An Asynchronous, Forward-Backward, Distributed Generalized Nash Equilibrium Seeking Algorithm (I)

Cenedese, Carlo; Belgioioso, Giuseppe; Grammatico, Sergio; Cao, Ming

Published in:
Proceedings of the European Control Conference 2019

DOI:
10.23919/ECC.2019.8795952

Publication date:
2019

2019 18th European Control Conference (ECC)
Napoli, Italy, June 25-28, 2019

An asynchronous, forward-backward, distributed generalized Nash equilibrium seeking algorithm

Carlo Cenedese[1] Giuseppe Belgioioso[2] Sergio Grammatico[3] Ming Cao[1]

***Abstract*— In this paper, we propose an asynchronous distributed algorithm for the computation of generalized Nash equilibria in noncooperative games, where the players interact via an undirected communication graph. Specifically, we extend the paper "Asynchronous distributed algorithm for seeking generalized Nash equilibria" by Yi and Pavel: we redesign the asynchronous update rule using auxiliary variables over the nodes rather than over the edges. This key modification renders the algorithm scalable for highly interconnected games. The derived asynchronous algorithm is robust against delays in the communication and it eliminates the idle times between computations, hence modeling a more realistic interaction between players with different update frequencies. We address the problem from an operator-theoretic perspective and design the algorithm via a preconditioned forward-backward splitting. Finally, we numerically simulate the algorithm for the Cournot competition in networked markets.**

I. INTRODUCTION

A. Motivation and literature overview

Noncooperative generalized games over networks is currently a very active research field, due to the spreading of multi-agent network systems in modern society. Such type of games emerge in several application domains, such as smart grids [1], [2], social networks [3] and robotics [4]. In a game setup the players, or agents, have a private and local objective function that depends on the decisions of some other players, which shall be minimized while satisfying both local and global, coupling, constraints. Typically each agent defines its decision, or strategy, based on some local information exchanged with a subset of other agents, called neighbors. Various authors proposed solutions to this problem [5], [3], [6]. So, all the agents shall wait until the slowest one in the network completes its update, before starting a new operation. This can slow down the convergence drammatically, especially in large scale and heterogeneous systems. On the other hand, adopting an asynchronous update reduces the idle times, increasing efficiency. In addition, it can also speed up the convergence, facilitate the insertion of new agents in the network and even increse robusteness w.r.t. communication faults [7]. During the past years, several asynchronous algorithms for distributed convex optimization were proposed [8], [9], converging under different assumptions. The novel work in [10], provides a simple framework (ARock) to develop a wide range of iterative fixed point algorithms based on nonexpansive operators and it is already adopted in [11] to seek variational GNE seeking under equality constraints and using edge variables.

In this paper, we propose an extension of the work in [11]. Specifically, we consider inequality coupling constraints and use a restricted set of auxiliary variables, namely, associated with the nodes rather than with the edges. Especially this latter upgrade is non-trivial and presents technical challenges in the asynchronous implementation of the algorithm, which we overcome by analyzing the influence of the delayed information on the update of the auxiliary variables. The use of node variables only, rather than edge variables, preserves the scalability of the algorithm.

B. NOTATION

We use the same basic and operator-theoretic notations as in [12]. In addition, for a square matrix $A \in \mathbb{R}^{n\times n}$, its transpose is $A^\top$, $[A]_i$ is the i-th row of the matrix and $[A]_{ij}$ represents the elements in the row i and column j. $A \succ 0$ ($A \succeq 0$) stands for positive definite (semidefinite) matrix, instead $>$ ($\geq$) describes element wise inequality. $\mathrm{diag}(A_1, \dots, A_N)$ describes a block-diagonal matrix with the matrices $A_1, \dots, A_N$ on the main diagonal. The null space of a matrix A is $\ker(A)$.

A set valued mapping $\mathcal{F} : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is (strictly) monotone if $\forall x, y \in \mathbb{R}^n \ \langle \mathcal{F}(x) - \mathcal{F}(y), x - y\rangle \geq (>) 0$ holds true, and maximally monotone if it does not exist a monotone operator with a graph that strictly contains graph of $\mathcal{F}$,[19, Def. 20.20]. The proofs are omitted due to space limitations.

II. PROBLEM FORMULATION

A. Mathematical formulation

We consider a set of N agents (players), involved in a noncooperative game subject to coupling constraints. Each player $i \in \mathcal{N} := \{1, \dots, N\}$ has a local decision variable (strategy) x_i that belongs to its private decision set $\Omega_i \subseteq \mathbb{R}^{n_i}$, the vector of all the strategies played is $\boldsymbol{x} := \mathrm{col}(x_1, \dots, x_N) \in \mathbb{R}^n$ where $n = \sum_{i\in\mathcal{N}} n_i$, and $\boldsymbol{x}_{-i} = \mathrm{col}(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_N)$ are the decision variables of all the players other than i. The aim of each agent i is to minimize its local cost function $f_i(x_i, \boldsymbol{x}_{-i}) : \Omega_i \times \boldsymbol{\Omega}_{-i} \to \overline{\mathbb{R}}$, where $\boldsymbol{\Omega}_{-i} = \prod_{i\in\mathcal{N}\setminus\{i\}} \Omega_i \subseteq \mathbb{R}^{n-n_i}$, that leads to a coupling between players. In this work we assume the

[1] Engineering and Technology Institute Groningen (ENTEG), Faculty of Science and Engineering, University of Groningen, The Netherlands c.cenedese@rug.nl and m.cao@rug.nl. The work of Cenedese, and Cao was supported in part by the European Research Council (ERC-CoG-771687) and the Netherlands Organization for Scientific Research (NWO-vidi-14134).

[2] Control System group, TU Eindhoven, 5600 MB Eindhoven, The Netherlands g.belgioioso@tue.nl.

[3] Delft Center for Systems and Control, TU Delft, The Netherlands s.grammatico@tudelft.nl. The work of Grammatico was partially supported by NWO, under research projects OMEGA (TOP 613.001.702) and P2P-TALES (ESI-BIDA 647.003.003) and by the ERC under research project COSMOS (ERC-StG 802348).

presence of affine constraints between the agent strategies. These shape the collective feasible decision set

$$\boldsymbol{X} := \boldsymbol{\Omega} \cap \{\boldsymbol{x} \in \mathbb{R}^n \mid A\boldsymbol{x} \leq b\} \,, \tag{1}$$

where $\boldsymbol{\Omega}_i = \prod_{i\in\mathcal{N}} \Omega_i$, $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. Then, the feasible set of each agent $i \in \mathcal{N}$ reads as

$$\mathcal{X}_i(\boldsymbol{x}_{-i}) := \left\{ y \in \Omega_i \mid A_i y - b_i \leq \textstyle\sum_{j\in\mathcal{N}\setminus\{i\}} b_j - A_j x_j \right\} \,,$$

where $A = [A_1, \ldots, A_N]$, $A_i \in \mathbb{R}^{m\times n_i}$ and $\sum_{j=1}^N b_j = b$. We note that both the local decision set Ω_i and how the player i is involved in the coupling constraints, i.e. A_i and b_i, are private information, hence will not be accessible to other agents. Assuming affine constraints is common in the literature on noncooperative games [13], [5]. In the following, we introduce some other common assumptions over the aforementioned sets and cost function.

