Distributed Nash Equilibrium Seeking via the Alternating Direction Method of Multipliers

Farzad Salehisadaghiani * **Lacra Pavel** *

* *Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON M5S 3G4, Canada (e-mails:* farzad.salehisadaghiani@mail.utoronto.ca, pavel@ece.utoronto.ca*).*

Abstract: In this paper, the problem of finding a Nash equilibrium of a multi-player game is considered. The players are only aware of their own cost functions as well as the action space of all players. We develop a relatively fast algorithm within the framework of inexact-ADMM. It requires a communication graph for the information exchange between the players as well as a few mild assumptions on cost functions. The convergence proof of the algorithm to a Nash equilibrium of the game is then provided. Moreover, the convergence rate is investigated via simulations.

Keywords: Nash games, Game theory, Distributed control,

1. INTRODUCTION

There is a close connection between the problem of finding a Nash equilibrium (NE) of a distributed game and a distributed optimization problem. In a distributed optimization problem with N agents that communicate over a connected graph, it is desired to minimize a global objective as follows:

$$\begin{cases} \underset{x}{\text{minimize}} & f(x) := \sum_{i=1}^{N} f_i(x) \\ \text{subject to} & x \in \Omega. \end{cases} \tag{1}$$

In this problem the agents cooperatively solve (1) over a common optimization variable x. In other words, all the agents are serving in the public interest in a way that they reduce the global loss. However, there are many real-world applications that involve selfishness of the players (agents) such as congestion control for Ad-hoc wireless networks and optical signal-to-noise ratio (OSNR) maximization in an optical network. In these applications, players *selfishly* desire to optimize their own performance even though the global objective may not be minimized, hence play a game. In this regard, we are interested in studying the (Nash) equilibrium of this game.

Considering the difference between distributed optimization and distributed Nash equilibrium (NE) seeking, we aim to employ an optimization technique referred to as *the alternating direction method of multipliers* (ADMM) to find an equilibrium point of a multi-player game. ADMM takes advantage of two different approaches used in solving optimization problems: 1) Dual Decomposition, and 2) Augmented Lagrangian Methods.

Dual decomposition is a special case of a dual ascent method for solving an optimization problem when the objective function is separable w.r.t. variable x, i.e., $f(x) := \sum_{i=1}^{N} f_i(x_i)$ where $x = [x_1, \ldots, x_N]^T$. This decomposition leads to N parallel dual ascent problems whereby each is to be solved for x_i, $i \in \{1, \ldots, N\}$. This parallelism makes the convergence faster.

The augmented Lagrangian method is more robust and relaxes the assumptions in the dual ascent method. This method involves a penalty term added to the normal Lagrangian.

In this work, we aim to exploit the benefits of ADMM in the context of finding an NE of a game. Here are the difficulties that we need to overcome:

- A Nash game can be seen as a set of parallel optimization problems, each of them associated with the minimization of a player's own cost function w.r.t. his variable. However, each optimization problem is dependent on the solution of the other parallel problems. This leads to have N Lagrangians whereby each is dependent on the other players' variables.
- Each player i updates only his own variable x_i, however, he requires also an estimate of all other variables $(x_j)_{j \in \{1, \ldots, N\}, j \neq i}$ and updates it in order to solve his optimization problem. This demands an extra step in the algorithm based on communications between players.
- Each optimization problem is not in the proper format of sum of separable functions to allow direct application of ADMM.

Related Works. Our work is related to the literature on distributed Nash games such as Yin et al. (2011); Alpcan and Başar (2005) and distributed optimization problems such as Nedic (2011); Johansson (2008). Finding NE in distributed games has recently drawn attention due to many real-world applications. To name only a few, Salehisadaghiani and Pavel (2014, 2016a); Frihauf et al. (2012); Gharesifard and Cortes (2013); Salehisadaghiani and Pavel (2016b); Pavel (2007); Pan and Pavel (2009). In Koshal et al. (2012) an algorithm has been designed based on

gossiping protocol to compute an NE in aggregate games. Zhu and Frazzoli (2016) study the problem of finding an NE in more general games by a gradient-based method over a complete communication graph. This problem is extended to the case with partially coupled cost functions (the functions which are not necessarily dependent on all the players' actions) in Bramoullé et al. (2014). Recently, Ye and Hu (2015) investigate distributed seeking of a time-varying NE with non-model-based costs for players. Computation of a time-varying NE is considered in Lou et al. (2016) in networked games consisting of two subnetworks with shared objectives. Parise et al. (2015) propose two different algorithms to solve for an NE in a large population aggregative game which is subject to heterogeneous convex constraints.

ADMM algorithms, which are in the scope of this paper, have been developed in 1970s to find an optimal point of distributed optimization problems. This method has become widely used after its re-introduction in Boyd et al. (2011) such as He and Yuan (2012); Goldstein et al. (2014); Wei and Ozdaglar (2012). Shi et al. (2014) investigate the linear convergence rate of an ADMM algorithm to solve a distributed optimization problem. ADMM algorithms are extended by Makhdoumi and Ozdaglar (2014) to the case when agents broadcast their outcomes to their neighbors. The problem of distributed consensus optimization is considered in Chang et al. (2015) which exploits *inexact-*ADMM to reduce the computational costs of a classical ADMM. Recently, an ADMM-like algorithm is proposed by Shi and Pavel (submitted) in order to find an NE of a game. It is shown that the algorithm converges faster than the gradient-based methods. However, the algorithm requires individual cocoercivity and is not developed, but rather postulated by mimicking of ADMM in distributed optimization according to the NE condition.

Contributions. In this paper, first, we reformulate the problem of finding an NE of a convex game as a set of distributed consensus optimization problems. Then we take advantage of a dummy variable to make the problem separable in the optimization variable. This technique can be used for any convex game which satisfies a set of relatively mild assumptions.

Second, we design a synchronous inexact-ADMM algorithm by which every player updates his action as well as his estimates of the other players' actions. This algorithm takes advantage of the speed and robustness of the classical ADMM and reduces the computational costs by using a linear approximation in players' action update rule (inexact-ADMM). Compared with gradient-based algorithms such as Koshal et al. (2012); Zhu and Frazzoli (2016); Li and Marden (2013), our ADMM algorithm has an extra penalty term (could be seen as an extra state) which is updated through the iterations and improves the convergence rate.

Third, we prove the convergence of the proposed algorithm toward the NE of the game and compare its convergence rate with a gradient-based method via simulation.

The paper is organized as follows. The problem statement and assumptions are provided in Section 2. In Section 3, an inexact-ADMM-like algorithm is proposed. Convergence of the algorithm to a Nash equilibrium of the game is

discussed in Section 4 while in Section 5 a simplified representation of the algorithm for implementation is provided. Simulation results are given in Section 6 and conclusions in Section 7.

2. PROBLEM STATEMENT

Consider $V = \{1, \ldots, N\}$ as a set of N players in a networked multi-player game. The game is denoted by $\mathcal{G}$ and defined as follows:

- $\Omega_i \subset \mathbb{R}$: Action set of player i, $\forall i \in V$,
- $\Omega = \prod_{i \in V} \Omega_i \subset \mathbb{R}^N$: Action set of all players,
- $J_i : \Omega \to \mathbb{R}$: Cost function of player i, $\forall i \in V$.

