Negotiation Between Dynamical Systems with Connectivity Constraints

Yaniv Ben Shoushan* and Daniel Zelazo†
Technion - Israel Institute of Technology, Haifa 32000, Israel

This work presents a real-time sub-optimal solution for an agreement problem in multi-agent systems. Each agent is modeled with integrator dynamics and has an associated objective function it wishes to minimizes. The agents must coordinate to reach an agreement on their state in finite time. Each agent is able to communicate with other neighbors according to a fixed connectivity structure, and each agent must ensure that it will maintain the necessary communication range with its neighbors, meaning that there will be no damage to the connectivity structure that may be caused by a larger distance between the agents than the communication radius defined. The main contribution of this work is to find a distributed solution to this problem in finite-time.

I. Introduction

There are many open challenges involved in the study of communication constraints between agents in multi-agent systems. Distributed aerospace systems, such as multiple spacecraft, fleets of autonomous rovers, unmanned aerial vehicles have been identified as a new paradigm for a wide array of applications [6,24,25]. It is envisioned that distributed aerospace technologies will enable the implementation of a spatially distributed network of vehicles that collaborate toward a single collective scientific, military, or civilian goal. Moreover, distributed systems lead to higher degrees of scalability and adaptability in response to changes in the mission goals and system capabilities [1, 3].

In recent years, there has been an increasing number of contributions dealing with the managing of communication in multi-agent systems. One of the biggest problems when dealing with multi-agent system is the communication between the agents and how to optimize their data transfer. Another problem is when trying to coordinate between all the agents when each agent has its own tasks and all of the agents must agree at a specific meeting point. Each agent will have to decide in real-time what to do in order to succeed at the group task and to perform its own task. A fast real-time communication channel is needed so all the agents can communicate and make real time decisions. Many articles have been published on this subject in the past few years; a lot of them deal with the transmission problem and stochastic problems [9, 12].

It is well known that in communication there is an effective transmission and receiving radius. In communication, it is desirable to ensure that the bit-error-rate (BER) is low,

*The Technion Autonomous Systems Program, Technion-Israel Institute of Technology, Haifa 32000, Israel (yanivbs1@gmail.com).

†Faculty of Aerospace Engineering, Technion-Israel Institute of Technology, Haifa 32000, Israel (dzelazo@technion.ac.il).

Preprint submitted to 57th Israel Annual Conference on Aerospace Sciences.
Received September 13, 2016.

and the signal-noise-ratio (SNR) is high. There is always a trade-off between the data rate and the transmission range [7, 12]. This trade-off creates an effective transmission radius leading to acceptable communication. For multi-agent systems, this is an important property that we want to explore in this work.

This work considers an agreement problem for multi-agent systems. The main idea is to use real-time communication to come to an agreement between the agents in finite time. Additionally, each agent will try to minimize their own objective function. Each agent must also ensure that it preserves its initial neighborhood defined by the communication graph so as not to exceed the communication radius. What will distinguish this work from others published to this point is the ability to control the group of agents distributedly and in real-time.

This research is based on the article [3] which presented a distributed real-time algorithm between agents in a multi-agents system. Each agent was assigned its own specific task and a finite time at which all the agents must agree upon a meeting point. The algorithm proposed in [3], however, did not take into consideration communication constrains between the agents. This work will focus on this point and extend the algorithm presented [3] to explicitly handle communication constraints of each agent.

II. The Distance Constrained Preference Agreement Problem

In this section, we formally introduce the preference agreement problem with connectivity constraints. Here we will present the optimization model for the problem and discuss its solution from a centralized perspective. We show that the problem can be modeled as a quadratic program and provide simulation results showing the behavior of the optimal solutions.

We consider a group of n agents that aim to minimize individual objective functions while satisfying a terminal constraint where all agents must meet at an agreed point in space in finite time. Furthermore, each agent must satisfy a distance constraint between its neighbors to ensure communication is not lost during the resulting trajectories. Each agent is modeled as a simple discrete-time integrator,

$$x(t+1) \;=\; x(t) + u(t),\; x_i(0) = x_{i0}, i = 1, \ldots, n, \tag{1}$$

where $x(t) = [x_1(t) \cdots x_n(t)]^T$ is the aggregate agent state and $u(t) = [u_1(t) \cdots u_n(t)]^T$ the control. The basic assumption is that the agents can communicate with one another according to a given and fixed connectivity graph, denoted $\mathcal{G}$. The graph is associated with an incidence matrix E, such that $[E]_{ij} = +1$ if there is an outgoing edge from node i to node j, $[E]_{ij} = -1$ if there is an incoming edge from node j to node i, and $[E]_{ij} = 0$ otherwise [13]. In this work we assume that the graph is a spanning tree, and thus $E \in \mathbb{R}^{n \times n-1}$ has full column rank and $E^T \mathbb{1} = 0$ by construction.[a] The communication between the agents is also assumed to be synchronous.

The self-interest of every agent is modeled by a quadratic objective function,

$$J_i(t_0, T, x_i, u_i) = \frac{1}{2}\left(\sum_{t=t0}^{T-1} q_i(x_i(t+1) - \xi_i)^2 + r_i u_i(t)^2\right), \tag{2}$$

where ξ_i is the preference state of agent i, $q_i > 0$ represents a state weight, and $r_i > 0$ represent the control weights. The terminal time constraint requires that all agents are

[a] The vector of all ones is denoted $\mathbb{1}$.

Preprint submitted to 57th Israel Annual Conference on Aerospace Sciences.
Received September 13, 2016.

in agreement at a finite time T, meaning $x_1(T) = x_2(T) = \cdots = x_n(T)$, or equivalently, $E^T x(T) = 0$. The distance constraint between agents can be expressed as $|E^T x(t)| \leq R\mathbb{1}$.

With the above model, the centralized optimal control problem (OCP) can therefore be stated as,

$$\begin{aligned} OCP(t_0, T, x_0): \quad \min_{x,u} \quad & \sum_{i=1}^{n} J_i(t_0, T, x_i, u_i) \\ s.t. \quad & x(t+1) = x(t) + u(t), x(t_0) = x_0 \\ & E^T x(T) = 0 \\ & -R\mathbb{1} \leq E^T x(t) \leq R\mathbb{1},\ t = t_0, \ldots, T. \end{aligned} \tag{3}$$

The OCP (3) can also be expressed as a quadratic program with equality and inequality constraints,

$$\begin{aligned} \min_{z} \quad & \frac{1}{2} z^T H z + f^T z \\ \text{s.t.} \quad & A_{eq} z = b_{eq} \\ & A_{ieq} z \leq b_{ieq}, \end{aligned}$$

where $z = [x^T \ u^T]^T$, and H, f, A_{eq}, b_{eq}, A_{ieq}, b_{ieq} can be defined directly from the description of the OCP in (3).

