}^*(t), \boldsymbol{x}^*(t)) \\ &- \sum_{i=1}^{s} \alpha_i L_i(\boldsymbol{u}(t), \boldsymbol{x}(t)) + \boldsymbol{p}^T(t) \boldsymbol{f}(\boldsymbol{u}(t), \boldsymbol{x}(t)) \\ &- \dot{\boldsymbol{p}}^T(t)(\boldsymbol{x}^*(t_f) - \boldsymbol{x}) \leq 0 \end{aligned} \tag{5}$$

for all $\boldsymbol{x} \in \mathbb{R}^n$ *and* $\boldsymbol{u} \in U$*, and almost all* $t \in [0, t_f]$*, and*

$$\boldsymbol{p}^T(t)(\boldsymbol{x}^*(t_f) - \boldsymbol{x}) \leq 0 \ \forall \ \boldsymbol{x} \in \Theta, \tag{6}$$

where Θ *is a target set in* $\mathbb{R}^n$*.*

Proof. Suppose we have a solution $\mathbf{u}$ with corresponding trajectory $\mathbf{x}$. Using the theorem and the state equations, we have

$$\begin{aligned} &\sum_{i=1}^{s} \alpha_i L_i(\mathbf{u}^*(t), \mathbf{x}^*(t)) - \sum_{i=1}^{s} \alpha_i L_i(\mathbf{u}(t), \mathbf{x}(t)) \\ &- \frac{d}{dt}\{\mathbf{p}^T(t)(\mathbf{x}^*(t_f) - \mathbf{x}) \leq 0 \end{aligned} \tag{7}$$

for almost all $t \in [t_0, t_f]$. Integrating the state equations with $\mathbf{x}^*(t) = \mathbf{x}(0) = \mathbf{x}^0$, and using the second component of the theorem, we get

$$J(\mathbf{x}^0, \mathbf{u}^*(t), \mathbf{x}^*(t)) \leq J(\mathbf{x}^0, \mathbf{u}(t), \mathbf{x}(t)) \tag{8}$$

where

$$J(\mathbf{x}^0, \mathbf{u}(t), \mathbf{x}(t)) = \sum_{i=1}^{s} \alpha_i \int_0^{t_f} L_i(\mathbf{u}(t), \mathbf{x}(t))dt \tag{9}$$

and $\mathbf{u} \in U$ is arbitrarily selected. Given (8) and the first condition of the theorem, we have satisfied the requirements of the first lemma; and therefore, $\mathbf{u}^*$ is Pareto optimal. □

Here the sufficient conditions for Pareto optimality have been successfully reduced to sufficient conditions for an optimum of the optimal control problem with scalar-valued cost:

$$\sum_{i=1}^{s} \alpha_i J_i(\mathbf{x}^0, \mathbf{u}(t), \mathbf{x}(t)). \tag{10}$$

If there exists $\alpha \in \mathbb{R}^s$ and if the scalar-valued objective function (10) has an optimum $\mathbf{u}^*$ at $\mathbf{x}^0$, then for the problem with a vector-valued objective function, $\mathbf{u}^*$ is also Pareto optimal at $\mathbf{x}^0$. Later, we illustrate the use of these sufficiency conditions in an example.

5. RELAXED CONTROLS

From the previous section, we see that to solve the multiobjective optimal control problem with state constraints, we first need to convert it into a single objective control problem of the form in equations (1a)–(1e). The conversion is commonly done using a weighted-sum method which basically assigns a weight to each objective function then sums them to get the new scalar cost function.

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 [1]. 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 5.1. 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

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

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,

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

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 $\pi_1, \ldots, \pi_k$ to be nonnegative and measurable on $[t_0, t_f]$ such that $\sum_{i=1}^k \pi_i = 1$ and take the functions $\mathbf{u}_1, \ldots, \mathbf{u}_k$ to be measurable on $[t_0, t_f]$ such that $u_i(t) \in U(t)$. For any Borel set E in U(t) let

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

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

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

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

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

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

To μ_t we may associate the vector $(\pi_1(t), \ldots, \pi_k(t), u_1(t), \ldots, u_k(t))$. In fact, every relaxed control μ corresponds to $(\pi_1(t), \ldots, \pi_{n+1}(t), u_1(t), \ldots, u_{n+1}(t))$ where $\pi_i \geq 0$ *a.e.*, $\sum_{i=1}^{n+1} \pi_i = 1$ *a.e.*, $u_i \in U$, $i = 1, \ldots, n+1$.