The game $\mathcal{G}(V, \Omega_i, J_i)$ is defined over the set of players, V, the action set of player $i \in V$, Ω_i and the cost function of player $i \in V$, J_i.

The players' actions are denoted as follows:

- $x = (x_i, x_{-i}) \in \Omega$: All players actions,
- $x_i \in \Omega_i$: Player i's action, $\forall i \in V$,
- $x_{-i} \in \Omega_{-i} := \prod_{j \in V \setminus \{i\}} \Omega_j$: All players' actions except player i's.

The game is played such that for a given $x_{-i} \in \Omega_{-i}$, each player $i \in V$ aims to minimize his own cost function selfishly w.r.t. x_i to find an optimal action,

$$\begin{cases} \underset{x_i}{\text{minimize}} & J_i(x_i, x_{-i}) \\ \text{subject to} & x_i \in \Omega_i. \end{cases} \quad (2)$$

Each optimization problem is run by a particular player i at the same time with other players.

An NE of a game is defined as follows:

Definition 1. Consider an N-player game $\mathcal{G}(V, \Omega_i, J_i)$, each player i minimizing the cost function $J_i : \Omega \to \mathbb{R}$. A vector $x^* = (x_i^*, x_{-i}^*) \in \Omega$ is called a Nash equilibrium of this game if

$$J_i(x_i^*, x_{-i}^*) \leq J_i(x_i, x_{-i}^*) \quad \forall x_i \in \Omega_i, \ \forall i \in V. \quad (3)$$

An NE lies at the intersection of all solutions of the set (2). The challenge is that each optimization problem in (2) is dependent on the solution of the other simultaneous problems. And since this game is distributed, no player is aware of the actions (solutions) of the other players (problems).

We assume that each player i maintains an estimate of the other players' actions. In the following, we define a few notations for players' estimates.

- $x^i = (x_i^i, x_{-i}^i) \in \Omega$: Player i's estimate of all players actions,
- $x_i^i \in \Omega_i$: Player i's estimate of his own action, $\forall i \in V$,
- $x_{-i}^i \in \Omega_{-i} := \prod_{j \in V \setminus \{i\}} \Omega_j$: Player i's estimate of all other players' actions except his action,
- $\underline{x} = [x^{1^T}, \ldots, x^{N^T}]^T \in \Omega^N$: Augmented vector of estimates of all players' actions

Note that player i's estimate of his action is indeed his action, i.e., $x_i^i = x_i$ for $i \in V$. Note also that all players actions $x = (x_i, x_{-i})$ can be interchangeably represented as $x = [x_i^i]_{i \in V}$.

We assume that the cost function J_i and the action set Ω are the only information available to player i. Thus, the players need to exchange some information in order

to update their estimates. An undirected *communication graph* $G_C(V, E)$ is defined where $E \subseteq V \times V$ denotes the set of communication links between the players. $(i,j) \in E$ if and only if players i and j exchange information. In the following, we have a few definitions for G_C:

- $N_i := \{j \in V | (i,j) \in E\}$: Set of neighbors of player i in G_C,
- $A := [a_{ij}]_{i,j \in V}$: Adjacency matrix of G_C where $a_{ij} = 1$ if $(i,j) \in E$ and $a_{ij} = 0$ otherwise,
- $D := \text{diag}\{|N_1|, \ldots, |N_N|\}$: Degree matrix of G_C.

The following assumption is used.

Assumption 1. G_C is a connected graph.

We aim to relate game (2) to the following problem whose solution can be based on the alternating direction method of multipliers (Bertsekas and Tsitsiklis (1997), page 255).

$$\begin{cases} \underset{x \in C_1, z \in C_2}{\text{minimize}} & G_1(x) + G_2(z) \\ \text{subject to} & Ax = z. \end{cases}$$

To this end, we reformulate game (2) so that the objective function is separable by employing estimates of the actions for each player $i \in V$ as x^i (the estimates are also interpreted as the local copies of x). Particularly, from (2), consider that for a given $x_{-i}^i \in \Omega_{-i}$, each player $i \in V$ minimizes his cost function selfishly w.r.t. his own action subject to an equality constraint, i.e., for all $i \in V$,

$$\begin{cases} \underset{x_i^i, x_i^j |_{j \in N_i} \in \Omega_i}{\text{minimize}} & J_i(x_i^i, x_{-i}^i) + \sum_{j \in N_i} g_j(x_i^j, x_{-i}^j) \\ \text{subject to} & x_i^i = x_i^j \quad \forall j \in N_i, \end{cases}$$

where $g_j(\cdot) = 0$ for $j \in N_i$. Note that, in order to update all elements in x_{-i}^i we need to augment the constraint space to an $N \times 1$ vector form $x^i = x^j$, $j \in N_i$. Moreover, we replace the constraints with $x^l = x^s$ $\forall l \in V, \forall s \in N_l$ which includes $x^i = x^j$, $j \in N_i$. Note that augmenting the constraints in this way does not affect the solutions of the problem. Then for a given $x_{-i}^i \in \Omega_{-i}$ and for all $i \in V$, we obtain,

$$\begin{cases} \underset{x_i^i \in \Omega_i}{\text{minimize}} & J_i(x_i^i, x_{-i}^i) \\ \text{subject to} & x^l = x^s \quad \forall l \in V, \forall s \in N_l. \end{cases} \quad (4)$$

The equality constraint along with Assumption 1 ensures that all the local copies of x are identical, i.e., $x^1 = x^2 = \ldots = x^N$. Hence (4) recovers (2).

By Assumption 1, the set of problems (4) are equivalent to the following set of optimization problems: for a given $x_{-i}^i \in \Omega_{-i}$ and for all $i \in V$,

$$\begin{cases} \underset{x_i^i \in \mathbb{R}}{\text{minimize}} & J_i(x_i^i, x_{-i}^i) + \mathcal{I}_{\Omega_i}(x_i^i) \\ \text{subject to} & x^l = t^{ls} \quad \forall l \in V, \forall s \in N_l, \\ & x^s = t^{ls} \quad \forall l \in V, \forall s \in N_l, \end{cases} \quad (5)$$

where $\mathcal{I}_{\Omega_i}(x_i^i) := \begin{cases} 0 & \text{if } x_i^i \in \Omega_i \\ \infty & \text{otherwise} \end{cases}$ is an indicator function of the feasibility constraint $x_i^i \in \Omega_i$ and t^{ls} is an intermediary variable to separate the equality constraints.

Note that one can regard the set of problems (5) as being the same as the set of problems (2) but considering N

estimates (local copies) of the players' actions for each player $i \in V$.

