2018 European Control Conference (ECC)
June 12-15, 2018. Limassol, Cyprus

Asynchronous distributed algorithm for seeking generalized Nash equilibria

Peng Yi and Lacra Pavel

Abstract— **In this paper, we propose an asynchronous distributed algorithm with delayed information for computing a generalized Nash equilibrium over multi-agent systems. We consider a game where all players' local decisions are coupled via a shared affine constraint. We assume each player can only access its local objective function, local constraint, and a local block matrix of the affine constraint. With the help of auxiliary edge variables and *edge Laplacian matrix*, each player can perform its local iteration in an asynchronous manner, using only local data and possibly *delayed neighbour information*, without any centralized clock coordination. Therefore, the algorithm fully exploits the local computation resource of each player, and reduces the idle time due to waiting for the "slowest" agent. The algorithm convergence is shown using asynchronous fixed-point iterations. Numerical studies verify its convergence.**

I. Introductions

In many large-scale noncooperative multi-agent networks, the agents (players) need to make local decisions to minimize/maximize their individual cost/utility, for e.g., power allocation in cognitive radio networks, [1], [2], demand response and electric-vehicle charging management in smart grids, [3]-[6], rate/power control over optical networks, [7], and opinion evolution over social networks, [8]. Each player's local objective function to be optimized depends on other players' decisions, and their feasible decision sets may also couple through shared/non-shared constraints. Generalized Nash Equilibrium (GNE) is proposed as a reasonable solution to noncooperative games with coupling constraints, where no player can decrease/increase its cost/utility by unilaterally changing its local decision to another feasible one. Interested readers can refer to [9] for a review on GNE.

Recently, distributed NE/GNE computation methods over large-scale networks have received increasing research attention, see [2]-[6] and [10]-[19]. When using distributed GNE seeking methods, each player keeps its local objective function, local feasible set and local constraints, and the data does not need to be transferred to a center. Various game models and information structures have been considered. [6] proposed a primal-dual approach for GNE seeking with diminishing step-sizes for pseudo-monotone games. [4], [5] and [11] proposed fixed-point iteration based approaches for NE/GNE computation of aggregative games with/without affine coupling constraints, but also adopted a centralized coordinator. [17] combined projected continuous-time dynamics and finite-time consensus dynamics for aggregative games and eliminated the requirement for centralized coordinator by introducing local multipliers and local estimators for aggregative variables. [13]-[15] considered gradient-based NE seeking in general games, where each player can estimate other players' decisions by local communication. [10] considered GNE seeking for monotone games based on a Tikhonov regularized primal-dual algorithm with a centralized multiplier coordinator. [18] proposed a center-free GNE seeking algorithm for strongly monotone games with fixed step-sizes, assuming each player has access to all players' decisions that affect its cost. [19] proposed GNE seeking for monotone games based on double-layer preconditioned proximal algorithms.

However, all the above works considered only *synchronous* GNE computation algorithms, which need a global clock or a coordinator to ensure that all players have finished their current iteration before executing the next one. This is quite computationally inefficient since all players have to wait for the "slowest" player to finish before carrying on their local iterations. Meanwhile, the centralized coordinator needs to guarantee that for their computations, all players utilize the most recent information at the same iteration index. Hence, the synchronicity requirement limits their application and efficiency in large-scale networks. In fact, asynchronous computation has always been an appealing feature for algorithms over multi-agent networks. For example, [20] proposed a distributed optimization algorithm with asynchrony and delays, which enjoys a high computational efficiency compared to the synchronous ones. [2] investigated asynchronous NE computation for monotone games, and [21] investigated asynchronous NE computation for stochastic games, both with proximal best-response algorithms. Moreover, naively running the synchronous algorithms in an asynchronous manner will not always converge. Hence, it is important to design and analyze asynchronous GNE seeking algorithms with convergence guarantees.

Motivated by the above, the main contribution of this paper is to propose a novel *asynchronous distributed GNE* seeking algorithm for noncooperative games with linear equality coupling constraints. The equality constraint is motivated by various task allocation problems when a global requirement should be exactly met by all players' decisions, like electric-vehicle charging management. The algorithm is motivated by the preconditioned forward-back operator splitting method, which was also adopted in [18]. However,

This work was supported by NSERC Discovery Grant (261764).