Definition 5.2 (Relaxed trajectory)**.** An absolutely continuous function $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_n)$ defined on $[t_0, t_f]$ is a relaxed trajectory [1] 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

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

We also write

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

as mentioned above.

Definition 5.3 (Admissible pair)**.** The pair [1] $(\boldsymbol{\phi}, \boldsymbol{\mu})$ 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

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

is integrable.

We can now state the *relaxed* version [1], [2] of the control problem in equations (1a)–(1e):

$$\min_{\boldsymbol{u_i} \in U} \mathbf{J}_r(\mathbf{z}) = \Phi(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} f^0(\mathbf{x}(t), \boldsymbol{\mu_t}) dt \tag{11a}$$

s.t.

$$\dot{\mathbf{x}}(t) = \mathbf{g}(\mathbf{x}(t), \boldsymbol{\mu_t}), \qquad \mathbf{x}(0) = \mathbf{x}^0 \tag{11b}$$

$$\mathbf{u}_i \in U \tag{11c}$$

$$\boldsymbol{\eta}(\mathbf{x}(t)) \leq 0 \tag{11d}$$

where

$$\boldsymbol{\mu}(t) = (\pi_1(t), \ldots, \pi_{n+1}(t), \boldsymbol{u}_1(t), \ldots, \boldsymbol{u}_{n+1}(t)),$$

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

Then

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

5.1. **Necessary Conditions for Optimal Controls.** 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 [7]. Fortunately, Berkovitz and Medhin [1] offer the following theorem concerning the necessary conditions for a relaxed pair to be an optimal solution of the control problem described in (1a)–(1e). We use these conditions to numerically solve a social network problem later in this paper.

Assumption 5.1. 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 [1].

Theorem 5.1. *Suppose Assumption* 5.1 *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 (1a)–(1e) 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* $\beta, \boldsymbol{\gamma}, \lambda^0, (\lambda^0 \geq 0)$, *such that*

1. $|p(t_f)| + \sum_l |\lambda_l(0^+)| + \sum_l |\beta_l| + \lambda^0 \neq 0$,
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. $\boldsymbol{p}^T(t_f) = \beta \Phi_x(\boldsymbol{\phi}_0(t_f)) + \sum_l \boldsymbol{\gamma}_l \nabla_{\boldsymbol{x}}[\eta_l(\boldsymbol{\phi}_0(t_f))]$

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

5.

$$-\left[\boldsymbol{p}-\sum_{l=1}^{p}\lambda_l(t)[\nabla_{\boldsymbol{x}}[\eta_l(\boldsymbol{\phi}_0(t))]]^T\right]\boldsymbol{g}(\boldsymbol{\phi}_0(t),\boldsymbol{z}_{0t})+\lambda^0 f^0(\boldsymbol{\phi}_0(t),\boldsymbol{z}_{0t})$$

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

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) [4] used DE to solve multimodal optimal control problems with great success and Feng-Sheng Wang and Ji-Pyng Chiou (1997) [21] 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 [19] 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 [12], [17].

6. NUMERICAL METHOD

Evolutionary algorithms (EA), like Differential Evolution (DE), are well-suited for solving multiobjective optimization problems.

6.1. **Differential Evolution.** Differential Evolution (DE) is a population-based search method developed by Storn and Price [19] to handle problems with multiple objectives over continuous domains. DE is an appealing approach for solving MOCPs because it eliminates the need to consider function continuity, convexity, or concavity unlike some traditional search techniques where the complexities must be given great attention. In addition, DE is capable of providing a complete set of Pareto-optimal solutions in a single run [18]. It is a stochastic population-based direct search method that improves some randomly generated initial population through *mutation, crossover*, and *selection*. The algorithm includes the following steps.