Standing Assumption 1 (Convex constraint sets): For each player $i \in \mathcal{N}$, the set Ω_i is convex, nonempty and compact, the feasible local set $\mathcal{X}_i(\boldsymbol{x}_{-i})$ satisfies Slater's constraint qualification. □

Standing Assumption 2 (Convex and diff. cost functions): For all $i \in \mathcal{N}$, the cost function f_i is continuous, β-Lipschitz continuous, continuously differentiable and convex in its first argument. □

In compact form, the game between players reads as

$$x_i \in \operatorname*{argmin}_{y\in\mathbb{R}^n} f_i(y, \boldsymbol{x}_{-i}) \quad \text{s.t.} \quad y \in \mathcal{X}_i(\boldsymbol{x}_{-i}) \,. \tag{2}$$

In this paper, we are interested in the *generalized Nash equilibia* (GNE) of the game in (2).

Definition 1 (Generalized Nash equilibrium): A collective strategy $\boldsymbol{x}^*$ is a GNE if, for each player i, it holds

$$x_i^* \in \operatorname*{argmin}_{y\in\mathbb{R}^n} f_i(y, \boldsymbol{x}_{-i}^*) \quad \text{s.t.} \quad y \in \mathcal{X}_i(\boldsymbol{x}_{-i}^*) \,. \tag{3}$$

□

B. Variational GNE

Let us introduce an interesting subset of GNE, the set of so called *variational GNE* (v-GNE), or *normalized equilibrium point*, of the game in (2) referring to the fact that all players share a common penalty in order to meet the constraints - see [14] and references therein. This set can be rephrased as solutions of a variational inequality (VI), as in [6].

First, we define the *pseudo-gradient* mapping of the game (2) as

$$F(\boldsymbol{x}) = \operatorname{col}\left(\{\nabla_{x_i} f_i(x_i, \boldsymbol{x}_{-i})\}_{i\in\mathcal{N}}\right) \,, \tag{4}$$

that gathers all the subdifferentials of the local cost functions of the agents. The following are some standard technical assumptions on F, see [15].

Standing Assumption 3: The pseudo-gradient F in (4) is ℓ-Lipschitz continuous and α-strongly monotone, for some $\ell, \alpha > 0$. □

Standing Assumption 2 implies that F is a single valued mapping, hence one can define VI($F, \boldsymbol{X}$) as the problem:

$$\text{find } \boldsymbol{x}^* \in \boldsymbol{X}, \text{ s.t. } \langle F(\boldsymbol{x}^*), \boldsymbol{x} - \boldsymbol{x}^* \rangle \geq 0 \,, \quad \forall \boldsymbol{x} \in \boldsymbol{X} \,. \tag{5}$$

Next, let us define the KKT conditions associated to the game in (2). Due to the convexity assumption, if $\boldsymbol{x}^*$ is a solution of (2), then there exist N dual variables $\lambda_i^* \in \mathbb{R}^m_{\geq 0}$, $\forall i \in \mathcal{N}$, such that the following inclusions are satisfied:

$$\begin{cases} \mathbf{0} & \in \nabla_{x_i} f_i(x_i) + A_i^\top \lambda_i^* + N_{\Omega_i}(x_i^*) \\ \mathbf{0} & \in b - A\boldsymbol{x}^* + N_{\mathbb{R}^m_{\geq 0}}(\lambda_i^*) \end{cases} \,, \ \forall i \in \mathcal{N} \,. \tag{6}$$

While in general the dual variables $\{\lambda_i\}_{i\in\mathcal{N}}$ can be different, here we focus on the subclass of equilibria sharing a common dual variable, i.e., $\lambda^* = \lambda_1^* = \cdots = \lambda_N^*$.

The KKT conditions for the VI($F, \boldsymbol{X}$) in (5) (see [6], [16]) read as

$$\begin{cases} \mathbf{0} & \in \nabla_{x_i} f_i(x_i) + A_i^\top \lambda^* + N_{\Omega_i}(x_i^*) \\ \mathbf{0} & \in b - A\boldsymbol{x}^* + N_{\mathbb{R}^m_{\geq 0}}(\lambda^*) \end{cases} \,, \ \forall i \in \mathcal{N} \,. \tag{7}$$

By (6) and (7), we deduce that every solution $\boldsymbol{x}^*$ of VI($F, \boldsymbol{X}$) is also a GNE of the game in (2), [6, Th. 3.1(i)]. In addition, if the pair $(\boldsymbol{x}^*, \lambda^*)$ satisfies the KKT conditions in (7), then $\boldsymbol{x}^*$ and the vectors $\lambda_1^* = \cdots = \lambda_N^* = \lambda^*$ satisfy the KKT conditions for the GNE, i.e. (6) [6, Th. 3.1(ii)].

Note that under Standing Assumptions 1–3 the set of v-GNE is guaranteed to be a singleton [16, Cor. 2.2.5; Th. 2.3.3].

III. Synchronous Distributed GNE Seeking

In this section, we describe the *Synchronous Distributed GNE Seeking Algorithm with Node variables* (SD-GENO).

A. Communication network

The communication between agents is described by an *undirected and connected* graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ where $\mathcal{N}$ is the set of players and $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ is the set of edges. We define $|\mathcal{E}| = M$, and $|\mathcal{N}| = N$. If an agent i shares information with j, then $(i, j) \in \mathcal{E}$, then we say that j belongs to the neighbours of i, i.e., $j \in \mathcal{N}_i$ where $\mathcal{N}_i$ is the neighbourhood of i. Let us label the edges e_l, for $l \in \{1, \ldots, M\}$. We denote by $E \in \mathbb{R}^{M\times N}$ the *incidence matrix*, where $[E]_{li}$ is equal to 1 (respectively -1) if $e_l = (i, \cdot)$ ($e_l = (\cdot, i)$) and 0 otherwise. By construction, $E\mathbf{1}_N = \mathbf{0}_N$. Then, we define $\mathcal{E}_i^{\text{out}}$ (respectively $\mathcal{E}_i^{\text{in}}$) as the set of all the indexes l of the edges e_l that start from (end in) node i, moreover $\mathcal{E}_i = \mathcal{E}_i^{\text{out}} \cup \mathcal{E}_i^{\text{in}}$. The *node Laplacian* $L \in \mathbb{R}^{N\times N}$ of an undirected graph is a symmetric matrix and can be expressed as $L = E^\top E$, [17, Lem. 8.3.2]. In the remainder of the paper, we exploit the fact that the Laplacian matrix is such that $L\mathbf{1}_N = \mathbf{0}_N$ and $\mathbf{1}_N^\top L = \mathbf{0}_N^\top$.