P. Yi is with Department of Electrical and Systems Engineering, Washington University in St. Louis, USA, and L. Pavel is with Department of Electrical and Computer Engineering, University of Toronto, Canada. yipeng@wustl.edu,pavel@control.toronto.edu

due to the sequential updating order of the algorithm in [18], it is not trivial to develop an asynchronous algorithm based on [18]. In this paper, we achieve this with the help of auxiliary edge variables and *edge Laplacian*, unlike [18], where nodal variables and Laplacian are used. We propose a totally asynchronous distributed algorithm, which can rely on delayed information, thus further eliminating the need for coordinating information consistence in terms of iteration index. Moreover, the proposed algorithm reduces idle time, and each player can carry on its local iteration as soon as the delayed neighbour information is available. Assumptions similar to those in [18], [16], [4] and [5] are adopted. The convergence analysis is based on [22] for asynchronous fixed-point iterations, under bounded delay assumption for a sufficient *fixed step-size* choice.

The paper is organized as follows. Section II gives the notations and preliminary background. Section III formulates the noncooperative game and assumptions. Section IV presents our asynchronous distributed GNE computation algorithm. Section V presents the algorithm development and convergence analysis. Section VI presents a numerical study, and Section VII draws the concluding remarks. Due to page limitations, the omitted proofs can be found in [23].

II. NOTATIONS AND PRELIMINARIES

In this section, we review the notations and preliminary notions in monotone and averaged operators from [24].

Notations: In the following, $\mathbf{R}^m$ ($\mathbf{R}^m_+$) denotes the $m-$dimesional (nonnegative) Euclidean space. For a column vector $x \in \mathbf{R}^m$ (matrix $A \in \mathbf{R}^{m\times n}$), x^T (A^T) denotes its transpose. $x^T y = \langle x, y\rangle$ denotes the inner product of x, y, and $||x|| = \sqrt{x^T x}$ denotes the norm induced by inner product $\langle\cdot,\cdot\rangle$. $||x||^2_G$ denotes $\langle x, Gx\rangle$ for a symmetric positive definite matrix G, and $||x||_G = \sqrt{\langle x, Gx\rangle}$ denotes $G-$induced norm. Denote $\mathbf{1}_m = (1, ..., 1)^T \in \mathbf{R}^m$ and $\mathbf{0}_m = (0, ..., 0)^T \in \mathbf{R}^m$. $diag\{A_1, ..., A_N\}$ represents the block diagonal matrix with matrices $A_1, ..., A_N$ on its main diagonal. Denote $col(x_1, ..., x_N)$ as the column vector stacked with vectors $x_1, ..., x_N$. I_n denotes the identity matrix in $\mathbf{R}^{n\times n}$. For a matrix $A = [a_{ij}]$, a_{ij} or $[A]_{ij}$ stands for the matrix entry in the ith row and jth column of A. Denote $int(\Omega)$ as the interior of Ω and $ri(\Omega)$ as the relative interior of Ω. Denote $\times_{i=1,...,N}\Omega_i$ or $\prod_{i=1}^N \Omega_i$ as the Cartesian product of the sets $\Omega_i, i = 1, ..., N$. For vectors (matrices) a and b, $a\otimes b$ is their Kronecker product, and $a\odot b$ is their Hadamard product when their dimensions are the same.

Let $\mathfrak{A} : \mathbf{R}^m \to 2^{\mathbf{R}^m}$ be a set-value operator. Denote Id as the identity operator, i.e, $\mathrm{Id}(x) = x$. The domain of $\mathfrak{A}$ is $dom\mathfrak{A} = \{x \in \mathbf{R}^m | \mathfrak{A}x \neq \emptyset\}$ where $\emptyset$ stands for the empty set, its range is $ran\mathfrak{A} = \{y|\exists x, y \in \mathfrak{A}x\}$, and its graph is $gra\mathfrak{A} = \{(x, u)|u \in \mathfrak{A}x\}$, then the graph of its inverse is $gra\mathfrak{A}^{-1} = \{(u, x)|(x, u) \in gra\mathfrak{A}\}$. The zero set of $\mathfrak{A}$ is $zer\mathfrak{A} = \{x|\mathbf{0} \in \mathfrak{A}x\}$. The sum of $\mathfrak{A}$ and $\mathfrak{B}$ is defined as $gra(\mathfrak{A} + \mathfrak{B}) = \{(x, y + z)|(x, y) \in gra\mathfrak{A}, (x, z) \in gra\mathfrak{B}\}$. Define the *resolvent* of $\mathfrak{A}$ as $R_{\mathfrak{A}} = (\mathrm{Id} + \mathfrak{A})^{-1}$.