6.1.1. *Steps for Differential Evolution (DE) Algorithm.*

- **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:

$$u_{j,i}^g = u_{j,i_{min}}^g + rand() * (u_{j,i_{\max}}^g - u_{j,i_{min}}^g), \tag{12}$$

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, $j = 1, 2, \ldots, NP$.

- **Step 2: Mutation**
  After population initialization, the population undergoes mutation. For each population vector, $\mathbf{u}_j^g$, $j = 1, \ldots, NP$, DE generates NP mutated vectors, $\hat{\mathbf{z}}_j^g$:

$$\hat{\mathbf{z}}_j^g = \mathbf{u}_{j_1}^g + W * (\mathbf{u}_{j_2}^g - \mathbf{u}_{j_3}^g) \tag{13}$$

  where j_1, j_2, j_3 are random mutually different vectors belonging to $\{1, 2, \ldots, NP\}$ 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 [19].
- **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, $\mathbf{u}_{j_1}^g, \mathbf{u}_{j_2}^g, \mathbf{u}_{j_3}^g$ involved in mutation or from the corresponding parent vector, $\mathbf{u}_j^g$ [19]. 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,

$$z_{j,i}^g = \begin{cases} u_{j_1,i}^g + W * (u_{j_2,i}^g - u_{j_3,i}^g) & \text{if } rand() < CR \text{ or } i = \hat{i}, \\ u_{j,i}^g & \text{otherwise.} \end{cases} \tag{14}$$

  where $rand()$ is a random number in $[0, 1]$ and $\hat{i}$ is a randomly selected index from $\{1, 2, \ldots, D\}$ [3].
- **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:

$$\mathbf{u}_j^{g+1} = \begin{cases} \mathbf{z}_j^g & \text{if } J(\mathbf{z}_j^g) \leq J(\mathbf{u}_j^g), \\ \mathbf{u}_j^g & \text{otherwise} \end{cases} \tag{15}$$

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.

In the numerical procedure that follow, 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_f which gives rise to a TPBVP which we must solve subject to the aforementioned constraints on the state and control variables. 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 the 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 5.1 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 5.1 using these fixed multipliers. When some desired stopping criteria is achieved, we stop and output the optimal control.

6.2. **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 5.1 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 6.2.1.

- **Step 1:** First, in the initial generation, we initialize a population of NP controls $\mathbf{u}^g = [\mathbf{u}^{(1)}, \mathbf{u}^{(2)}, \ldots, \mathbf{u}^{(NP)}]$. For the i-th member of the population, $i \in \{1, 2, \ldots, NP\}$, 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 $\mathbf{u}^{(i)}(t_k)$, $k = 0, 1, \ldots, N-1$.

  Note: We randomly initialize controls to meet the control constraints in (**??**) and we only choose controls that produce state variables that do not violate the state constraints in (11d). 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 $\mathbf{x}^g(t_k) = [\mathbf{x}^{(1)}(t_k), \mathbf{x}^{(2)}(t_k), \ldots, \mathbf{x}^{(NP)}(t_k)]$, $k = 0, 1, \ldots, N-1$.
- **Step 3:** In the initial generation, we randomly guess NP sets of $\boldsymbol{\gamma}$ and $\boldsymbol{\lambda}$ values: $\boldsymbol{\gamma}^g = [\boldsymbol{\gamma}^{(1)}, \boldsymbol{\gamma}^{(2)}, \ldots, \boldsymbol{\gamma}^{(NP)}]$, $k = 0, 1, \ldots, N-1$ and $\boldsymbol{\lambda}^g(t_k) = [\boldsymbol{\lambda}^{(1)}(t_k), \boldsymbol{\lambda}^{(2)}(t_k), \ldots, \boldsymbol{\lambda}^{(NP)}(t_k)]$, $k = 0, 1, \ldots, N-1$. For each set

of $\boldsymbol{\lambda}$ multipliers, we satisfy the conditions that $\boldsymbol{\lambda}^{(i)}(t_k) \geq 0$, $i = 1, \ldots, NP$ 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 $\mathbf{p}^g(t_k) = [\mathbf{p}^{(1)}(t_k), \mathbf{p}^{(2)}(t_k), \ldots, \mathbf{p}^{(NP)}(t_k)]$, $k = 1, \ldots, N-1$. Use $\mathbf{p}^{(i)}(t_f)$, $i = 1, \ldots, NP$, 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, $J(\mathbf{x}^{(i)}(t_k), \mathbf{u}^{(i)}(t_k))$, $i = 1, \ldots, NP$. 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.