A characterization of the NE for game (2) could be obtained by finding KKT conditions on the set of problems (5). Let $\{u^{ls}, v^{ls}\}_{l \in V, s \in N_l}$ with $u^{ls}, v^{ls} \in \mathbb{R}^N$ be the Lagrange multipliers associated with the two constraints in (5), respectively. The corresponding Lagrange function for player i, $\forall i \in V$ is as follows:

$$L_i\left(x^i, \{t^{ls}\}_{l \in V, s \in N_l}, \{u^{ls}\}_{l \in V, s \in N_l}, \{v^{ls}\}_{l \in V, s \in N_l}\right)$$

$$:= J_i(x_i^i, x_{-i}^i) + \mathcal{I}_{\Omega_i}(x_i^i) + \sum_{l \in V} \sum_{s \in N_l} u^{ls^T}(x^l - t^{ls})$$

$$+ \sum_{l \in V} \sum_{s \in N_l} v^{ls^T}(x^s - t^{ls}), \quad (6)$$

Let $(x^{i*})_{i \in V}$ and $\{u^{ls*}, v^{ls*}\}_{l \in V, s \in N_l}$ be a pair of optimal primal and dual solutions to (5). The KKT conditions are summarized as follows:

$$\nabla_i J_i(x^{i*}) + \partial_i \mathcal{I}_{\Omega_i}(x_i^{i*}) + \sum_{j \in N_i} u^{ij*} + v^{ji*} = 0 \quad \forall i \in V, \quad (7)$$

$$x^{i*} = x^{j*} \quad \forall i \in V, \forall j \in N_i, \quad (8)$$

$$u^{ij*} + v^{ij*} = \mathbf{0}_N \quad \forall i \in V, \forall j \in N_i, \quad (9)$$

where $\nabla_i J_i(\cdot)$ is gradient of J_i w.r.t. x_i and $\partial_i \mathcal{I}_{\Omega_i}(\cdot)$ is a subgradient of $\mathcal{I}_{\Omega_i}$ at x_i. Note that the index of v^{ji*} in (7) is inverse of v^{ij*} in (9). By (8) and Assumption 1, $x^{1*} = \ldots = x^{N*} := x^*$. Thus, $x^* := (x_i^*, x_{-i}^*)$ is a solution of (5) if and only if,

$$\begin{cases} \nabla_i J_i(x^*) + \partial \mathcal{I}_{\Omega_i}(x_i^*) + \sum_{j \in N_i} u^{ij*} + v^{ji*} = 0 \quad \forall i \in V, \\ u^{ij*} + v^{ij*} = \mathbf{0}_N \quad \forall i \in V, \forall j \in N_i. \end{cases} \quad (10)$$

We state a few assumptions for the existence and the uniqueness of an NE.

Assumption 2. For every $i \in V$, the action set Ω_i is a non-empty, compact and convex subset of $\mathbb{R}$. $J_i(x_i, x_{-i})$ is a continuously differentiable function in x_i, jointly continuous in x and convex in x_i for every x_{-i}.

The convexity of Ω_i implies that $\mathcal{I}_{\Omega_i}$ is a convex function. This yields that there exists at least one bounded subgradient $\partial \mathcal{I}_{\Omega_i}$.

Assumption 3. Let $F : \Omega^N \to \mathbb{R}^N$, $F(\underline{x}) := [\nabla_i J_i(x^i)]_{i \in V}$ be the the pseudo-gradient vector (game map) where $\underline{x} := [x^{1^T}, \ldots, x^{N^T}]^T \in \Omega^N$. F is cocoercive $\forall \underline{x} \in \Omega^N$ and $\underline{y} \in \Omega^N$, i.e.,

$$\left(F(\underline{x}) - F(\underline{y})\right)^T (x - y) \geq \sigma_F \|F(\underline{x}) - F(\underline{y})\|^2, \quad (11)$$

where $\sigma_F > 0$.

Remark 1. Assumption 2 is a standard assumption in the literature of NE seeking. Assumption 3 is relatively stronger than the (strong) monotonicity of the game map (pseudo-gradient vector) (see Zhu and Frazzoli (2016); Koshal et al. (2012)). However, as we will show, this leads to an algorithm with the benefits of ADMM algorithms (speed).

Remark 2. Assumption 3 is not usually required in distributed optimization problems; there instead the (strong) convexity of the objective function is assumed to be w.r.t.

the full vector x and also the gradient of the objective function is assumed to be Lipschitz continuous (see Chang et al. (2015)).

Our objective is to find an ADMM-like[1] algorithm for computing an NE of $\mathcal{G}(V, \Omega_i, J_i)$ using only imperfect information over the communication graph $G_C(V, E)$.

3. DISTRIBUTED INEXACT ADMM ALGORITHM

We propose a distributed algorithm, using an *inexact*[2] *consensus ADMM*. We obtain an NE of $\mathcal{G}(V, \Omega_i, J_i)$ by solving the set of problems (5) by an ADMM-like approach.

The mechanism of the algorithm can be briefly explained as follows: Each player maintains an estimate of the actions of all players and locally communicates with his neighbors over G_C. Then, he takes average of his neighbors' information and uses it to update his estimates.

The algorithm is elaborated in the following steps:
1- ***Initialization Step:*** Each player $i \in V$ maintains an initial estimate for all players, $x^i(0) \in \Omega$. The initial values of $u^{ij}(0)$ and $v^{ij}(0)$ are set to be zero for all $i \in V$, $j \in N_i$.
2- ***Communication Step:*** At iteration $T(k)$, each player $i \in V$ exchanges his estimate of the other players' actions with his neighbors j, $\forall j \in N_i$. Then, he takes average of the received information with his estimate and updates his estimate as follows:

$$x^i_{-i}(k) = \frac{1}{2}\Big(x^i_{-i}(k-1) + \underbrace{\frac{1}{|N_i|}\sum_{j \in N_i} x^j_{-i}(k-1)}_{\text{RECEIVED INFORMATION}}\Big)$$
$$-\underbrace{\frac{1}{2c|N_i|}\sum_{j \in N_i}(u^{ij}_{-i}(k) + v^{ji}_{-i}(k))}_{\text{PENALTY TERM}}, \qquad (12)$$

where $c > 0$ is a scalar coefficient, and $\forall i \in V, j \in N_i$,

$$u^{ij}(k) = u^{ij}(k-1) + \frac{c}{2}\big(x^i(k-1) - x^j(k-1)\big), \quad (13)$$

$$v^{ij}(k) = v^{ij}(k-1) + \frac{c}{2}\big(x^j(k-1) - x^i(k-1)\big). \quad (14)$$

Equations (13), (14) are the dual Lagrange multipliers update rules. Note that in (12), a penalty factor $\sum_{j \in N_i}(u^{ij}_{-i}(k) + v^{ji}_{-i}(k))$ is subtracted, which is associated with the difference between the estimates of the neighboring players (Equations (13), (14)).

Remark 3. Unlike distributed optimization algorithms where the minimization is w.r.t. x, here each player minimizes his cost function w.r.t. x^i_i. To update x^i_i, each player requires the estimate of the other players x^i_{-i} at each iteration. Thus, the communication step is inevitable to update x^i_{-i} for the next iteration.