Operator $\mathfrak{A}$ is called monotone if $\forall(x, u), \forall(y, v) \in gra\mathfrak{A}$, we have $\langle x - y, u - v\rangle \geq 0$. Moreover, it is maximally monotone if $gra\mathfrak{A}$ is not *strictly* contained in the graph of any other monotone operator. A skew-symmetric matrix $A = -A^T$ defines a maximally monotone operator Ax (Example 20.30 of [24]). Suppose $\mathfrak{A}$ and $\mathfrak{B}$ are maximally monotone operators and $0 \in int(dom\mathfrak{A} - dom\mathfrak{B})$, then $\mathfrak{A} + \mathfrak{B}$ is also maximally monotone. For a closed convex set Ω, denote $N_\Omega(x)$ as the normal cone operator of Ω, and $N_\Omega(x) = \{v|\langle v, y - x\rangle \leq 0, \forall y \in \Omega\}$ and $domN_\Omega = \Omega$. N_Ω is maximally monotone. Define the projection of x onto set Ω as $P_\Omega(x) = \arg\min_y ||x - y||^2_2$, then $P_\Omega(x) = R_{N_\Omega}(x)$.

For a single-valued operator $T : \Omega \subset \mathbf{R}^m \to \mathbf{R}^m$, a point $x \in \Omega$ is a fixed point of T if $Tx = x$, and the set of fixed points of T is denoted as $FixT$. T is nonexpansive if it is $1-$Lipschitzian, i.e., $||T(x) - T(y)|| \leq ||x - y||, \forall x, y \in \Omega$. Let $\alpha \in (0, 1)$, then T is $\alpha-$averaged if there exists a nonexpansive operator T' such that $T = (1 - \alpha)\mathrm{Id} + \alpha T'$. Denote the class of $\alpha-$averaged operators as $\mathcal{A}(\alpha)$. If $T \in \mathcal{A}(\frac{1}{2})$, then T is called firmly nonexpansive. An operator T is $\beta-$cocoercive if $\beta T \in \mathcal{A}(\frac{1}{2})$. If operator $\mathfrak{A}$ is maximally monotone, then $T = R_{\mathfrak{A}} = (\mathrm{Id} + \mathfrak{A})^{-1} \in \mathcal{A}(\frac{1}{2})$ and $domR_{\mathfrak{A}} = \mathbf{R}^m$ (Proposition 23.7 of [24]).

III. GAME FORMULATION

Consider a set of players (agents) $\mathcal{N} = \{1, \cdots, N\}$ involved in the following noncooperative game with shared coupling constraints. Player $i \in \mathcal{N}$ controls its own decision (strategy or action) $x_i \in \Omega_i \subset \mathbf{R}^{n_i}$, where Ω_i is its private feasible decision set. Let $\mathbf{x} = col(x_1, \cdots, x_N) \in \mathbf{R}^n$ denote the decision profile, i.e., the stacked vector of all agents' decisions, with $\sum_{i=1}^N n_i = n$. Let $\mathbf{x}_{-i} = col(x_1, \cdots, x_{i-1}, x_{i+1}, \cdots, x_N)$ denote the decision profile of all agents except player i. Player i aims to optimize its own objective function within its feasible set, $f_i(x_i, \mathbf{x}_{-i}) : \bar{\Omega} \to \mathbf{R}$ where $\bar{\Omega} = \prod_{i=1}^N \Omega_i \subset \mathbf{R}^n$. Note that $f_i(x_i, \mathbf{x}_{-i})$ is coupled with other players' decisions $\mathbf{x}_{-i}$. Moreover, all players' decisions are coupled together through a *globally shared set* $X \subset \mathbf{R}^n$. Hence, player i has a set-valued map $X_i(\mathbf{x}_{-i}) : \mathbf{R}^{n-n_i} \to 2^{\mathbf{R}^{n_i}}$ that specifies its feasible set defined as $X_i(\mathbf{x}_{-i}) := \{x_i \in \Omega_i | (x_i, \mathbf{x}_{-i}) \in X\}$. Given $\mathbf{x}_{-i}$, player i's best-response strategy is

$$\min_{x_i} \ f_i(x_i, \mathbf{x}_{-i}), \ s.t., \ x_i \in X_i(\mathbf{x}_{-i}). \tag{1}$$