We now present a numerical example for the OCP with the distance and terminal constraints defined in (3). In this example, we consider $n = 7$ agents and a terminal time of $T = 10$ seconds. The radius defined is $R = 35$ with communication graph shown in Figure 1, and the problem data for each agent is given as

$$r = \begin{bmatrix} 3 \\ 7 \\ 9 \\ 4 \\ 8 \\ 7 \\ 1 \end{bmatrix}, \quad q = \begin{bmatrix} 6 \\ 4 \\ 9 \\ 1 \\ 5 \\ 5 \\ 5 \end{bmatrix}, \quad \xi = \begin{bmatrix} -27 \\ 17 \\ -29 \\ 2 \\ 45 \\ 32 \\ -22 \end{bmatrix}, \ x_0 = \begin{bmatrix} 18.5821 \\ 43.6214 \\ 13.9127 \\ -17.2734 \\ 37.7576 \\ 36.9890 \\ 5.5656 \end{bmatrix}.$$

The optimal trajectories were found in MATLAB using the quadratic programming formulation. Figures 2(b) and 2(a) shows the resulting trajectories. We can see that each agent pursuing its preference, while at the terminal time T, all of the agents go to a common meeting point ($x(T) = 0.4059\mathbb{1}$) In Figure 2(a) we can see that each agent is trying to pursue its preference value while they are satisfying the distance constraint between adjacent agents for the entire trajectory. As we can see in Figure 2(c), there are 3 edges which the absolute distance is larger than R, meaning this optimization works and the maximal distance between the agents was not passed the defined radius. The optimization prevent from the agent to fulfil their own individual objective so the group of agents can stand by the distance constraints.

This work will be based on the understanding of how the OCP behaves once the initial conditions are changed, and how that will affect the rest of the trajectory. In this direction, it is also useful to study the corresponding *dual problem* associated with (3).

Preprint submitted to 57th Israel Annual Conference on Aerospace Sciences.
Received September 13, 2016.

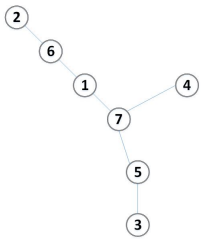
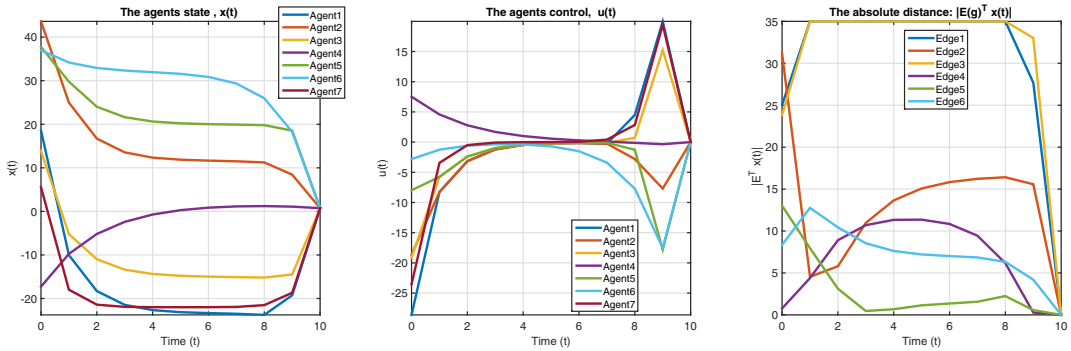


Figure 1. A spanning tree on 7 nodes.



(a) Trajectories of $\overline{\mathbf{x}}$ for OCP with distance constraints.

(b) Trajectories of $\overline{\mathbf{u}}$ for OCP with distance constraints.

(c) A plot of $|E^T\mathbf{x}(t)|$.

Figure 2. Optimal trajectories for the optimal control problem (3)**.**

We define the relaxed Lagrangian function to be

$$\begin{aligned}\mathcal{L}(\mathbf{x},\mathbf{u},\delta,\zeta,\mu) &= \sum_{i=1}^{n} J_i(t_0,T,x_i,u_i) + \mu^T E^T (I_n \otimes e_{T,T})^T \mathbf{x} + \\ &\quad \delta^T\left(\overline{E}^T\mathbf{x} - R\mathbb{1}\right) + \zeta^T\left(-\overline{E}^T\mathbf{x} - R\mathbb{1}\right). \end{aligned} \tag{4}$$

Here, μ, δ, and ζ are the Lagrange multipliers associated with system constraints. The notation $e_{m,n}$ denotes the n-dimensional unit vector with $[e_{m,n}]_m = 1$, and 0 otherwise. The matrix $\overline{E} = E \otimes I_{T-t_0}$ is an inflated version of the incidence matrix.

The dual function is obtained by minimizing the partial Lagrangian subject to the dynamical constraints,

$$g(\mu,\zeta,\delta) = \min_{\mathbf{x},\mathbf{u}} \quad \mathcal{L}(\mathbf{x},\mathbf{u},\delta,\zeta,\mu) \tag{5}$$

$$\text{s.t.} \quad x(t+1) = x(t) + u(t), x(t_0) = x_0. \tag{6}$$

The dual problem is obtained by maximizing the dual function,

$$\max_{\mu,\zeta\geq 0,\delta\geq 0} g(\mu,\zeta,\delta). \tag{7}$$

The optimal solution of the primal and dual problems are denoted by $(\bar{\mathbf{x}},\bar{\mathbf{u}},\bar{\mu},\bar{\delta},\bar{\zeta})$. Let $J(t_0,T,x,u) = \sum_{i=1}^{n} J_i(t_0,T,x_i,u_i)$ be the sum of objectives functions. Since $\text{OCP}(t_0,T,x_0)$ is a convex problem with linear constraints, we have strong duality which implies

$$g(\bar{\mu},\bar{\zeta},\bar{\delta}) = J(t_0,T,\bar{\mathbf{x}},\bar{\mathbf{u}}). \tag{8}$$

Preprint submitted to 57th Israel Annual Conference on Aerospace Sciences.
Received September 13, 2016.

For a given initial time t and initial condition $x(t)$, we denote the optimal primal and dual solutions associated with the $OCP(t, T, x(t))$ by

$$(\bar{x}^{(t,x(t))}, \bar{u}^{(t,x(t))}, \bar{\mu}^{(t,x(t))}, \bar{\zeta}^{(t,x(t))}, \bar{\delta}^{(t,x(t))}). \tag{9}$$

Thus, for example, $\bar{x}^{(t,x(t))}$ denotes the optimal state trajectory beginning at time t with initial condition $x(t)$. Using this notation we will state a principle of optimality result for dynamic programming in the context of OCP to study the relation between the optimal trajectories of different instances of OCP.

Lemma 1. *(Principle of optimality) . The optimal trajectories generated by the $OCP(t, T, z)$ and $OCP(t+1, T, w)$ with $w = z + \bar{u}^{(t,z)}(t)$ satisfy*

$$\begin{aligned}(\bar{x}^{(t,z)}(\tau), \bar{u}^{(t,z)}(\tau), \bar{\mu}^{(t,z)}(\tau), \bar{\zeta}^{(t,z)}(\tau), \bar{\delta}^{(t,z)}(\tau)) \quad = \\ (\bar{x}^{(t+1,w)}(\tau), \bar{u}^{(t+1,w)}(\tau), \bar{\mu}^{(t+1,w)}(\tau), \bar{\zeta}^{(t+1,w)}(\tau), \bar{\delta}^{(t+1,w)}(\tau)), \ , \tau = t+1 \dots T.\end{aligned}$$