3- ***Action Update Step***

At this moment all the players update their actions via an ADMM-like approach developed as follows. For each player i, $\forall i \in V$ let the augmented Lagrange function associated to problem (5) be as follows:

$$L^a_i\Big(x^i, \{t^{ls}\}_{l \in V, s \in N_l}, \{u^{ls}\}_{l \in V, s \in N_l}, \{v^{ls}\}_{l \in V, s \in N_l}\Big)$$
$$:= J_i(x^i_i, x^i_{-i}) + \mathcal{I}_{\Omega_i}(x^i_i)$$
$$+ \sum_{l \in V}\sum_{s \in N_l} u^{ls^T}(x^l - t^{ls}) + \sum_{l \in V}\sum_{s \in N_l} v^{ls^T}(x^s - t^{ls})$$
$$+ \frac{c}{2}\sum_{l \in V}\sum_{s \in N_l}(\|x^l - t^{ls}\|^2 + \|x^s - t^{ls}\|^2). \qquad (15)$$

where $c > 0$ is a scalar coefficient which is also used in (12), (13) and (14). Consider the ADMM algorithm associated to problem (5) based on (15):

$$x^i_i(k) = \arg\min_{x^i_i \in \mathbb{R}} L^a_i\Big((x^i_i, x^i_{-i}(k-1)), \{t^{ls}(k-1)\}_{l \in V, s \in N_l},$$
$$\{u^{ls}(k)\}_{l \in V, s \in N_l}, \{v^{ls}(k)\}_{l \in V, s \in N_l}\Big)$$
$$= \arg\min_{x^i_i \in \mathbb{R}}\Big\{J_i(x^i_i, x^i_{-i}(k-1)) + \mathcal{I}_{\Omega_i}(x^i_i)$$
$$+ \sum_{j \in N_i}(u^{ij}(k) + v^{ji}(k))^T(x^i_i, x^i_{-i}(k-1))$$
$$+ c\sum_{j \in N_i}\Big\|(x^i_i, x^i_{-i}(k-1)) - t^{ij}(k-1)\Big\|^2\Big\} \quad \forall i \in V, \qquad (16)$$

The update rule for the auxiliary variable t^{ij} $\forall i \in V, j \in N_i$ is based on (15),

$$t^{ij}(k) = \arg\min_{t^{ij}} L^a_i\Big((x^i_i(k), x^i_{-i}(k-1)), \{t^{ls}\}_{l \in V, s \in N_l},$$
$$\{u^{ls}(k)\}_{l \in V, s \in N_l}, \{v^{ls}(k)\}_{l \in V, s \in N_l}\Big)$$
$$= \arg\min_{t^{ij}}\Big\{-(u^{ij}(k) + v^{ij}(k))^T t^{ij}$$
$$+ \frac{c}{2}(\|(x^i_i(k), x^i_{-i}(k-1)) - t^{ij}\|^2$$
$$+ \|(x^j_i(k), x^j_{-i}(k-1)) - t^{ij}\|^2)\Big\} = \frac{1}{2c}(u^{ij}(k) + v^{ij}(k))$$
$$+ \frac{1}{2}((x^i_i(k), x^i_{-i}(k-1)) + (x^j_i(k), x^j_{-i}(k-1))). \qquad (17)$$

The initial conditions $u^{ij}(0) = v^{ij}(0) = \mathbf{0}_N$ $\forall i \in V, j \in N_i$ along with (13) and (14) suggest that $u^{ij}(k) + v^{ij}(k) = \mathbf{0}_N$ $\forall i \in V, j \in N_i, k > 0$. Then,

$$t^{ij}(k) = \frac{(x^i_i(k), x^i_{-i}(k-1)) + (x^j_i(k), x^j_{-i}(k-1))}{2}. \qquad (18)$$

Using (18) in (16), one can derive the local estimate update for all $i \in V$ as follows:

$$x^i_i(k) = \arg\min_{x^i_i \in \mathbb{R}}\Big\{J_i(x^i_i, x^i_{-i}(k-1)) + \mathcal{I}_{\Omega_i}(x^i_i)$$
$$+ \sum_{j \in N_i}(u^{ij}(k) + v^{ji}(k))^T(x^i_i, x^i_{-i}(k-1))$$
$$+ c\sum_{j \in N_i}\Big\|(x^i_i, x^i_{-i}(k-1)) \qquad (19)$$
$$- \frac{(x^i_i(k-1), x^i_{-i}(k-2)) + (x^j_i(k-1), x^j_{-i}(k-2))}{2}\Big\|^2\Big\}$$

We simplify (19) by using a proximal first-order approximation for $J_i(x_i^i, x_{-i}^i(k-1))$ around $x^i(k-1)$; thus using inexact ADMM it follows:

$$x_i^i(k) = \arg \min_{x_i^i \in \mathbb{R}} \Big\{ \nabla_i J_i(x^i(k-1))^T (x_i^i - x_i^i(k-1))$$

$$+ \frac{\beta_i}{2} \|x_i^i - x_i^i(k-1)\|^2 + \mathcal{I}_{\Omega_i}(x_i^i) + \sum_{j \in N_i} (u_i^{ij}(k) + v_i^{ji}(k)) x_i^i$$

$$+ c \sum_{j \in N_i} \Big\| x_i^i - \frac{x_i^i(k-1) + x_i^j(k-1)}{2} \Big\|^2 \Big\} \quad \forall i \in V, \qquad (20)$$

where $\beta_i > 0$ is a penalty factor for the proximal first-order approximation of each player i's cost function.

At this point, the players are ready to begin a new iteration from step 2. To sum up, the algorithm consists of (12), (13), (14) and (20) which are the update rule for the players' estimates except their own actions, the update rules for the Lagrange multipliers and the update rule for player's action, respectively.

4. CONVERGENCE PROOF

Theorem 1. Let $\beta_{\min} := \min_{i \in V} \beta_i > 0$ [3] be the minimum penalty factor of the approximation in the inexact ADMM algorithm which satisfies

$$\sigma_F > \frac{1}{2(\beta_{\min} + c\lambda_{\min}(D+A))}, \qquad (21)$$

where σ_F is a positive constant for the cocoercive property of F, and D and A are the degree and adjacency matrices of G_C, respectively. Under Assumptions 1-3, the sequence $\{x^i(k)\} \forall i \in V$, generated by the algorithm (12), (13), (14) and (20), converges to x^* NE of game (2).

Proof. The optimality condition of (20) yields:

$$\nabla_i J_i(x^i(k-1)) + \beta_i(x_i^i(k) - x_i^i(k-1))$$

$$+ \partial_i \mathcal{I}_{\Omega_i}(x_i^i(k)) + \sum_{j \in N_i} (u_i^{ij}(k) + v_i^{ji}(k))$$

$$+ 2c \sum_{j \in N_i} \Big(x_i^i(k) - \frac{x_i^i(k-1) + x_i^j(k-1)}{2} \Big) = 0. \qquad (22)$$

We combine (22) with (10) which represents the equations associated with the solutions of the set of problems (5) (NE of game (2)). Then we obtain,

$$\nabla_i J_i(x^i(k-1)) - \nabla_i J_i(x^*) + \beta_i(x_i^i(k) - x_i^i(k-1))$$

$$+ \partial_i \mathcal{I}_{\Omega_i}(x_i^i(k)) - \partial_i \mathcal{I}_{\Omega_i}(x_i^*)$$

$$+ \sum_{j \in N_i} (u_i^{ij}(k) + v_i^{ji}(k) - u_i^{ij^*} - v_i^{ji^*})$$

$$+ 2c \sum_{j \in N_i} \Big(x_i^i(k) - \frac{x_i^i(k-1) + x_i^j(k-1)}{2} \Big) = 0. \qquad (23)$$

We multiply both sides by $(x_i^i(k) - x_i^*)$ and then add and subtract $x_i^i(k-1)$ as follows:

$$\Big(\nabla_i J_i(x^i(k-1)) - \nabla_i J_i(x^*) \Big)^T (x_i^i(k-1) - x_i^*)$$

[3] In order to have a fully distributed algorithm, one can consider a network-wide known lower bound $\tilde{\beta}_{\min}$, $\tilde{\beta}_{\min} \le \beta_i \forall i \in V$ and use it instead of $\beta_{\min}$.