B. Algorithm design

Now, we present a distributed algorithm with convergence guarantees to the unique v-GNE of the game in (2). The KKT system in (6), can be cast in compact form as

$$\begin{aligned} \mathbf{0} &\in F(\boldsymbol{x}) + \Lambda^\top \boldsymbol{\lambda} + N_{\boldsymbol{\Omega}}(\boldsymbol{x}) \\ \mathbf{0} &\in \bar{b} - \Lambda\boldsymbol{x} + N_{\mathbb{R}^{mN}_{\geq 0}}(\boldsymbol{\lambda}) \end{aligned} \,, \tag{8}$$

where $\boldsymbol{\lambda} = \mathrm{col}(\lambda_1, \dots, \lambda_N) \in \mathbb{R}^{mN}$, $\Lambda = \mathrm{diag}(A_1, \dots, A_N) \in \mathbb{R}^{mN \times n}$ and $\bar{b} = \mathrm{col}(b_1, \dots, b_N) \in \mathbb{R}^{mN}$. To enforce consensus among the dual variables, hence obtain a v-GNE, we introduce the auxiliary variables σ_l, $l \in \{1, \dots, M\}$, one for every edge of the graph. Defining $\boldsymbol{\sigma} = \mathrm{col}(\sigma_1, \dots, \sigma_M) \in \mathbb{R}^{mM}$ and using $\boldsymbol{E} = E \otimes I_m \in \mathbb{R}^{mM \times mN}$, the inclusions in (8) become

$$\begin{aligned} \mathbf{0} &\in F(\boldsymbol{x}) + \Lambda^\top \boldsymbol{\lambda} + N_{\boldsymbol{\Omega}}(\boldsymbol{x}) \\ \mathbf{0} &\in \bar{b} - \Lambda \boldsymbol{x} + N_{\mathbb{R}^{mN}_{\geq 0}}(\boldsymbol{\lambda}) + \boldsymbol{E}^\top \boldsymbol{\sigma} \\ \mathbf{0} &\in -\boldsymbol{E}\boldsymbol{\lambda} \,. \end{aligned} \tag{9}$$

The variables $\{\sigma_l\}_{l \in \{1 \dots M\}}$ are used to simplify the analysis, but we will show how we decrease their number to one for each node, increasing the scalability of the algorithm, especially for dense networks.

From an operator theoretic perspective, a solution $\varpi^* = \mathrm{col}(\boldsymbol{x}^*, \boldsymbol{\sigma}^*, \boldsymbol{\lambda}^*)$ to (9) can be interpreted as a zero of the sum of two operators, $\mathcal{A}$ and $\mathcal{B}$, defined as

$$\begin{aligned} \mathcal{A} : \varpi &\mapsto \begin{bmatrix} 0 & 0 & \Lambda^\top \\ 0 & 0 & -\boldsymbol{E} \\ -\Lambda & \boldsymbol{E}^\top & 0 \end{bmatrix} + \begin{bmatrix} N_{\boldsymbol{\Omega}}(\boldsymbol{x}) \\ 0 \\ N_{\mathbb{R}^{mN}_{\geq 0}}(\boldsymbol{\lambda}) \end{bmatrix} \\ \mathcal{B} : \varpi &\mapsto \begin{bmatrix} F(\boldsymbol{x}) \\ 0 \\ \bar{b} \end{bmatrix} . \end{aligned} \tag{10}$$

In fact, $\varpi^* \in \mathrm{zer}(\mathcal{A} + \mathcal{B})$ if and only if ϖ^* satisfies (9).

Next, we show that the zeros of $\mathcal{A} + \mathcal{B}$ are actually the v-GNE of the initial game.

Proposition 1: Let $\mathcal{A}$ and $\mathcal{B}$ be as in (10). Then the following hold:

(i) $\mathrm{zer}(\mathcal{A} + \mathcal{B}) \neq \varnothing$,

(ii) if $\mathrm{col}(\boldsymbol{x}^*, \boldsymbol{\sigma}^*, \boldsymbol{\lambda}^*) \in \mathrm{zer}(\mathcal{A} + \mathcal{B})$ then $(\boldsymbol{x}^*, \lambda^*)$ satisfies the KKT conditions in (7), hence $\boldsymbol{x}^*$ is the v-GNE for the game in (2). □

The proof used an argument analogue to the one used in [5, Th. 4.5] and the properties of the incidence matrix E.

The problem of finding the zeros of the sum of two monotone operators is widely studied in literature and a plethora of different splitting method can be used to iteratively solve the problem [18], [19, Ch. 26]. A necessary first step is to prove the monotonicity of the defined operators.

Lemma 1: The mappings $\mathcal{A}$ and $\mathcal{B}$ in (10) are maximally monotone. Moreover, $\mathcal{B}$ is $\frac{\alpha}{\ell^2}$-cocoercive. □

The splitting method chosen here to find $\mathrm{zer}(\mathcal{A} + \mathcal{B})$ is the *preconditioned forward-backward* splitting (PFB), which can be applied thanks to the properties stated in Lemma 1. The iteration of the algorithm takes the form of the so called Krasnosel'skiĭ iteration, namely

$$\begin{aligned} \tilde{\varpi}^k &= T\varpi^k \\ \varpi^{k+1} &= \varpi^k + \eta(\tilde{\varpi}^k - \varpi^k) \end{aligned} \tag{11}$$

where $\varpi^k = \mathrm{col}(\boldsymbol{x}^k, \boldsymbol{\sigma}^k, \boldsymbol{\lambda}^k)$, $\eta > 0$ and T is the PFB splitting operator

$$T = \mathrm{J}_{\gamma \Phi^{-1} \mathcal{A}} \circ (\mathrm{Id} - \gamma \Phi^{-1} \mathcal{B}) \,, \tag{12}$$

where $\gamma > 0$ is a step size. The so-called preconditioning matrix Φ is defined as

$$\Phi := \begin{bmatrix} \boldsymbol{\tau}^{-1} & 0 & -\Lambda^\top \\ 0 & \delta^{-1} I_{mM} & \boldsymbol{E} \\ -\Lambda & \boldsymbol{E}^\top & \boldsymbol{\varepsilon}^{-1} \end{bmatrix} \tag{13}$$

where $\delta \in \mathbb{R}_{>0}$, $\boldsymbol{\varepsilon} = \mathrm{diag}(\varepsilon_1, \dots, \varepsilon_N) \otimes I_m$ with $\varepsilon_i > 0$, $\forall i \in \mathcal{N}$ and $\boldsymbol{\tau}$ is defined in a similar way.