Proof. Concerning the primal solution, the initial condition for $OCP(t+1, T, w)$ corresponds to the point $\bar{x}^{(t,z)}(t+1)$. The remaining statement is a direct application of the principle of optimality for dynamic programming [5], and its uniqueness is due to the strict convexity of the problem statement . The statement concerning the dual solution is a direct consequence of the first statement, in particular, we have $\bar{x}^{(t,z)}(T) = \bar{x}^{(t+1,w)}(T)$. A necessary condition for optimality (the KKT conditions) is

$$0 = \frac{\partial}{\partial \bar{x}(T)} \mathcal{L}(\mathbf{x}, \mathbf{u}, \delta, \zeta, \mu) \quad = \quad Q(\bar{x}(T) - \xi) + E\overline{\mu} + \overline{E}\delta - \overline{E}\zeta.$$

This system of n equations admits a unique solution since the incidence matrix E is full column-rank (the communication graph is a tree). Similarly,

$$0 = \frac{\partial}{\partial \bar{x}(t)} \mathcal{L}(\mathbf{x}, \mathbf{u}, \delta, \zeta, \mu) \quad = \quad Q(\bar{x}(t) - \xi) + (I_n \otimes e_{t,T})^T \overline{E}\delta - (I_n \otimes e_{t,T})^T \overline{E}\zeta,$$

for $t = t_0, \dots, T-1$, which also admits a unique solution. □

III. Distributed and Finite-time Algorithms

The previous sections address the optimal control problem from a centralized perspective. The main goal of this work is to develop *distributed* algorithms for solving the optimal control problem. First, we propose a distributed approach based on a *dual decomposition sub-gradient algorithm* [5]. We show how (3) be solved using this asymptotic method by distributing the computation to each individual and through communication. Second, we propose a sub-optimal modification to this algorithm that attempts to solve (3) in *finite-time*. This solution approach, based on the *shrinking horizon preference agreement problem* introduced in [3], is based on a modification of the dual-decomposition sub-gradient algorithm. Both approaches are then demonstrated with some numerical examples.

Preprint submitted to 57th Israel Annual Conference on Aerospace Sciences.
Received September 13, 2016.

A. Dual Decomposition Sub-Gradient Algorithm

A decomposition method for solving (3) is an algorithm that attempts to divide the main problem into smaller sub-problems that coordinate with each other to solve the original problem. In this section, we present a dual decomposition sub-gradient algorithm [5] for distributedly solving (3).

As described in Chapter II, the OCP (3) can be solved using centralized methods. The challenge in finding a distributed solution that can be solved by each agent is that the constraints couple neighboring agents together. Note however, that the objective function is already in a separable form.

Observe also that the Lagrangian function (4) associated with (3) is also *not* separable. In particular, the multipliers μ, δ, ζ are associated with the *edges* in the graph. However, exploiting the properties of the incidence matrix, we can define new variables associated with the nodes in the graph. In particular, let

$$\gamma := E\mu \tag{10}$$
$$\lambda := \overline{E}\delta \tag{11}$$
$$\beta := \overline{E}\zeta. \tag{12}$$

This transformation is unique, since E (and also $\overline{E}$) has full column-rank. Consequently, it is also possible to transform the node variables back to edges with the following expression,

$$\mu = (E^T E)^{-1} E^T \gamma. \tag{13}$$

Similar transformations can be used for δ and ζ.

After this variable transformation, we can now write the partial Lagrangian in a separable form,

$$\begin{aligned} \mathcal{L}(\mathbf{x}, \mathbf{u}, \delta, \zeta, \mu) &= \sum_{i=1}^{n} \tilde{\mathcal{L}}_i(x_i, u_i, \lambda_i, \beta_i, \gamma_i) - R(\delta + \zeta)^T \mathbb{1}, \\ \tilde{\mathcal{L}}_i(x_i, u_i, \lambda_i, \beta_i, \gamma_i) &= J_i(t_0, T, x_i, u_i) + \gamma_i e_{T,T}^T x_i + (\lambda_i - \beta_i)^T x_i. \end{aligned} \tag{14}$$

Observe that

$$\arg\min_{\mathbf{x},\mathbf{u}} \mathcal{L}(\mathbf{x}, \mathbf{u}, \delta, \zeta, \mu) = \arg\min_{x_i, u_i} \sum_{i=1}^{n} \tilde{\mathcal{L}}_i(x_i, u_i, \lambda_i, \beta_i, \gamma_i). \tag{15}$$

We are now prepared to describe the dual-decomposition sub-gradient algorithm. At each iteration of the algorithm, each agent computes the dual function for a fixed value of the multiplier. So for iteration k, agent i solves the problem

$$\begin{aligned} \begin{bmatrix} \overline{x}_i^{[k+1]} & \overline{u}_i^{[k+1]} \end{bmatrix} &= \arg\min_{x_i, u_i} \tilde{\mathcal{L}}_i(x_i, u_i, \lambda_i^{[k]}, \beta_i^{[k]}, \gamma_i^{[k]}) \\ &\qquad s.t. \begin{bmatrix} I & -B_{T-t_0} \end{bmatrix} \begin{bmatrix} x_i \\ u_i \end{bmatrix} = \mathbb{1} x_{i0} \\ &= \arg\min_{x_i, u_i} \Big(J_i(t_0, T, x_i, u_i) + \gamma_i^{[k]} e_{T,T}^T x_i + (\lambda_i^{[k]} - \beta_i^{[k]})^T x_i \Big) \\ &\qquad s.t. \begin{bmatrix} I & -B_{T-t_0} \end{bmatrix} \begin{bmatrix} x_i \\ u_i \end{bmatrix} = \mathbb{1} x_{i0}; \end{aligned} \tag{16}$$

Preprint submitted to 57th Israel Annual Conference on Aerospace Sciences.
Received September 13, 2016.

In the above QP, the constraint is the linear matrix representation of the dynamic constraints for each agent. The minimization is only over the functions $\tilde{\mathcal{L}}_i$ due to (15). Note that the minimization above is an equality constrained quadratic program. We refer to this sub-problem solved by each agent as QP_i. The notation $\overline{x}_i^{[k+1]}$ indicates the optimal solution computed in the kth iteration.

The next step in the algorithm is to propagate the multipliers in the direction of the positive gradient of the Lagrangian function with respect to each multiplier. Thus,

$$\begin{aligned}\mu^{[k+1]} &= \mu^{[k]} + \alpha_\mu \nabla_\mu \mathcal{L}(\mathbf{x}, \mathbf{u}, \delta, \zeta, \mu) \\ &= \mu^{[k]} + \alpha_\mu E^T \begin{bmatrix} e_{T,T}^T \overline{x}_i^{[k+1]} \\ \vdots \\ e_{T,T}^T \overline{x}_n^{[k+1]} \end{bmatrix} = \mu^{[k]} + \alpha_\mu E^T \overline{\mathbf{x}}^{[k+1]}(T), \end{aligned} \tag{17}$$

where we slightly abuse notation above. From the transformation in (10), the multiplier update can be expressed equivalently for the nodes as

$$E\mu^{[k+1]} = \gamma^{[k+1]} = \gamma^{[k]} + \alpha_\mu E E^T \overline{\mathbf{x}}^{[k+1]}(T). \tag{18}$$

Observe that EE^T is the graph Laplacian matrix [1], and the multiplier update can be achieved distributedly by exchanging the values $e_{T,T}\overline{x}_i^{[k+1]}$ to neighbors over the network.

