Distributed Solution of Networked Convex-Concave Games with Coupling Constraints

Jianghai Hu, *Member, IEEE,* Yingying Xiao, *Student Member, IEEE,* and Xiaodong Hou

***Abstract*—In this paper, we study the convex-concave games played by two teams on a network. The two teams have their decision variables scattered across all the nodes of the network. At each node there is a local convex-concave (saddle) payoff function that depends on the decision variables of not only that node but also its neighboring nodes. In addition, there are constraints that couple the decision variables of each team at nodes within a neighborhood and/or across the whole network. The goal is to design distributed algorithms so that each team iteratively updates its decision variables using only locally available information so that collectively the decision variables of both teams converge to a Nash equilibrium (saddle point) of the global payoff function (sum of local payoff functions) while satisfying all the coupling constraints. Using the notion of saddle differential operators and operator splitting techniques, we propose several distributed algorithms with guaranteed convergence to saddle point solutions under mild assumptions. As a special instance, these algorithms can lead to new primal-dual solution algorithms for the distributed optimization problems on networks. Several examples, including the networked Cournot competition, are provided to illustrate the proposed algorithms.**

***Index Terms*—Convex-concave games, networked optimization, distributed algorithms, operator splitting.**

I. INTRODUCTION

Saddle point problems have many important applications in, e.g., constrained optimization, zero-sum games, partial differential equations, machine learning (see the survey [1]). Methods developed for computing saddle points include iterative methods such as the Arrow-Hurwicz and Uzawa method [2] and their variants, Krylov subspace methods and associated preconditioning techniques [3], Hermitian and skew-Hermitian splitting based methods [4], best response dynamics [5], to name a few. Many (sub)gradient-based algorithms have also been developed, e.g., smoothing techniques [6], prox method [7], primal-dual gradient flows [8], and approximating the saddle point using the running average with a constant stepsize [9]. The saddle point problems studied and the algorithms proposed in these work are all centralized.

This paper studies the distributed solution of saddle point problems on a network. We consider zero-sum games between two teams (x-team and y-team) whose decision variables (x_i's and y_i's) are distributed among all the nodes of the network. Each node i is associated with a local payoff function K_i that depends on the decision variables of both team not only at node i but also at neighboring nodes. In addition, for each team, its decision variables could be subject to either local coupling constraints involving nodes from the same neighborhood or global coupling constraints involving all nodes of the network. The goal is to compute a Nash equilibrium, or equivalently, a saddle point of the global payoff function $K = \sum_i K_i$ satisfying all coupling constraints via properly designed distributed algorithms that update the local decision variables at each node using only information at the node and from its neighboring nodes. A special instance of the game is when each node is allowed to have the decision variables from only one team. In this case, the network nodes are partitioned into two disjoint groups; nodes in the same group form a coalition and coordinate with one another to compete against nodes in the other group. Another special instance is, when studying networked constrained optimization problems, the Lagrange functions can be deemed as the payoff functions for a game between the primal and dual variables.

Compared with existing work, a distinct feature of the networked games studied here is that they are truly ubiquitous over the network: not only is the global payoff function distributed as local payoff functions, each player's decision variables are also distributed on all the nodes. The latter allows for the coexistence of cooperation and competition among nodes and, together with general network topology, gives arise to intriguing local and global behaviors in the resulting Nash equilibriums. In comparison, in the games studied in [10]–[14], each player's decision variables are confined to a single node that selfishly competes against the rest of the nodes. The aggregate games studied in [15]–[18] can also be considered the same with the introduction of a central (and neutral) node that stores the aggregate information. In [19], [20], the games considered are played between two subnetworks with bipartite connections. As described in the previous paragraph, they are special instances of the games studied here.

Another notable feature of our game formulation is that it allows general coupling constraints both locally and globally: the decision variables of both teams at nodes in the same neighborhood can have arbitrary convex constraints as in (9); the decision variables of the same team from the whole network can be subject to a sum-of-convex-functions type of global constraint as in (26). The presence of these coupling constraints, especially the global ones, makes it challenging to design distributed algorithms where the update by each team at any node can rely on only local information, e.g., only a local portion of the global constraints relevant to that node. Among the existing work, [11], [17], [19], [20] did not consider coupling constraints; [12] considered only local coupling constraints; [16] considered constraints on the aggregate value; and [13]–[15], [18] considered only global

J. Hu, Y. Xiao, and X. Hou were with the Department of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, 47907 USA e-mail: {jianghai,xiao106,hou39}@purdue.edu.

constraints that are affine in the decision variables. It should be pointed out that the coupling constraints in these existing work involve decision variables from different players, while in our case they only involve decision variables from the same player (though located at different network nodes). Inter-player couplings are necessary when different players compete for the same pool of common resource; and intra-player inter-node couplings can model situations where each player has its own limited resource to be allocated across the network, e.g., army deploying forces over strategic locations, company managing inventory across multiple markets.

In this paper, we focus on zero-sum convex-concave games where each local payoff function K_i (hence the global payoff K) is convex in the variables of one team and concave in the variables of the other. Such games were also studied in [5], [19]–[21]. Recently, non-zero-sum multi-player games and in particular monotone games on networks have received a lot of attention [22]. Their solutions, called Generalized Nash Equilibriums (GNEs) [23], are in general not unique and difficult to compute. A common strategy is to formulate the games as variational inequality problems [24] and solve for the so-called variational GNEs, a special class of GNEs with economic interpretation. Many distributed algorithms have been proposed for computing variational GNEs [10], [12], [14], [25]. Our focus on convex-concave games has the following motivations. First, their saddle points exist under very mild conditions (see, e.g., Sion's Minimax Theorem [26] in Proposition 1) and can be computed by distributed algorithms with guaranteed convergence under much weaker assumptions than monotone games. Second, convex-concave games encompass several general classes of important problems. For example, the distributed optimization problems on node networks [27], [28] formulated in the primal-dual framework is equivalent to finding a saddle point of the Lagrange function K that is convex-affine in the primal and dual variables [29]. Thus, the distributed algorithms developed in this paper can be adopted directly for solving the networked optimization problems with local and global coupling constraints. It should be noted that convex-concave games are much broader than networked optimization with many practical applications such as, e.g., power allocation in multi-channel communication [21], networked Cournot game [30].

The main contributions of the paper consist of: 1) formulation of a new class of games on networks with general local and global coupling constraints; 2) development of synchronous and randomized distributed algorithms that can compute Nash equilibrium points under local coupling constraints; 3) development of synchronous distributed algorithms for computing the Nash equilibrium points with the presence of both local and global coupling constraints. This paper significantly extends the preliminary results in the authors' conference paper [31]; particularly, the consideration of coupling constraints and networked Cournot competitions are new additions. The main theoretical tool used for algorithm development is the operator splitting techniques, specifically, the preconditioned Douglas-Rachford splitting methods [32], [33] ([31] used canonical D-R splitting). Such techniques have been successfully applied in the distributed optimization and game solution [14], [16]. Compared to (sub)gradient-based methods such as the ones in [9], [10], [12], [18], [19], our method has several advantages: general nonsmooth payoff functions, general convex constraints (not necessarily bounded), constant step size (no need for tuning), to name a few. A similar preconditioned operator splitting method has been proposed in a recent work [14] for the distributed computation of GNEs for monotone games on networks with coupling constraints. Comparing our work with [14], the games under study and the operator splitting methods used are different (Douglas-Rachford splitting vs. forward-backward splitting). The algorithms proposed in [14], while more computationally efficient, converge under more stringent conditions: differentiable payoff functions with strongly monotone and Lipschitz continuous pseudo-gradients. None of these conditions are required for the convergence of our algorithms.

This paper is organized as follows. Some useful facts on convex-concave (or saddle) functions and saddle points are reviewed in Section II. In Section III, operator splitting methods are introduced to compute the saddle points of composite saddle functions. Section IV formulates the convex-concave games on networks with only local coupling constraints and propose two distributed solution algorithms. The case with global coupling constraints is studied in Section V. Simulation results are given in Section VI. Finally, Section VII concludes the paper.

II. Saddle Functions and Saddle Differential Operators

A. Preliminaries

We first briefly review some basic facts from convex analysis. An extended-real-valued function $f : \mathbb{R}^n \to \overline{\mathbb{R}} := \mathbb{R} \cup \{\pm\infty\}$ is convex if, for all $x_1, x_2 \in \mathbb{R}^n$ and all $\lambda \in [0,1]$, $f(\lambda x_1 + (1-\lambda)x_2) \le \lambda f(x_1) + (1-\lambda) f(x_2)$ holds whenever $\{f(x_1), f(x_2)\} \neq \{\pm\infty\}$. A convex function f is called *proper* if: (i) $f(x) > -\infty$ for all x; and (ii) $f(x) < +\infty$ for at least one x. It is called *closed* if its epigraph $\{(x,t) \,|\, f(x) \le t\}$ is a closed set, or equivalently, if f is lower semicontinuous. A convex function satisfying both the above properties is called convex, closed and proper (CCP). For a convex subset $C \subset \mathbb{R}^n$, its (convex) indicator function $\mathbf{1}_C(\cdot)$ is defined such that $\mathbf{1}_C(x) = 0$ if $x \in C$ and $\mathbf{1}_C(x) = +\infty$ if otherwise. $\mathbf{1}_C(\cdot)$ is closed if and only if C is closed and proper if $C \neq \emptyset$.

Let $Z = \mathbb{R}^n$. A set-valued operator $T : Z \to 2^Z$ has the domain $\mathrm{dom}\, T := \{x \,|\, T(x) \neq \emptyset\}$. Its inverse map $T^{-1} : Z \to 2^Z$ is such that $y \in T^{-1}(x)$ if and only if $x \in T(y)$. T is *monotone* if $(w_2 - w_1)^T (z_2 - z_1) \ge 0$, $\forall z_1, z_2 \in Z,\ w_1 \in T(z_1),\ w_2 \in T(z_2)$. It is *maximally monotone* if it is monotone and its graph $\{(x,y) \,|\, y \in T(x)\}$ is not properly contained in that of any other monotone operator. A well known fact is that the subdifferential operator ∂f of a convex function $f : Z \to \overline{\mathbb{R}}$, where $\partial f(x)$ consists of all the subgradients of f at x, is monotone and, if f is further CCP, maximally monotone. See [34] for more details.

For an operator $T : Z \to 2^Z$ and a constant $\rho > 0$, the *resolvent* of T is the set-valued operator $J_{\rho T}$ defined by $J_{\rho T} = (I + \rho T)^{-1}$ where I is the identity operator on Z; and

its *reflected resolvent* is $R_{\rho T} := 2J_{\rho T} - I$. For a monotone operator T, it can be shown that $R_{\rho T}$ is nonexpansive w.r.t. the Euclidean norm $\|\cdot\|$, namely, $\|R_{\rho T}(x) - R_{\rho T}(y)\| \leq \|x - y\|$ for all applicable x and y. Thus, $J_{\rho T}$ is an *averaged operator*, namely, $J_{\rho T} = \alpha I + (1-\alpha)S$ for some nonexpansive operator $S : Z \to Z$ and some $\alpha \in (0, 1)$ (in this particular case we have $S = R_{\rho T}$ and $\alpha = \frac{1}{2}$). Moreover, $J_{\rho T}$ and $R_{\rho T}$ have the same fixed point set, $\mathrm{Fix}(J_{\rho T}) = \mathrm{Fix}(R_{\rho T})$, which also coincides with the zero set of T, $\mathrm{zer}\, T := \{x \,|\, 0 \in T(x)\}$. If T is further maximally monotone, then $\mathrm{dom}\, J_{\rho T} = Z$. In this case the resolvent iteration $x^{k+1} = J_{\rho T}(x^k)$ is well defined for all k and will converge to a point $x^* \in \mathrm{zer}\, T$ if $\mathrm{zer}\, T \neq \emptyset$. This has important implications in optimization as the problem of finding a minimizer of a convex function f on Z is equivalent to finding a point in the zero set of its subdifferential operator ∂f, which is in turn equivalent to finding a fixed point of the resolvent $J_{\rho \partial f}$. It is well known that the resolvent $J_{\rho \partial f}$ for a CCP function f is given by its *proximal operator*:

$$\mathrm{prox}_{\rho f}(x) := \arg\min_z f(z) + \tfrac{1}{2\rho}\|z - x\|^2, \quad \forall x \in Z.$$

For instance, if $f = \mathbf{1}_C$ is the convex indicator function of a closed nonempty convex subset $C \subset Z$, $\mathrm{prox}_{\rho f}$ becomes the orthogonal projection operator Π_C onto C. If f has a minimizer, then the resolvent iteration of ∂f, $x^{k+1} = \mathrm{prox}_{\rho f}(x^k)$, will converge to a minimizer of f. This is the famed proximal point algorithm [35].

B. Saddle Functions

Let $K : X \times Y \to \overline{\mathbb{R}}$ be an extended-real-valued function defined on the product of two Euclidean spaces X and Y.

Definition 1 ([36]). *$K(x, y)$ is called a* saddle function *if it is a convex function of x for each fixed y and a concave function of y for each fixed x. The effective domain of K is defined as $\mathrm{dom}\, K := \{(x, y) \in X \times Y \,|\, K(x, y') < +\infty,\ \forall y' \in Y \text{ and } K(x', y) > -\infty,\ \forall x' \in X\}$. A saddle function K is* proper *if $\mathrm{dom}\, K \neq \emptyset$, and it is* closed *if $K(x, y)$ is lower semicontinuous in x for each fixed y and upper semicontinuous in y for each fixed x*[1].

Example 1.

(i) $K(x, y) = f(x) - g(y) + x^T A y$ is a saddle function where $f \in \overline{\mathbb{R}}$ and $g \in \mathbb{R}$ are convex functions and A is a matrix of proper dimension. It is closed and proper if both f and g are CCP.

(ii) For the optimization problem $\min_{x \in \mathbb{R}^n}\{f(x) \,|\, g(x) \leq 0\}$ where $f \in \mathbb{R}$ and $g \in \mathbb{R}^m$ are convex functions, the Lagrange function $L(x, y) = f(x) + \langle y, g(x)\rangle - \mathbf{1}_D(y)$ with the set $D = \{y \in \mathbb{R}^m \,|\, y \geq 0\}$ is a saddle function. It is closed and proper if both f and g are CCP functions.

(iii) Let $C \subset X$ and $D \subset Y$ be convex subsets. The function $\mathbf{1}_C(x) - \mathbf{1}_D(y)$ is not well defined on $C^c \times D^c$ where $C^c = X \setminus C$ and $D^c = Y \setminus D$ as $\infty - \infty$ is indefinite. We can forcibly set its values on $C^c \times D^c$ to $+\infty$ to define the following function:

$$\mu_{C \times D}(x, y) := \begin{cases} 0 & \text{if } x \in C \text{ and } y \in D \\ -\infty & \text{if } x \in C \text{ and } y \notin D \\ +\infty & \text{if } x \notin C, \end{cases} \tag{1}$$

which is a saddle function on $X \times Y$ with the domain $C \times D$. It is closed and proper if C and D are nonempty and closed sets. In the rest of the paper, we will follow the same rule as above when dealing with indefinite function values. For instance, we set $\mu_{C \times D} + \mu_{C' \times D'}$ to be $\mu_{(C \cap C') \times (D \cap D')}$ for subsets $C, C' \subset X$ and $D, D' \subset Y$.

A saddle function K has a *saddle point* at $(x^*, y^*) \in X \times Y$ if $K(x^*, y) \leq K(x^*, y^*) \leq K(x, y^*)$ for all $x \in X$, $y \in Y$, i.e., x^* is a minimizer of $K(\cdot, y^*)$ and y^* is a maximizer of $K(x^*, \cdot)$. In this case, we have $\sup_y \inf_x K(x, y) = \inf_x \sup_y K(x, y) = K(x^*, y^*)$. When K is the payoff function of a zero-sum two-player game, its saddle points are exactly the Nash equilibrium points. Existence of saddle points is not guaranteed for general saddle functions; however, a sufficient condition is given by the Sion's Minimax Theorem [26].