From (12), we note that $\mathrm{fix}(T) = \mathrm{zer}(\mathcal{A} + \mathcal{B})$, indeed $\varpi \in \mathrm{fix}(T) \Leftrightarrow \varpi \in T\varpi \Leftrightarrow 0 \in \Phi^{-1}(\mathcal{A} + \mathcal{B})\varpi \Leftrightarrow \varpi \in \mathrm{zer}(\mathcal{A} + \mathcal{B})$, [19, Th. 26.14]. Thus, the zero-finding problem is translated into the fixed point problem for T in (12).

At this point, we calculate from (11) the explicit update rules of the variables. We focus on the first part of the update, i.e., $\tilde{\varpi}^k = T\varpi^k$. It can be rewritten as $\tilde{\varpi}^k \in \mathrm{J}_{\gamma \Phi^{-1} \mathcal{A}} \circ (\mathrm{Id} - \gamma \Phi^{-1} \mathcal{B}) \varpi^k \Leftrightarrow \Phi(\varpi^k - \tilde{\varpi}^k) \in \mathcal{A}\tilde{\varpi}^k + \mathcal{B}\varpi^k$ and finally

$$\mathbf{0} \in \mathcal{A}\tilde{\varpi}^k + \mathcal{B}\varpi^k + \Phi(\tilde{\varpi}^k - \varpi^k) \,, \tag{14}$$

here $\tilde{\varpi}^k := \mathrm{col}(\tilde{\boldsymbol{x}}^k, \tilde{\boldsymbol{\sigma}}^k, \tilde{\boldsymbol{\lambda}}^k)$. For ease of notation, we drop the time superscript k. By solving the first row block of (14), i.e. $\mathbf{0} \in F(\boldsymbol{x}) + N_{\boldsymbol{\Omega}}(\tilde{\boldsymbol{x}}) + \boldsymbol{\tau}^{-1}(\tilde{\boldsymbol{x}} - \boldsymbol{x}) + \Lambda^\top \boldsymbol{\lambda}$, we obtain

$$\tilde{\boldsymbol{x}} = \mathrm{J}_{N_{\boldsymbol{\Omega}}} \circ \big(\boldsymbol{x} - \boldsymbol{\tau}(F(\boldsymbol{x}) + \Lambda^\top \boldsymbol{\lambda})\big) \,. \tag{15}$$

The third row block of (14) instead reads as $\mathbf{0} \in \bar{b} + N_{\mathbb{R}^{mN}_{\geq 0}}(\tilde{\boldsymbol{\lambda}}) + \Lambda(2\tilde{\boldsymbol{x}} - \boldsymbol{x}) + \boldsymbol{E}^\top(2\tilde{\boldsymbol{\sigma}} - \boldsymbol{\sigma}) + \boldsymbol{\varepsilon}^{-1}(\tilde{\boldsymbol{\lambda}} - \boldsymbol{\lambda})$ that leads to

$$\tilde{\boldsymbol{\lambda}} = \mathrm{J}_{N_{\mathbb{R}^{mN}_{\geq 0}}} \circ \big(\boldsymbol{\lambda} - \boldsymbol{\varepsilon}(\Lambda(2\tilde{\boldsymbol{x}} - \boldsymbol{x}) - \bar{b} - \boldsymbol{E}^\top(2\tilde{\boldsymbol{\sigma}} - \boldsymbol{\sigma}))\big) \,. \tag{16}$$

The second row block of (14) defines the simple update $\tilde{\boldsymbol{\sigma}} = \boldsymbol{\sigma} + \delta \boldsymbol{E}\boldsymbol{\lambda}$. We note that in the update (16) of $\tilde{\boldsymbol{\lambda}}$, only $\boldsymbol{E}^\top \boldsymbol{\sigma}$ is used, hence an agent i needs only an aggregated information over the edge variables $\{\sigma_l\}_{l \in \mathcal{E}_i}$, to update its state and the dual variables. We exploit this property by replacing the edge variables with $\boldsymbol{z} = \boldsymbol{E}^\top \boldsymbol{\sigma} \in \mathbb{R}^{Nm}$. In this way, the auxiliary variables are one for each agent, instead of being one for each edge. Using the property $\boldsymbol{E}^\top \boldsymbol{E} = L \otimes I_m = \boldsymbol{L}$, we cast the update rule of these new auxiliary variables as

$$\begin{aligned} \tilde{\boldsymbol{z}}^k &= \boldsymbol{z}^k + \delta \boldsymbol{L} \boldsymbol{\lambda}^k \\ \boldsymbol{z}^{k+1} &= \boldsymbol{z}^k + \eta(\tilde{\boldsymbol{z}}^k - \boldsymbol{z}^k) \,. \end{aligned} \tag{17}$$

By introducing $\boldsymbol{z}$ in (16), we then have

$$\tilde{\boldsymbol{\lambda}} = \mathrm{J}_{N_{\mathbb{R}^{mN}_{\geq 0}}} \circ \big(\boldsymbol{\lambda} + \boldsymbol{\varepsilon}(\Lambda(2\tilde{\boldsymbol{x}} - \boldsymbol{x}) - \bar{b} - 2\tilde{\boldsymbol{z}} + \boldsymbol{z})\big) \,. \tag{18}$$

The next statement shows that an equilibrium of the new mapping is a v-GNE.

Theorem 1: If $(\boldsymbol{x}^*, \boldsymbol{z}^*, \boldsymbol{\lambda}^*)$ is a solution to the equations (15), (17) and (18), with $\mathbf{1}^\top \boldsymbol{z}^* = 0$, then $\boldsymbol{x}^*$ is a v-GNE. □

Remark 1: The change of auxiliary variables, from $\boldsymbol{\sigma}$ to $\boldsymbol{z}$, is particularly useful in large non-so-sparse networks and it is in general convenient when the number of edges higher than the number of nodes. In fact, for dense networks, we have one auxiliary variable for each player, hence the scalability of the algorithm is preserved.