7. SOCIAL NETWORKS

Several key concepts [22] form the basis of social network analysis and are fundamental to our study of social networks.

7.1. **Methodology for Social Networks.** *Nodes* form the basis of social networks and are often referred to as actors, actors 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.

There are two tools in particular which are often seen in the literature to represent social networks: *matrices* and *graphs*. In this work, we'll use both in illustrative examples of friendship networks. 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 $1, 2, \ldots, N$. 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.

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.

7.2. **Social Forces Model for Social Networks.** Different modeling approaches have been developed to model social networks and social interaction. In this work, we take a more physical approach inspired by Helbing's social forces model for pedestrian walking behavior. We adapt Helbing's model to describe social interaction and ultimately, formulate a friendship model mathematically using the notion of social forces. In essence, actors interact as though they were subject to acceleration and repulsive forces when making their friendship choices. This approach assumes 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. In the context of friendship networks, social forces theory assumes that each actor possesses a specific attitude toward making friends, a desire to befriend those who share their preferences and attributes and that they respect the private space of others. Consequently, following Helbing and Molnar's theory, these rules describing social interaction can be placed into a set of equations of motion [8].

7.2.1. *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 [9] in our model of network dynamics:

- All actors consider the same attributes when attempting to make friends.
- Actors do not change categories within a particular attribute.
- Relationships between actors depend on shared preferences for attributes and categories.
- 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.
- Each actor attempts to maximize his status in the social group, i.e, he wishes to form as many relationships as possible.
- Finally, the objective functional of each actor decreases with an increase in shared attribute preferences and categories.

7.2.2. *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, $1, \ldots, m$

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

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

$\mathbf{u}_i(t)$ – vector describing actor i's control for each attribute, $1, \ldots, m$

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 Section 3.3 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, τ_i will be small for those who are more reluctant to change their attribute preferences permanently.

8. NUMERICAL EXPERIMENT

The aim of this experiment is to formulate an MOCP using a small social network model consisting of just three actors ($i = 1, 2, 3$) and two attributes ($m = 2$). To solve the problem, we translate the MOCP into an optimal control problem with a single cost function using the weighted-sum method. The we *relax* the problem and use

Algorithm 6.2.1 to solve the relaxed problem numerically. The objective here is not to go into great detail regarding the model's development. The reader is directed to [13] for a thorough explanation of the model. 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 some fixed time interval, $[t_0, t_f]$. 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.

8.1. **Problem Statement.** Consider a social group with $N = 3$ actors ($i = 1, 2, 3$ and $j = 1, 2, 3$) 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 governs social interaction is given by

$$\text{(16a)} \qquad \dot{\mathbf{r}}_i = \mathbf{v}_i$$

$$\text{(16b)} \qquad \begin{aligned} \dot{\mathbf{v}}_i = {} & \frac{1}{\tau_i}(\mathbf{v}_i^0 - \mathbf{v}_i) - \nabla_{\mathbf{r}_i}\Big[\sum_{j\neq i} \|\mathbf{u}_i - \mathbf{u}_j\|^2 \\ & \cdot(1 + ((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\|)^2 - \|\mathbf{v}_j\Delta t\|^2)) \\ & \cdot \exp\{-l_{ij}((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\|)^2 - \|\mathbf{v}_j\Delta t\|^2)\}\Big] \end{aligned}$$

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

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

$$\text{(16c)} \qquad \mathbf{r}_i(0) - \boldsymbol{\delta}_{i_{min}} \leq \mathbf{r}_i(t) \leq \mathbf{r}_i(0) + \boldsymbol{\delta}_{i_{\max}}$$

$$\text{(16d)} \qquad -\boldsymbol{\delta}_{i_{min}} \leq \mathbf{u}_i(t) \leq \boldsymbol{\delta}_{i_{\max}}$$

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:

$$\text{(16e)} \qquad J_i = \sum_{j\neq i} \|\mathbf{r}_i(t_f) - \mathbf{r}_j(t_f)\|^2 + \mathcal{N}_i \int_{t_0}^{t_f} \|\mathbf{u}_i(t)\|^2 \, dt$$

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

The multiobjective optimal control problem can be stated as: find the 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 of controls.