$$+ \Big(\nabla_i J_i(x^i(k-1)) - \nabla_i J_i(x^*) \Big)^T (x_i^i(k) - x_i^i(k-1))$$

$$+ \beta_i(x_i^i(k) - x_i^i(k-1))^T(x_i^i(k) - x_i^*)$$

$$+ (\partial \mathcal{I}_{\Omega_i}(x_i^i(k)) - \partial \mathcal{I}_{\Omega_i}(x_i^*))^T(x_i^i(k) - x_i^*)$$

$$+ \sum_{j \in N_i} (u_i^{ij}(k) + v_i^{ji}(k) - u_i^{ij^*} - v_i^{ji^*})^T(x_i^i(k) - x_i^*) \qquad (24)$$

$$+ 2c \sum_{j \in N_i} \Big(x_i^i(k) - \frac{x_i^i(k-1) + x_i^j(k-1)}{2} \Big)^T (x_i^i(k) - x_i^*) = 0.$$

As discussed in Remark 3, in addition to updating their own actions, the players need to update their estimates as well. In the following, we explain how to bring in the update rule of x_{-i}^i into (24).

Note that by (12), one can obtain,

$$\sum_{j \in N_i} (u_{-i}^{ij}(k) + v_{-i}^{ji}(k)) \qquad (25)$$

$$+ 2c \sum_{j \in N_i} \Big(x_{-i}^i(k) - \frac{x_{-i}^i(k-1) + x_{-i}^j(k-1)}{2} \Big) = \mathbf{0}_{N-1}.$$

Multiplying (25) by $(x_{-i}^i(k) - x_{-i}^*)$, one can arrive at,

$$\sum_{j \in N_i} (u_{-i}^{ij}(k) + v_{-i}^{ji}(k))^T (x_{-i}^i(k) - x_{-i}^*)$$

$$+ 2c \sum_{j \in N_i} \Big(x_{-i}^i(k) - \frac{x_{-i}^i(k-1) + x_{-i}^j(k-1)}{2} \Big)^T$$

$$\cdot (x_{-i}^i(k) - x_{-i}^*) = 0. \qquad (26)$$

Adding (26) to (24) and using (13), (14), yilds $\forall i \in V$,

$$\Big(\nabla_i J_i(x^i(k-1)) - \nabla_i J_i(x^*) \Big)^T (x_i^i(k-1) - x_i^*)$$

$$+ \Big(\nabla_i J_i(x^i(k-1)) - \nabla_i J_i(x^*) \Big)^T (x_i^i(k) - x_i^i(k-1))$$

$$+ \beta_i(x_i^i(k) - x_i^i(k-1))^T(x_i^i(k) - x_i^*)$$

$$+ (\partial \mathcal{I}_{\Omega_i}(x_i^i(k)) - \partial \mathcal{I}_{\Omega_i}(x_i^*))^T(x_i^i(k) - x_i^*)$$

$$+ \sum_{j \in N_i} (u_i^{ij}(k+1) + v_i^{ji}(k+1) - u_i^{ij^*} - v_i^{ji^*})^T(x_i^i(k) - x_i^*)$$

$$+ \sum_{j \in N_i} (u_{-i}^{ij}(k+1) + v_{-i}^{ji}(k+1))^T (x_{-i}^i(k) - x_{-i}^*)$$

$$+ 2c \sum_{j \in N_i} \Big(\frac{x^i(k) + x^j(k)}{2} - \frac{x^i(k-1) + x^j(k-1)}{2} \Big)^T$$

$$\cdot (x^i(k) - x^*) = 0. \qquad (27)$$

The second and the third terms are bounded as follows:

$$\Big(\nabla_i J_i(x^i(k-1)) - \nabla_i J_i(x^*) \Big)^T (x_i^i(k) - x_i^i(k-1)) \ge \quad (28)$$

$$\frac{-1}{2\rho} \|\nabla_i J_i(x^i(k-1)) - \nabla_i J_i(x^*)\|^2 - \frac{\rho}{2} \|x_i^i(k) - x_i^i(k-1)\|^2,$$

for any $\rho > 0$ $\forall i \in V$. By the convexity of $\mathcal{I}_{\Omega_i}$ (Assumption 2), we have for the fourth term,

$$(\partial \mathcal{I}_{\Omega_i}(x_i^i(k)) - \partial \mathcal{I}_{\Omega_i}(x_i^*))^T(x_i^i(k) - x_i^*) \ge 0. \qquad (29)$$

Using (28) and (29) in (27) and summing over $i \in V$, we obtain,

$$\left(F(\underline{x}(k-1)) - F(\underline{x}^*)\right)^T (x(k-1) - x^*)$$

$$-\frac{1}{2\rho}\|F(\underline{x}(k-1)) - F(\underline{x}^*)\|^2 - \frac{1}{2}\|\underline{x}(k) - \underline{x}(k-1)\|_{M_1}^2$$

$$+(\underline{x}(k) - \underline{x}(k-1))^T \mathrm{diag}((\beta_i e_i e_i^T)_{i \in V})(\underline{x}(k) - \underline{x}^*)$$

$$+\sum_{i \in V}\sum_{j \in N_i}(u_i^{ij}(k+1)+v_i^{ji}(k+1)-u_i^{ij*}-v_i^{ji*})^T(x_i^i(k)-x_i^*)$$

$$+\sum_{i \in V}\sum_{j \in N_i}(u_{-i}^{ij}(k+1) + v_{-i}^{ji}(k+1))^T(x_{-i}^i(k) - x_{-i}^*)$$

$$+c(\underline{x}(k) - \underline{x}(k-1))^T((D+A) \otimes I_N)(\underline{x}(k) - \underline{x}^*) \le 0, \quad (30)$$

where $M_1 := \mathrm{diag}((\rho e_i e_i^T)_{i \in V})$ and $\underline{x}^* = [x^{1*T}, \ldots, x^{N*T}]^T$. We bound the first term using Assumption 3,

$$\left(F(\underline{x}(k-1)) - F(\underline{x}^*)\right)^T (x(k-1) - x^*)$$

$$\ge \sigma_F\|F(\underline{x}(k-1)) - F(\underline{x}^*)\|^2. \quad (31)$$

We also simplify the fifth and the sixth terms in (30). Since G_C is an undirected graph, for any $\{a_{ij}\}$, $\sum_{i \in V}\sum_{j \in N_i} a_{ij} = \sum_{i \in V}\sum_{j \in N_i} a_{ji}$. Then,

$$\sum_{i \in V}\sum_{j \in N_i}(u_i^{ij}(k+1)+v_i^{ji}(k+1)-u_i^{ij*}-v_i^{ji*})^T(x_i^i(k)-x_i^*)$$

$$+\sum_{i \in V}\sum_{j \in N_i}(u_{-i}^{ij}(k+1) + v_{-i}^{ji}(k+1))^T(x_{-i}^i(k) - x_{-i}^*)$$

$$=\sum_{i \in V}\sum_{j \in N_i}(u_i^{ij}(k+1) - u_i^{ij*})^T(x_i^i(k) - x_i^*)$$

$$+\sum_{i \in V}\sum_{j \in N_i}(v_i^{ij}(k+1) - v_i^{ij*})^T(x_i^j(k) - x_i^*)$$

$$+\sum_{i \in V}\sum_{j \in N_i}u_{-i}^{ij}(k+1)^T(x_{-i}^i(k) - x_{-i}^*)$$

$$+\sum_{i \in V}\sum_{j \in N_i}v_{-i}^{ij}(k+1)^T(x_{-i}^j(k) - x_{-i}^*). \quad (32)$$