Algorithm 1: SD-GENO

Input: $k=0$, $\boldsymbol{x}^0 \in \mathbb{R}^n$, $\boldsymbol{\lambda}^0 \in \mathbb{R}^{mN}$, $\boldsymbol{z}^0 = \mathbf{0}_{mN}$, and chose η, δ, $\boldsymbol{\varepsilon}$, $\boldsymbol{\tau}$ as in Theorem 2.

for $i \in \mathcal{N}$ **do**

$$\tilde{x}_i^k = \mathrm{proj}_{\Omega_i}\big(x_i^k - \tau_i(\nabla_{x_i} f_i(x_i^k, \boldsymbol{x}_{-i}^k) + A_i^\top \lambda_i^k)\big)$$
$$\tilde{z}_i^k = z_i^k + \delta \textstyle\sum_{j\in\mathcal{N}_i}(\lambda_i^k - \lambda_j^k)$$
$$\tilde{\lambda}_i^k = \mathrm{proj}_{\mathbb{R}^m_{\geq 0}}\big(\lambda_i^k + \varepsilon_i\big(A_i(2\tilde{x}_i^k - x_i^k) - b_i + z_i^k - 2\tilde{z}_i^k\big)\big)$$
$$x_i^{k+1} = x_i^k + \eta(\tilde{x}_i^k - x_i^k)$$
$$z_i^{k+1} = z_i^k + \eta(\tilde{z}_i^k - z_i^k)$$
$$\lambda_i^{k+1} = \lambda_i^k + \eta(\tilde{\lambda}_i^k - \lambda_i^k)$$
$$k \leftarrow k+1$$

C. Synchronous, distributed algorithm with node variables (SD-GENO)

We are now ready to state the update rules defining the synchronous version of the proposed algorithm. The update rule is obtained by gathering (15), (17), (18) and by modifying the second part of (11) via the variables $\boldsymbol{z}$:

$$\begin{aligned}
\tilde{\boldsymbol{x}}^k &= \mathrm{proj}_{\boldsymbol{\Omega}}\big(\boldsymbol{x}^k - \boldsymbol{\tau}(F(\boldsymbol{x}^k) + \Lambda^\top \boldsymbol{\lambda}^k)\big)\\
\tilde{\boldsymbol{z}}^k &= \boldsymbol{z}^k + \delta \boldsymbol{L}\boldsymbol{\lambda}^k\\
\tilde{\boldsymbol{\lambda}}^k &= \mathrm{proj}_{\mathbb{R}^{mN}_{\geq 0}}\big(\boldsymbol{\lambda}^k + \boldsymbol{\varepsilon}(\Lambda(2\tilde{\boldsymbol{x}}^k - \boldsymbol{x}^k) - \bar{b} - 2\tilde{\boldsymbol{z}}^k + \boldsymbol{z}^k)\big)\\
\boldsymbol{x}^{k+1} &= \boldsymbol{x}^k + \eta(\tilde{\boldsymbol{x}}^k - \boldsymbol{x}^k)\\
\boldsymbol{z}^{k+1} &= \boldsymbol{z}^k + \eta(\tilde{\boldsymbol{z}}^k - \boldsymbol{z}^k)\\
\boldsymbol{\lambda}^{k+1} &= \boldsymbol{\lambda}^k + \eta(\tilde{\boldsymbol{\lambda}}^k - \boldsymbol{\lambda}^k)\,,
\end{aligned} \tag{19}$$

See also Algorithm 1, for the local updates.

The $\boldsymbol{x}$ variable, generated by Algorithm 1, converges to the v-GNE of the game in (2) in view of the following statement.

Theorem 2: Let $\vartheta > \frac{\ell^2}{2\alpha}$, $\boldsymbol{\varepsilon}$, δ, $\boldsymbol{\tau} > 0$ such that $\Phi - \vartheta I \succeq 0$ and $\eta \in (0, \frac{4\alpha\vartheta - \ell^2}{2\alpha\vartheta})$. Then, Algorithm 1 converges to the v-GNE of the game in (2). □

IV. Asynchronous Distributed Algorithm

In this section, we present the main contribution of the paper, the *Asynchronous Distributed GNE Seeking Algorithm with Node variables* (AD-GENO), namely, the asynchronous counterpart of Algorithm 1. We first define a preliminary version of the algorithm using the edge auxiliary variables $\boldsymbol{\sigma}$, and then we derive the final formulation via the variable $\boldsymbol{z}$. To achieve an asynchronous update of the agent variables, we adopt the "ARock" framework [10].

A. Algorithm design

We modify the update rule in (11) to describe the asynchronism, in the local update of the agent i, as follows

$$\varpi^{k+1} = \varpi^k + \eta \Upsilon_i (T\varpi^k - \varpi^k)\,, \tag{20}$$

where Υ_i is a real diagonal matrix of dimension $n + (N+M)m$, where the element $[\Upsilon_i]_{jj}$ is 1 if the j-th element of $\mathrm{col}(\boldsymbol{x}, \boldsymbol{\sigma}, \boldsymbol{\lambda})$ is an element of $\mathrm{col}(x_i, \{\sigma_l\}_{l\in\mathcal{E}_i^{\mathrm{out}}}, \lambda_i)$ and 0 otherwise.We assume that the choice of which agent performs the update at iteration $k \in \mathbb{N}_{\geq 0}$ is ruled by an i.i.d. random variable ζ^k, that takes values in $\boldsymbol{\Upsilon} := \{\Upsilon_i\}_{i\in\mathcal{N}}$. Given a discrete probability distribution $(p_1, \ldots, p_N)$, let $\mathbb{P}[\zeta^k = \Upsilon_i] = p_i$, $\forall i \in \mathcal{N}$. Therefore, we rephrase the previous update rule as

$$\varpi^{k+1} = \varpi^k + \eta \zeta^k (T\varpi^k - \varpi^k)\,. \tag{21}$$

We also consider the possibility of delayed information, namely the update (21) can be performed with outdated values of ϖ^k. We refer to [10, Sec. 1] for a more complete overview on the topic. Due to the structure of the Υ_i, the update of x_i, λ_i and $\{\sigma_l\}_{l\in\mathcal{E}_i^{\mathrm{out}}}$ are performed at the same moment, hence they share the same delay φ_i^k at k.

We denote the vector of possibly delayed information at time k as $\hat{\varpi}^k$, hence the reformulation of (21) reads as

$$\varpi^{k+1} = \varpi^k + \eta \zeta^k (T - \mathrm{Id})\hat{\varpi}^k\,. \tag{22}$$

We impose that the maximum delay is uniformly bounded.