Proposition 1. *([26]) Let K be a real-valued closed saddle function on $X \times Y$ and $C \subset X$ and $D \subset Y$ be two nonempty compact convex subsets. Then $K + \mu_{C \times D}$ has a saddle point.*

Note that $K + \mu_{C \times D}$ is a saddle function that agrees with K on $C \times D$ and by definition (1) can only have saddle point on $C \times D$. Thus, by adding $\mu_{C \times D}$ to any saddle function, one can effectively enforce the constraints $x^* \in C$ and $y^* \in D$ for saddle points (x^*, y^*). In this sense, the μ-functions play a similar role as convex indicator functions do for converting constrained optimization problems into unconstrained ones.

C. Saddle Subdifferential Operators

For a saddle function K on $X \times Y$, a set-valued operator $T_K : X \times Y \to 2^{X \times Y}$ can be defined by

$$T_K(x, y) = \begin{bmatrix} \partial_x K(x, y) \\ \partial_y(-K)(x, y) \end{bmatrix}, \quad \forall (x, y) \in X \times Y.$$

Here, $\partial_x K(x, y)$ denotes the subdifferentials of the convex function $K(\cdot, y)$ at the point x; similarly for $\partial_y(-K)(x, y)$. T_K is called the *saddle subdifferential operator* of K and in particular the KKT operator when K is the Lagrangian of a constrained optimization problem [37]. T_K has the domain $\mathrm{dom}\, T_K := \{(x, y) \,|\, T_K(x, y) \neq \emptyset\}$ and the zero set $\mathrm{zer}\, T_K := \{(x, y) \,|\, 0 \in T_K(x, y)\}$. A point $(x^*, y^*) \in \mathrm{dom}\, T_K$ is a saddle point of K if and only if $0 \in \partial_x K(x^*, y^*)$ and $0 \in \partial_y(-K)(x^*, y^*)$, i.e., $(x^*, y^*) \in \mathrm{zer}\, T_K$. Thus, the set of saddle points of K is given exactly by $\mathrm{zer}\, T_K$.

Remark 1. *In general, for $(x, y) \in \mathrm{dom}\, T_K$, $(p, q) \in T_K(x, y)$ if and only if (x, y) is a saddle point of the saddle function $K(x, y) - \langle p, x\rangle + \langle q, y\rangle$ (see [36]).*

Proposition 2 ([36]). *Let K be a saddle function on $X \times Y$. If K is proper, then T_K is a monotone operator with the domain $\mathrm{dom}\, T_K \subset \mathrm{dom}\, K$. If K is proper and closed, then T_K is a maximally monotone operator.*

[1]The definition of closedness given here is stronger and easier to check than the one originally given in [36]

By the above result, for a closed proper saddle function K, T_K is maximally monotone. Hence, the resolvent of T_K for given $\rho > 0$, which we denote by $J_{\rho K}$, is an averaged operator defined everywhere on $X \times Y$. The fixed point set $\mathrm{Fix}(J_{\rho K})$ of $J_{\rho K}$, being the same as $\operatorname{zer} T_K$, is exactly the set of saddle points of K. Thus, finding a saddle point of K is equivalent to finding a fixed point of the averaged operator $J_{\rho K}$. Since repeated iterations using an averaged operator converges to one of its fixed points (if it exists), a saddle point of K can be computed via the iteration $(x^{k+1}, y^{k+1}) = J_{\rho K}(x^k, y^k)$. In the following, we list several explicit characterizations of $J_{\rho K}$. For any $(x, y) \in X \times Y$, $(p, q) = J_{\rho K}(x, y)$ if and only if $(x, y) \in (I + \rho T_K)(p, q)$, or equivalently,

$$\begin{cases} 0 \in \partial_p K(p,q) + (p-x)/\rho \\ 0 \in \partial_q(-K)(p,q) + (q-y)/\rho \end{cases} \tag{2a}$$

$$\Leftrightarrow \quad \begin{cases} p = \arg\min_{p \in X} \; K(p,q) + \frac{1}{2\rho}\|p-x\|^2 \\ q = \arg\max_{q \in Y} \; K(p,q) - \frac{1}{2\rho}\|q-y\|^2 \end{cases} . \tag{2b}$$

Yet another equivalent condition is that (p, q) is a saddle point of $K(p,q) + \frac{1}{2\rho}\left(\|p-x\|^2 - \|q-y\|^2\right)$.

Example 2.

(i) Let $K(x,y) = f(x) + \langle y, Ax - b\rangle$ be the Lagrange function for the optimization problem $\min\{f(x) \,|\, Ax = b\}$. Then, (2a) becomes $0 \in (p-x)/\rho + \partial f(p) + A^T q$ and $(q-y)/\rho = Ap - b$. These imply

$$\begin{cases} p &= \arg\min_z \mathcal{L}_\rho(z) + \frac{1}{2\rho}\|z - x\|^2 \\ q &= y + \rho(Ap - b). \end{cases}$$

Here, $\mathcal{L}_\rho(z) = f(z) + \langle y, Az - b\rangle + \frac{\rho}{2}\|Az - b\|^2$ is the augmented Lagrange function [37].

(ii) Consider the bilinear saddle function

$$K(x,y) = \frac{1}{2}\begin{bmatrix} x \\ y \end{bmatrix}^T \begin{bmatrix} \Sigma_1 & \Sigma_2 \\ \Sigma_2^T & -\Sigma_3 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}^T \begin{bmatrix} x \\ y \end{bmatrix} \tag{3}$$

where $\Sigma_1, \Sigma_2, \Sigma_3$ are matrices of proper dimensions with $\Sigma_1, \Sigma_3 \succeq 0$. Let $\Sigma := \begin{bmatrix} \Sigma_1 & \Sigma_2 \\ -\Sigma_2^T & \Sigma_3 \end{bmatrix}$. Then (2a) leads to

$$J_{\rho K}(x,y) = (I + \rho\Sigma)^{-1}\begin{bmatrix} x - \rho b_1 \\ y + \rho b_2 \end{bmatrix}. \tag{4}$$

If $\Sigma_1, \Sigma_3 \succ 0$, then $\Sigma^{-1}\begin{bmatrix} -b_1^T & b_2^T \end{bmatrix}^T$ is the unique fixed point of $J_{\rho K}$ and the unique saddle point of K.

(iii) For $K = \mu_{C \times D}$ in (1), $J_{\rho K}$ for any $\rho > 0$ is $\Pi_C \times \Pi_D$ where Π_C and Π_D are the orthogonal projections onto the closed convex sets C and D, respectively.

The proof of the following result is straightforward and hence omitted.

Proposition 3. *Suppose $K(x,y) = K_1(x_1, y_1) + \cdots + K_m(x_m, y_m)$. Here, $x = (x_1, \ldots, x_m) \in X = X_1 \times \cdots \times X_m$, $y = (y_1, \ldots, y_m) \in Y = Y_1 \times \cdots \times Y_m$, and $K_i(x_i, y_i)$ is a closed proper saddle function on $X_i \times Y_i$ for each i. Then, $(p, q) = J_{\rho K}(x, y)$ is given by $p = (p_1, \ldots, p_m)$ and $q = (q_1, \ldots, q_m)$ where $(p_i, q_i) = J_{\rho K_i}(x_i, y_i)$ for each i.*

III. Operator Splitting Methods

Suppose $K = K_1 + K_2$ is the sum of two closed and proper saddle functions K_1 and K_2 on $X \times Y$. Then K is also a closed and proper saddle function. Assume K has a saddle point (while K_1 and K_2 may have none). By Proposition 2, the saddle differential operators T_K, T_{K_1}, and T_{K_2} are maximally monotone, and their resolvents $J_{\rho K}$, $J_{\rho K_1}$, and $J_{\rho K_2}$ are averaged operators defined everywhere on $X \times Y$. In this section we focus on the case when $J_{\rho K_1}$ and $J_{\rho K_2}$ are much easier to compute than J_K, and look for iterative algorithms that compute a saddle point of K using $J_{\rho K_1}$ and $J_{\rho K_2}$.

Example 3.

(i) For $K(x,y) = f(x) - g(y) + y^T Ax$ we have $T_K = \begin{bmatrix} \partial f(x) + A^T y \\ \partial g(y) - Ax \end{bmatrix}$. Even if both f and g are differentiable, computing $J_{\rho K}$ using (2a) entails the solution of two coupled nonlinear equations. Write K as $K = K_1 + K_2$ with $K_1 = f(x) + y^T Ax$ and $K_2 = -g(y)$. Then, $J_{\rho K_1}$ can be computed as in Example 2 (i), while $J_{\rho K_2} = I \times \mathrm{prox}_{\rho g}$.

(iii) Let $K(x,y) = G_0(x) + \sum_{i=1}^m G_i(x_i, y)$ where $x = (x_1, \ldots, x_m) \in X = X_1 \times \cdots \times X_m$; $y \in Y$; G_0 is a real-valued CCP function on X; and $G_1, \ldots, G_m$ are real-valued closed proper saddle function on $X_i \times Y$. By introducing variables $y_1, \ldots, y_m$, K can be rewritten as $K = K_1 + K_2$ where $K_1 = \sum_{i=1}^m G_1(x_i, y_i)$ is separable and $K_2 = G_0(x) - \mathbf{1}_{\mathcal{A}}(y_1, \ldots, y_m)$ with $\mathcal{A} = \{(y_1, \ldots, y_m) \,|\, y_1 = \cdots = y_m\}$. Then, $J_{\rho K_1}$ can be computed by Proposition 3 and $J_{\rho K_2} = \mathrm{prox}_{\rho G_0} \times \Pi_{\mathcal{A}}$ where $\Pi_{\mathcal{A}}(y_1, \ldots, y_m) = (\bar{y}, \ldots, \bar{y})$ with $\bar{y} = \frac{1}{m}(y_1 + \cdots + y_m)$.

To find a saddle point of $K = K_1 + K_2$, we employ the Douglas-Rachford splitting [38], [39], which is a classical technique for finding a zero point of an operator $T = T_1 + T_2$ for two monotone operators T_1, T_2 on $Z = \mathbb{R}^n$. In the following we present a generalized version that will be needed in Section V. For this purpose, we first generalize the definition of the resolvent. Suppose $P = P^T \in \mathbb{R}^{n \times n}$ is positive definite. Denote by $\langle z, z' \rangle_P = z^T P z'$ and $\|z\|_P = \sqrt{z^T P z}$ the inner product and the norm on Z induced by P. For a set-valued operator $T : Z \to 2^Z$ define its generalized resolvent and reflected resolvent (preconditioned by P) as

$$J_T^P := (I + P^{-1}T)^{-1}, \quad R_T^P := 2J_T^P - I. \tag{5}$$

It can be easily verified that $J_T^P = (P+T)^{-1}P$. The canonical $J_{\rho T}$ and $R_{\rho T}$ defined in Section II-A are special instances with $P = \rho^{-1} I$. If T is maximally monotone, then both J_T^P and R_T^P have the domain Z; R_T^P is nonexpansive w.r.t. the norm $\|\cdot\|_P$; and thus J_T^P is a $(1/2)$-averaged operator w.r.t. $\|\cdot\|_P$. As a result, the generalized resolvent iteration $z^{k+1} = J_T^P(z^k)$ converges to a point in the set $\mathrm{Fix}(J_T^P) = \mathrm{Fix}(R_T^P) = \operatorname{zer} T$ provided that this set is nonempty. See [34] for further details.

Proposition 4 (Pre-conditioned Douglas-Rachford Splitting)**.** *Suppose $T_1, T_2 : Z \to 2^Z$ are maximally monotone operators on $Z = \mathbb{R}^n$ with $\operatorname{zer}(T_1 + T_2) \neq \emptyset$ and $P \in \mathbb{R}^{n \times n}$ is positive*

definite. Then $w^* \in \operatorname{zer}(T_1+T_2)$ *if and only if* $w^* = J^P_{T_2}(z^*)$ *for some* $z^* \in \operatorname{Fix}(R^P_{T_1} \circ R^P_{T_2})$. *Further, the iteration*

$$w^k = J^P_{T_2}(z^k) \tag{6a}$$
$$z^{k+1} = z^k + 2\alpha\left(J^P_{T_1}(2w^k - z^k) - w^k\right) \tag{6b}$$

starting from any z^0 *yields a sequence* w^k *that converges to some* $w^* \in \operatorname{zer}(T_1+T_2)$ *as* $k \to \infty$ *for any* $\alpha \in (0,1)$.

The proof of Proposition 4, a simple extension of the canonical proof, is included in Appendix A. In the rest of this section, we consider the classical case ($P = \rho^{-1} I$).

To find a saddle point of $K = K_1 + K_2$, i.e., a zero point of the operator $T_K = T_{K_1} + T_{K_2}$, since T_{K_1} and T_{K_2} are maximally monotone, we can apply (6a) to obtain:

$$(\tilde{x}^k, \tilde{y}^k) = J_{\rho K_2}(x^k, y^k) \tag{7a}$$
$$(x^{k+1}, y^{k+1}) = (x^k, y^k) + 2\alpha\big(J_{\rho K_1}(2\tilde{x}^k - x^k, 2\tilde{y}^k - y^k) - (\tilde{x}^k, \tilde{y}^k)\big) \tag{7b}$$

for $k = 0, 1, \ldots$. By Proposition 4, for $\alpha \in (0,1)$ and starting from any (x^0, y^0), the sequence $(\tilde{x}^k, \tilde{y}^k)$ will converge to a saddle point (x^*, y^*) of K. This algorithm will be the basis of our algorithm design in Sections IV and V. Note that in (7), the roles of K_1 and K_2 can be switched.

Example 4 (Sparse bilinear games)**.** Let $K(x,y) = x^T A y + b_1^T x + b_2^T y + \beta_1\|x\|_1 - \beta_2\|y\|_1$ where $\|\cdot\|_1$ is the L_1-norm and $\beta_1, \beta_2 > 0$. The L_1-regulation terms are added to promote sparsity of K's saddle points $w^* = (x^*, y^*)$. Write $K = K_1 + K_2$ where $K_1 = x^T A y + b_1^T x + b_2^T y$ and $K_2 = \beta_1\|x\|_1 - \beta_2\|y\|_1$. Then $J_{\rho K_1}$ can be computed as in Example 2 (ii) and $J_{\rho K_2} = \operatorname{prox}_{\beta_1\rho\|\cdot\|_1} \times \operatorname{prox}_{\beta_2\rho\|\cdot\|_1}$ where $\operatorname{prox}_{\eta\rho\|\cdot\|_1}$, $\eta \in \{\beta_1, \beta_2\}$, are given by the elementwise soft thresholding operator $(s_{\eta\rho}(v))_i := \max\{v_i - \eta\rho, 0\} - \max\{-v_i - \eta\rho, 0\}$ with i indexing the entries of v and $s_{\eta\rho}(v)$ (see [40]).

Consider the following numerical example: $A = \begin{bmatrix} 1 & 3 & 2 \\ 6 & 5 & 4 \\ 9 & 8 & 7 \end{bmatrix}$, $b_1 = \begin{bmatrix} -30 \\ -33 \\ -60 \end{bmatrix}$, and $b_2 = \begin{bmatrix} -117 \\ -126 \\ -45 \end{bmatrix}$. In this case K_1 has a unique saddle point $(x_1^*, y_1^*) = (30, 124, -73, -5, 7, 7)$. The saddle point of K computed by the algorithm in (7) is (x_1^*, y_1^*) for $\beta_1 = \beta_2 = 0$, $(0, 0, 5.75, 0, -2.5, 0)$ for $\beta_1 = \beta_2 = 80$, and $(0, 0)$ for $\beta_1 = \beta_2 = 130$. With larger β's, the sparsity of the computed saddle point increases.

IV. Convex-Concave Games on Networks

In this section, we formulate the convex-concave games on networks and propose distributed iterative algorithms to compute Nash equilibrium points. We first consider games with local but no global coupling constraints. The case with global constraints will be discussed in Section V.