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

TABLE 1. Parameters for each actor: $i = 1, 2, 3$

i	l_{ij}	τ_i	$\mathcal{N}_i$	Δt
1	0.3	1/20	1	0.02
2	0.1	1/10	1	0.02
3	0.2	1/15	1	0.02

TABLE 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. Initial rate of change, $\mathbf{v}_i^0$, of attribute preferences for each actor $i = 1, 2, 3$ at $t = 0$

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

8.2. **Implementation.** 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*. The new single objective optimal control problem is

$$\begin{aligned}\min_{\mathbf{u}} J = &\frac{1}{3}\left\{\|\mathbf{r}_1(t_f) - \mathbf{r}_2(t_f)\|^2 + \|\mathbf{r}_1(t_f) - \mathbf{r}_3(t_f)\|^2 + \mathcal{N}_1 \int_{t_0}^{t_f} \|\mathbf{u}_1(t)\|^2 \, dt\right\} \\ &+\frac{1}{3}\left\{\|\mathbf{r}_2(t_f) - \mathbf{r}_1(t_f)\|^2 + \|\mathbf{r}_2(t_f) - \mathbf{r}_3(t_f)\|^2 + \mathcal{N}_2 \int_{t_0}^{t_f} \|\mathbf{u}_2(t)\|^2 \, dt\right\} \\ &+\frac{1}{3}\left\{\|\mathbf{r}_3(t_f) - \mathbf{r}_1(t_f)\|^2 + \|\mathbf{r}_3(t_f) - \mathbf{r}_2(t_f)\|^2 + \mathcal{N}_3 \int_{t_0}^{t_f} \|\mathbf{u}_3(t)\|^2 \, dt\right\}\end{aligned} \tag{17a}$$

s.t.

$$\dot{\mathbf{r}}_i = \mathbf{v}_i \tag{17b}$$

$$\text{(17c)}\qquad \begin{aligned}\dot{\mathbf{v}}_i &= \frac{1}{\tau_i}(\mathbf{v}_i^0 - \mathbf{v}_i) - \nabla_{\mathbf{r}_i}\Big[\sum_{j\neq i}\|\mathbf{u}_i - \mathbf{u}_j\|^2 \\ &\quad \cdot(1 + ((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\|)^2 - \|\mathbf{v}_j\Delta t\|^2) \\ &\quad \cdot \exp\{-l_{ij}((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\|)^2 - \|\mathbf{v}_j\Delta t\|^2)\}\Big],\end{aligned}$$

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

and

$$\text{(17d)}\qquad \mathbf{r}_i(0) - \boldsymbol{\delta}_{i_{min}} \leq \mathbf{r}_i(t) \leq \mathbf{r}_i(0) + \boldsymbol{\delta}_{i_{\max}}$$

$$\text{(17e)}\qquad -\boldsymbol{\delta}_{i_{min}} \leq \mathbf{u}_i(t) \leq \boldsymbol{\delta}_{i_{\max}}.$$

We derive the necessary conditions as follows for solving the optimal control problem using Theorem 5.1. 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

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

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

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

By combining these two, we get

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

In the remainder of this section, we will use $\mathbf{x} = [\mathbf{r}, \mathbf{v}]^T$ to simplify the notation. The Hamiltonian is

$$\begin{aligned}H = &- [\mathbf{p} - \lambda_{1,11}[\nabla\eta_{1,11}]^T + \lambda_{1,12}[\nabla\eta_{1,12}]^T + \lambda_{1,21}[\nabla\eta_{1,21}]^T + \lambda_{1,22}[\nabla\eta_{1,22}]^T \\ &+ \lambda_{1,31}[\nabla\eta_{1,31}]^T + \lambda_{1,32}[\nabla\eta_{1,32}]^T + \lambda_{2,11}[\nabla\eta_{2,11}]^T + \lambda_{2,12}[\nabla\eta_{2,12}]^T\end{aligned}$$