Standing Assumption 4 (Bounded maximum delay): The delays are uniformly upper bounded, i.e. there exists $\bar{\varphi} > 0$ such that $\sup_{k\in\mathbb{N}_{\geq 0}} \max_{i\in\mathcal{N}}\{\varphi_i^k\} \leq \bar{\varphi} < +\infty$. □

From the computational perspective, we assume that each player i has a public and a private memory. The first stores the information obtained by the neighbours $\mathcal{N}_i$. The private is instead used during the update of i at time k and it is an unchangeable copy of the public memory at iteration k. The local update rules in Algorithm 2 are obtained similarly to Sec. III-B for SD-GENO, hence by using the definition of T. The obtained algorithm resembles ADAGNES in [11, Alg. 1], therefore we name it E-ADAGNES.

The convergence of the update (22) can be derived by relying on the theoretical results in [10] for the Krasnosel'skiĭ asynchronous iteration.

Theorem 3: Let $\eta \in (0, \frac{4\alpha\vartheta - \ell^2}{\alpha\vartheta}\frac{cNp_{\min}}{4\bar{\varphi}\sqrt{p_{\min}}+1}]$, where $p_{\min} := \min\{p_i\}_{i\in\mathcal{N}}$ and $c \in (0,1)$. Then, the sequence $\{\boldsymbol{x}^k\}_{k\in\mathbb{N}_{\geq 0}}$ defined by Algorithm 2 converges to the v-GNE of the game in (2) almost surely. □

B. Asynchronous, distributed algorithm with node variables (AD-GENO)

With Algorithm 2 as starting point, we show that the change from auxiliary variables over the edges to variables over the nodes does not affect the dynamics of the pair $(\boldsymbol{x}, \boldsymbol{\lambda})$, thus preserving the convergence.

However, in this case, we need to introduce an extra variable for each node i, i.e., $\mu_i \in \mathbb{R}^m$. This is an aggregate information that groups all the changes of the neighbours dual variables from the previous update of i to the present iteration. We highlight that these variables are updated during the writing phase of the neighbours, therefore they do not require extra communications between the agents.

Algorithm 2: E-ADAGNES

Input: $k=0$, $\boldsymbol{x}^0 \in \mathbb{R}^n$, $\boldsymbol{\lambda}^0 \in \mathbb{R}^{mN}$, $\boldsymbol{\sigma}^0 = \mathbf{0}_{mM}$, chose η, δ, $\boldsymbol{\varepsilon}$, $\boldsymbol{\tau}$ as in Theorem 2.

Iteration k**:** Select the agent i^k with probability $\mathbb{P}(\zeta^k = \Upsilon_{i^k}) = p_{i^k}$

Reading: Agent i^k copies in its private memory the current values of the public memory, i.e. $x_j^{k-\varphi_j^k}$, $\lambda_j^{k-\varphi_j^k}$ for $j \in \mathcal{N}_{i^k}$ and $\sigma_l^{k-\psi_l^k}$, $\forall l \in \mathcal{E}_{i^k}$

Update:

$$\tilde{x}_{i^k}^k = \mathrm{proj}_{\Omega_{i^k}}\big(x_{i^k}^{k-\varphi_{i^k}^k} - \tau_{i^k}(\nabla_{x_{i^k}} f_{i^k}(x_{i^k}^{k-\varphi_{i^k}^k}, \hat{\boldsymbol{x}}_{-i^k}^k) + A_{i^k}^\top \lambda_{i^k}^{k-\varphi_{i^k}^k})\big)$$

$$\tilde{\sigma}_l^k = \sigma_l^{k-\varphi_{i^k}^k} + \delta([E]_l \otimes I_m)\hat{\boldsymbol{\lambda}}^k, \quad \forall l \in \mathcal{E}_{i^k}^{\mathrm{out}}$$

$$\tilde{\lambda}_{i^k}^k = \mathrm{proj}_{\mathbb{R}^m_{\geq 0}}\big(\lambda_{i^k}^{k-\varphi_{i^k}^k} + \varepsilon_{i^k}(A_{i^k}(2\tilde{x}_{i^k}^k - x_{i^k}^{k-\varphi_{i^k}^k}) - b_{i^k} - ([E^\top]_{i^k} \otimes I_m)(2\tilde{\boldsymbol{\sigma}}^k - \hat{\boldsymbol{\sigma}}^k))\big)$$

$$x_i^{k+1} = x_i^{k-\varphi_{i^k}^k} + \eta(\tilde{x}_i^k - x_i^k - \varphi_{i^k}^k)$$

$$\sigma_l^{k+1} = \sigma_l^{k-\varphi_{i^k}^k} + \eta(\tilde{\sigma}_l^k - \sigma_l^{k-\psi_l^k}), \quad \forall l \in \mathcal{E}_{i^k}$$

$$\lambda_i^{k+1} = \lambda_i^{k-\varphi_{i^k}^k} + \eta(\tilde{\lambda}_i^k - \lambda_i^{k-\varphi_{i^k}^k})$$

Writing: Agent i^k writes in the public memories of $j \in \mathcal{N}_{i^k}$ the new values of $x_{i^k}^{k+1}$, $\lambda_{i^k}^{k+1}$ and $\{\sigma_l^{k+1}\}_{l \in \mathcal{E}_{i^k}^{\mathrm{out}}}$

$k \leftarrow k+1$

Remark 2: The need for $\mu_i \in \mathbb{R}^m$ arises from the different update frequency between $\{\sigma_l\}_{l \in \mathcal{E}_i}$ and z_i. Therefore, we cannot characterize the dynamics of $\boldsymbol{\sigma}$, if we define $\boldsymbol{z} = \boldsymbol{E}^\top \boldsymbol{\sigma}$ only.

Algorithm 3 presents AD-GENO, where μ_i are rigorously defined.

The convergence of AD-GENO is proven by the following statement. Essentially, we show that introducing $\boldsymbol{z}$ does not change the dynamics of $(\boldsymbol{x}, \boldsymbol{\lambda})$.