A generalized Nash equilibrium (GNE) $\mathbf{x}^*$ is defined at the intersection of all players' best-response sets, i.e., $\forall i \in \mathcal{N}$,

$$x_i^* \in \arg\min_{x_i} f_i(x_i, \mathbf{x}^*_{-i}), \ s.t., \ x_i \in X_i(\mathbf{x}^*_{-i}). \tag{2}$$

In this work, we consider the following coupling set,

$$X := \prod_{i=1}^N \Omega_i \bigcap \{\mathbf{x} \in \mathbf{R}^n | \sum_{i=1}^N A_i x_i = \sum_{i=1}^N b_i\}. \tag{3}$$

where $A_i \in \mathbf{R}^{m\times n_i}$ and $b_i \in \mathbf{R}^m$ as well as Ω_i are private data of player i. Thereby, the shared set X couples all players' feasible sets, but is **not known** by any agent. We consider the following assumption on the game in (1).

Assumption 1: Ω_i is a closed convex set with nonempty interiors, and X in (3) *has nonempty relative interiors. Given*

any $\mathbf{x}_{-i} \in \prod_{j=1,j\neq i}^{N} \Omega_i$*, the set* $X_i(\mathbf{x}_{-i})$ *has nonempty relative interiors. For player i,* $f_i(x_i, \mathbf{x}_{-i})$ *is a continuously differentiable convex function with respect to* x_i *given any fixed* $\mathbf{x}_{-i}$*. The pseudo-gradient of the game* (1)*, denoted as* $F(\mathbf{x})$ *and defined as*

$$F(\mathbf{x}) = col(\nabla_{x_1} f_1(x_1, \mathbf{x}_{-1}), \cdots, \nabla_{x_N} f_N(x_N, \mathbf{x}_{-N})). \tag{4}$$

is $\upsilon-$*strongly monotone over* Ω*:* $\langle F(\mathbf{x}) - F(\mathbf{y}), \mathbf{x} - \mathbf{y}\rangle \geq \upsilon||\mathbf{x}-\mathbf{y}||_2^2, \forall \mathbf{x}, \mathbf{y} \in \Omega$*, and* $\chi-$*Lipschitz continuous over* Ω*:* $||F(\mathbf{x}) - F(\mathbf{y})||_2 \leq \chi||\mathbf{x} - \mathbf{y}||_2, \forall \mathbf{x}, \mathbf{y} \in \Omega$.

With $F(\mathbf{x})$, we can define the *variational inequality* (VI) problem as follows,

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

Under Assumption 1, based on the Lagrangian duality for VI, $\mathbf{x}^*$ is a solution of VI (5) iff $\exists \lambda^* \in \mathbf{R}^m$ such that

$$\begin{aligned} &\mathbf{0} \in \nabla_{x_i} f_i(x_i^*, \mathbf{x}_{-i}^*) + A_i^T \lambda^* + N_{\Omega_i}(x_i^*), \quad \forall i \in \mathcal{N}, \\ &\mathbf{0} = \textstyle\sum_{i=1}^{N} (A_i x_i^* - b_i). \end{aligned} \tag{6}$$

Therefore, any solution to VI (5) is a GNE of the game in (1), which is termed as a *variational GNE* with all players having the same local multiplier.

Not every GNE of the game in (1) is a solution to (5). Since the variational GNE has an economic interpretation of no price discrimination and enjoys a stability property (refer to [1] and [18]), *we aim to propose a novel asynchronous distributed algorithm for computing a variational GNE.*

IV. Asynchronous distributed GNE computation algorithm

In this section, we propose an asynchronous distributed algorithm that players can use to find a solution of VI in (5), hence to compute a variational GNE of game in (1).