A. Game Formulation

Consider a network consisting of m nodes indexed by $[m] := \{1, \ldots, m\}$. The local variable (x_i, y_i) of each node $i \in [m]$ consists of two parts: a local x-variable $x_i \in \mathbb{R}^{\check{n}_i}$ and a local y-variable $y_i \in \mathbb{R}^{\hat{n}_i}$ for some $\check{n}_i, \hat{n}_i \in \{0, 1, \ldots\}$. It is possible that $\check{n}_i = 0$ and/or $\hat{n}_i = 0$, in which case we write $x_i = \varnothing$ and/or $y_i = \varnothing$ to indicate the absence of the corresponding local variable. Denote by $[m]_x = \{i \in [m] \,|\, x_i \neq \varnothing\}$ and $[m]_y = \{i \in [m] \,|\, y_i \neq \varnothing\}$ the groups of nodes with non-empty local x- and y-variables, respectively. The game under study has a global payoff function $K(x,y)$ to be defined shortly and is played between two teams: one team (*x-team*) controls all the local x-variables from the group of nodes $[m]_x$, concatenated as $x = (x_i)_{i\in[m]_x} \in \mathbb{R}^{\check{n}}$, and tries to minimize K while the other team (*y-team*) controls all the local y-variables $y = (y_i)_{i\in[m]_y} \in \mathbb{R}^{\hat{n}}$ from the group of nodes $[m]_y$ and tries to maximize K.

The global payoff function K is defined by

$$K(x,y) = \textstyle\sum_{i\in[m]} K_i \tag{8}$$

where, for each $i \in [m]$, K_i is an extended-real-valued function called the local payoff function of node i. We assume that the local payoff functions are coupled: K_i depends on not only the local variable (x_i, y_i) of node i but also the variables of neighboring nodes, specifically, x_j for $j \in \check{\mathcal{N}}_i^+$ and y_l for $l \in \hat{\mathcal{N}}_i^+$. The two (possibly different) sets $\check{\mathcal{N}}_i^+, \hat{\mathcal{N}}_i^+ \subset [m]$ are called the x-in-neighbor and y-in-neighbor sets of node i, respectively. Thus, K_i is of the form $K_i(x_i, (x_j)_{j\in\check{\mathcal{N}}_i^+}, y_i, (y_l)_{l\in\hat{\mathcal{N}}_i^+})$. We further assume that each K_i is a saddle function, i.e., it is convex in $(x_i, (x_j)_{j\in\check{\mathcal{N}}_i^+})$ for each fixed $(y_i, (y_l)_{l\in\hat{\mathcal{N}}_i^+})$ and concave in $(y_i, (y_l)_{l\in\hat{\mathcal{N}}_i^+})$ for each fixed $(x_i, (x_j)_{j\in\check{\mathcal{N}}_i^+})$. Then, $K(x,y)$ is also a saddle function. Note that this formulation allows the existence of locally coupled constraints for each node i of the form

$$(x_i, (x_j)_{j\in\check{\mathcal{N}}_i^+}) \in D_i^x, \quad (y_i, (y_l)_{l\in\hat{\mathcal{N}}_i^+}) \in D_i^y \tag{9}$$

for nonempty closed convex sets D_i^x and D_i^y. In the rest of this paper we assume that constraints (9) are incorporated into the local payoff functions via the additive terms $\mu_{D_i^x \times D_i^y}$.

The dependency structure of the local payoff functions on the local variables can be represented by two directed graphs. The x-dependency graph $([m], \mathcal{E}_x)$ has the edge set $\mathcal{E}_x \subset [m] \times [m]$ so that an edge $(j,i) \in \mathcal{E}_x$ exists if and only if K_i depends on x_j. For node i, its x-in-neighbor set becomes $\check{\mathcal{N}}_i^+ = \{j | (j,i) \in \mathcal{E}_x\}$ and its x-out-neighbor set can be defined as $\check{\mathcal{N}}_i^- = \{j | (i,j) \in \mathcal{E}_x\}$. Similarly, we can define the y-dependency graph $([m], \mathcal{E}_y)$ and the y-out-neighbor set $\hat{\mathcal{N}}_i^-$ for each node i. As an example, in the game shown in the top of Fig. 1, node 1 has the x-in-neighbor set $\check{\mathcal{N}}_1^+ = \{2\}$, the x-out-neighbor set $\check{\mathcal{N}}_1^- = \{2\}$, the y-in-neighbor set $\hat{\mathcal{N}}_1^+ = \{3\}$, and the y-out-neighbor set $\hat{\mathcal{N}}_1^- = \emptyset$. Note that node 2 has no local y-variable and its local payoff function K_2 only depends on x-variables, both of which are allowed in our formulation.

We aim to solve the following problem in this section.

Problem 1. *Find a saddle point of* $K(x,y) = \sum_{i\in[m]} K_i$.

Problem 1 is equivalent to finding a Nash equilibrium point of a zero-sum game played by two teams controlling two separate sets of the local variables on a network. A special case is when each node has local variables from at most one team, i.e., $[m]_x \cap [m]_y = \emptyset$. Then the game is being played between two distinct groups of nodes, with friendly nodes in one group coordinating their decisions against the (also coordinated)

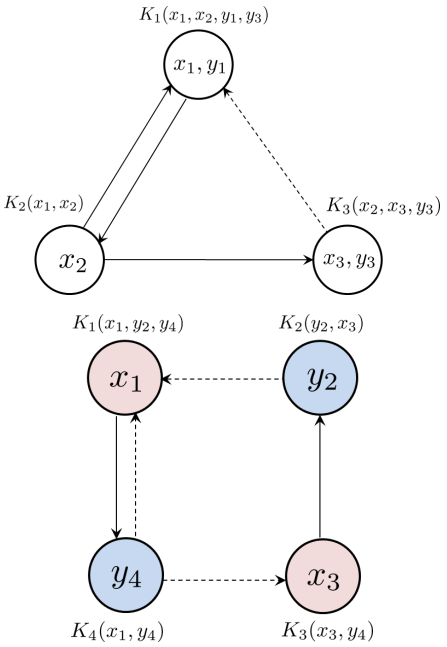


Fig. 1: Two examples of games on networks. Each circle represents an node, with its local variables marked inside the circle and its local payoff function marked next to the circle. Solid and dashed arrows represent the edges of the x-dependency graph and the y-dependency graph, respectively.

enemy nodes in the other group. The bottom game in Fig. 1 is one such example, where nodes belonging to the two groups are labeled with blue and red colors, respectively.

Example 5 (Networked Cournot Competition)**.** Consider two firms, Firm X and Firm Y, producing and supplying a single homogeneous good on multiple interconnected markets indexed by $[m]$. At each market i, let x_i and y_i be the amounts of good supplied by the two firms, and let the convex functions $f_i(x_i)$ and $g_i(y_i)$ be the corresponding local production costs of the two firms, respectively. The market clearing price p_i of the good in market i is given by $p_i(x_i + y_i + \sum_{j\in\check{\mathcal{N}}_i^+} \alpha_{ij}x_j + \sum_{l\in\hat{\mathcal{N}}_i^+} \beta_{il}y_l)$ for some $\alpha_{ij}, \beta_{il} \in [0, 1]$ and some convex decreasing function $p_i(\cdot)$. Note that the price depends on not only the total local supply $x_i + y_i$ in market i but also (to a lesser degree) the supply in neighboring markets due to local buyers' actual or perceived option of buying from neighboring markets. The total profits of the two firms in all the markets are thus $\sum_i(p_i x_i - f_i(x_i))$ and $\sum_i(p_i y_i - g_i(y_i))$, respectively. The global payoff function is the profit difference between the two firms: $K = \sum_{i\in[m]}[p_i(y_i - x_i) + f_i(x_i) - g_i(y_i)]$. Note that the game defined above is different from the ones studied in [30] and [14] due to the couplings of local clearing price with neighboring markets' supply and the distribution of decision variables from each team across the whole network.

Take the left of Figure 1 as an example. Assume $p_i(s) = \omega_i - \gamma_i s$ with $\omega_i = 10$ and $\gamma_i = 1$, $f_i(x_i) = x_i^2$, $g_i(y_i) = y_i^2$, and $\alpha_{ij} = \alpha_{il} = 0.5$ for all i and applicable j, l. Then $K_1 = (10 - x_1 - 0.5x_2 - y_1 - 0.5y_3)(y_1 - x_1) + x_1^2 - y_1^2$, $K_2 = (10 - 0.5x_1 - x_2)(-x_2) + x_2^2$, and $K_3 = (10 - 0.5x_2 - x_3 - y_3)(y_3 - x_3) + x_3^2 - y_3^2$. In this case, $K = K_1 + K_2 + K_3$ is of the form (3) and has a unique Nash equilibrium at $(x_1^*, x_2^*, x_3^*, y_1^*, y_2^*) = (0.82, 1.16, 1.10, 0.97, 1.09)$. At this equilibrium the local market clearing prices of the good at the three markets are $p_1^* = 2.08$, $p_2^* = 3.43$, $p_3^* = 2.23$. Not surprisingly p_2^* is the highest due to market 2 being captive to Firm X, while p_1^* is the lowest since market 1 is the most contested market. For Firm X, its total production is 3.09 and total profit is 4.91; for Firm Y, these two numbers are 2.06 and 2.32, respectively.

To ensure the existence of a solution to Problem 1, the following assumption is made.

Assumption 1. *Each K_i is closed and proper; and $K = \sum_{i\in[m]} K_i$ has at least one saddle point.*

Assumption 1 implies that the sets D_i^x and D_i^y in the local coupling constraints (9) must be closed. However, it does not require each local saddle function K_i to have saddle points.

Assumption 2 (Communicability)**.** *Two nodes i and j can communicate local variables with each other whenever $(i, j) \in \mathcal{E}_x \cup \mathcal{E}_y$ or $(j, i) \in \mathcal{E}_x \cup \mathcal{E}_y$. Further, the communicated information of each team is available to the other team.*

Assumption 2 means that two neighboring nodes in either the x- or y-dependency graph can exchange their local x- and y-variables via *bidirectional* communications. Using a simple two-node network with a single directed dependency edge, it is easy to see why bidirectional communications are needed for the convergence of any distributed algorithm. Also, the communications by both teams between neighboring nodes are assumed to be in pubic and available to the opposing teams. On the other hand, even for the same team, its decision makers at non-neighboring nodes cannot exchange information directly.

Our goal is to design a distributed iterative algorithm to solve Problem 1 so that each node uses only locally available information permitted by Assumption 2 for update at every iteration and that the collected variables of all nodes converge asymptotically to a saddle point (x^*, y^*) of K.

B. Problem Reformulation

For each node i, we introduce the local auxiliary variables $(x_{ij})_{j\in\check{\mathcal{N}}_i^+}$ and $(y_{il})_{l\in\hat{\mathcal{N}}_i^+}$ representing the copies held by node i for the x-variables of its x-in-neighbors and the y-variables of its y-in-neighbors, respectively. Together with (x_i, y_i), these are exactly the variables that the local payoff function K_i depends on. Denote by $\mathbf{x}_i := (x_i, (x_{ij})_{j\in\check{\mathcal{N}}_i^+}) \in \mathbb{R}^{\check{N}_i}$ and $\mathbf{y}_i := (y_i, (y_{il})_{l\in\hat{\mathcal{N}}_i^+}) \in \mathbb{R}^{\hat{N}_i}$ the augmented x-variable and y-variable of node i, and by $\mathbf{x} := (\mathbf{x}_i)_{i\in[m]} \in \mathbb{R}^{\check{N}}$ and $\mathbf{y} := (\mathbf{y}_i)_{i\in[m]} \in \mathbb{R}^{\hat{N}}$ their concatenations. Define the x-consensus and y-consensus subspaces for $\mathbf{x}$ and $\mathbf{y}$ as

$$\begin{aligned}\mathcal{A}_x &:= \{\mathbf{x} \,|\, x_i = x_{ji},\ \forall i \in [m],\ j \in \check{\mathcal{N}}_i^-\},\\ \mathcal{A}_y &:= \{\mathbf{y} \,|\, y_i = y_{li},\ \forall i \in [m],\ l \in \hat{\mathcal{N}}_i^-\}.\end{aligned} \tag{10}$$

For each $(\mathbf{x}, \mathbf{y}) \in \mathcal{A} := \mathcal{A}_x \times \mathcal{A}_y$, since all the auxiliary variables are faithful copies, the saddle function

$$K_a(\mathbf{x}, \mathbf{y}) := \textstyle\sum_{i\in[m]} K_i(\mathbf{x}_i, \mathbf{y}_i) \tag{11}$$

has the true value of the global payoff function $K(x,y)$ where (x,y) is obtained from $(\mathbf{x},\mathbf{y})$ by pruning all the auxiliary variables. Thus, Problem 1 is equivalent to the following.

Problem 2. *Find a saddle point* $(\mathbf{x}^*,\mathbf{y}^*)$ *of* $K_a(\mathbf{x},\mathbf{y}) + \mu_{\mathcal{A}_x\times\mathcal{A}_y}(\mathbf{x},\mathbf{y})$.

By Assumption 1, the saddle function K_a is closed and proper; and a solution to Problem 2 exists. In the following, we focus on solving Problem 2 instead of Problem 1 since, unlike $K(x,y)$, $K_a(\mathbf{x},\mathbf{y})$ is separable; hence the resolvent $J_{\rho K_a}$ of its saddle differential operator T_{K_a} can be computed in parallel by individual nodes according to Proposition 3. Further, $J_{\rho\mu_{\mathcal{A}_x\times\mathcal{A}_y}} = \Pi_{\mathcal{A}_x}\times\Pi_{\mathcal{A}_y}$. Here, the orthogonal projection operator $\Pi_{\mathcal{A}_x}$ has the explicit form $\Pi_{\mathcal{A}_x}(\mathbf{x}) = (\bar{\mathbf{x}}_i)_{i\in[m]}$ where each $\bar{\mathbf{x}}_i = (\bar{x}_i, (\bar{x}_{ij})_{j\in\check{\mathcal{N}}_i^+})$ is given by

$$\bar{x}_i = \tfrac{1}{1+|\check{\mathcal{N}}_i^-|}(x_i + \textstyle\sum_{j\in\check{\mathcal{N}}_i^-} x_{ji}),\ \ \bar{x}_{ij} = \bar{x}_j,\ \forall j\in\check{\mathcal{N}}_i^+. \quad (12)$$

Similarly, $\Pi_{\mathcal{A}_y}(\mathbf{y}) = (\bar{\mathbf{y}}_i)_{i\in[m]}$ with each $\bar{\mathbf{y}}_i = (\bar{y}_i, (\bar{y}_{il})_{l\in\hat{\mathcal{N}}_i^+})$ given by

$$\bar{y}_i = \tfrac{1}{1+|\hat{\mathcal{N}}_i^-|}(y_i + \textstyle\sum_{l\in\hat{\mathcal{N}}_i^-} y_{li}),\ \ \bar{y}_{il} = \bar{y}_l,\ \forall l\in\hat{\mathcal{N}}_i^+. \quad (13)$$

C. Distributed Solution Algorithm

Applying the algorithm (7) to the splitting $K_a + \mu_{\mathcal{A}_x\times\mathcal{A}_y}$, we obtain

$$\bar{\mathbf{x}}^k = \Pi_{\mathcal{A}_x}(\mathbf{x}^k),\ \ \bar{\mathbf{y}}^k = \Pi_{\mathcal{A}_y}(\mathbf{y}^k); \quad (14a)$$

$$(\mathbf{x}_i^{k+1},\mathbf{y}_i^{k+1}) = (\mathbf{x}_i^k,\mathbf{y}_i^k) + 2\alpha\Big(J_{\rho K_i}(2\bar{\mathbf{x}}_i^k - \mathbf{x}_i^k, 2\bar{\mathbf{y}}_i^k - \mathbf{y}_i^k) - (\bar{\mathbf{x}}_i^k,\bar{\mathbf{y}}_i^k)\Big),\ \ i\in[m]. \quad (14b)$$

The overall algorithm is summarized in Algorithm 1 below. Each iteration of the algorithm consists of two stages. The first stage (14a) requires two synchronous rounds of communications between neighboring nodes: each node i first collects variables x_{ji}'s from its x-out-neighbors $j\in\check{\mathcal{N}}_i^-$ and y_{li}'s from its y-out-neighbors $l\in\hat{\mathcal{N}}_i^-$; it then computes $\bar{x}_i$ by (12) and $\bar{y}_i$ by (13); finally it sends $\bar{x}_i$ and $\bar{y}_i$ to its out-neighbors as the updated values of x_{ji}'s and y_{li}'s. The second stage (14b) requires no inter-node communications.