The multiplier update for δ and ζ are similar, but because of the non-negativity constraint we use a projected update,

$$\begin{aligned}\delta^{[k+1]} &= \max\left\{\delta^{[k]} + \alpha_\delta \nabla_\delta \mathcal{L}(\mathbf{x}, \mathbf{u}, \delta, \zeta, \mu), 0\right\} \\ &= \max\left\{\delta^{[k]} + \alpha_\delta \left(\overline{E}^T \overline{\mathbf{x}}^{[k+1]} - R\mathbb{1}\right), 0\right\} \qquad (19) \\ \zeta^{[k+1]} &= \max\left\{\zeta^{[k]} + \alpha_\zeta \nabla_\delta \mathcal{L}(\mathbf{x}, \mathbf{u}, \delta, \zeta, \mu), 0\right\} \\ &= \max\left\{\zeta^{[k]} + \alpha_\zeta \left(-\overline{E}^T \overline{\mathbf{x}}^{[k+1]} - R\mathbb{1}\right), 0\right\} \qquad (20)\end{aligned}$$

For these updates, multiplication by $\overline{E}$ does not immediately lead to a distributed computation because of the projection operator. However, the update is still only based on relative quantities between each agent and their neighbors, so a distributed implementation can still be possible (by, for example, having each agent keep track of the edge multipliers for all edges it is incident to). After the edge multiplier update, it can be converted to nodes using $\overline{E}$,

$$\lambda^{[k+1]} = \overline{E}\delta^{[k+1]}, \quad \beta^{[k+1]} = \overline{E}\zeta^{[k+1]},$$

to be used in the next step of the algorithm. Observe that $\lambda^{[k+1]}$ and $\beta^{[k+1]}$ do not need to be non-negative.

Note the choice of the step-sizes for the updates, $\alpha_\mu, \alpha_\delta, \alpha_\zeta$ are also important for the convergence of the algorithm. The choice of step-size is beyond the scope of this work, but can be chosen using standard rules [5]. For this work, we assume constant step-sizes.

Note that this algorithm is an *asymptotic* algorithm, and with the correct choice of step-size it converges to the optimal solution of (3),

$$\lim_{k\to\infty} (\overline{\mathbf{x}}^{[k]}, \overline{\mathbf{u}}^{[k]}, \gamma^{[k]}, \beta^{[k]}, \lambda^{[k]}) = (\bar{\mathbf{x}}^{(t_0,x_0)}, \bar{\mathbf{u}}^{(t_0,x_0)}, E\bar{\mu}^{(t_0,x_0)}, \overline{E}\bar{\zeta}^{(t_0,x_0)}, \overline{E}\bar{\delta}^{(t_0,x_0)}).$$

Preprint submitted to 57th Israel Annual Conference on Aerospace Sciences.
Received September 13, 2016.

As seen above, the sub-gradient method has a significant advantages due to the fact that it can run distributedly among the agents. Its disadvantage is its asymptotic behaviour. If we wish to use the the dual sub-gradient algorithm, we need to run the algorithm infinity iteration to get the optimal trajectory of each agent.

We can see that in order us to get the optimal results, this algorithm must be run off-line, before the agents started their trajectory, and infinity iterations, it will promise convergence but not at finite time.

B. A Finite-Time Distributed Algorithm

As we explored the dual decomposition sub-gradient algorithm, we saw that it can achieve good results. However, the algorithm must be run "off-line" before the agents are released to do their tasks. If we want good accuracy for the trajectories, we need to run the algorithm for potentially many iterations.

The primary goal of this work is to find an on-line finite-time algorithm for solving (3). We also want to have the iteration step of the algorithm correspond to real-time. So after each step in the algorithm, the agents should already propagate their physical state forward in a direction they think is optimal. After T steps the algorithm will terminate.

The algorithm we develop here is based on the *shrinking horizon preference agreement* (SHPA) algorithm originally developed in [3]. In this section we present a modification of the SHPA algorithm to incorporate the distance constraints of (3). We solve this problem for the 2-agent case.

1. The SHPA Algorithm

In this subsection we will introduce the SHPA algorithm as described in [3], with the addition of the distance constraints. The algorithm is stated in Algorithm 1. The main differences between this algorithm and the dual sub-gradient algorithm is it is a finite time algorithm, and at each step of the algorithm, agent i solves a quadratic program corresponding to an optimal control problem where the initial time changes. In this way, each agent has its physical state, $x_i(t)$, and the multiplier values $\mu(t), \delta(t)$, and $\zeta(t)$ at time t. Thus, the QP_i is

$$\left[\begin{array}{cc} \overline{x}_i^{[t+1]} & \overline{u}_i^{[t+1]} \end{array}\right] = \arg\min_{x_i,u_i} J_i(t,T,x_i,u_i) + \gamma_i(t) e_{\tau,\tau}^T x_i + (\lambda_i(t) - \beta_i(t))^T x_i \tag{21}$$

$$s.t. \left[\begin{array}{cc} I & -B_\tau \end{array}\right] \left[\begin{array}{c} x_i \\ u_i \end{array}\right] = \mathbb{1} x_i(t),$$

The horizon is defined as $\tau = T - t$. Note that at each iteration, the horizon effectively "shrinks" (the length of the trajectory) and the initial condition (the physical state $x_i(t)$) corresponds to the propagated state from the previous iteration. The solution of the QP_i is then used to physically propagate that agent forward using the optimal control,

$$x_i(t+1) = x_i(t) + e_{1,\tau}^T \overline{u}_i^{[t+1]},$$

and the multipliers are updated according to the sub-gradient as in the dual - decomposition algorithm.

We now examine the performance of Algorithm 1 compared to the dual - decomposition sub-gradient algorithm and the optimal solution of (3).

Preprint submitted to 57th Israel Annual Conference on Aerospace Sciences.
Received September 13, 2016.

Algorithm 1: SHPA Algorithm

Data: Initial conditions : $x_i(t_0) = x_{i0}$, $\mu(t_0) = \mu_0$, $\zeta(t_0) = \zeta_0$ and $\delta(t_0) = \delta_0$, Parameters $\alpha_\delta, \alpha_\zeta, \alpha_\mu, R$ given.

begin

for $t := t_0$ **to** $T - 1$ **do**

$\tau = T - t,\ \gamma(t) = E\mu(t) \in \mathbb{R}^n,\ \beta(t) = \overline{E}\zeta(t) \in \mathbb{R}^{n\tau},\ \lambda(t) = \overline{E}\delta(t) \in \mathbb{R}^{n\tau}$

for $i := 1$ **to** n **do**

Each agent solved its own $QP_i(t)$

$$[\overline{x}_i\ \overline{u}_i] = \arg\min_{x_i, u_i}(J_i(t, T, x_i, u_i) + \gamma_i(t)e_{\tau,\tau}^T x_i + (\lambda_i(t) - \beta_i(t))^T x_i$$
$$\text{s.t } x_i(t+1) = x_i(t) + e_{1,\tau}^T \overline{u}_i$$

After solving the $QP_i(t)$ the multipliers will be updated :

$$\begin{aligned}
\mu(t+1) &= \mu(t) + \alpha_\mu E^T (I_n \otimes e_{\tau,\tau})^T \overline{\mathbf{x}} \\
\delta(t+1) &= \max\left\{\delta(t) + \alpha_\delta \left[\overline{E}^T \overline{\mathbf{x}} - R\mathbb{1}_{(n-1)\tau}\right], 0\right\} \\
\zeta(t+1) &= \max\left\{\zeta(t) + \alpha_\zeta \left[-\overline{E}^T \overline{\mathbf{x}} - R\mathbb{1}_{(n-1)\tau}\right], 0\right\}
\end{aligned}$$