We focus on asynchronous distributed algorithms with delayed information because of following reasons. Firstly, player i can only manipulate its local $f_i(x_i, \mathbf{x}_{-i})$, A_i, b_i and Ω_i for local computation, since that local data contains its private information. Secondly, we assume there is no central node that has bidirectional communications with all players, which is vulnerable to single point failure. Thirdly, it is preferred that the players can compute asynchronously such that they perform local computations without waiting for the slowest agent to finish. By using asynchronous GNE seeking algorithms, there is no need for a global iteration counter, the idle time is reduced, and the computational resource of each player is fully exploited. Moreover, the local computation can rely on delayed information. In the following, we first introduce the communication graph and related auxiliary variables, and then we give the asynchronous distributed algorithm.

To facilitate the distributed computation, players are able to communicate with their neighbours through a *connected and undirected* graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$. The edge set $\mathcal{E} \subset \mathcal{N} \times \mathcal{N}$ contains all the information interactions. $(i, j) \in \mathcal{E}$ if agent i and j can share information with each other, and j, i are called neighbours. A path of graph $\mathcal{G}$ is a sequence of distinct agents in $\mathcal{N}$ such that any consecutive agents in the sequence are neighbours. Agent j is said to be connected to agent i if there is a path from j to i. $\mathcal{G}$ is connected since any two agents are connected.

Obviously, $|\mathcal{N}| = N$, and we denote $|\mathcal{E}| = M$. The edges are also labeled with e_l, $l = 1, ..., M$. Without loss of generality, $e_l = (i, j)$ is *arbitrarily* ordered as $i \to j$. Define $\mathcal{E}_i^{in}$ and $\mathcal{E}_i^{out}$ for agent i as follows: $e_l \in \mathcal{E}_i^{in}$ if agent i is the targeted point of e_l; $e_l \in \mathcal{E}_i^{out}$ if agent i is the starting point of e_l. Then denote $\mathcal{E}_i = \mathcal{E}_i^{in} \bigcup \mathcal{E}_i^{out}$ as the set of edges adjoint to agent i. Define the incidence matrix of $\mathcal{G}$ as $V \in \mathbf{R}^{N\times M}$ with $V_{il} = 1$ if $e_l \in \mathcal{E}_i^{in}$, and $V_{il} = -1$ if $e_l \in \mathcal{E}_i^{out}$, otherwise $V_{il} = 0$. We have $\mathbf{1}_N^T V = \mathbf{0}_M^T$, and $V^T x = \mathbf{0}_M$ if and only if $x \in \{\alpha \mathbf{1}_N | \alpha \in \mathbf{R}\}$ since $\mathcal{G}$ is connected. Denote $\mathcal{N}_l = \{i, j\}$ as the pair of agents connected by edge $e_l = (i, j)$. We also define the node neighbour set of each agent i as $\mathbb{N}_i = \{j | \mathcal{E}_i \cap \mathcal{E}_j \neq \emptyset\}$.

With the incidence matrix V, we can define the *edge Laplacian* of $\mathcal{G}$ as $L^e = V^T V$. Clearly $L^e \in \mathbf{R}^{M\times M}$ is symmetric, and $[L^e]_{ll} = 2$; $[L^e]_{lp} \neq 0$ for $l \neq p$ if e_l and e_p are connected to the same node i, otherwise $[L^e]_{lp} = 0$. Moreover, $[L^e]_{lp} = 1$ if e_l and e_p share the same direction in term of i, that is either both point to i or both start from i; and $[L^e]_{lp} = -1$ if e_l and e_p have opposite directions in term of i, that is one points to i and the other starts from i. Define an edge neighbour set of e_l as $\mathcal{N}_l^e = \{e_p | [L^e]_{lp} \neq 0\}$. Notice that $e_l \in \mathcal{N}_l^e$. Moreover, we know that $\mathcal{N}_l^e = \bigcup_{j\in\mathcal{N}_l} \mathcal{E}_j$.