$$+ \lambda_{2,21}[\nabla\eta_{2,21}]^T + \lambda_{2,22}[\nabla\eta_{2,22}]^T + \lambda_{2,31}[\nabla\eta_{2,31}]^T + \lambda_{2,32}[\nabla\eta_{2,32}]^T]\mathbf{f}(\mathbf{x},\mathbf{u})$$

and

$$\mathbf{f}(\mathbf{x},\mathbf{u}) = \begin{bmatrix} \dot{r}_{11} = v_{11} \\ \dot{r}_{12} = v_{12} \\ \dot{r}_{21} = v_{21} \\ \dot{r}_{22} = v_{22} \\ \dot{r}_{31} = v_{31} \\ \dot{r}_{32} = v_{32} \\ \dot{r}_{32} = v_{32} \\ \dot{v}_{11} = \frac{1}{\tau_1}(v_{11}^0 - v_{11}) - \nabla_{r_{11}}[F_{12} + F_{13}] \\ \dot{v}_{12} = \frac{1}{\tau_1}(v_{12}^0 - v_{12}) - \nabla_{r_{12}}[F_{12} + F_{13}] \\ \dot{v}_{21} = \frac{1}{\tau_2}(v_{21}^0 - v_{21}) - \nabla_{r_{21}}[F_{21} + F_{23}] \\ \dot{v}_{22} = \frac{1}{\tau_2}(v_{22}^0 - v_{22}) - \nabla_{r_{22}}[F_{21} + F_{23}] \\ \dot{v}_{31} = \frac{1}{\tau_3}(v_{31}^0 - v_{32}) - \nabla_{r_{31}}[F_{31} + F_{32}] \\ \dot{v}_{32} = \frac{1}{\tau_3}(v_{32}^0 - v_{32}) - \nabla_{r_{32}}[F_{31} + F_{32}] \end{bmatrix}$$

where

$$F_{ij} = \Big[\|\mathbf{u}_i - \mathbf{u}_j\|^2 (1 + ((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\|)^2 - \|\mathbf{v}_j\Delta t\|^2) \cdot \exp\{-l_{ij}((\|\mathbf{r}_i - \mathbf{r}_j\| + \|\mathbf{r}_i - \mathbf{r}_j - \mathbf{v}_j\Delta t\|)^2 - \|\mathbf{v}_j\Delta t\|^2)\}\Big]$$

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

where $\mathbf{f}_{\mathbf{x}}$ is a $n \times n$ jacobian matrix. We also have the costate boundary conditions

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

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

with $\beta \geq 0$.

The optimal solution must satisfy the condition

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

8.2.1. *Numerical Results and Analysis.* We used the above algorithm to solve the weighted-sum problem with the following parameters and stopping criteria:

1. **DE Parameters:** $NP = 15,000$, $W = 0.5$, 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 [13] which was 0.2251.

The algorithm was implemented in C++ and took approximately ? to run. The fourth-order Runge Kutta method was used to numerically solve the TPBVP. Using this method, J^* was 0.0720 which is better than 0.2251 produced by the gradient algorithm.

Figure 1 plots the optimal trajectory for the evolution of actor preferences over time and Figure 2 plots the social distance between actor 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 , these figures indicate that actors 1 and 2 are mutually close; actors 1 and 3 are mutually close; and actors 2 and 3 are not considered close.

9. CONCLUSION

We began by converting the multiobjective optimal control problem to one with a single objective using a weighted-sum method with equal weights. Employing necessary conditions of optimality we could structure a numerical method in which we used Differential Evolution to minimize the Hamiltonian and ultimately the objective