Algorithm 1 Synchronous Algorithm

1: Initialize $\mathbf{x}^0$ and $\mathbf{y}^0$, and let $k\leftarrow 0$;
2: **repeat**
3: $\quad\bar{\mathbf{x}}^k\leftarrow\Pi_{\mathcal{A}_x}(\mathbf{x}^k)$; $\bar{\mathbf{y}}^k\leftarrow\Pi_{\mathcal{A}_y}(\mathbf{x}^k)$;
4: $\quad$**for** $i=1,\ldots,m$ **do**
5: $\quad\quad(\mathbf{x}_i^{k+1},\mathbf{y}_i^{k+1}) \leftarrow (\mathbf{x}_i^k - 2\alpha\bar{\mathbf{x}}_i^k, \mathbf{y}_i^k - 2\alpha\bar{\mathbf{y}}_i^k) + 2\alpha J_{\rho K_i}(2\bar{\mathbf{x}}_i^k - \mathbf{x}_i^k, 2\bar{\mathbf{y}}_i^k - \mathbf{y}_i^k)$;
6: $\quad k\leftarrow k+1$;
7: **until** $|(\mathbf{x}^k,\mathbf{y}^k) - (\mathbf{x}^{k-1},\mathbf{y}^{k-1})|$ is sufficiently small
8: **return** $(\bar{\mathbf{x}}^k,\bar{\mathbf{y}}^k)$

The following result follows directly from the discussions in Section IV-B and Proposition 4.

Theorem 1. *Under Assumptions 1 and 2 and with* $\alpha\in(0,1)$, $\rho>0$, *the sequence* $(\bar{\mathbf{x}}^k,\bar{\mathbf{y}}^k)$ *generated by Algorithm 1 starting from any initial* $\mathbf{x}^0$ *and* $\mathbf{y}^0$ *converges as* $k\to\infty$ *to a saddle point* $(\mathbf{x}^*,\mathbf{y}^*)$ *of* $K_a + \mu_{\mathcal{A}_x\times\mathcal{A}_y}$, *which corresponds to a saddle point* (x^*,y^*) *of* K.

Algorithm convergence is guaranteed for any constant (implicit) step size $\rho>0$ and with no further assumptions on payoff functions such as differentiability, strong monotonicity and Lipschitzness of its gradients. The locally coupling constraints (9) can be specified by arbitrary closed convex sets.

D. Randomized Implementation

Algorithm 1 can be implemented via randomization as a way to combat practical issues such as lack of synchronous clocks among nodes and heterogeneity in node capacity. We first introduce a relevant result.

Proposition 5 ([41]). *Let* $T:\mathbb{R}^n\to\mathbb{R}^n$ *be an averaged operator with* $\mathrm{Fix}(T)\neq\emptyset$. *Partition* $z\in\mathbb{R}^n$ *into* $(z_1,\ldots,z_m)$ *and* Tz *into* $(T_1z,\ldots,T_mz)$ *where* $z_i, T_iz\in\mathbb{R}^{n_i}$ *for* $i\in[m]$. *Consider the following iteration. At each step* $k=0,1,\ldots$, *first an index* $i^k\in[m]$ *is chosen randomly and independently with the probabilities* $\mathbb{P}\left(i^k=i\right)=p_i>0$; *then* z^k *is updated to* z^{k+1} *where* $z_{i^k}^{k+1} = T_{i^k}z^k$ *and* $z_l^{k+1} = z_l^k$ *for* $l\neq i^k$. *Then,* z^k *converges almost surely to some* $z^*\in\mathrm{Fix}(T)$ *as* $k\to\infty$.

Recall that in (14) the iteration from $(\mathbf{x}^k,\mathbf{y}^k)$ to $(\mathbf{x}^{k+1},\mathbf{y}^{k+1})$ is via an averaged operator (see the proof of Proposition 4). Applying Proposition 5 to this results in the following algorithm: at round k, one node $i^k = i\in[m]$ is activated with the i.i.d. probability $p_i>0$; it first computes $(\bar{\mathbf{x}}_i^k,\bar{\mathbf{y}}_i^k)$; then updates $(\mathbf{x}_i^{k+1},\mathbf{y}_i^{k+1}) = (\mathbf{x}_i^k,\mathbf{y}_i^k) + 2\alpha\Big(J_{\rho K_i}(2\bar{\mathbf{x}}_i^k - \mathbf{x}_i^k, 2\bar{\mathbf{y}}_i^k - \mathbf{y}_i^k) - (\bar{\mathbf{x}}_i^k,\bar{\mathbf{y}}_i^k)\Big)$. Note that to compute $(\bar{\mathbf{x}}_i^k,\bar{\mathbf{y}}_i^k) = (\bar{x}_i^k, (\bar{x}_j^k)_{j\in\check{\mathcal{N}}_i^+}, \bar{y}_i^k, (\bar{y}_l^k)_{l\in\hat{\mathcal{N}}_i^+})$, node i needs to collect $\bar{x}_j^k$ from its x-in-neighbors $j\in\check{\mathcal{N}}_i^+$ and $\bar{y}_l^k$ from its y-in-neighbors $l\in\hat{\mathcal{N}}_i^+$, whose values in turn require these in-neighbors to communicate bidirectionally with their own out-neighbors. To avoid such two-hop communications, we let every node $i\in[m]$ hold two extra variables: $\bar{x}_i$, the latest average of x_i and x_{ji} for $j\in\check{\mathcal{N}}_i^-$, and $\bar{y}_i$, the latest average of y_i and y_{li} for $l\in\hat{\mathcal{N}}_i^-$. After the randomly activated node has updated its local variables, the affected extra variables will also be updated to the latest averaged values. With this modification, a randomized version of Algorithm 1 is summarized in Algorithm 2. In each round, the activated node i communicates bidirectionally with its x- and y-in-neighbors in step 7 to collect the latest averages $\bar{x}_j^k$ and $\bar{y}_l^k$; and in steps 11-14 to send back the differences $x_{ij}^{k+1} - x_{ij}^k$, $y_{il}^{k+1} - y_{il}^k$. No two-hop communications are needed.

By Propositions 4 and 5, we have the following result.

Corollary 1. *Suppose Assumptions 1 and 2 hold and* $\alpha\in(0,1)$, $\rho>0$, $p_i>0$, $\forall i\in[m]$. *Starting from any initial* $\mathbf{x}^0$ *and* $\mathbf{y}^0$, *the sequence* $(\bar{x}^k,\bar{y}^k)$ *returned by Algorithm 2 converges with probability one to a saddle point of* K.

E. Numerical Example

Consider an example on a 7-node network as shown in Fig. 2. The network nodes are partitioned into two groups:

Algorithm 2 Randomized Algorithm

Choose any $\mathbf{x}^0$ and $\mathbf{y}^0$, and let $k \leftarrow 0$
for $i = 1, \ldots, m$ **do**
  $\bar{x}_i^0 \leftarrow (x_i^0 + \sum_{j \in \check{\mathcal{N}}_i^-} x_{ji}^0)/(1 + |\check{\mathcal{N}}_i^-|)$;
  $\bar{y}_i^0 \leftarrow (y_i^0 + \sum_{l \in \hat{\mathcal{N}}_i^-} y_{li}^0)/(1 + |\hat{\mathcal{N}}_i^-|)$;
repeat
  Pick $i \in [m]$ with i.i.d. probability $p_i > 0$
  $\bar{\mathbf{x}}_i^k \leftarrow (\bar{x}_i^k, (\bar{x}_j^k)_{j \in \check{\mathcal{N}}_i^+})$; $\bar{\mathbf{y}}_i^k \leftarrow (\bar{y}_i^k, (\bar{y}_l^k)_{l \in \hat{\mathcal{N}}_i^+})$;
  $(\mathbf{x}_i^{k+1}, \mathbf{y}_i^{k+1}) \leftarrow (\mathbf{x}_i^k - 2\alpha \bar{\mathbf{x}}_i^k, \mathbf{y}_i^k - 2\alpha \bar{\mathbf{y}}_i^k) + 2\alpha J_{\rho K_i}(2\bar{\mathbf{x}}_i^k - \mathbf{x}_i^k, 2\bar{\mathbf{y}}_i^k - \mathbf{y}_i^k)$;
  $\bar{x}_i^{k+1} \leftarrow \bar{x}_i^k + (x_i^{k+1} - x_i^k)/(|\check{\mathcal{N}}_i^-| + 1)$;
  $\bar{y}_i^{k+1} \leftarrow \bar{y}_i^k + (y_i^{k+1} - y_i^k)/(|\hat{\mathcal{N}}_i^-| + 1)$;
  for $j \in \check{\mathcal{N}}_i^+$ **do**
    $\bar{x}_j^{k+1} \leftarrow \bar{x}_j^k + (x_{ij}^{k+1} - x_{ij}^k)/(|\check{\mathcal{N}}_j^-| + 1)$;
  for $l \in \hat{\mathcal{N}}_i^+$ **do**
    $\bar{y}_l^{k+1} \leftarrow \bar{y}_l^k + (y_{il}^{k+1} - y_{il}^k)/(|\hat{\mathcal{N}}_l^-| + 1)$;
  $k \leftarrow k + 1$;
until k is sufficiently large
return $\bar{x}^k = (\bar{x}_i^k)_{i \in [m]}$ and $\bar{y}^k = (\bar{y}_i^k)_{i \in [m]}$

$\{1, 2, 3, 4, 5\}$ and $\{6, 7\}$. Nodes in the first group have only local x-variables while nodes in the second group have only local y-variables. The local payoff function K_i of each node i depends on the variables of its in-neighbors (as depicted by the incoming arrows in Fig. 2) and is assumed to be of the bilinear form in Example 2 (ii) with randomly generated parameters. It is ensured that K_i's associated with nodes in the x-group are strictly convex in the x-variables and affine in the y-variables while K_i's belonging to nodes in the y-group are strictly concave in the y-variables and affine in the x-variables; so each K_i has no saddle points but the sum $K = \sum_{i=1}^7 K_i$ has a unique saddle point (x^*, y^*). We first apply Algorithm 1 to solve this game with four sets of parameters, $(\rho, \alpha) = (0.01, 0.5), (1, 0.5), (1, 0.98), (100, 0.5)$, respectively. As shown in Fig. 2, starting from a randomly generated initial point $\mathbf{z}^0$, the iteration result $(\bar{x}^k, \bar{y}^k)$ extracted from $\bar{\mathbf{z}}^k$ computed in Step 3 of Algorithm 1 converges to the unique saddle point (x^*, y^*) at linear convergence rates in all cases. For this example, a moderate value of the parameter $\rho = 1$ appears to result in the fastest convergence. Algorithm 2 is then applied to solve the same problem with equal node activation probabilities $p_1 = \cdots = p_1 = \frac{1}{7}$. Compared to Algorithm 1 using the same parameters $(\rho, \alpha) = (1, 0.5)$, Algorithm 2 requires more iterations to achieve the same convergence accuracy (10^{-8}) as at each round only one node updates as compared to all seven nodes in Algorithm 1.

V. Convex-Concave Games on Networks with Global Constraints

The games studied in Problem 1 in the previous section allow locally coupled constraints of the form (9) involving decision variables from nodes in the same neighborhood. In many applications there are also global constraints involving variables from all of the nodes, e.g., limits on the total available resources or actuation capacity. In this section, we

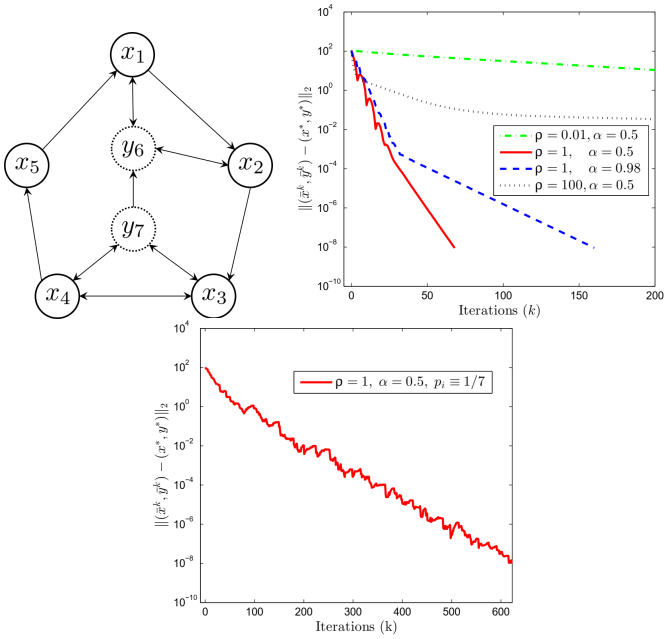


Fig. 2: Bilinear game on a 7-node network. Top left: dependence graphs; Top right: convergence of Algorithm 1 with different parameters (ρ, α); Bottom: Convergence of Algorithm 2 with $\rho = 1$, $\alpha = 0.5$, and $p_i = \frac{1}{7}$, $\forall i$.

study the convex-concave games on networks with the same global payoff function $K(x, y) = \sum_i K_i$ as defined in (8), and with the following additional global constraints[2]:

$$x \in \mathcal{F} := \{x \mid \textstyle\sum_{i \in [m]} F_i x_i = c_0\},$$
$$y \in \mathcal{G} := \{y \mid \textstyle\sum_{i \in [m]} G_i y_i = d_0\}. \tag{15}$$

Here, $F_i \in \mathbb{R}^{n_f \times \check{n}_i}$, $c_0 \in \mathbb{R}^{n_f}$, $G_i \in \mathbb{R}^{n_g \times \hat{n}_i}$, and $d_0 \in \mathbb{R}^{n_g}$ are given constant matrices and vectors. By letting $F = \begin{bmatrix} F_1 & \cdots & F_m \end{bmatrix}$ and $G = \begin{bmatrix} G_1 & \cdots & G_m \end{bmatrix}$, the two global constraints can be written compactly as $Fx = c_0$ and $Gy = d_0$. For the algorithm design, it is often convenient to distribute the global constraints to individual nodes. Write $c_0 = c_1 + \cdots + c_m$ and $d_0 = d_1 + \cdots + d_m$ for some $c_i \in \mathbb{R}^{n_f}$, $d_i \in \mathbb{R}^{n_g}$, $i \in [m]$. Then the global constraints become $\sum_i (F_i x_i - c_i) = 0$ and $\sum_i (G_i y_i - d_i) = 0$. We emphasize that node i does not need to satisfy the constraints $F_i x_i - c_i = 0$ or $G_i y_i - d_i = 0$.

Assumption 3. *The data (F_i, c_i) and (G_i, d_i) are private to node i and not shared with decision makers at other nodes, even those from the same team.*

Thus, each node only knows a portion of the global constraints and keeps that information from other nodes. In place of Assumption 1, the following is assumed in this section.

Assumption 4. *Each K_i is closed and proper; and $K + \mu_{\mathcal{F} \times \mathcal{G}}$ has at least one saddle point.*

The problem to be studied in this section is defined below.