We introduce the variables used. Firstly, each player controls its local decision x_i and its *local multiplier* λ_i. Each player i has a local variable $B_i = A_i x_i - b_i$. According to KKT (6), in the steady-state all players should have the same local multipliers, i.e., $\lambda_i = \lambda^*, \forall i \in \mathcal{N}$. To facilitate the coordination for the consensus of local multipliers and to ensure the coupling constraint, we introduce an auxiliary variable $Z_l \in \mathbf{R}^m$ related to edge e_l of graph $\mathcal{G}$. Intuitively speaking, Z_l can be regarded as a resource "flow" on each edge to reach the balance of the equality constraint. Notice that $\mathcal{G}$ is undirected and the edges are *arbitrarily* ordered, therefore, any agent from $\mathcal{N}_l$ can maintain Z_l. For clarity, we consider that Z_l will be maintained by agent i if $e_l \in \mathcal{E}_i^{out}$, but Z_l is shared/known by the two players in $\mathcal{N}_l$. We assume that each player i has an output buffer where he writes to, which can be read by its neighbouring players $j \in \mathbb{N}_i$, and an input buffer he reads from, which contains $Z_l, e_l \in \mathcal{E}_i^{in}$ and can be written by its neighbour $j \in \mathbb{N}_i$, $j \in \mathcal{N}_l \backslash i$. We assume that player i can read the decision x_j from the output buffer of player j if $f_i(x_i, \mathbf{x}_{-i})$ *explicitly* depends on x_j, and that it has a local oracle that returns $\nabla_{x_i} f_i(x_i, \mathbf{x}_{-i})$, given $\mathbf{x}_{-i}$.

To describe the Asynchronous Distributed Algorithm for GNE Computation with Delays (ADAGNES), we assume that each agent has a (virtual) Poisson variable (clock), which can model the number of iterations in a given time period. Then the ADAGNES is given as:

Algorithm 1 (ADAGNES):

Initialization: Player i picks $x_{i,0} \in \Omega_i$, $Z_{l,0} \in \mathbf{R}^m$, $e_l \in \mathcal{E}_i^{out}$ and $\lambda_{i,0} \in \mathbf{R}^m$, and has a local variable $B_{i,0} = A_i x_{i,0} - b_i$ and a Poisson clock with rata ς_i.

Iteration at k: Suppose player i_k's clock ticks at time k, then player i_k is active and updates its local variables:

Reading phase: Get information from its neighbours' output buffer and its input buffer. Duplicate its input buffer to its output buffer. Read $x_{j,k-\delta_j^k}$ if $f_{i_k}(x_{i_k}, \mathbf{x}_{-i_k})$ directly depends on x_j. If $j \in \mathbb{N}_{i_k}$, read $\lambda_{j,k-\delta_j^k}$ from the output buffer of j. If $e_l \in \mathcal{E}_{i_k}^{out}$ and $j \in \mathcal{N}_l \backslash i_k$, then read $B_{j,k-\delta_j^k}$ and $Z_{q,k-\pi_q^k}$, $e_q \in \mathcal{E}_j$ from the output buffer of j.

Computing phase:

i): update λ_{i_k},

$$\begin{aligned}\tilde{\lambda}_{i_k,k} &= \lambda_{i_k,k-\delta_{i_k}^k} + \sigma(A_{i_k} x_{i_k,k-\delta_{i_k}^k} - b_{i_k} \\ &\quad + \textstyle\sum_{e_l \in \mathcal{E}_{i_k}} V_{i_k l} Z_{l,k-\pi_l^k}) \\ \lambda_{i_k,k+1} &= \lambda_{i_k,k-\delta_{i_k}^k} + \eta(\tilde{\lambda}_{i_k,k} - \lambda_{i_k,k-\delta_i^k})\end{aligned} \tag{7}$$

ii): update x_{i_k},

$$\begin{aligned}\tilde{x}_{i_k,k} &= P_{\Omega_{i_k}}\big[x_{i_k,k-\delta_{i_k}^k} - \tau(\nabla_{i_k} f_{i_k}(x_{i_k,k-\delta_{i_k}^k}, \mathbf{x}_{-i_k,k-\delta^k}) \\ &\quad + A_{i_k}^T(2\tilde{\lambda}_{i_k,k} - \lambda_{i_k,k-\delta_{i_k}^k}))\big] \\ x_{i_k,k+1} &= x_{i_k,k-\delta_{i_k}^k} + \eta(\tilde{x}_{i_k,k} - x_{i_k,k-\delta_{i_k}^k})\end{aligned} \tag{8}$$

iii): If $e_l \in \mathcal{E}_{i_k}^{out}$ and $j_k \in \mathbb{N}_{i_k}$, $j_k \in \mathcal{N}_l \backslash i_k$, then update $Z_{l,k}$:

$$\begin{aligned}Z_{l,k+1} &= Z_{l,k-\pi_l^k} - \eta\gamma(\lambda_{j_k,k-\delta_{j_k}^k} - \lambda_{i_k,k-\delta_{i_k}^k}) \\ &\quad -2\eta\sigma\gamma\big[(B_{j_k,k-\delta_{j_k}^j} - B_{i_k,k-\delta_{i_k}^k}) + \textstyle\sum_{q\in\mathcal{N}_l^e} L_{lq}^e Z_{q,k-\pi_q^k}\big]\end{aligned} \tag{9}$$