Note that by (13) and (14) as well as the initial conditions for Lagrange multipliers $u^{ij}(0) = v^{ij}(0) = \mathbf{0}_N \ \forall i \in V, j \in N_i$, we obtain,

$$u^{ij}(k) + v^{ij}(k) = \mathbf{0}_N \quad \forall i \in V, j \in N_i, k > 0. \quad (33)$$

Substituting (33) into (32) and using (8), we obtain,

$$\sum_{i \in V}\sum_{j \in N_i}(u_i^{ij}(k+1) - u_i^{ij*})^T(x_i^i(k) - x_i^j(k))$$

$$+\sum_{i \in V}\sum_{j \in N_i}u_{-i}^{ij}(k+1)^T(x_{-i}^i(k) - x_{-i}^j(k))$$

$$=\sum_{i \in V}\sum_{j \in N_i}(u^{ij}(k+1) - u_i^{ij*}e_i)^T(x^i(k) - x^j(k))$$

$$=\frac{2}{c}\sum_{i \in V}\sum_{j \in N_i}(u^{ij}(k+1) - u_i^{ij*}e_i)^T(u^{ij}(k+1) - u^{ij}(k))$$

$$:= \frac{2}{c}(\underline{u}(k+1) - \underline{u}^*)^T(\underline{u}(k+1) - \underline{u}(k)), \quad (34)$$

where $\underline{u} = (u^i)_{i \in V} \in \mathbb{R}^{N\sum_{i \in V}|N_i|}$ and $u^i = (u^{ij})_{j \in N_i} \in \mathbb{R}^{N|N_i|}$ and also $\underline{u}^* = (u^{i*})_{i \in V} \in \mathbb{R}^{N\sum_{i \in V}|N_i|}$ and $u^{i*} = (u_i^{ij*})_{j \in N_i} \otimes e_i \in \mathbb{R}^{N|N_i|}$.

Using (31) and (34), for $\rho = \frac{1}{2\sigma_F}$ (30) becomes,

$$-\frac{1}{2}\|\underline{x}(k) - \underline{x}(k-1)\|_{M_1}^2$$

$$+(\underline{x}(k) - \underline{x}(k-1))^T M_2(\underline{x}(k) - \underline{x}^*)$$

$$+\frac{2}{c}(\underline{u}(k+1) - \underline{u}^*)^T(\underline{u}(k+1) - \underline{u}(k)) \le 0, \quad (35)$$

where $M_2 := \mathrm{diag}((\beta_i e_i e_i^T)_{i \in V}) + c((D+A) \otimes I_N)$. Note that $\mathrm{diag}((\beta_i e_i e_i^T)_{i \in V}) \succeq 0$. Note also that,

$$c((D+A) \otimes I_N) = c((2D-L) \otimes I_N)$$
$$= c((D^{\frac{1}{2}}(2I - D^{-\frac{1}{2}}LD^{-\frac{1}{2}})D^{\frac{1}{2}}) \otimes I_N)$$
$$= c((D^{\frac{1}{2}}(2I - L_N)D^{\frac{1}{2}}) \otimes I_N), \quad (36)$$

where $L := D - A$, $D^{\frac{1}{2}}$, $D^{-\frac{1}{2}}$ and $L_N := D^{-\frac{1}{2}}LD^{-\frac{1}{2}}$ are the Laplacian of G_C, the square root and reciprocal square root of D and the normalized Laplacian of G_C, respectively. Since $D \succ 0$, $D^{-\frac{1}{2}}$ exist, it is shown in Chung (1997) that $\lambda_{\max}(L_N) \le 2$. Then (36) yields that $c((D+A) \otimes I_N) \succeq 0$. This concludes $M_2 \succeq 0$.

We use the following inequality in (35) for every $\{a(k)\}$ and $M \succeq 0$:

$$(a(k) - a(k-1))^T M(a(k) - a^*) = \frac{1}{2}\|a(k) - a^*\|_M^2$$

$$+\frac{1}{2}\|a(k) - a(k-1)\|_M^2 - \frac{1}{2}\|a(k-1) - a^*\|_M^2. \quad (37)$$

Then, (35) becomes,

$$\frac{1}{2}\|\underline{x}(k) - \underline{x}^*\|_{M_2}^2 + \frac{1}{c}\|\underline{u}(k+1) - \underline{u}^*\|^2 \le$$

$$\frac{1}{2}\|\underline{x}(k-1) - \underline{x}^*\|_{M_2}^2 + \frac{1}{c}\|\underline{u}(k) - \underline{u}^*\|^2 \quad (38)$$

$$-\frac{1}{2}\|\underline{x}(k) - \underline{x}(k-1)\|_{M_2 - M_1}^2 - \frac{1}{c}\|\underline{u}(k+1) - \underline{u}(k)\|^2.$$

By the condition (21), $M_2 - M_1 \succ 0$. Then (38) implies the following two results:

(1) $\frac{1}{2}\|\underline{x}(k) - \underline{x}^*\|_{M_2}^2 + \frac{1}{c}\|\underline{u}(k+1) - \underline{u}^*\|^2 \to \theta$, for some $\theta \ge 0$,

(2) $\begin{cases} \underline{x}(k) - \underline{x}(k-1) \to \mathbf{0}_{N^2} \\ \underline{u}(k+1) - \underline{u}(k) \to \mathbf{0}_{N\sum_{i \in V}|N_i|} \end{cases}$.

Result 1 implies that the sequences $\{x^i(k)\}$ and $\{u^{ij}(k)\}$ (similarly $\{v^{ij}(k)\}$) are bounded and have limit points denoted by $\tilde{x}^i$ and $\tilde{u}^{ij}$ ($\tilde{v}^{ij}$), respectively. Then, we obtain,

$$\theta = \frac{1}{2}\|\underline{\tilde{x}} - \underline{x}^*\|_{M_2}^2 + \frac{1}{c}\|\underline{\tilde{u}} - \underline{u}^*\|^2 \quad (39)$$

Result 2 yields that $\tilde{x}^i = \tilde{x}^j := \tilde{x}$ for all $i \in V, j \in N_i$ since by (13) we have,

$$\frac{c}{2}\left(x^i(k) - x^j(k)\right) = u^{ij}(k+1) - u^{ij}(k) \to \mathbf{0}_N$$

$$\Rightarrow \tilde{x}^i = \tilde{x}^j \quad \forall i \in V, j \in N_i. \quad (40)$$

Moreover, by (33) we arrive at,

$$\tilde{u}^{ij} + \tilde{v}^{ij} = \mathbf{0}_N \quad \forall i \in V, j \in N_i. \quad (41)$$