Problem 3. *Find a saddle point of $K + \mu_{\mathcal{F} \times \mathcal{G}}$.*

[2]More general global constraints will be discussed in Section V-C

Example 6. *Consider the convex-concave game with the same bilinear payoff function as in* (3) *and with the additional global constraints* $Fx = c_0$ *and* $Gy = d_0$. *In this case, the Nash equilibrium* (x^*, y^*) *can be computed from*

$$\begin{bmatrix} x^* \\ y^* \\ p^* \\ q^* \end{bmatrix} = \begin{bmatrix} \Sigma_1 & \Sigma_2 & -F^T & 0 \\ -\Sigma_2^T & \Sigma_3 & 0 & G^T \\ F & 0 & 0 & 0 \\ 0 & G & 0 & 0 \end{bmatrix}^{-1} \begin{bmatrix} -b_1 \\ b_2 \\ c_0 \\ d_0 \end{bmatrix}, \tag{16}$$

where p^* *and* q^* *are the optimal values of the dual variables w.r.t. the two global constraints.*

A. Distributed Algorithms with Coordinator

A simple solution to Problem 3 is to introduce an additional node called the coordinator node (with the index 0) that has no local variables and a local payoff function $K_0 = \mu_{\mathcal{F}\times\mathcal{G}}$. This new node is an x-out-neighbor as well as a y-out-neighbor of every other node. Then $K + \mu_{\mathcal{F}\times\mathcal{G}} = \sum_{i=0}^m K_i$ and Problem 3 is reduced to Problem 1. The algorithm (14) developed in Section IV can be applied after proper modifications to accommodate the new node. In each iteration of the algorithm, the coordinator node needs to collect the local variables from all the other nodes and evaluates $J_{\rho K_0} = \Pi_{\mathcal{F}} \times \Pi_{\mathcal{G}}$ where $\Pi_{\mathcal{F}}(\mathbf{x}) = \mathbf{x} - F^\dagger(F\mathbf{x} - c_0)$ and $\Pi_{\mathcal{G}}(\mathbf{y}) = \mathbf{y} - G^\dagger(G\mathbf{y} - d_0)$ are the orthogonal projections onto the affine subspaces $\mathcal{F}$ and $\mathcal{G}$. The matrices $F^\dagger$ and $G^\dagger$ require private data F_i and G_i from all the other nodes and could have large dimension.

We next present a different distributed algorithm also with a coordinator node that has lower computation burden for the coordinator and better privacy preservation for the other nodes. To this purpose, introduce dual variables $\xi_0 \in \mathbb{R}^{n_g}$ and $\eta_0 \in \mathbb{R}^{n_f}$ for the two global constraints and define

$$K_b(x, \xi_0, y, \eta_0) = K(x, y) + \eta_0^T(Fx - c_0) + \xi_0^T(Gy - d_0). \tag{17}$$

It is easy to see that K_b is a saddle function in the pair of variables (x, ξ_0) and (y, η_0).

Lemma 1. (x^*, y^*) *is a saddle point of* $K + \mu_{\mathcal{F}\times\mathcal{G}}$ *if and only if* $(x^*, \xi_0^*, y^*, \eta_0^*)$ *is a saddle point of* K_b *for some* ξ_0^* *and* η_0^*.

Proof. For the "if" part, note that $(x^*, \xi_0^*, y^*, \eta_0^*)$ being a saddle point of K_b is equivalent to

$$\begin{aligned} &0 \in \partial_x K(x^*, y^*) + F^T\eta_0^*,\ 0 \in \partial_y(-K)(x^*, y^*) - G^T\xi_0^*, \\ &Gy^* - d_0 = 0, \quad Fx^* - c_0 = 0. \end{aligned} \tag{18}$$

The last two conditions imply that $(x^*, y^*) \in \mathcal{F} \times \mathcal{G}$. By (1), this further implies that $\partial_x \mu_{\mathcal{F}\times\mathcal{G}}(x^*, y^*) = \partial \mathbf{1}_{\mathcal{F}}(x^*)$ and $\partial_y(-\mu_{\mathcal{F}\times\mathcal{G}})(x^*, y^*) = \partial \mathbf{1}_{\mathcal{G}}(y^*)$. Recall that $\partial \mathbf{1}_{\mathcal{F}}(x^*)$ is the normal cone of the set $\mathcal{F}$ at x^* ([34]). As $\mathcal{F} = \{x \,|\, Fx = c_0\}$ is an affine subspace, its normal cone at x^* is exactly the range space of F^T. We then have $F^T\eta_0^* \in \partial_x \mu_{\mathcal{F}\times\mathcal{G}}(x^*, y^*)$. Thus the first condition in (18) can be written as $0 \in \partial_x K(x^*, y^*) + \partial_x \mu_{\mathcal{F}\times\mathcal{G}}(x^*, y^*)$. Similarly, the second condition in (18) results in $0 \in \partial_y(-K)(x^*, y^*) + \partial_y(-\mu_{\mathcal{F}\times\mathcal{G}})(x^*, y^*)$. These two facts show that (x^*, y^*) is a saddle point of $K + \mu_{\mathcal{F}\times\mathcal{G}}$. Proof of the "only if" part is similar and hence omitted. □

Thus, Problem 3 is equivalent to finding a saddle point of K_b, which exists by Assumption 4. In the game corresponding to K_b, in addition to the variables x_i's and y_i's, each of the two teams has an additional decision variable ξ_0 or η_0 for enforcing the global constraint on the other team. Using the decomposition $c_0 = c_1 + \cdots + c_m$ and $d_0 = d_1 + \cdots + d_m$, we can rewrite K_b as

$$\begin{aligned} K_b(x, \xi_0, y, \eta_0) &= \sum_{i\in[m]} L_i \\ &:= \sum_{i\in[m]} \left[K_i + \eta_0^T(F_i x_i - c_i) + \xi_0^T(G_i y_i - d_i)\right]. \end{aligned}$$

We designate (ξ_0, η_0) to be the local decision variables of the coordinator node 0, which has a trivial payoff function $L_0 = 0$. The payoff function of each node $i \in [m]$ is L_i as defined above. In addition to the original dependency relations corresponding to K_i's, the coordinator node is an x-in-neighbor as well as a y-in-neighbor of every other node. For the augmented variables, we have $\mathbf{x}_0 = \xi_0$ and $\mathbf{y}_0 = \eta_0$ for the coordinator node, and $\mathbf{x}_i = (x_i, (x_{ij})_{j\in\check{\mathcal{N}}_i^+}, \xi_{i0})$ and $\mathbf{y}_i = (y_i, (y_{ij})_{j\in\hat{\mathcal{N}}_i^+}, \eta_{i0})$ for node $i \in [m]$. The consensus subspace $\mathcal{A}_x = \{\mathbf{x} \,|\, \xi_{i0} = \xi_0,\ x_i = x_{ji},\ \forall i \in [m], j \in \check{\mathcal{N}}_i^-\}$ with $\bar{\mathbf{x}} = \Pi_{\mathcal{A}_x}(\mathbf{x})$ such that $\bar{\xi}_0 = \bar{\xi}_{i0} = (\xi_0 + \sum_{i\in[m]} \xi_{i0})/(m+1)$ and $\bar{x}_i$ and $\bar{x}_{ij}$ are as defined in (12). Similarly we can define $\mathcal{A}_y$ and $\Pi_{\mathcal{A}_y}$. Algorithm (14) then becomes

$$\bar{\mathbf{x}}^k = \Pi_{\mathcal{A}_x}(\mathbf{x}^k),\ \ \bar{\mathbf{y}}^k = \Pi_{\mathcal{A}_y}(\mathbf{y}^k), \tag{19a}$$

$$\begin{aligned} (\mathbf{x}_i^{k+1}, \mathbf{y}_i^{k+1}) &= (\mathbf{x}_i^k - 2\alpha\bar{\mathbf{x}}_i^k, \mathbf{y}_i^k - 2\alpha\bar{\mathbf{y}}_i^k) \\ &\quad + 2\alpha J_{\rho L_i}(2\bar{\mathbf{x}}_i^k - \mathbf{x}_i^k, 2\bar{\mathbf{y}}_i^k - \mathbf{y}_i^k),\ \ i \in [m], \end{aligned} \tag{19b}$$

$$(\xi_0^{k+1}, \eta_0^{k+1}) = (1 - 2\alpha)(\xi_0^k, \eta_0^k) + 2\alpha(\bar{\xi}_0^k, \bar{\eta}_0^k). \tag{19c}$$

In the last step we have used the fact that $J_{\rho L_0}$ is the identity operator. The computation of $J_{\rho L_i}$ for $i \in [m]$ is different from that of $J_{\rho K_i}$ but of comparable complexity.

The following theorem follows directly from Proposition 4.

Theorem 2. *Suppose Assumptions 2, 3 and 4 hold, and that the coordinator node can communicate with any other node* $i \in [m]$ *bidirectionally. Let* $\alpha \in (0, 1)$ *and* $\rho > 0$. *Starting from any initial* $(\mathbf{x}^0, \mathbf{y}^0)$, $(\bar{\mathbf{x}}^k, \bar{\mathbf{y}}^k)$ *obtained from the algorithm in* (19) *converges asymptotically to some* $(\mathbf{x}^*, \mathbf{y}^*)$ *for which the corresponding* (x^*, y^*) *is a solution to Problem 3.*

In the algorithm (19) the coordinator node only collects copies of the dual variables from other nodes and none of their private data; and the computation load of the coordinator node is minimum. On the other hand, the need of a coordinator node increases the system's vulnerability to single-point failures and attacks. In the next subsection, we will develop distributed solution algorithms without a coordinator node.

B. Distributed Algorithms without Coordinators

Note that in the algorithm in (19) the role of the coordinator node is to ensure that copies of the dual variables ξ_0 and η_0 kept by the other nodes are in consensus. This consensus can be achieved without a coordinator via information exchanges between neighboring nodes on a connected graph.

Assumption 5. *There is a connected undirected graph $([m], \mathcal{E}_c)$ with no self-loops so that node pairs (i, j) in the edge set $\mathcal{E}_c$ can communicate with each other bidirectionally.*

We call $([m], \mathcal{E}_c)$ the *consensus graph* and denote by $\mathcal{N}_i = \{j \,|\, (i, j) \in \mathcal{E}_c\}$ the set of neighbors of node i in this graph. Associate each edge (i, j) in $\mathcal{E}_c$ with a positive weight $w_{ij} = w_{ji} > 0$ and let $W = W^T \in \mathbb{R}^{m\times m}$ be the corresponding graph Laplacian matrix: $W_{ij} = -w_{ij}$ for $(i, j) \in \mathcal{E}_c$; $W_{ij} = 0$ for $(i, j) \notin \mathcal{E}_c$ and $i \neq j$; and $W_{ii} = \sum_{j\neq i} w_{ij}$ for $i \in [m]$. Since $([m], \mathcal{E}_c)$ is connected, W has a simple eigenvalue at 0 with the eigenvector $\mathbf{1}_m$ with all entry ones.

Remark 2. *The consensus graph can be different from the dependency graphs: it is undirected as opposed to directed; and it needs to be connected while the dependency graphs are not necessarily so. On the other hand, one can always choose $\mathcal{E}_c$ to include as many edges in $\mathcal{E}_x \cup \mathcal{E}_y$ as possible to take advantage of the communication capability in Assumption 2.*

Let $\xi = (\xi_i)_{i\in[m]} \in \mathbb{R}^{mn_g}$ and $\eta = (\eta_i)_{i\in[m]} \in \mathbb{R}^{mn_f}$ be the vectors concatenating the local copies ξ_i and η_i kept by nodes $i \in [m]$ for the dual variables $\xi_0 \in \mathbb{R}^{n_g}$ and $\eta_0 \in \mathbb{R}^{n_f}$. By our assumption on W, a sufficient and necessary condition for ξ_i's to be in consensus (i.e., $\xi = \mathbf{1}_m \otimes \xi_0$ for some ξ_0) is $(W \otimes I_{n_g})\xi = 0$. Similarly, η_i's are in consensus if and only if $(W \otimes I_{n_f})\eta = 0$. We can incorporate these two consensus constraints into the saddle function via the new dual variables $u = (u_i)_{i\in[m]} \in \mathbb{R}^{mn_f}$ and $v = (v_i)_{i\in[m]} \in \mathbb{R}^{mn_g}$, where $u_i \in \mathbb{R}^{n_f}$ and $v_i \in \mathbb{R}^{n_g}$. Denote

$$\check{F} := \mathrm{diag}(F_1, \ldots, F_m) \in \mathbb{R}^{mn_f\times\check{n}},$$
$$\hat{G} := \mathrm{diag}(G_1, \ldots, G_m) \in \mathbb{R}^{mn_g\times\hat{n}}.$$

The following saddle function in the pair of variables (x, ξ, u) and (y, η, v) can then be defined:

$$\begin{aligned} K_c(x, \xi, u, y, \eta, v) &:= K(x, y) + \eta^T(\check{F}x - c) + \xi^T(\hat{G}y - d) \\ &\quad + u^T(W \otimes I_{n_f})\eta + v^T(W \otimes I_{n_g})\xi \\ &= K(x, y) + \sum_{i\in[m]} \Big(\eta_i^T(F_i x_i - c_i) + \xi_i^T(G_i y_i - d_i) \\ &\quad + \sum_{j\in\mathcal{N}_i} w_{ij}\big(u_i^T(\eta_i - \eta_j) + v_i^T(\xi_i - \xi_j)\big)\Big). \end{aligned} \tag{20}$$

The saddle differential operator of K_c is

$$T_{K_c}(x, \xi, u, y, \eta, v) = \begin{bmatrix} \partial_x K(x, y) + \check{F}^T\eta \\ \hat{G}y - d + (W \otimes I_{n_g})v \\ (W \otimes I_{n_f})\eta \\ \partial_y(-K)(x, y) - \hat{G}^T\xi \\ -\check{F}x + c - (W \otimes I_{n_f})u \\ -(W \otimes I_{n_g})\xi \end{bmatrix}.$$

Lemma 2. *$(x^*, \xi_0^*, y^*, \eta_0^*)$ is a saddle point of K_b if and only if $(x^*, \xi^*, u^*, y^*, \eta^*, v^*)$ is a saddle point of K_c for some ξ^*, u^*, η^*, and v^*.*

Proof. For the "if" part, suppose $(x^*, \xi^*, u^*, y^*, \eta^*, v^*)$ is a saddle point of K_c. From $0 \in T_{K_c}$, we have $(W \otimes I_{n_f})\eta^* = 0$ and $(W \otimes I_{n_g})\xi^* = 0$. By our assumption on W, these imply that $\xi^* = \mathbf{1}_m \otimes \xi_0^*$ and $\eta^* = \mathbf{1}_m \otimes \eta_0^*$ for some ξ_0^* and η_0^*. Thus, $\check{F}^T\eta^* = F^T\eta_0^*$ and $\hat{G}^T\xi^* = G^T\xi_0^*$. The first and fourth rows of $0 \in T_{K_c}$ can then be rewritten as $0 \in \partial_x K(x^*, y^*) + F^T\eta_0^*$ and $0 \in \partial_y(-K)(x^*, y^*) - G^T\xi_0^*$, which are exactly the first two conditions in (18). By multiplying the second and fifth rows of $0 \in T_{K_c}$ from the left by $\mathbf{1}_m^T \otimes I_{n_g}$ and $\mathbf{1}_m^T \otimes I_{n_f}$, respectively, and using the fact that $\mathbf{1}_m^T W = 0$ due to W being the graph Laplacian, we derive the last two conditions in (18). This shows that $(x^*, \xi_0^*, y^*, \eta_0^*)$ is a saddle point of K_b.