Theorem 4: Let $\eta \in (0, \frac{4\alpha\vartheta - \ell^2}{\alpha\vartheta} \frac{cNp_{\min}}{4\bar{\varphi}\sqrt{p_{\min}+1}}]$ with $p_{\min} := \min\{p_i\}_{i \in \mathcal{N}}$ and $c \in (0,1)$. Then, the sequence $\{\boldsymbol{x}^k\}_{k \in \mathbb{N}_{\geq 0}}$ defined by Algorithm 3 converges to the v-GNE of the game in (2) almost surely. □

V. Illustrative example

This section presents the implementation of AD-GENO to solve a network Cournot game, that models the interaction of N companies competing over m markets. The problem is widely studied and we adopt a set-up similar to the one in [20], [5]. We chose $N = 8$ companies, each operating 4 strategies, i.e., $x_i \in \mathbb{R}^4$, $\forall i \in \mathcal{N}$. It ranges in $0 \leq x_i \leq \Omega_i$, where $\Omega_i \in \mathbb{R}^4$ and its elements are randomly drawn from $[10, 45]$. The markets are $m = 4$, named A, B, C and D. Two companies are neighbors if they share a market. The constraint matrix is $\boldsymbol{A} = [A_1, \ldots, A_N] \in \mathbb{R}^{4 \times 32}$ and the columns k of A_i have a nonzero element in position

Algorithm 3: AD-GENO

Input: $k=0$, $\boldsymbol{x}^0 \in \mathbb{R}^n$, $\boldsymbol{\lambda}^0 \in \mathbb{R}^{mN}$, $\boldsymbol{z}^0 = \mathbf{0}_{mN}$, chose η, δ, $\boldsymbol{\varepsilon}$, $\boldsymbol{\tau}$ as in Theorem 2. For all $i \in \mathcal{N}$ and $\mu_i = \mathbf{0}_m$.

Iteration k**:** Select the agent i^k with probability $\mathbb{P}[\zeta^k = \Upsilon_{i^k}] = p_{i^k}$

Reading: Agent i^k copies in its private memory the actual values of the public memory, i.e. $x_j^{k-\varphi_j^k}$, $\lambda_j^{k-\varphi_j^k}$, $z_j^{k-\varphi_j^k}$ for $j \in \mathcal{N}_{i^k}$ and μ_i. Reset the public values of μ_i to $\mathbf{0}_m$.

Update:

$$\tilde{x}_{i^k}^k = \mathrm{proj}_{\Omega_{i^k}}\big(x_{i^k}^{k-\varphi_{i^k}^k} - \tau_{i^k}(\nabla_{x_{i^k}} f_{i^k}(x_{i^k}^{k-\varphi_{i^k}^k}, \hat{\boldsymbol{x}}_{-i^k}^k) + A_{i^k}^\top \lambda_{i^k}^{k-\varphi_{i^k}^k})\big)$$

$$z_{i^k}^k = z_{i^k}^{k-\varphi_i^k} + \delta\eta\mu_{i^k}$$

$$\tilde{\lambda}_{i^k}^k = \mathrm{proj}_{\mathbb{R}^m_{\geq 0}}\Big(\lambda_{i^k}^{k-\varphi_{i^k}^k} + \varepsilon_{i^k}(A_{i^k}(2\tilde{x}_{i^k}^k - x_{i^k}^{k-\varphi_{i^k}^k}) - b_{i^k} - \tilde{z}_{i^k}^{k-\varphi_i^k} - 2\delta\textstyle\sum_{j \in \mathcal{N}_i \setminus \{i\}}(\lambda_{i^k}^{k-\varphi_{i^k}^k} - \lambda_j^{k-\varphi_j^k})\Big)$$

$$x_i^{k+1} = x_{i^k}^{k-\varphi_{i^k}^k} + \eta(\tilde{x}_{i^k}^k - x_{i^k}^{k-\varphi_{i^k}^k})$$

$$z_{i^k}^{k+1} = \tilde{z}_{i^k}^k + \eta\delta\textstyle\sum_{l \in \mathcal{E}_{i_k}^{\mathrm{out}}}([E]_l \otimes I_m)\hat{\boldsymbol{\lambda}}^k$$

$$\lambda_{i^k}^{k+1} = \lambda_{i^k}^{k-\varphi_{i^k}^k} + \eta(\tilde{\lambda}_{i^k}^k - \lambda_{i^k}^{k-\varphi_{i^k}^k})$$

Writing: Agent i^k writes in the public memories of $j \in \mathcal{N}_{i^k}$ the new values of $x_{i^k}^{k+1}$ and $\lambda_{i^k}^{k+1}$, for $j \in \mathcal{N}_{i^k} \setminus \{i^k\}$ the player i^k also overwrites μ_j as

$\mu_j \leftarrow \mu_j + \lambda_j^{k-\varphi_j^k} - \lambda_i^{k-\varphi_{i^k}^k}$

$k \leftarrow k+1$

j if the k-th strategy of player i is applied to market j. The nonzero values are randomly chosen from $[0.6, 1]$. The elements of $b \in \mathbb{R}^4$ are the markets' maximal capacities and are randomly chosen from $[20, 100]$. The arising inequality coupling constraint is $\boldsymbol{A}\boldsymbol{x} \leq b$. The local cost function is $f_i(x_i, \boldsymbol{x}_{-i}) = c_i(\boldsymbol{x}) - P(\boldsymbol{x})^\top A_i x_i$, where $c_i(\boldsymbol{x})$ is the cost of playing a certain strategy and $P(\boldsymbol{x})$ the price obtained by the market. We define the markets price as a linear function $P(\boldsymbol{x}) = \bar{P} - DA\boldsymbol{x}$, where $\bar{P} \in \mathbb{R}^4$ and $D \in \mathbb{R}^{4 \times 4}$ is a diagonal matrix, the values of their elements are randomly chosen respectively from $[250, 500]$ and $[1, 5]$. The cost function is quadratic $c_i(\boldsymbol{x}) = x_i^\top Q_i x_i + q_i^\top x_i$, where the elements of the diagonal matrix $Q_i \in \mathbb{R}^{4 \times 4}$ and the vector $q_i \in \mathbb{R}^4$ are randomly drawn respectively from $[1, 8]$ and $[1, 4]$. We propose two setups, the case of communication over a ring graph with alphabetic order and the case of random communication (in Figure 1, respectively the blue and red trajectories). In the latter, we only ensure that every $20N$ iterations all the agents performed a similar number of updates. The edges of the graph are arbitrarily oriented. We assume that the agents update with uniform probability, i.e., $P[\zeta^k = \Upsilon_i] = \frac{1}{N}$. The step sizes δ, ε, τ in AD-GENO