Theorem 2. *Let $\overline{\mu}, \overline{\delta}, \overline{\zeta}$ denote the optimal Lagrange multipliers associated with* (3)*. If Algorithm 1 is initialized with these multipliers, then the trajectories generated by the algorithm are the optimal trajectories of* (3)*.*

Proof. This is a direct consequence of the statement of Lemma 1. □

We now demonstrate Theorem 2 with a numerical example. We consider $n = 7$ agents, and a terminal time of $T = 10$ seconds. The graph shown in Figure 3 along with the following preferences, weights, and initial conditions,

$$r = \begin{bmatrix} 4 \\ 5 \\ 2 \\ 6 \\ 6 \\ 4 \\ 9 \end{bmatrix}, \quad q = \begin{bmatrix} 4 \\ 5 \\ 3 \\ 8 \\ 3 \\ 6 \\ 6 \end{bmatrix}, \quad \xi = \begin{bmatrix} -37 \\ -44 \\ 44 \\ -48 \\ -14 \\ 42 \\ 28 \end{bmatrix}, \ x_0 = \begin{bmatrix} -10.1751 \\ 38.9336 \\ -20.8393 \\ -26.1990 \\ 26.2277 \\ -4.9053 \\ 4.4257 \end{bmatrix}.$$

The step-size for the multiplier updates are $\alpha_\mu = 2.2$, $\alpha_\zeta = 1.2$, $\alpha_\delta = 1.2$, and$R = 35$. We want to test the behavior of the agents when initialized with the optimal multiplier values, and compare that to a case when the multipliers are initialized to zero. In the following figures, solid lines denote trajectories generated by the algorithm, and dashed lines are the optimal trajectories of the OCP. In Figure 4 it can be seen that the algorithm matches the optimal trajectories of the OCP, validating the results of Theorem 2. However, when the multipliers are initialized to 0, we are not able to generate the optimal trajectories, as shown in Figure 5. Most noticeable is that the distance constraints are

Preprint submitted to 57th Israel Annual Conference on Aerospace Sciences.
Received September 13, 2016.

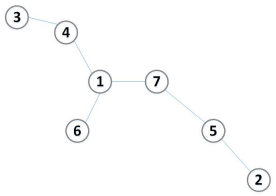
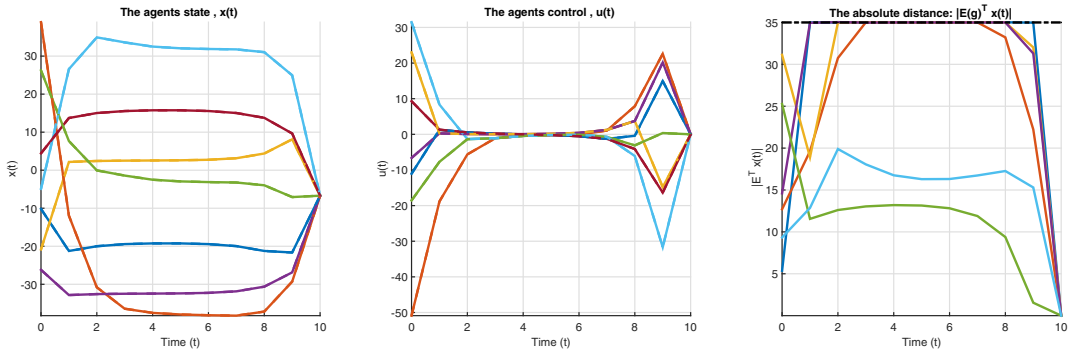
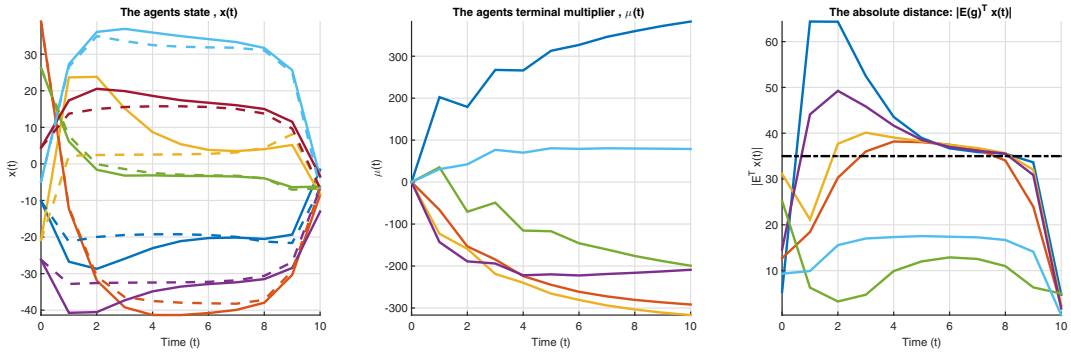


Figure 3. A spanning tree on 7 nodes.



(a) Trajectories of $\mathbf{x}(t)$. (b) Trajectories of $\overline{\mathbf{u}}$. (c) Plot of constraint $|E^T x(t)|$.

Figure 4. Trajectories generated by Algorithm 1 when initialized with the optimal multiplier values.



(a) Trajectories of $\mathbf{x}(t)$. (b) Trajectories of $\mu(t)$. (c) Plot of constraint $|E^T x(t)|$.

Figure 5. Trajectories generated by Algorithm 1 when multipliers initialized with zero.

violated by the solution trajectories, shown in Figure 5(c). In fact, we see that already in the first step ($t = 1$) the distance constraint is violated.

We now aim to correct this flaw in the algorithm by attempting to control the multiplier associated with the distance constraint in the algorithm. In fact, because the quadratic program each agent must solve has an analytic solution, we can find an explicit condition on the initial conditions for the multipliers that will ensure the distance constraint is not violated. We present the solution here for the 2 agent case ($n = 2$).

Theorem 3. *Assume that the initial conditions, $x_i(0)$ of each agent ($i = 1, 2$) satisfies $|E^T x(0)| \leq R\mathbb{1}$. Then there exists initial values for the multipliers $\lambda(0), \beta(0)$ and $\gamma(0)$*

Preprint submitted to 57th Israel Annual Conference on Aerospace Sciences.
Received September 13, 2016.

that guarantees that $|E^T x(1)| \leq R\mathbb{1}$.