Writing phase: Write $x_{i_k,k+1}$, $\lambda_{i_k,k+1}$, $Z_{l,k+1}, e_l \in \mathcal{E}_{i_k}^{out}$ and $B_{i_k,k+1} = A_{i_k} x_{i_k,k+1} - b_{i_k}$ to its output buffer, and write $Z_{l,k+1}$, $e_l \in \mathcal{E}_{i_k}^{out}$ to the input buffer of player $j_k \in \mathcal{N}_l \backslash i_k$.

All other players keep their variables (output buffers) unchanged, and only increase k to $k+1$.

In the foregoing, σ, γ, τ and η are real positive step-sizes, and $x_{i,k}, \lambda_{i,k}$ and $Z_{l,k}$ are x_i, λ_i and Z_l, respectively, at iteration k. δ_i^k and π_l^k are the delays of x_i (as well as λ_i and B_i) and Z_l, respectively, to characterize the information used by the active player i at time k. $\mathbf{x}_{-i,k-\delta^k}$ is the vector composed with the delayed variable $x_{j,k-\delta_j^k}$ which directly influences $f_i(x_i, \mathbf{x}_{-i})$ of the active player i at time k.

Remark 1: The delays arise from various sources. It is possible that one agent has finished the writing phase (index k will increase 1) before another agent finishes its computation, in which case the index of the information used by the agents in computation phase gets delayed by one. Overall, the delays account for the time due to reading buffers, local computation and writing buffers. The algorithm is asynchronous since an agent does not need to wait for its neighbours to finish before doing its local iteration.

V. Algorithm development and analysis

In this section, we first show based on an operator theoretic approach how the algorithm without asynchrony and delays is developed. Then we give the convergence analysis for the asynchronous one with delays, based on fixed-point iterations, [22].

A. Algorithm development without asynchrony and delays

Let us consider the *Synchronous Distributed GEN Computation Algorithm without Delays*, denoted as SYDNEY. In this case, the update of agent i is given as:

Algorithm 2 (SYDNEY):

$$\begin{aligned}&\tilde{\lambda}_{i,k} = \lambda_{i,k} + \sigma(A_i x_{i,k} + \textstyle\sum_{e_l \in \mathcal{E}_i} V_{il} Z_{l,k} - b_i) \\ &\lambda_{i,k+1} = \lambda_{i,k} + \eta(\tilde{\lambda}_{i,k} - \lambda_{i,k}) \\ &\tilde{x}_{i,k} = P_{\Omega_i}\big[x_{i,k} - \tau(\nabla_i f_i(x_{i,k}, \mathbf{x}_{-i,k}) + A_i^T(2\tilde{\lambda}_{i,k} - \lambda_{i,k}))\big] \\ &x_{i,k+1} = x_{i,k} + \eta(\tilde{x}_{i,k} - x_{i,k}) \\ &Z_{l,k+1} = Z_{l,k} - \eta\gamma(\lambda_{j,k} - \lambda_{i,k}) - 2\eta\sigma\gamma(B_{j,k} - B_{i,k}) \\ &\quad -2\eta\sigma\gamma \textstyle\sum_{q\in\mathcal{N}_l^e} L_{lq}^e Z_{q,k}\end{aligned}$$

The notations are the same as in ADAGNES.

In fact, SYDNEY is derived by a preconditioned forward backward splitting algorithm for finding zeros of a sum of monotone operators, like [18]. Denote $\mathbf{x}_k = col(x_{1,k}, \cdots, x_{N,k})$, then $\bar{\lambda}_k$, $\mathbf{Z}_k$, and $\bar{b}$ are defined similarly. Denote $\Lambda = diag\{A_1, \cdots, A_N\}$ and $\bar{V} = V \otimes I_m$. Denote $\varpi = col(\bar{\lambda}, \mathbf{Z}, \mathbf{x})$, and define operators $\mathfrak{A}$ and $\mathfrak{B}$ as follows,

$$\mathfrak{A} : \varpi \mapsto \begin{pmatrix} -\Lambda\mathbf{x} - \bar{V}\mathbf{Z} \\ \bar{V}^T\bar{\lambda} \\ \Lambda^T\bar{\lambda} + N_{\bar{\Omega}}\mathbf{x} \end{pmatrix}, \mathfrak{B} : \varpi \mapsto \begin{pmatrix} \bar{b} \\ \mathbf{0} \\ F(\mathbf{x}) \end{pmatrix} \tag{10}$$