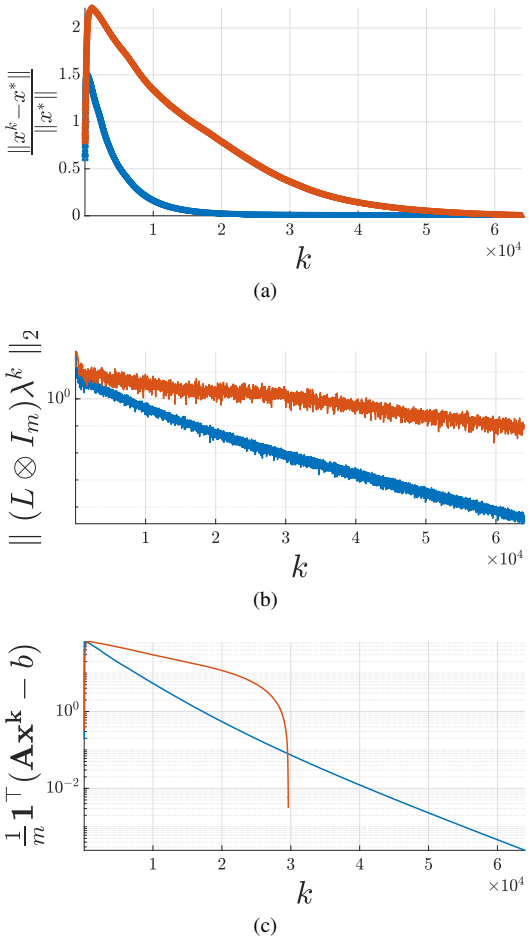

Fig. 1: Communication in alphabetic order (blue) versus random communication (red): (a) Normalized distance from equilibrium, (b) Norm of the disagreement vector, (c) Averaged constraints violation (the negative values are omitted).

are randomly chosen, the first from $[0.5, 0.2]$ and the others from $[0.5, 0.03]$, in order to ensure $\Phi \succ 0$ and $\eta = 0.35$. The maximum delay is assumed $\bar{\varphi} = 4$, therefore $\hat{\varpi}^k$ in (22) is $\hat{\varpi}^k = \text{col}(\hat{\varpi}_1^k, \dots, \hat{\varpi}_N^k)$ where each $\hat{\varpi}_i^k$ is randomly chosen from $\{\hat{\varpi}_i^{k-\bar{\varphi}}, \dots, \hat{\varpi}_i^k\}$.

The results of the simulations are shown in Figure 1. In particular, Figure 1a presents the convergence of the collective strategy $\boldsymbol{x}^k$ to the v-GNE $\boldsymbol{x}^*$. Furthermore, Figure 1b highlights the convergence of the Lagrangian multipliers to consensus. We noticed that a simple update sequence, as the alphabetically ordered one, leads to a faster convergence than a random one. In general, the more the agents' updates are well mixed the faster the algorithm converge.

VI. Conclusion

This work propose a variant of the forward-backward splitting algorithm to solve generalized Nash equilibrium problems via asynchronous and distributed information exchange, that is robust to communication delays. A change of variables based on the node Laplacian matrix of the information-exchange graph allows one to preserve the scalability of the solution algorithm in the number of nodes (as opposed to the number of edges). A theoretical and numerical comparison between the proposed algorithm and that in [11] is left as future work.

References

[1] F. Dörfler, J. Simpson-Porco, and F. Bullo, "Breaking the hierarchy: Distributed control and economic optimality in microgrids," *IEEE Trans. on Control of Network Systems*, vol. 3, no. 3, pp. 241–253, 2016.

[2] F. Parise, M. Colombino, S. Grammatico, and J. Lygeros, "Mean field constrained charging policy for large populations of plug-in electric vehicles," in *Proc. of the IEEE Conference on Decision and Control*, Los Angeles, California, USA, 2014, pp. 5101–5106.

[3] S. Grammatico, "Proximal dynamics in multi-agent network games," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 4, pp. 1707–1716, 2018.

[4] S. Martínez, F. Bullo, J. Cortés, and E. Frazzoli, "On synchronous robotic networks – Part i: Models, tasks, and complexity," *IEEE Trans. on Automatic Control*, vol. 52, pp. 2199–2213, 2007.

[5] P. Yi and L. Pavel, "A distributed primal-dual algorithm for computation of generalized nash equilibria via operator splitting methods," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, Dec 2017, pp. 3841–3846.

[6] F. Facchinei, A. Fischer, and V. Piccialli, "On generalized Nash games and variational inequalities," *Operations Research Letters*, vol. 35, pp. 159–164, 2007.

[7] D. P. Bertsekas and J. N. Tsitsiklis, "Some aspects of parallel and distributed iterative algorithmsa survey," *Automatica*, vol. 27, no. 1, pp. 3 – 21, 1991.

[8] P. Combettes and J. Pesquet, "Stochastic quasi-fejr block-coordinate fixed point iterations with random sweeping," *SIAM Journal on Optimization*, vol. 25, no. 2, pp. 1221–1248, 2015.

[9] A. Nedic, "Asynchronous broadcast-based convex optimization over a network," *IEEE Transactions on Automatic Control*, vol. 56, no. 6, pp. 1337–1351, 2011.

[10] Z. Peng, Y. Xu, M. Yan, and W. Yin, "Arock: an algorithmic framework for asynchronous parallel coordinate updates," *SIAM Journal on Scientific Computing*, vol. 38, no. 5, pp. A2851–A2879, 2016.

[11] P. Yi and L. Pavel, "Asynchronous distributed algorithm for seeking generalized nash equilibria," *Proc. of the IEEE European Control Conference (ECC)*, pp. 2164–2169, June 2018.

[12] G. Belgioioso and S. Grammatico, "Projected-gradient algorithms for generalized equilibrium seeking in aggregative games arepreconditioned forward-backward methods," in *2018 European Control Conference (ECC)*. IEEE, 2018, pp. 2188–2193.

[13] D. Paccagnan, B. Gentile, F. Parise, M. Kamgarpour, and J. Lygeros, "Distributed computation of generalized nash equilibria in quadratic aggregative games with affine coupling constraints," pp. 6123–6128, Dec 2016.

[14] A. A. Kulkarni and U. Shanbhag, "On the variational equilibrium as a refinement of the generalized Nash equilibrium," *Automatica*, vol. 48, pp. 45–55, 2012.

[15] A. Dreves, F. Facchinei, C. Kanzow, and S. Sagratella, "On the solution of the kkt conditions of generalized nash equilibrium problems," *SIAM Journal on Optimization*, vol. 21, no. 3, pp. 1082–1108, 2011.

[16] F. Facchinei and J. Pang, *Finite-dimensional variational inequalities and complementarity problems*. Springer Verlag, 2003.

[17] C. Godsil and G. Royle, *Algebraic Graph Theory*, ser. Graduate Texts in Mathematics. Springer Science & Business Media, 2013, vol. 207.

[18] J. Eckstein, "Splitting methods for monotone operators with applications to parallel optimization," Ph.D. dissertation, Massachusetts Institute of Technology, 1989.

[19] H. H. Bauschke, P. L. Combettes *et al.*, *Convex analysis and monotone operator theory in Hilbert spaces*. Springer, 2011, vol. 408.

[20] C.-K. Yu, M. van der Schaar, and A. H. Sayed, "Distributed learning for stochastic generalized nash equilibrium problems," *IEEE Transactions on Signal Processing*, vol. 65, no. 15, pp. 3893–3908, 2017.