Result 2 also implies that by (22) and (40),

$$\nabla_i J_i(\tilde{x}) + \partial \mathcal{I}_{\Omega_i}(\tilde{x}_i) + \sum_{j \in N_i}(\tilde{u}_i^{ij} + \tilde{v}_i^{ji}) = 0. \quad (42)$$

Comparing (41), (42) with (10), it follows $\forall i \in V, j \in N_i$,

$$\tilde{x}^i = x^* \quad (\underline{\tilde{x}} = \underline{x}^*), \tag{43}$$

$$\tilde{u}^{ij} = u^{ij^*} \quad (\underline{\tilde{u}} = \underline{u}^*). \tag{44}$$

Using (43) and (44) in (39), it follows that $\theta = 0$. Thus, one can conclude from Result 1 that, $\frac{1}{2}\|\underline{x}(k) - \underline{x}^*\|_{M_2}^2 + \frac{1}{c}\|\underline{u}(k+1) - \underline{u}^*\|^2 \to 0$ which completes the proof. ∎

Remark 4. Assumption 3 is only used in equation (31). It is straightforward to verify that (31) can be satisfied by Assumption 3 for $y = x^*$ (similar to Assumption 4.4 in Frihauf et al. (2012)).

5. IMPLEMENTATION OF ALGORITHM

For the purpose of implementation, we simplify the algorithm to a more compact representation. One may begin with (12) and (20) as follows:

(1) Let $w^i := \sum_{j \in N_i} u^{ij} + v^{ji}$. Then by (13) and (14),

$$w^i(k) = w^i(k-1) + c\sum_{j \in N_i}(x^i(k-1) - x^j(k-1)). \tag{45}$$

(2) By replacing $\sum_{j \in N_i}(u^{ij}_{-i}(k) + v^{ji}_{-i}(k))$ with $w^i_{-i}(k)$ and using (45) in (12), after a few manipulations we obtain,

$$x^i_{-i}(k) = \frac{1}{|N_i|}\sum_{j \in N_i} x^j_{-i}(k-1)) - \frac{1}{2c|N_i|}w^i_{-i}(k-1).$$

(3) By differentiating of (20) w.r.t. x^i_i and equating it to 0, one can verify that x^i_i can be obtained as:

$$x^i_i(k) = \arg\min_{x^i_i \in \mathbb{R}}\Big\{\mathcal{I}_{\Omega_i}(x^i_i) + \frac{\alpha_i}{2}\Big\|x^i_i - \alpha_i^{-1}\Big(\beta_i x^i_i(k-1) - w^i_i(k)$$

$$-\nabla_i J_i(x^i(k-1)) + c\sum_{j \in N_i}\big(x^i_i(k-1) + x^j_i(k-1)\big)\Big)\Big\|^2\Big\},$$

where $\alpha_i = \beta_i + 2c|N_i|$. Let $\text{prox}_g^a[s] := \arg\min_x\{g(x) + \frac{a}{2}\|x - s\|^2\}$ be the proximal operator for the non-smooth function g. Note that $\text{prox}_{\mathcal{I}_{\Omega_i}}^{\alpha_i}[s] = T_{\Omega_i}[s]$ where $T_{\Omega_i} : \mathbb{R} \to \Omega_i$ is an Euclidean projection. Then for each player i, $\forall i \in V$ we obtain,

$$x^i_i(k) = T_{\Omega_i}\Big[\alpha_i^{-1}(\beta_i + c|N_i|)x^i_i(k-1)$$

$$-\alpha_i^{-1}\Big(w^i_i(k) + \nabla_i J_i(x^i(k-1)) - c\sum_{j \in N_i} x^j_i(k-1)\Big)\Big].$$

Then the ADMM algorithm is as follows:

Algorithm 1 ADMM Algorithm for Implementation

1: **initialization** $x^i(0) \in \Omega$, $w^i(0) = \mathbf{0}_N$ $\forall i \in V$
2: **for** $k = 1, 2, \ldots$ **do**
3:     **for** each player $i \in V$ **do**
4:       players exchange estimates with the neighbors
5:       $w^i(k) = w^i(k-1) + c\sum_{j \in N_i}(x^i(k-1) - x^j(k-1))$
6:       $x^i_{-i}(k) = \frac{\sum_{j \in N_i} x^j_{-i}(k-1))}{|N_i|} - \frac{w^i_{-i}(k-1)}{2c|N_i|}$
7:       $x^i_i(k) = T_{\Omega_i}\Big[\frac{\beta_i + c|N_i|}{\alpha_i}x^i_i(k-1) - \alpha_i^{-1}\Big(w^i_i(k)$
               $+\nabla_i J_i(x^i(k-1)) - c\sum_{j \in N_i} x^j_i(k-1)\Big)\Big]$
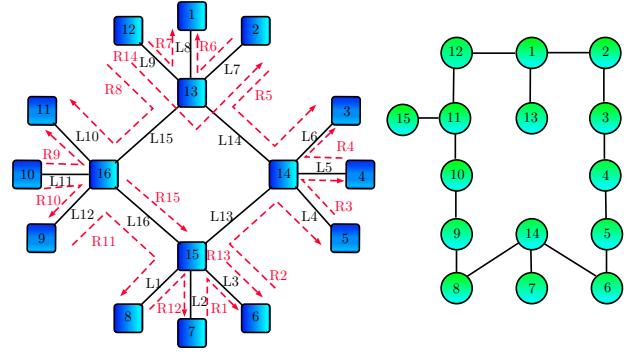8:   **end for**
9: **end for**

Fig. 1. (a) Wireless Ad-Hoc Network (left). (b) Communication graph G_C (right).

6. SIMULATION RESULTS

In this section, we compare our algorithm with the gradient-based one proposed in Salehisadaghiani and Pavel (2016a).

We consider a wireless ad-hoc network (WANET) with 16 nodes and 16 multi-hop communication links as in Salehisadaghiani and Pavel (2016b). There are 15 users who aim to transfer data from a source node to a destination node via this WANET. Fig. 1 (a) shows the topology of the WANET in which solid lines represent links and dashed lines display paths that assigned to users to transfer data. Each link has a positive capacity that restricts the users' data flow . Here is the list of WANET notations:

(1) L_j: Link j, $j \in \{1, \ldots, 16\}$,
(2) R_i: The path assigned to user i, $i \in \{1, \ldots, 15\}$,
(3) $C_j > 0$: The capacity assigned to each link j, $j \in \{1, \ldots, 16\}$,
(4) $0 \leq x_i \leq 10$: The data flow of user i, $i \in \{1, \ldots, 15\}$.

Note that each path consists of a set of links, e.g., $R_1 = \{L_2, L_3\}$.

For each user i, a cost function J_i is defined as in Salehisadaghiani and Pavel (2016b):

$$J_i(x_i, x_{-i}) := \sum_{j : L_j \in R_i} \frac{\kappa}{C_j - \sum_{w : L_j \in R_w} x_w} - \chi_i \log(x_i + 1),$$

where $\kappa > 0$ and $\chi_i > 0$ are network-wide known and user-specific parameters, respectively.