For the "only if" part, suppose $(x^*, \xi_0^*, y^*, \eta_0^*)$ is a saddle point of K_b, i.e., condition (18) holds. Set $\xi^* = \mathbf{1}_m \otimes \xi_0^*$ and $\eta^* = \mathbf{1}_m \otimes \eta_0^*$. For $0 \in T_{K_c}$ to hold, the only nontrivial rows are the second and the fifth ones. For the second row, note that the symmetric matrix $W \otimes I_{n_g}$ has the null space $\{\mathbf{1}_m \otimes w \,|\, w \in \mathbb{R}^{n_g}\}$. Its range space is the orthogonal complement of its null space, which is exactly the null space of $\mathbf{1}_m^T \otimes I_{n_g}$. Since $(\mathbf{1}_m^T \otimes I_{n_g})(\hat{G}y - d) = \sum_{i\in[m]}(G_i y_i - d_i) = 0$, $\hat{G}y - d$ belongs to the range space of $W \otimes I_{n_g}$. This shows that the second row of $0 \in T_{K_c}$ can be satisfied by a proper choice of v. Similar arguments can be applied for the fifth row of $0 \in T_{K_c}$. □

By Lemma 2, Problem 3 can be equivalently solved by finding a saddle point of K_c. From its definition in (20), K_c has several non-separable terms: $K(x, y)$ due to the couplings of K_i's, and the last two terms involving η_j and ξ_j from neighboring nodes j in the consensus graph. To deal with these, we first follow the strategy in Section IV and augment the local variables x_i and y_i of each node i to $\mathbf{x}_i = (x_i, (x_{ij})_{j\in\check{\mathcal{N}}_i^-})$ and $\mathbf{y}_i = (y_i, (y_{il})_{l\in\hat{\mathcal{N}}_i^-})$. Denote by $\check{S}_i$ and $\hat{S}_i$ the selection matrices that select the original local variables from their augmented versions: $x_i = \check{S}_i\mathbf{x}_i$ and $y_i = \hat{S}_i\mathbf{y}_i$. Note that $\check{S}_i \in \mathbb{R}^{\check{n}_i\times\check{N}_i}$ and $\hat{S}_i \in \mathbb{R}^{\hat{n}_i\times\hat{N}_i}$ have all entries being either 0 or 1. Denote

$$\check{S} = \mathrm{diag}(\check{S}_1, \ldots, \check{S}_m), \quad \hat{S} = \mathrm{diag}(\hat{S}_1, \ldots, \hat{S}_m).$$

Then $x = \check{S}\mathbf{x}$ and $y = \hat{S}\mathbf{y}$. Define the consensus subspaces $\mathcal{A}_x$ and $\mathcal{A}_y$ as in (10). For $(\mathbf{x}, \mathbf{y}) \in \mathcal{A}_x \times \mathcal{A}_y$, $K_a(\mathbf{x}, \mathbf{y})$ defined in (11) agrees with $K(x, y) = K(\check{S}\mathbf{x}, \hat{S}\mathbf{y})$. Define

$$\begin{aligned} K_d(\mathbf{x}, \xi, u, \mathbf{y}, \eta, v) &:= K_a(\mathbf{x}, \mathbf{y}) + \eta^T(\check{F}\check{S}\mathbf{x} - c) \\ &\quad + \xi^T(\hat{G}\hat{S}\mathbf{y} - d) + u^T(W \otimes I_{n_f})\eta + v^T(W \otimes I_{n_g})\xi \\ &= \sum_{i\in[m]} \Big(K_i(\mathbf{x}_i, \mathbf{y}_i) + \eta_i^T(F_i\check{S}_i\mathbf{x}_i - c_i) + \xi_i^T(G_i\hat{S}_i\mathbf{y}_i - d_i) \\ &\quad + \sum_{j\in\mathcal{N}_i} w_{ij}\big(u_i^T(\eta_i - \eta_j) + v_i^T(\xi_i - \xi_j)\big)\Big). \end{aligned} \tag{21}$$

Obviously, for $(\mathbf{x}, \mathbf{y}) \in \mathcal{A}_x \times \mathcal{A}_y$, we have $K_d(\mathbf{x}, \xi, u, \mathbf{y}, \eta, v) = K_c(x, \xi, u, y, \eta, v)$ with $x = \check{S}\mathbf{x}$ and $y = \hat{S}\mathbf{y}$. This implies the following result.

Lemma 3. *There is a one-to-one correspondence between the saddle points $(x^*, \xi^*, u^*, y^*, \eta^*, v^*)$ of K_c and the saddle points $(\mathbf{x}^*, \xi^*, u^*, \mathbf{y}^*, \eta^*, v^*)$ of the following saddle function:*

$$K_e(\mathbf{x}, \xi, u, \mathbf{y}, \eta, v) = K_d(\mathbf{x}, \xi, u, \mathbf{y}, \eta, v) + \mu_{\mathcal{A}_x\times\mathcal{A}_y}(\mathbf{x}, \mathbf{y}).$$

Thus, from this point on we will focus on finding a saddle point of K_e, or equivalently, finding a point in the zero set of its saddle differential operator scaled by a constant of 2:

$$\begin{aligned}2T_{K_e} &= T_{K_{e1}} + T_{K_{e2}}\\ &:= \begin{bmatrix} 2\partial_{\mathbf{x}} K_a(\mathbf{x},\mathbf{y}) + (\check{F}\check{S})^T\eta \\ \hat{G}\hat{S}\mathbf{y} - d + (W\otimes I_{n_g})v \\ (W\otimes I_{n_f})\eta \\ 2\partial_{\mathbf{y}}(-K_a)(\mathbf{x},\mathbf{y}) - (\hat{G}\hat{S})^T\xi \\ -\check{F}\check{S}\mathbf{x} + c - (W\otimes I_{n_f})u \\ -(W\otimes I_{n_g})\xi \end{bmatrix} + \begin{bmatrix} 2\partial \mathbf{1}_{\mathcal{A}_x}(\mathbf{x}) + (\check{F}\check{S})^T\eta \\ \hat{G}\hat{S}\mathbf{y} - d + (W\otimes I_{n_g})v \\ (W\otimes I_{n_f})\eta \\ 2\partial \mathbf{1}_{\mathcal{A}_y}(\mathbf{y}) - (\hat{G}\hat{S})^T\xi \\ -\check{F}\check{S}\mathbf{x} + c - (W\otimes I_{n_f})u \\ -(W\otimes I_{n_g})\xi \end{bmatrix}. \end{aligned} \tag{22}$$

In the above, $2T_{K_e}$ is split into two operators $T_{K_{e1}}$ and $T_{K_{e2}}$ with a similar structure. As will be seen in Lemma 5, this enables us to compute their resolvents in a distributed way.

Lemma 4. *$T_{K_{e1}}$ and $T_{K_{e2}}$ are maximally monotone operators.*

Proof. Note that $T_{K_{e1}}$ can be written as the sum of $2\begin{bmatrix}(\partial_{\mathbf{x}}K_a(\mathbf{x},\mathbf{y}))^T & 0 & 0 & (\partial_{\mathbf{y}}(-K_a)(\mathbf{x},\mathbf{y}))^T & 0 & 0\end{bmatrix}^T$ and an affine operator defined by a skew symmetric transformation matrix. The former being the saddle differential operator of the closed proper saddle function $2K_a$ is maximally monotone by Proposition 2. The latter is also maximally monotone and defined everywhere. As a result, their sum is also maiximally monotone [34]. Similar proof holds for $T_{K_{e2}}$ using the fact that $\mathcal{A}_x$ and $\mathcal{A}_y$ are nonempty closed convex sets. □

Define the following symmetric preconditioning matrix

$$P := \begin{bmatrix} P_1 & 0 & 0 & 0 & (\check{F}\check{S})^T & 0 \\ 0 & P_2 & 0 & -\hat{G}\hat{S} & 0 & -(W\otimes I_{n_g}) \\ 0 & 0 & P_3 & 0 & W\otimes I_{n_f} & 0 \\ 0 & \star & 0 & P_4 & 0 & 0 \\ \star & 0 & \star & 0 & P_5 & 0 \\ 0 & \star & 0 & 0 & 0 & P_6 \end{bmatrix}. \tag{23}$$

where $\star$'s indicate transposes of the corresponding blocks above the main diagonal. Here, we have

$$\begin{aligned} &P_1 = \rho^{-1}I_{\check{N}},\ \ P_2 = \mathrm{diag}(\check{\delta}_1,\ldots,\check{\delta}_m)^{-1}\otimes I_{N_g},\\ &P_3 = \mathrm{diag}(\check{\epsilon}_1,\ldots,\check{\epsilon}_m)^{-1}\otimes I_{N_f},\\ &P_4 = \rho^{-1}I_{\hat{N}},\ \ P_5 = \mathrm{diag}(\hat{\delta}_1,\ldots,\hat{\delta}_m)^{-1}\otimes I_{N_f},\\ &P_6 = \mathrm{diag}(\hat{\epsilon}_1,\ldots,\hat{\epsilon}_m)^{-1}\otimes I_{N_g}, \end{aligned}$$

where ρ, $\check{\delta}_i$, $\hat{\delta}_i$, $\check{\epsilon}_i$, and $\hat{\epsilon}_i$, $i\in[m]$, are positive constants chosen small enough so that $P\succ 0$ is positive definite. With this preconditioning matrix P, we now characterize the generalized resolvents of $T_{K_{e1}}$ and $T_{K_{e2}}$ as defined in (5), which we denote as $\tilde{J}_{K_{e1}}$ and $\tilde{J}_{K_{e2}}$, respectively.

Lemma 5. *$(\mathbf{x}',\xi',u',\mathbf{y}',\eta',v') = \tilde{J}_{K_{e1}}(\mathbf{x},\xi,u,\mathbf{y},\eta,v)$ can be computed according to*

$$\xi' = \xi - P_2^{-1}\left(\hat{G}\hat{S}\mathbf{y} - d + (W\otimes I_{n_g})v\right), \tag{24a}$$
$$\eta' = \eta + P_5^{-1}\left(\check{F}\check{S}\mathbf{x} - c + (W\otimes I_{n_f})u\right), \tag{24b}$$
$$u' = u - P_3^{-1}(W\otimes I_{n_f})(2\eta'-\eta), \tag{24c}$$
$$v' = v + P_6^{-1}(W\otimes I_{n_g})(2\xi'-\xi), \tag{24d}$$
$$(\mathbf{x}',\mathbf{y}') = J_{2\rho K_a}\big(\mathbf{x} - \rho(\check{F}\check{S})^T(2\eta'-\eta),\ \mathbf{y} + \rho(\hat{G}\hat{S})^T(2\xi'-\xi)\big). \tag{24e}$$

Here, $J_{2\rho K_a}$ is the canonical resolvent of $2\rho T_{K_a}$ (i.e., no preconditioning). Similarly, $(\mathbf{x}',\xi',u',\mathbf{y}',\eta',v') = \tilde{J}_{K_{e2}}(\mathbf{x},\xi,u,\mathbf{y},\eta,v)$ can be computed as above, with (24e) *replaced by: $\mathbf{x}' = \Pi_{\mathcal{A}_x}\big(\mathbf{x} - \rho(\check{F}\check{S})^T(2\eta'-\eta)\big)$, and $\mathbf{y}' = \Pi_{\mathcal{A}_y}\big(\mathbf{y} + \rho(\hat{G}\hat{S})^T(2\xi'-\xi)\big)$.*

Proof. From the definition of $\tilde{J}_{K_{e1}}$, we have $P(\mathbf{x},\xi,u,\mathbf{y},\eta,v) \in P(\mathbf{x}',\xi',u',\mathbf{y}',\eta',v') + T_{K_{e1}}(\mathbf{x}',\xi',u',\mathbf{y}',\eta',v')$, or explicitly,

$$\begin{bmatrix} P_1\mathbf{x} + (\check{F}\check{S})^T\eta \\ P_2\xi - \hat{G}\hat{S}\mathbf{y} - (W\otimes I_{n_g})v \\ P_3u + (W\otimes I_{n_f})\eta \\ P_4\mathbf{y} - (\hat{G}\hat{S})^T\xi \\ P_5\eta + \check{F}\check{S}\mathbf{x} + (W\otimes I_{n_f})u \\ P_6v - (W\otimes I_{n_g})\xi \end{bmatrix} \in \begin{bmatrix} P_1\mathbf{x}' + 2\partial_{\mathbf{x}}K_a(\mathbf{x}',\mathbf{y}') + 2(\check{F}\check{S})^T\eta' \\ P_2\xi' - d \\ P_3u' + 2(W\otimes I_{n_f})\eta' \\ P_4\mathbf{y}' + 2\partial_{\mathbf{y}}(-K_a)(\mathbf{x}',\mathbf{y}') - 2(\hat{G}\hat{S})^T\xi' \\ P_5\eta' + c \\ P_6v' - 2(W\otimes I_{n_g})\xi' \end{bmatrix}.$$

The second and the fifth rows lead to (24a) and (24b), while (24c) and (24d) follow from the third and the sixth rows. The first and fourth rows can be written as $\big(\mathbf{x}-\rho(\check{F}\check{S})^T(2\eta'-\eta), \mathbf{y}+\rho(\hat{G}\hat{S})^T(2\xi'-\xi)\big) \in (I+2\rho T_{K_a})(\mathbf{x}',\mathbf{y}')$, which results in (24e). For $\tilde{J}_{K_{e2}}$, the proof is entirely similar, with the only change being that K_a is replaced with $\mathbf{1}_{\mathcal{A}_x\times\mathcal{A}_y}$. □

All of the steps in (24) can be completed by individual nodes in a distributed way. This is due to the fact that W is the graph Laplacian and all other matrices are block diagonal. As an example, (24a) is equivalent to $\xi_i' = \xi_i - \check{\delta}_i\big(G_iy_i - d_i + \sum_{j\in\mathcal{N}_i} w_{ij}(v_i - v_j)\big)$, $\forall i\in[m]$.

By applying generalized Douglas-Rachford splitting to $T_{K_{e1}} + T_{K_{e2}}$, we have the following algorithm:

$$\bar{\mathbf{z}}^k = \tilde{J}_{K_{e2}}(\mathbf{z}^k) \tag{25a}$$
$$\mathbf{z}^{k+1} = \mathbf{z}^k + 2\alpha\left(\tilde{J}_{K_{e1}}(2\bar{\mathbf{z}}^k - \mathbf{z}^k) - \bar{\mathbf{z}}^k\right), \tag{25b}$$

where $\mathbf{z}^k = (\mathbf{x}^k,\xi^k,u^k,\mathbf{y}^k,\eta^k,v^k)$ and $\bar{\mathbf{z}}^k = (\bar{\mathbf{x}}^k,\bar{\xi}^k,\bar{u}^k,\bar{\mathbf{y}}^k,\bar{\eta}^k,\bar{v}^k)$. Using Lemma 5, the detailed executions of (25) are described in Algorithm 3. There are five **for** loops in each iteration, the first three carrying out step (25a) and the last two carrying out step (25b). Each loop corresponds to one round of synchronous communications among neighboring nodes. The second loop is the most

communication intensive: each node i collects $\bar{\zeta}_j$ and $\bar{\chi}_j$ from its neighboring nodes j in the consensus graph and $\bar{p}_{li}$ and $\bar{q}_{li}$ from its out-neighboring nodes l in the dependency graphs.