Proof. In the first iteration of the SHPA algorithm, corresponding to $t = t_0 = 0$, each agent solves an equality constrained quadratic program, and therefore admits an analytic solution. The QP expressed in standard form can be given as

$$\begin{array}{ll} \min\limits_{z} & \frac{1}{2} z_i^T H_i z_i + f_i^T z_i \\ \text{s.t.} & A_{eq} z_i = b_{eqi}; \end{array}$$

Here, $z_i = [x_i^T \; u_i^T]^T$, $H_i = \mathbf{diag}(q_i I_T, r_i I_T)$, and

$$f_i \;=\; \underbrace{\begin{bmatrix} -q_i\xi_i \\ \vdots \\ -q_i\xi_i \\ 0 \\ \vdots \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \vdots \\ \gamma_i(t_0) \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{f_{1,i}} + \underbrace{\begin{bmatrix} \lambda_i(t_0+1) - \beta_i(t_0+1) \\ \vdots \\ \lambda_i(T-t_0) - \beta_i(T-t_0) \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{f_{2,i}}. \tag{22}$$

The analytic expression for the solution of the QP can be given as [15]

$$\bar{z}_i = -H_i^{-1} f_i + H_i^{-1} A_{eq}^T (A_{eq} H_i^{-1} A_{eq}^T)^{-1} (A_{eq} H_i^{-1} f_i + b_{eqi}),$$

and thus $\bar{u}_i(t_0) = e_{T+1,2T}^T \bar{z}_i$. The complete expression is described as,

$$\bar{u}_i \;=\; \underbrace{-e_{T+1,2T}^T H_i^{-1} f_i}_{=0} + e_{T+1,2T}^T \underbrace{H_i^{-1} A_{eq}^T (A_{eq} H_i^{-1} A_{eq}^T)^{-1}}_{W_i} (A_{eq} H^{-1} f_i + b_{eqi}).$$

Using the expression (22) for f_i, we further have

$$\bar{u}_i \;=\; \underbrace{e_{T+1,2T}^T W_i A_{eq} H_i^{-1}}_{v_i^T} f_{2,i} + \underbrace{e_{T+1,2T}^T W_i A_{eq} H_i^{-1} f_{1,i} + e_{T+1,2T}^T W_i b_{eqi}}_{C_i}.$$

Note that the vector v_i and scalar C_i associated with each agent can be computed *offline* and are functions of the problem data only.

The constraint that we wish to satisfy between the agents at the first step $t = 1$ is $|x_1(1) - x_2(1)| \leq R$. Using the analytic expression for the control, we obtain

$$-R \leq \underbrace{(x_1(0) + C_1)}_{\tilde{C}_1} + v_1^T f_{2,1} - v_2^T f_{2,2} - \underbrace{(x_2(0) + C_2)}_{\tilde{C}_2} \leq R. \tag{23}$$

Note also that due to the properties of the incidence matrix E, we have that $\mathbb{1}^T \beta = \mathbb{1}^T \overline{E} \zeta = 0$, and thus, $f_{2,1} + f_{2,2} = 0$.

As can be seen, the quantity $v_i^T f_{2,i}$ can be used to enforce the distance constraint. While the vector $f_{2,i}$ holds the multiplier values λ_i and β_i for the entire time horizon, we will eliminate these many degrees of freedom by projecting λ_i and β_i onto the all ones vector $\mathbb{1}$, allowing us to focus on choosing only one parameter for controlling the distance constraint. In this direction, we denote the projection of $\lambda_i - \beta_i$ onto the all-ones vector

Preprint submitted to 57th Israel Annual Conference on Aerospace Sciences.
Received September 13, 2016.

as $S_i\mathbb{1}$, with $S_i \in \mathbb{R}$, and we now focus on choosing the scalar variables S_i to ensure the distance constraint is not violated.

In this direction, let $M_i = v_i^T[\mathbb{1}^T \ \mathbf{0}^T]^T$, then the inequality in (23) can be expressed as

$$-R + \tilde{C}_2 - \tilde{C}_1 \leq \begin{bmatrix} M_1 & -M_2 \end{bmatrix} \begin{bmatrix} S_1 \\ S_2 \end{bmatrix} \leq R + \tilde{C}_2 - \tilde{C}_1.$$

From the constraint $f_{2,1} + f_{2,2} = 0$, we additionally require that $S_1 + S_2 = 0$. Thus, the set

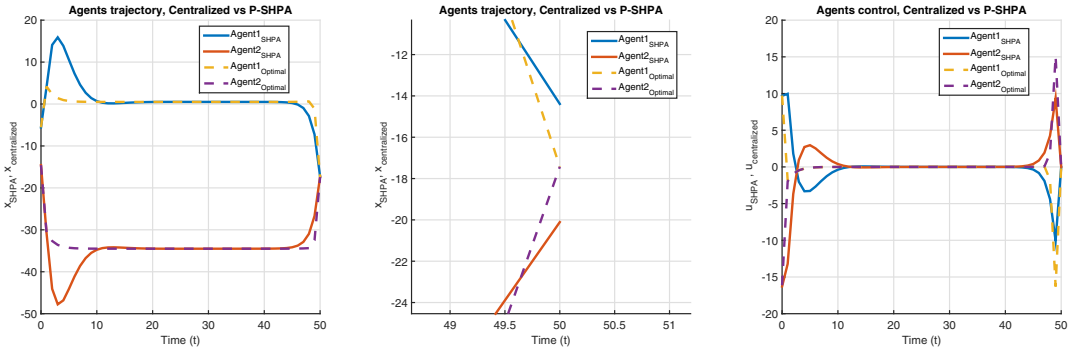
$$\Omega = \left\{ (S_1, S_2) \in \mathbb{R}^2 \,|\, S_1 + S_2 = 0,\ -R + \tilde{C}_2 - \tilde{C}_1 \leq \begin{bmatrix} M_1 & -M_2 \end{bmatrix} \begin{bmatrix} S_1 \\ S_2 \end{bmatrix} \leq R + \tilde{C}_2 - \tilde{C}_1 \right\},$$

defining a line-segment in $\mathbb{R}^2$, characterizes all initial conditions for the multipliers of the form $\lambda_i - \beta_i = S_i\mathbb{1}$ guarantees that (23) is satisfied. Thus, λ_i and β_i can be chosen in any way so long as they satisfy the previous equation. Finally, we note that the choice of the initial condition for the multiplier γ is in fact arbitrary, since the set Ω is computed based on any choice of γ. The edge multipliers can then be reconstructed by the relation (13). □

To demonstrate the results of Theorem 3, we run a numerical example with two agents. The parameters of the agents are given below

$$r = \begin{bmatrix} 9 \\ 10 \end{bmatrix}, \quad q = \begin{bmatrix} 5 \\ 5 \end{bmatrix}, \quad \xi = \begin{bmatrix} 25 \\ -59 \end{bmatrix}, \ x_0 = \begin{bmatrix} -5.5628 \\ -14.5777 \end{bmatrix}.$$

The remaining parameters are given as $R = 35$, $T = 50$, and $\alpha_\mu = 1$, $\alpha_\zeta = 1$, $\alpha_\delta = 1$. Using the results of Theorem 3, we chose the values $\delta(t_0) = 59.1904\mathbb{1}$ and $\zeta(t_0) = \mathbf{0}$ (equivalently, $S_1(t_0) = 59.1904$, $S_2(t_0) = -59.1904$). As can be seen in Figures 6(a) and 6(c), the algorithm does not produce optimal trajectories. However, as shown in Figure 7, in the first step of the algorithm, the distance constrain is not violated. After updating the multipliers according to Algorithm 1, we still observe a violation of the distance constraint in subsequent steps.



(a) Trajectories of $\mathbf{x}(\mathbf{t})$. (b) Trajectories of $\mathbf{x}(\mathbf{t})$ zoomed into final time. (c) Trajectories of $\mathbf{u}(\mathbf{t})$.

Figure 6. Trajectories generated by Algorithm 1 when multipliers initialized according to Theorem 3.

Preprint submitted to 57th Israel Annual Conference on Aerospace Sciences.
Received September 13, 2016.

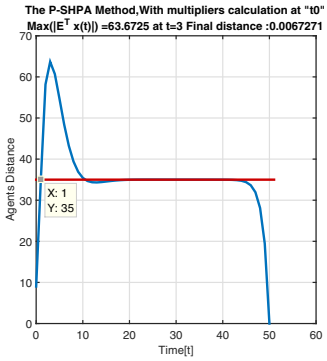


Figure 7. A plot of $\max(|E^T\mathbf{x}(t)|)$ generated by Algorithm 1 when multipliers initialized according to Theorem 3.