The problem is to find an NE of the game which is played over a communication graph G_C (depicted in Fig. 1 (b)). It is straightforward to check the Assumptions 1,2 and 3 [4] on G_C and the cost functions. We aim to compare the convergence rate of our algorithm with the one proposed in Salehisadaghiani and Pavel (2016a). The results of Algorithm 1 and the algorithm in Salehisadaghiani and Pavel (2016a) are shown in Fig. 2, for $\chi_i = 10$ $\forall i \in \{1, \ldots, 15\}$ and $C_j = 10$ $\forall j \in \{1, \ldots, 16\}$ (Fig. 2). The simulation results verify that the proposed algorithm is 70 times faster than the one in Salehisadaghiani and Pavel (2016a). The factors that lead to this improvement are as follows:

- We used the difference between the estimates of the users as a penalty term to update each user's action and estimates.

---
[4] It is sufficient for the cost functions that only satisfy equation (31) (see Remark 4).
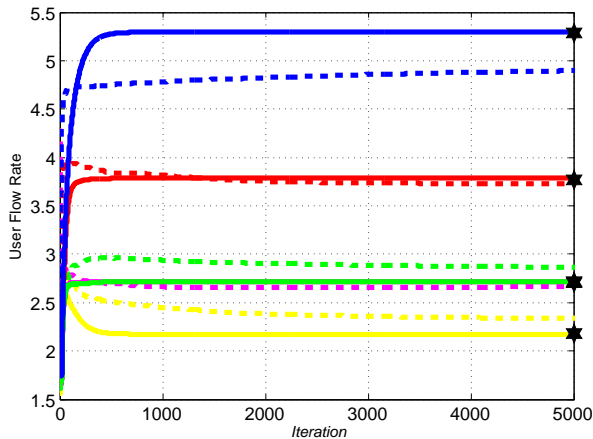
Fig. 2. Flow rates of users 1, 3, 5, 8, 13 using our algorithm (solid lines) vs. proposed algorithm in Salehisadaghiani and Pavel (2016a) (dashed lines). NE points are represented by black stars.

- We used a synchronous algorithm by which every user updates his action and estimates at the same time with the other users.
- Unlike gossiping protocol, which is used in Salehisadaghiani and Pavel (2016a), every user communicates with all of the neighboring users (not only one of them) at each iteration.

7. CONCLUSIONS

A distributed NE seeking algorithm is designed using inexact-ADMM to achieve more speed and robustness. The game is reformulated within the framework of inexact-ADMM. The communications between the players are defined to exchange the estimates. An inexact-ADMM-like approach is then designed and its convergence to an NE of the game is analyzed. Eventually, the convergence rate of the algorithm is compared with an existing gossip-based NE seeking algorithm.

REFERENCES

Alpcan, T. and Başar, T. (2005). Distributed algorithms for Nash equilibria of flow control games. In *Advances in Dynamic Games*, 473–498. Springer.

Bertsekas, D.P. and Tsitsiklis, J.N. (1997). *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, Belmont, Massachusetts.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1), 1–122.

Bramoullé, Y., Kranton, R., and D'amours, M. (2014). Strategic interaction and networks. *The American Economic Review*, 104(3), 898–930.

Chang, T.H., Hong, M., and Wang, X. (2015). Multi-agent distributed optimization via inexact consensus admm. *IEEE Transactions on Signal Processing*, 63(2), 482–497.

Chung, F.R. (1997). *Spectral graph theory*, volume 92. American Mathematical Soc.

Frihauf, P., Krstic, M., and Basar, T. (2012). Nash equilibrium seeking in noncooperative games. *IEEE Transactions on Automatic Control*, 57(5), 1192–1207.

Gharesifard, B. and Cortes, J. (2013). Distributed convergence to Nash equilibria in two-network zero-sum games. *Automatica*, 49(6), 1683–1692.

Goldstein, T., O'Donoghue, B., Setzer, S., and Baraniuk, R. (2014). Fast alternating direction optimization methods. *SIAM Journal on Imaging Sciences*, 7(3), 1588–1623.

He, B. and Yuan, X. (2012). On the o(1/n) convergence rate of the douglas-rachford alternating direction method. *SIAM Journal on Numerical Analysis*, 50(2), 700–709.

Johansson, B. (2008). On distributed optimization in networked systems. *Ph.D. Dissertation*.

Koshal, J., Nedic, A., and Shanbhag, U.V. (2012). A gossip algorithm for aggregative games on graphs. In *IEEE 51st Conference on Decision and Control (CDC)*, 4840–4845.

Li, N. and Marden, J.R. (2013). Designing games for distributed optimization. *IEEE Journal of Selected Topics in Signal Processing*, 7(2), 230–242.

Lou, Y., Hong, Y., Xie, L., Shi, G., and Johansson, K.H. (2016). Nash equilibrium computation in subnetwork zero-sum games with switching communications. *IEEE Transactions on Automatic Control*, 61(10), 2920–2935.

Makhdoumi, A. and Ozdaglar, A. (2014). Broadcast-based distributed alternating direction method of multipliers. In *Communication, Control, and Computing (Allerton), 2014 52nd Annual Allerton Conference on*, 270–277. IEEE.

Nedic, A. (2011). Asynchronous broadcast-based convex optimization over a network. *IEEE Transactions on Automatic Control*, 56(6), 1337–1351.

Pan, Y. and Pavel, L. (2009). Games with coupled propagated constraints in optical networks with multi-link topologies. *Automatica*, 45(4), 871–880.

Parikh, N., Boyd, S.P., et al. (2014). Proximal algorithms. *Foundations and Trends in optimization*, 1(3), 127–239.

Parise, F., Gentile, B., Grammatico, S., and Lygeros, J. (2015). Network aggregative games: Distributed convergence to Nash equilibria. In *2015 54th IEEE Conference on Decision and Control (CDC)*, 2295–2300. IEEE.

Pavel, L. (2007). An extension of duality to a game-theoretic framework. *Automatica*, 43(2), 226–237.

Salehisadaghiani, F. and Pavel, L. (2014). Nash equilibrium seeking by a gossip-based algorithm. In *IEEE 53rd Conference on Decision and Control (CDC)*, 1155–1160.

Salehisadaghiani, F. and Pavel, L. (2016a). Distributed Nash equilibrium seeking: A gossip-based algorithm. *Automatica*, 72, 209–216.

Salehisadaghiani, F. and Pavel, L. (2016b). Distributed Nash equilibrium seeking by gossip in games on graphs. *arXiv preprint arXiv:1610.01896*.

Shi, W., Ling, Q., Yuan, K., Wu, G., and Yin, W. (2014). On the linear convergence of the admm in decentralized consensus optimization. *IEEE Transactions on Signal Processing*, 62(7), 1750–1761.

Shi, W. and Pavel, L. (submitted). LANA: an ADMM-like Nash equilibrium seeking algorithm in decentralized environment.

Wei, E. and Ozdaglar, A. (2012). Distributed alternating direction method of multipliers. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*,

5445–5450. IEEE.

Ye, M. and Hu, G. (2015). Distributed seeking of time-varying Nash equilibrium for non-cooperative games. *IEEE Transactions on Automatic Control*, 60(11), 3000–3005.

Yin, H., Shanbhag, U.V., and Mehta, P.G. (2011). Nash equilibrium problems with scaled congestion costs and shared constraints. *IEEE Transactions on Automatic Control*, 56(7), 1702–1708.

Zhu, M. and Frazzoli, E. (2016). Distributed robust adaptive equilibrium computation for generalized convex games. *Automatica*, 63, 82–91.