Algorithm 3 Coordinator-Free Algorithm

Initialize $\mathbf{z}^0 = (\mathbf{x}^0, \xi^0, u^0, \mathbf{y}^0, \eta^0, v^0)$, and let $k \leftarrow 0$
repeat
  for $i = 1, \ldots, m$ **do**
    $\bar{\xi}_i^k \leftarrow \xi_i^k - \check{\delta}_i\big(G_i y_i^k - d_i + \sum_{j\in\mathcal{N}_i} w_{ij}(v_i^k - v_j^k)\big)$;
    $\bar{\eta}_i^k \leftarrow \eta_i^k + \hat{\delta}_i\big(F_i x_i^k - c_i + \sum_{j\in\mathcal{N}_i} w_{ij}(u_i^k - u_j^k)\big)$;
    $\bar{\chi}_i^k \leftarrow 2\bar{\xi}_i^k - \xi_i^k$; $\bar{\zeta}_i^k \leftarrow 2\bar{\eta}_i^k - \eta_i^k$;
    $\bar{p}_i^k = x_i^k - \rho F_i^T \bar{\zeta}_i^k$; $\bar{q}_i^k = y_i^k + \rho G_i^T \bar{\chi}_i^k$;
  for $i = 1, \ldots, m$ **do**
    $\bar{u}_i^k \leftarrow u_i^k - \check{\epsilon}_i \sum_{j\in\mathcal{N}_i} w_{ij}(\bar{\zeta}_i^k - \bar{\zeta}_j^k)$;
    $\bar{v}_i^k \leftarrow v_i^k + \hat{\epsilon}_i \sum_{j\in\mathcal{N}_i} w_{ij}(\bar{\chi}_i^k - \bar{\chi}_j^k)$;
    $\bar{x}_i^k \leftarrow (\bar{p}_i^k + \sum_{l\in\check{\mathcal{N}}_i^-} x_{li}^k)/(1 + |\check{\mathcal{N}}_i^-|)$;
    $\bar{y}_i^k \leftarrow (\bar{q}_i^k + \sum_{l\in\hat{\mathcal{N}}_i^-} y_{li}^k)/(1 + |\hat{\mathcal{N}}_i^-|)$;
  for $i = 1, \ldots, m$ **do**
    $\bar{x}_{ij}^k \leftarrow \bar{x}_j^k$, $\forall j \in \check{\mathcal{N}}_i^+$; $\bar{y}_{ij}^k \leftarrow \bar{y}_j^k$, $\forall j \in \hat{\mathcal{N}}_i^+$;
    $\bar{\mathbf{z}}_i^k \leftarrow (\bar{\mathbf{x}}_i^k, \bar{\xi}_i^k, \bar{u}_i^k, \bar{\mathbf{y}}_i^k, \bar{\eta}_i^k, \bar{v}_i^k)$;
    $\tilde{\mathbf{z}}_i^k = (\tilde{\mathbf{x}}_i^k, \tilde{\xi}_i^k, \tilde{u}_i^k, \tilde{\mathbf{y}}_i^k, \tilde{\eta}_i^k, \tilde{v}_i^k) \leftarrow 2\bar{\mathbf{z}}_i^k - \mathbf{z}_i^k$;
  for $i = 1, \ldots, m$ **do**
    $\hat{\xi}_i^k \leftarrow \tilde{\xi}_i^k - \check{\delta}_i\big(G_i \tilde{y}_i^k - d_i + \sum_{j\in\mathcal{N}_i} w_{ij}(\tilde{v}_i^k - \tilde{v}_j^k)\big)$;
    $\hat{\eta}_i^k \leftarrow \tilde{\eta}_i^k + \hat{\delta}_i\big(F_i \tilde{x}_i^k - c_i + \sum_{j\in\mathcal{N}_i} w_{ij}(\tilde{u}_i^k - \tilde{u}_j^k)\big)$;
    $\hat{\chi}_i^k \leftarrow 2\hat{\xi}_i^k - \tilde{\xi}_i^k$; $\hat{\zeta}_i^k \leftarrow 2\hat{\eta}_i^k - \tilde{\eta}_i^k$;
    $(\hat{\mathbf{x}}_i^k, \hat{\mathbf{y}}_i^k) = J_{2\rho K_i}\Big(\tilde{\mathbf{x}}_i^k - \rho\check{S}_i^T F_i^T \hat{\zeta}_i^k, \tilde{\mathbf{y}}_i^k + \rho\hat{S}_i^T G_i^T \hat{\chi}_i^k\Big)$;
  for $i = 1, \ldots, m$ **do**
    $\hat{u}_i^k \leftarrow \tilde{u}_i^k - \check{\epsilon}_i \sum_{j\in\mathcal{N}_i} w_{ij}(\hat{\zeta}_i^k - \hat{\zeta}_j^k)$;
    $\hat{v}_i^k \leftarrow \tilde{v}_i^k + \hat{\epsilon}_i \sum_{j\in\mathcal{N}_i} w_{ij}(\hat{\chi}_i^k - \hat{\chi}_j^k)$;
    $\mathbf{z}_i^{k+1} \leftarrow \mathbf{z}_i^k + 2\alpha((\hat{\mathbf{x}}_i^k, \hat{\xi}_i^k, \hat{u}_i^k, \hat{\mathbf{y}}_i^k, \hat{\eta}_i^k, \hat{v}_i^k) - \bar{\mathbf{z}}_i^k)$;
  $k \leftarrow k + 1$
until $|\mathbf{z}^{k+1} - \mathbf{z}^k|$ is sufficiently small
return $\bar{\mathbf{z}}^k = (\bar{\mathbf{x}}^k, \bar{\xi}^k, \bar{u}^k, \bar{\mathbf{y}}^k, \bar{\eta}^k, \bar{v}^k)$

Theorem 3. *Suppose Assumptions 2, 3, 4, and 5 hold, and that the positive parameters ρ, $\check{\delta}_i$, $\hat{\delta}_i$, $\check{\epsilon}_i$, and $\hat{\epsilon}_i$ for $i \in [m]$ are chosen such that P defined in (23) is positive definite. Let $\alpha \in (0,1)$. Then, starting from any initial guess $(\mathbf{x}^0, \xi^0, u^0, \mathbf{y}^0, \eta^0, v^0)$, the sequence $(\bar{\mathbf{x}}^k, \bar{\xi}^k, \bar{u}^k, \bar{\mathbf{y}}^k, \bar{\eta}^k, \bar{v}^k)$ obtained by Algorithm 3 converges asymptotically to a saddle point $(\mathbf{x}^*, \xi^*, u^*, \mathbf{y}^*, \eta^*, v^*)$ of K_e, which corresponds to a saddle point solution $(\check{S}\mathbf{x}^*, \hat{S}\mathbf{y}^*)$ of Problem 3.*

Proof. Since $P \succ 0$, by Propostion 4, the sequence $(\bar{\mathbf{x}}^k, \bar{\xi}^k, \bar{u}^k, \bar{\mathbf{y}}^k, \bar{\eta}^k, \bar{v}^k)$ converges to a saddle point $(\mathbf{x}^*, \xi^*, u^*, \mathbf{y}^*, \eta^*, v^*)$ of K_e. By Lemmas 1, 2, and 3, $(\check{S}\mathbf{x}^*, \hat{S}\mathbf{y}^*)$ is a solution to Problem 3. □

C. General Global Constraints

We now briefly discuss how the proposed methods can be extended to certain cases of more general global coupling constraints. We state the conclusions directly as they can be proved by slight modifications of our existing arguments.

a) Linear inequality constraints.: Suppose the global constraints are of the form $Fx \le c_0$ and $Gy \le d_0$. Then the dual variables in (17) should satisfy $\xi_0 \in \mathbb{R}^{n_g}_{\le 0}$ and $\eta_0 \in \mathbb{R}^{n_f}_{\ge 0}$, and correspondingly $\xi \in \mathbb{R}^{mn_g}_{\le 0}$ and $\eta \in \mathbb{R}^{mn_f}_{\ge 0}$ in (20). Here, $\mathbb{R}^{\ell}_{\ge 0} := \{\alpha \in \mathbb{R}^{\ell} \,|\, \alpha_i \ge 0,\, i = 1, \ldots, \ell\}$. Similarly for $\mathbb{R}^{\ell}_{\le 0}$. By introducing the corresponding μ functions in the definitions of K_b, K_c, and K_d to enforce these constraints, the conclusions of Lemmas 1, 2, and 3 remain valid. In the splitting (22) of $2T_{K_e}$, $T_{K_{e1}}$ will have an extra term $\partial \mathbf{1}_{\mathbb{R}^{mn_g}_{\le 0}}(\xi)$ in the second row and an extra term $\partial \mathbf{1}_{\mathbb{R}^{mn_f}_{\ge 0}}(\eta)$ in the fifth row. The conclusions of Lemma 5 also remain valid once (24a) and (24b) are modified to the following:

$$\xi' = \Pi_{\mathbb{R}^{mn_g}_{\le 0}}\Big(\xi - P_2^{-1}\big(\hat{G}\hat{S}\mathbf{y} - d + (W \otimes I_{n_g})v\big)\Big),$$
$$\eta' = \Pi_{\mathbb{R}^{mn_f}_{\ge 0}}\left(\eta + P_5^{-1}\big(\check{F}\check{S}\mathbf{x} - c + \big(W \otimes I_{n_f}\big)\, u\big)\right).$$

Here, the orthogonal projection $\Pi_{\mathbb{R}^{mn_g}_{\le 0}}$ and $\Pi_{\mathbb{R}^{mn_f}_{\ge 0}}$ are computed entrywise via the functions $\Pi_{\mathbb{R}_{\le 0}}(z) = \min\{0, z\}$ and $\Pi_{\mathbb{R}_{\ge 0}}(z) = \max\{0, z\}$, respectively. By making these changes in steps 16 and 17 in Algorithm 3, we obtain a distributed solution algorithm to the convex-concave games on networks with global inequality constraints $Fx \le c_0$ and $Gy \le d_0$.

b) Sum-of-convex-functions constraints.: Consider now a global constraint of the form

$$h_1(\mathbf{x}_1) + \cdots + h_m(\mathbf{x}_m) \le h_0 \tag{26}$$

for some convex closed proper functions $h_i(\cdot) \in \mathbb{R}^{n_h}$, $i \in [m]$, and some constant $h_0 \in \mathbb{R}^{n_h}$. Here we recall that $\mathbf{x}_i$ is the concatenation of the local x-variables of node i and its x-in-neighbors. A special case of the above constraint is given by $h_1(x_1) + \cdots + h_m(x_m) \le h_0$ where the left hand side is a separable convex function. By augmenting $\mathbf{x}_i$ with an additional variable $r_i \in \mathbb{R}^{n_h}$ for each node i, the global constraint (26) is equivalent to

$$h_i(\mathbf{x}_i) \le r_i,\ \forall i \in [m], \ \text{ and } \ r_1 + \cdots + r_m \le h_0.$$

The first m constraints are locally coupled ones of the form (9); indeed, $h_i(\mathbf{x}_i) \le r_i$ specifies a closed convex feasible set for $(\mathbf{x}_i, r_i)$. Thus, they can be incorporated into the local payoff functions of individual nodes. The remaining constraint $r_1 + \cdots + r_m \le h_0$ is a global linear inequality constraint we just discussed above. Similar procedures can be adopted if there are constraints of the form (26) for the y-variables. As a result, Algorithm 3 after slight modifications can be used to solve the games with sum-of-convex-functions type of global constraints for both teams.

VI. Numerical Examples

In this section, we consider the Cournot game on a network of 50 markets as shown in Figure 3. The locations q_i of each market i is generated uniformly at random on the square $[0,1] \times [0,1] \subset \mathbb{R}^2$. For any pair of markets i and j with the distance $d_{ij} = \|q_i - q_j\| \le d_{min} = \frac{1}{4}$, the directed edge (i,j) has a 70% chance of being in the x-dependency graph with the weight $\alpha_{ij} = 0.7e^{-2d_{ij}/d_{min}}$ and a 60% chance of being in the y-dependency graph with the weight $\beta_{ij} = 0.7e^{-2d_{ij}/d_{min}}$; similarly for the directed edge (j,i).

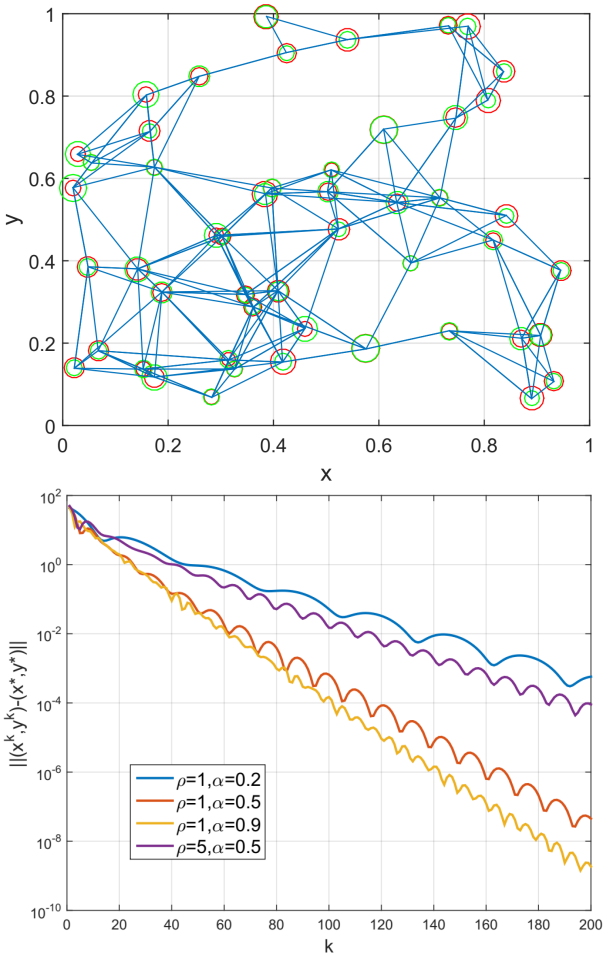


Fig. 3: Cournot competition with no global constraint on a randomly generated 50-node network. Left: unique Nash equilibrium allocation (x^*, y^*) as indicated by the sizes of circles (green circle for x_i's and red circles for y_i's); Right: convergence of $(\bar{x}^k, \bar{y}^k)$ obtained by Algorithm 1 to (x^*, y^*) under different parameters ρ and α.

All random choices are assumed to be independent. Suppose $\omega_i = 20$ and $\gamma_i = 1$ are identical for all markets i, and the local production costs $f_i(x_i) = \eta_i x_i^2$ and $g_i(y_i) = \zeta_i y_i^2$ for some random η_i and ζ_i uniformly distributed in the interval $[2, 10]$. Figure 3 (left) shows one instance of the random network and the associated Nash equilibrium allocations: at each node i, the sizes of the green and the red circles represent respectively the Nash equilibrium allocations x_i^* and y_i^* as specified by the unique saddle point solution (x^*, y^*) to the network Cournot game. The x-team and y-team have the total productions $\sum_i x_i^* = 70.92$ and $\sum_i y_i^* = 72.43$, total revenues $\sum_i p_i^* x_i^* = 1003.60$ and $\sum_i p_i^* y_i^* = 1033.92$, and total profits 509.66 and 524.68, respectively. Figure 3 (right) shows the convergence of the sequence (x^k, y^k) generated by Algorithm 1 to the solution (x^*, y^*) under different choices of the parameters ρ and α. In all cases linear convergence are observed, with some choices (e.g., $\rho = 1$, $\alpha = 0.9$) resulting in significantly faster convergence than others.

Next, we consider the same Cournot game as above, but now with the global total production constraints $\sum_i x_i = c_0 = 60$ and $\sum_i y_i = d_0 = 30$. In this case, the unique Nash equilibrium allocation (x^*, y^*) can be computed from (16) and is plotted on the left of Figure 4. The two teams have the total revenues 973.42 and 492.13, and the total profits 619.70 and 404.54, respectively. Compared to the Nash equilibrium solution in the unconstrained case, each team has lower total revenue due to reduced production enforced by the constraints, and the total profit is increased for the x-team while reduced for the y-team due to the more stringent production constraint on the latter. On the right of Figure 4 we plot the results of applying Algorithm 3 to solve this constrained network Cournot game. We assume that the edge set $\mathcal{E}_c$ of the consensus graph is the union set of the undirected edges of the x-dependency and y-dependency graphs, i.e., $(i, j) \in \mathcal{E}_c$ whenever either $(i, j) \in \mathcal{E}_x \cup \mathcal{E}_y$ or $(j, i) \in \mathcal{E}_x \cup \mathcal{E}_y$. It is clear from the left of Figure 4 that the consensus graph is connected. Each edge (i, j) in the consensus graph is assumed to have the same weight $w_{ij} = 1$. We choose the parameters $\rho = 1$, $\alpha = 0.9$, $\check{\epsilon}_i = \hat{\epsilon}_i = \check{\delta}_i = \hat{\delta}_i = 0.05$, and $c_i = 60/50$, $d_i = 30/50$, $\forall i$. It is easily verified that the preconditioning matrix P defined in (23) is positive definite. Thus, the conditions in Theorem 3 are satisfied. On the right of Figure 4, the solid line indicates the convergence of the sequence $(\bar{x}^k, \bar{y}^k)$ computed by Algorithm 3 to the Nash equilibrium (x^*, y^*), and the dash line indicates that the total global constraint violation as measured by $|\sum_i \bar{x}_i^k - c_0| + |\sum_i \bar{y}_i^k - d_0|$ converges to zero as $k \to \infty$. In both cases, the convergence are still roughly linear, though at noticeably slower rates than those achieved by Algorithm 1 on the unconstrained Cournot game (right of Figure 3). One reason for this is due to the extra time induced by the consensus process for ξ_i's and η_i's variables on the network in order to satisfy the global constraints.