In the next subsection, we use the results of Theorem 3 to propose a modification to the SHPA algorithm that ensures that at each time step, the distance constraint is not violated.

2. The Projected SHPA Algorithm

The SHPA algorithm in Algorithm 1 is based on the dual-decomposition sub-gradient algorithm. The update of the multipliers corresponding to the distance constraints had to be projected onto the non-negative orthant. In Theorem 3, we saw that in the first step, we could determine a good set of multipliers by picking values from a special set. The main idea of the Projected SHPA algorithm (P-SHPA) is to repeat the process presented in Theorem 3 at every step of the algorithm.

The P-SHPA algorithm is presented in Algorithm 2. In the first step of the algorithm, we determine initial values for the multipliers according to Theorem 3. At each step in the algorithm, the set Ω is computed as in the Theorem. The main difference here is now this set also changes in time; this reflects the "shrinking horizon" aspect of the algorithm. To reflect this, we denote the set as a function of time, $\Omega(t)$. A point not explicitly mentioned in the algorithm is *which* point to chose from the set $\Omega(t)$. For this work, we always chose one of the endpoints of the set (as it is always a line segment). In fact, this choice is arbitrary, and it is also possible to perform a projection onto this set. In this implementation, therefore, the propagation of the edge multipliers is not entirely necessary. We now present a result showing that Algorithm 2 indeed ensures that the agents do not violate the distance constraints during its evolution.

Theorem 4. *Assume that the initial conditions of the agent satisfy $|E^T x(t_0)| < R$. Then $|E^T x(t)| \leq R$, $\forall t = t_0 \dots T-1$ when $x(t)$ are the trajectories generated by the P-SHPA algorithm for any initiated values of the multipliers.*

Proof. The proof is a direct consequence of Theorem 3. □

We now examine the performance of Algorithm 2 compared to centralized solution, which is the optimal solution of (3). In this example we consider the following parameters:

$$r = \begin{bmatrix} 9 \\ 10 \end{bmatrix}, \quad q = \begin{bmatrix} 5 \\ 5 \end{bmatrix}, \quad \xi = \begin{bmatrix} 25 \\ -59 \end{bmatrix}, \; x_0 = \begin{bmatrix} -5.5628 \\ -14.5777 \end{bmatrix}.$$

Preprint submitted to 57th Israel Annual Conference on Aerospace Sciences.
Received September 13, 2016.

Algorithm 2: Projected SHPA Algorithm

Data: Initial conditions : $x_i(t_0) = x_i(0)$ and given parameters $\alpha_\delta, \alpha_\zeta, \alpha_\mu, R, T$.
Determine initial multiplier values based on Theorem 3.

begin

for $t := 0$ **to** T **do**

$\tau = T - t,\ \gamma(t) = E(\mathcal{G})\mu(t),\ \beta(t) = \overline{E}\zeta(t),\ \lambda(t) = \overline{E}\delta(t)$

Determine the set $\Omega(t)$,

$$\Omega(t) = \left\{ \begin{array}{l} (S_1, S_2) \in \mathbb{R}^2 \,|\, S_1 + S_2 = 0, \\ -R + \tilde{C}_2(t) - \tilde{C}_1(t) \leq \begin{bmatrix} M_1(t) & -M_2(t) \end{bmatrix} \begin{bmatrix} S_1 \\ S_2 \end{bmatrix} \leq R + \tilde{C}_2(t) - \tilde{C}_1(t) \end{array} \right\}$$

Each agent solves its own QP_i using S_i,

$$\begin{aligned} [\overline{x}_i\ \overline{u}_i] = \arg\min_{x_i, u_i} & (J_i(t, T, x_i, u_i) + \gamma_i(t) e_{\tau,\tau}^T x_i + S_i \mathbb{1}^T x_i \\ \text{s.t } & x_i(t+1) = x_i(t) + e_{1,\tau}^T \overline{u}_i \end{aligned}$$

Determine values for $\lambda_i(t), \beta_i(t)$ using S_i, and compute edge multipliers,
$\zeta = (\overline{E}^T\overline{E})^{-1}\overline{E}^T\beta(t)$.
$\delta = (\overline{E}^T\overline{E})^{-1}\overline{E}^T\lambda(t)$
Propagate the multipliers:
$\mu(t+1) = \mu(t) + \alpha_\mu E(\mathcal{G})^T x(T)$
$\delta(t+1) = \max\{\delta(t) + \alpha_\delta \nabla_\delta \mathcal{L}(x, u, \delta, \zeta, \mu), 0\}$
$\zeta(t+1) = \max\{\zeta(t) + \alpha_\zeta \nabla_\zeta \mathcal{L}(x, u, \delta, \zeta, \mu), 0\}$

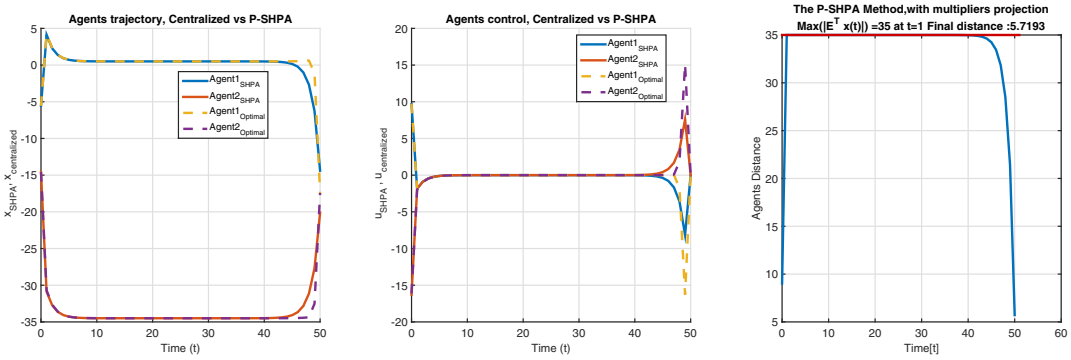
The radius is defined as $R = 35$, $T = 50$ the multipliers step values are $\alpha_\mu = 1$, $\alpha_\zeta = 1$, $\alpha_\delta = 1$. As can be seen in Figure 8, the algorithm performs quite well as compared to the optimal trajectories. Most importantly, the algorithm ensures that there are distance constraints are satisfied throughout the trajectory. As discussed, our choice of choosing multipliers from the boundaries of the set $\Omega(t)$ result in a trajectory that keeps the agents near the maximal allowable distance.

Figure 9 shows the evolution of the multiplier values. As expected by the algorithm description, the multipliers take some sporadic values to ensure the satisfaction of the distance constraints.