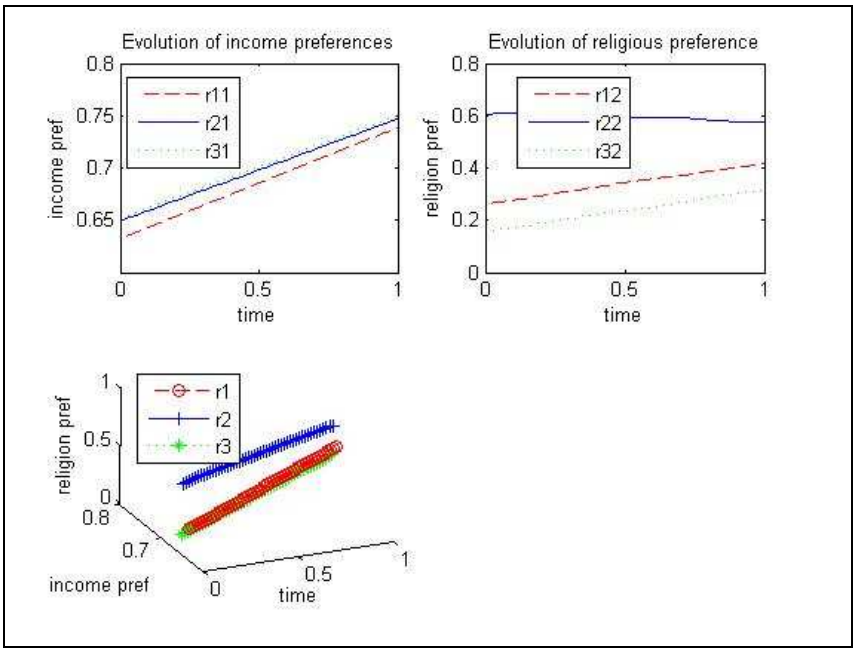

FIGURE 1. Optimal Trajectory - Actor Preferences

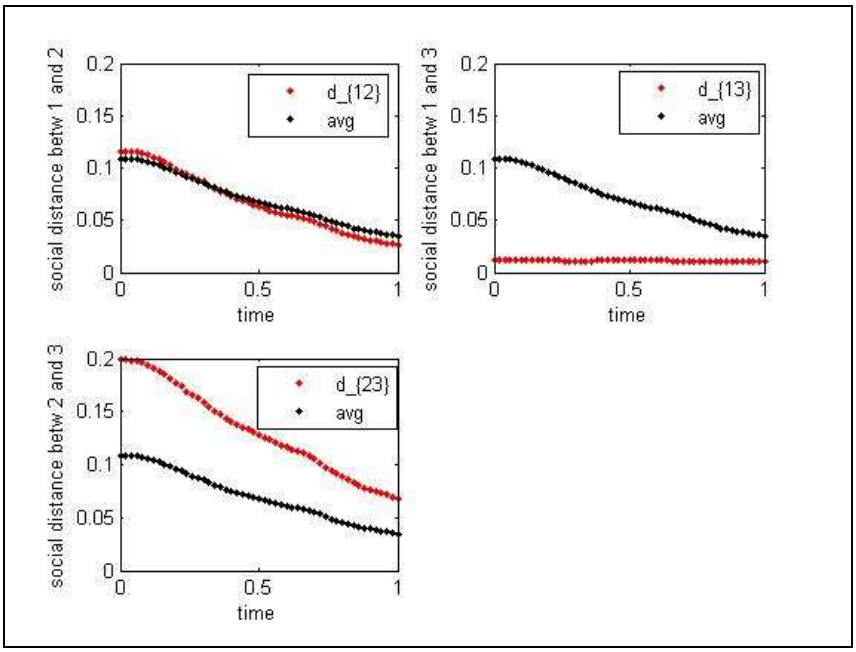

FIGURE 2. Distance between actor preferences using Algorithm 2

function value which was lower than that obtained using a gradient method used in an earlier paper [13]. Using the necessary conditions in the manner shown here considerably enhances the effectiveness of the DE method both in speed and accuracy.

REFERENCES

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

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

[3] 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.

[4] 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.

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

[6] M. Gopal. *Modern Control System Theory*. Number 2 in Wiley-Eastern Limited. John Wiley and Sons, New York, 1993.

[7] 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.

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

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

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

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

[12] 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.

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

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

[15] 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.

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

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

[18] R. Sarker and H.A. Abbass. Differential evolution for solving multi-objective optimization problems. report, University Of New South Wales, Northcott Drive, Canberra ACT, 2600, Australia.

[19] 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.

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

[21] 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.

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