Remark 3. *Interestingly if we modify the global constraints to $\sum_i x_i = 50$ and $\sum_i y_j = 60$, the Nash equilibrium allocation (x^*, y^*) computed by (16) satisfies that the x-team and the y-team have the total profits 531.60 and 587.16, respectively. Compared to the unconstrained case, both firms increase their total profits despite having lower total productions. This is not a contradiction as the Cournot game considered here is designed so that each firm tries to maximize its profit differential with the other firm instead of its own profit.*

VII. CONCLUSIONS AND FUTURE WORKS

This paper formulates a general class of games on networks and proposes several synchronous and randomized distributed solution algorithms with guaranteed convergence to Nash equilibrium solutions in the presence of local and global coupling constraints. In future work we will quantify the proposed algorithms' convergence rates in terms of the local payoff functions' convexity/concavity and the connectivity of the dependency and consensus graphs and extend the results to multi-player games.

APPENDIX
PROOF OF PROPOSITION 4

First note the relation $R_T^P(I + P^{-1}T) = [2(I + P^{-1}T)^{-1} - I](I + P^{-1}T) = 2I - (I + P^{-1}T) = I - P^{-1}T$. The

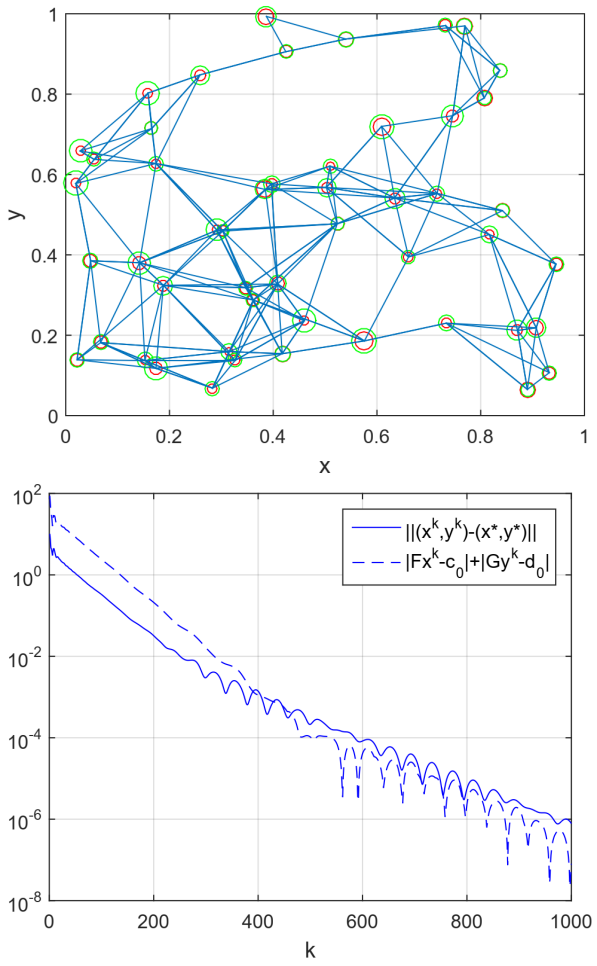


Fig. 4: Cournot competition on the same randomly generated 50-node network as in Fig. 3 with the global constraints $\sum_i x_i = c_0 = 60$ and $\sum_i y_i = d_0 = 30$. Left: unique Nash equilibrium allocation (x^*, y^*) computed by (16) as indicated by the sizes of circles (green for x_i's and red for y_i's); Right: convergence of Algorithm 3 with the parameters $\rho = 1$, $\alpha = 0.9$, $\check{\epsilon}_i = \hat{\epsilon}_i = \check{\delta}_i = \hat{\delta}_i = 0.05$, and $c_i = 60/50$, $d_i = 30/50$, $\forall i$. The solid line indicates the convergence of $(\bar{x}^k, \bar{y}^k)$ to (x^*, y^*) and the dash line indicates the convergence of the total global constraint violation as measured by $|\sum_i \bar{x}_i^k - c_0| + |\sum_i \bar{y}_i^k - d_0|$ to zero.

condition $0 \in T_1(w^*) + T_2(w^*)$ is then equivalent to $0 \in (I + P^{-1}T_1)w^* - (I - P^{-1}T_2)w^* = (I + P^{-1}T_1)w^* - R^P_{T_2}(I+P^{-1}T_2)w^*$. This in turn is equivalent to the existence of some $z^* \in (I + P^{-1}T_2)w^*$ (hence $w^* = J^P_{T_2}z^*$) so that $0 \in (I+P^{-1}T_1)w^* - R^P_{T_2}z^*$, i.e., $R^P_{T_2}z^* \in (I+P^{-1}T_1)J^P_{T_2}z^*$. The last condition can be rewritten as $^P_{T_2}z^* \in J^P_{T_1}R^P_{T_2}z^*$ which, after multiplied by two, is further equivalent to $(I + R^P_{T_2})z^* \in (I + R^P_{T_1})R^P_{T_2}z^*$, namely, $z^* \in R^P_{T_1}R^P_{T_2}z^*$. This completes the proof of the first half.

For the second part note that the iteration from z^k to z^{k+1} is via the averaged operator $I + 2\alpha J^P_{T_1}(2J^P_{T_2} - I) - 2\alpha J^P_{T_2} = (1-\alpha)I + \alpha R^P_{T_1}R^P_{T_2}$ w.r.t. the norm $\|\cdot\|_P$. This shows that $z^k \to z^*$ for some $z^* \in \mathrm{Fix}(R^P_{T_1} \circ R^P_{T_2})$. Using the first part, we conclude that $w^k \to J^P_{T_2}(z^*) \in \mathrm{zer}(T_1 + T_2)$.

REFERENCES

[1] M. Benzi, G. H. Golub, and J. Liesen, "Numerical solution of saddle point problems," *Acta numerica*, vol. 14, pp. 1–137, 2005.

[2] K. J. Arrow, L. Hurwicz, and H. Uzawa, "Studies in linear and non-linear programming," 1958.

[3] M. Rozložník and V. Simoncini, "Krylov subspace methods for saddle point problems with indefinite preconditioning," *SIAM Journal on Matrix Analysis and Applications*, vol. 24, no. 2, pp. 368–391, 2002.

[4] Z.-Z. Bai and G. H. Golub, "Accelerated Hermitian and skew-Hermitian splitting iteration methods for saddle-point problems," *IMA Journal of Numerical Analysis*, vol. 27, no. 1, pp. 1–23, 2007.

[5] J. Hofbauer and S. Sorin, "Best response dynamics for continuous zero-sum games," *Discrete and Continuous Dynamical Systems Series B*, vol. 6, no. 1, p. 215, 2006.

[6] Y. Nesterov, "Smooth minimization of non-smooth functions," *Mathematical Programming*, vol. 103, no. 1, pp. 127–152, 2005.

[7] A. Nemirovski, "Prox-method with rate of convergence $o(1/t)$ for variational inequalities with Lipschitz continuous monotone operators and smooth convex-concave saddle point problems," *SIAM Journal on Optimization*, vol. 15, no. 1, pp. 229–251, 2004.

[8] D. Feijer and F. Paganini, "Stability of primal–dual gradient dynamics and applications to network optimization," *Automatica*, vol. 46, no. 12, pp. 1974–1981, 2010.

[9] A. Nedić and A. Ozdaglar, "Subgradient methods for saddle-point problems," *Journal of Optimization Theory and Applications*, vol. 142, no. 1, pp. 205–228, 2009.

[10] H. Yin, U. V. Shanbhag, and P. G. Mehta, "Nash equilibrium problems with scaled congestion costs and shared constraints," *IEEE Transactions on Automatic Control*, vol. 56, no. 7, pp. 1702–1708, 2011.

[11] L. M. Briceno-Arias and P. L. Combettes, "Monotone operator methods for Nash equilibria in non-potential games," in *Computational and Analytical Mathematics*. Springer, 2013, pp. 143–159.

[12] M. Zhu and E. Frazzoli, "Distributed robust adaptive equilibrium computation for generalized convex games," *Automatica*, vol. 63, pp. 82–91, 2016.

[13] P. Yi and L. Pavel, "Distributed generalized Nash equilibria computation of monotone games via double-layer preconditioned proximal-point algorithms," *IEEE Transactions on Control of Network Systems*, vol. 6, no. 1, pp. 299–311, 2018.

[14] ——, "An operator splitting approach for distributed generalized Nash equilibria computation," *Automatica*, vol. 102, pp. 111–121, 2019.

[15] S. Liang, P. Yi, and Y. Hong, "Distributed Nash equilibrium seeking for aggregative games with coupled constraints," *Automatica*, vol. 85, pp. 179–185, 2017.

[16] S. Grammatico, "Dynamic control of agents playing aggregative games with coupling constraints," *IEEE Transactions on Automatic Control*, vol. 62, no. 9, pp. 4537–4548, 2017.

[17] M. Ye and G. Hu, "Game design and analysis for price-based demand response: An aggregate game approach," *IEEE Transactions on Cybernetics*, vol. 47, no. 3, pp. 720–730, 2017.

[18] D. Paccagnan, B. Gentile, F. Parise, M. Kamgarpour, and J. Lygeros, "Nash and Wardrop equilibria in aggregative games with coupling constraints," *IEEE Transactions on Automatic Control*, vol. 64, no. 4, pp. 1373–1388, 2018.

[19] B. Gharesifard and J. Cortés, "Distributed convergence to Nash equilibria in two-network zero-sum games," *Automatica*, vol. 49, no. 6, pp. 1683–1692, 2013.

[20] Y. Lou, Y. Hong, L. Xie, G. Shi, and K. H. Johansson, "Nash equilibrium computation in subnetwork zero-sum games with switching communications," *IEEE Transactions on Automatic Control*, vol. 61, no. 10, pp. 2920–2935, 2015.

[21] A. Ghosh and S. Boyd, "Minimax and convex-concave games," *Lecture Notes for Course EE392, Stanford Univ., Stanford, CA*, 2003.

[22] G. Scutari, D. P. Palomar, F. Facchinei, and J.-s. Pang, "Convex optimization, game theory, and variational inequality theory," *IEEE Signal Processing Magazine*, vol. 27, no. 3, pp. 35–49, 2010.

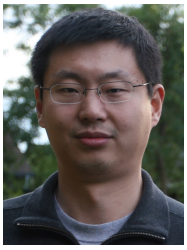
[23] F. Facchinei and C. Kanzow, "Generalized Nash equilibrium problems," *Annals of Operations Research*, vol. 175, no. 1, pp. 177–211, 2010.

[24] F. Facchinei and J.-S. Pang, *Finite-dimensional variational inequalities and complementarity problems*. Springer Science & Business Media, 2007.

[25] A. Kannan and U. V. Shanbhag, "Distributed computation of equilibria in monotone Nash games via iterative regularization techniques," *SIAM Journal on Optimization*, vol. 22, no. 4, pp. 1177–1205, 2012.

[26] M. Sion *et al.*, "On general minimax theorems." *Pacific Journal of mathematics*, vol. 8, no. 1, pp. 171–176, 1958.

[27] B. Johansson, T. Keviczky, M. Johansson, and K. H. Johansson, "Subgradient methods and consensus algorithms for solving convex optimization problems," in *Proc. 47th IEEE Conference on Decision and Control*. IEEE, 2008, pp. 4185–4190.
[28] A. Nedic, A. Ozdaglar, and P. A. Parrilo, "Constrained consensus and optimization in multi-agent networks," *IEEE Transactions on Automatic Control*, vol. 55, no. 4, pp. 922–938, 2010.
[29] M. Zhu and S. Martínez, "On distributed convex optimization under inequality and equality constraints," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 151–164, 2011.
[30] K. Bimpikis, S. Ehsani, and R. Ilkılıç, "Cournot competition in networked markets," *Management Science*, 2019.
[31] Y. Xiao, X. Hou, and J. Hu, "Distributed solutions of convex-concave games on networks," in *Proc. American Control Conference*, 2019, to appear.
[32] J. Eckstein and D. P. Bertsekas, "On the Douglas–Rachford splitting method and the proximal point algorithm for maximal monotone operators," *Mathematical Programming*, vol. 55, no. 1-3, pp. 293–318, 1992.
[33] K. Bredies and H. Sun, "Preconditioned Douglas–Rachford splitting methods for convex-concave saddle-point problems," *SIAM Journal on Numerical Analysis*, vol. 53, no. 1, pp. 421–444, 2015.
[34] H. H. Bauschke and P. L. Combettes, *Convex analysis and monotone operator theory in Hilbert spaces*. Springer, 2011, vol. 408.
[35] R. T. Rockafellar, "Monotone operators and the proximal point algorithm," *SIAM Journal on Control and Optimization*, vol. 14, no. 5, pp. 877–898, 1976.
[36] ——, "Monotone operators associated with saddle-functions and minimax problems," *Nonlinear Functional Analysis*, vol. 18, no. Part 1, pp. 397–407, 1970.
[37] E. K. Ryu and S. Boyd, "Primer on monotone operator methods," *Appl. Comput. Math*, vol. 15, no. 1, pp. 3–43, 2016.
[38] D. W. Peaceman and H. H. Rachford, Jr, "The numerical solution of parabolic and elliptic differential equations," *Journal of the Society for Industrial and Applied Mathematics*, vol. 3, no. 1, pp. 28–41, 1955.
[39] J. Douglas and H. H. Rachford, "On the numerical solution of heat conduction problems in two and three space variables," *Trans. American Mathematical Society*, vol. 82, no. 2, pp. 421–439, 1956.
[40] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.
[41] P. Bianchi, W. Hachem, and F. Iutzeler, "A coordinate descent primal-dual algorithm and application to distributed asynchronous optimization," *IEEE Transactions on Automatic Control*, vol. 61, no. 10, pp. 2947–2957, 2016.



Jianghai Hu Jianghai Hu (S'99-M'04) received the B.E. degree in automatic control from Xi'an Jiaotong University, P.R. China, in 1994, and the M.A. degree in Mathematics and the Ph.D. degree in Electrical Engineering from the University of California, Berkeley, in 2002 and 2003, respectively. He is currently an Associate Professor with the School of Electrical and Computer Engineering, Purdue University. His research interests include hybrid systems, multi-agent systems, control of systems with uncertainty, and control applications. He is an Associate Editor of *SIAM Journal on Control and Optimization* and *Nonlinear Analysis: Hybrid Systems*. He served as the IEEE Control System Society Electronic Publications Chair and editor of the monthly E-letters from 2016 to 2018.

Yingying Xiao received the B.S. and M.S. degrees in aerospace engineering from Harbin Institute of Technology, Harbin, China, in 2012 and 2014, respectively.

She is currently working toward a Ph.D. degree in School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA. Her research interests include distributed computation on network, multi-agent systems, control application in buildings.

Xiaodong Hou received his B.E. degree in Control Science and Engineering from Zhejiang University, Hangzhou, China, in 2013, and his Ph.D. degree in Electrical and Computer Engineering from Purdue University, in 2019. His research interests include distributed optimization over agent networks, and its applications to control and machine learning. Currently, he is a Machine Learning Data Scientist at Glassdoor, Inc., San Francisco, CA.