IV. Conclusions

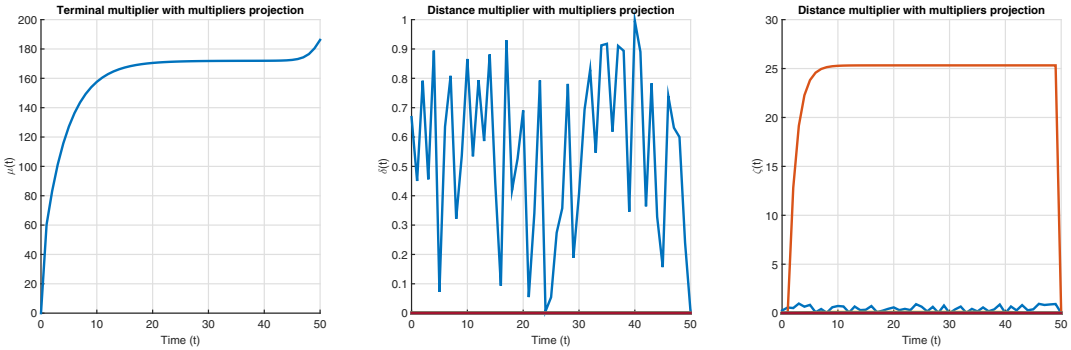
There is no doubt that the field of multi-agent system will be studied in more depth in the coming years. Most of the research dealing with multi-agent system is based on Graph Theory. This research can provide a steadier environment for the study; due to the radius keeping constraint we can know with certainty that no agent will disappear from the connectivity graph. Additionally, we know that no agent will lose its communication with the adjacent agents. This important property means that the connectivity graph staying the same throughout the trajectory is primarily an assumption, but this research can cause it to happen. From the manager of the multi-agent system point of view, it has significance. No lost agents, no graph damage, means a stable multi-agents network. For

Preprint submitted to 57th Israel Annual Conference on Aerospace Sciences.
Received September 13, 2016.



(a) Trajectories of $\mathbf{x}(\mathbf{t})$. (b) Trajectories of $\mathbf{u}(\mathbf{t})$. (c) A plot of $\max(|E^T\mathbf{x}(\mathbf{t})|)$.

Figure 8. Trajectories generated by the P-SHPA algorithm compared against the optimal solution.



(a) Trajectories of $\mu(\mathbf{t})$. (b) Trajectories of $\delta(\mathbf{t})$. (c) Trajectories of $\zeta(\mathbf{t})$.

Figure 9. Trajectories of the multipliers generated by the P-SHPA algorithm.

example, if we use a multi-agent system, with the connection topology of a star, every agent is connected to the central agent. There may be a possibility that the central agent will lose its connectivity to the rest of the group. This will have an immediate impact on the rest of the agents. Most of the agents will probably not know what to do once they become lost. In this specific case, if there is no reconnecting algorithm to join the lost agents, they are lost. The primary achievement of this research is that it will not harm the connectivity graph of the multi-agent system. The secondary achievement is that it is all done on-line, distributed, at finite time, meaning that every agent can work as an individual and still be connected to the rest of the group.

References

1. M. Mesbahi and M. Egerstedt *Graph theoretic methods in multiagent networks*, in Princeton University Press, 2010.

2. L. R. Foulds *Graph theory with applications*, in Springer Science and Business Media, 2012.

3. D. Zelazo, M. Bürger and F. Allgöwer, *A Finite-Time Dual Method for Negotiation*

Preprint submitted to 57th Israel Annual Conference on Aerospace Sciences.
Received September 13, 2016.

between Dynamical Systems, in SIAM Journal on Control and Optimization 51.1 (2013): 172-194.

4. B. Liu, T. Chu, L. Wang and G. Xie, *Controllability of a leader–follower dynamic network with switching topology*. IEEE Transactions on Automatic Control, 53.4 (2008): 1009-1013.

5. A. Ruszczynski , *Nonlinear Optimization*. Princeton University Press, Princeton, NJ, 2006.

6. R. Olfati-Saber, *Flocking for multi-agent dynamic systems: Algorithms and theory*. IEEE Transactions on Automatic Control, 51.3 (2006): 401-420.

7. D. P. Spanos and R. M. Murray, *Motion planning with wireless network constraints*, in Proceedings of the American Control Conference. Vol. 1. 2005.

8. S. P. Bhat and D. S. Bernstein, *Finite-time stability of continuous autonomous systems*. SIAM Journal on Control and Optimization 38.3 (2000): 751-766.

9. T. Li, M. Fu, L. Xie and J. F. Zhang, *Distributed consensus with limited communication data rate*. IEEE Transactions on Automatic Control, 56.2 (2011): 279-292.

10. M. Franceschelli, M. Egerstedt,A. Giua and C. Mahulea, *Constrained invariant motions for networked multi-agent systems*. American Control Conference, 2009. ACC'09. IEEE, 2009.

11. R. Olfati-Saber,J. A. Fax and R. M. Murray, *Consensus and cooperation in networked multi-agent systems*,in Proceedings of the IEEE 95.1 (2007): 215-233.

12. W. Wang and M. Zhao, *Joint effects of radio channels and node mobility on link dynamics in wireless networks* , in INFOCOM 2008. The 27th Conference on Computer Communications. IEEE (pp. 933-941).

13. C. D. Godsil and G. Royle , *Algebraic graph theory*, Springer , New York, 2001.

14. R. A. Horn and C. R. Johnson, *Topics in matrix analysis*. Cambridge university press, 1991.

15. S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

16. L. Wang and F. Xiao, *Finite-Time Consensus Problems for Networks of Dynamic Agents*. -, 2008.

17. R. Saber and R. M. Murray *Consensus Protocols for Networks of Dynamic Agents*. Control and Dynamical Systems California Institute of Technology Pasadena, 2003.

18. T. D. Tran and A. Y. Kibangou *Distributed Design of Finite-time Average Consensus Protocols*. Gipsa-Lab, CNRS, University Joseph Fourier, 2013.

19. Y. Zhu , X. Guan1 and X. Luo *Finite-time Consensus for Multi-agent Systems via Nonlinear Control Protocols*. Journal of Automation and Computing, 2013.

20. S. E. Parsegov , A. E. Polyakov and P. S. Shcherbakov *Finite-time Consensus for Multi-agent Systems via Nonlinear Control Protocols*. 4th IFAC Workshop on Distributed Estimation and Control in Networked Systems , 2013.

21. S. Kar and J. M. F. Moura *Consensus Algorithms in Sensor Networks With Imperfect Communication: Link Failures and Channel Noise*.IEEE Transactions on signal processing, VOL. 57, NO. 1, 2009

22. Z. Xue, , Z. Liu, C. Feng and J. Zeng *Target Tracking for Multi-agent Systems with Leader-Following Strategy*. Guangzhou, P. R. China, 29-31,July 2010, pp. 240-243

Preprint submitted to 57th Israel Annual Conference on Aerospace Sciences.
Received September 13, 2016.

23. G. Shi, Y. Hong, and K. H. Johansson *Connectivity and Set Tracking of Multi-Agent Systems Guided by Multiple Moving Leaders.*IEEE Transactions on automatic control , Vol. 57, no. 3, March 2012

24. G. Yang, Q. Yang, V. Kapila, D. Palmer and R. Vaidyanathan *Fuel optimal manoeuvres for multiple spacecraft formation reconfiguration using multi-agent optimization.*DOI: 10.1002/rnc.684 , 12 FEB 2002

25. Y. Kim, M. Mesbahi, F. Y. Hadaegh *Multiple-Spacecraft Reconfiguration Through Collision Avoidance, Bouncing, and Stalemate.* Journal of Optimization Theory and Applications, August 2004, Volume 122, Issue 2, pp 323-343

Preprint submitted to 57th Israel Annual Conference on Aerospace Sciences.
Received September 13, 2016.