Then, SYDNEY can be written in a compact form as

$$\varpi_{k+1} = \varpi_k + \eta(T\varpi_k - \varpi_k). \tag{11}$$

where $\varpi_k = col(\bar{\lambda}_k, \mathbf{Z}_k, \mathbf{x}_k)$, and the matrix Φ and operator T are defined as follows,

$$\Phi = \begin{pmatrix} \sigma^{-1}I & \bar{V} & \Lambda \\ \bar{V}^T & \gamma^{-1}I & \mathbf{0} \\ \Lambda^T & \mathbf{0} & \tau^{-1}I \end{pmatrix}, \tag{12}$$

$$T = (\mathrm{Id} + \Phi^{-1}\mathfrak{A})^{-1}(\mathrm{Id} - \Phi^{-1}\mathfrak{B}). \tag{13}$$

Therefore, SYDNEY can be regarded as a fixed point iteration for operator T. Suppose σ, γ and τ are chosen such that Φ in (12) is positive definite. Then, we have $FixT = zer(\mathfrak{A} + \mathfrak{B})$ since $\varpi = T\varpi \Leftrightarrow \Phi\varpi - \mathfrak{B}\varpi \in \mathfrak{A}\varpi + \Phi\varpi \Leftrightarrow \mathbf{0} \in (\mathfrak{A} + \mathfrak{B})\varpi$. Next result relates the zero of $\mathfrak{A} + \mathfrak{B}$ to the variational GNE of the game.

Theorem 1: Suppose that Assumption 1 holds for game (1). Then, any zero $col(\bar{\lambda}^*, \mathbf{Z}^*, \mathbf{x}^*)$ of $\mathfrak{A} + \mathfrak{B}$ has the $\mathbf{x}^*$ component as a variational GNE of (1), and $\bar{\lambda}^* = \mathbf{1}_N \otimes \lambda^*$. $\mathbf{x}^*$ and λ^* satisfy KKT condition (6).

Under Assumption 1, it can be shown that T is an averaged operator under a proper chosen norm with Lemma 1 and Lemma 2.

Lemma 1: Suppose Assumption 1 holds. Operator $\mathfrak{A}$ in (10) is maximally monotone. Operator $\mathfrak{B}$ in (10) is $\frac{v}{\chi^2}$-cocoercive;

Lemma 2: Suppose Assumption1 holds. Take $\theta > \frac{\chi^2}{2\upsilon}$, and take the step-sizes τ, γ, σ in Algorithm 1 such that $\Phi - \theta I$ is positive semi-definite. Then under the $\Phi-$induced norm $||\cdot||_\Phi$, we have, (i): $\Phi^{-1}\mathfrak{A}$ is maximally monotone, and $T_1 = R_{\Phi^{-1}\mathfrak{A}} = (\mathrm{Id} + \Phi^{-1}\mathfrak{A})^{-1} \in \mathcal{A}(\frac{1}{2})$. (ii): $\Phi^{-1}\mathfrak{B}$ is $\frac{\theta\upsilon}{\chi^2}-$cocoercive, and $T_2 = \mathrm{Id} - \Phi^{-1}\mathfrak{B} \in \mathcal{A}(\frac{\chi^2}{2\theta\upsilon})$. (iii): T in (13) is an averaged operator, and $T \in \mathcal{A}(\frac{2\theta\upsilon}{4\theta\upsilon-\chi^2})$.

Hence, the convergence of (11) follows from the Krasnosel'skiĭ-Mann (K-M) algorithm by choosing $0 < \eta < \frac{4\theta\upsilon-\chi^2}{2\theta\upsilon}$ according to Theorem 5.14 of [24].

B. Convergence analysis for the asynchronous ADAGNES

In this subsection, we will give the convergence analysis of the asynchronous ADAGNES, Algorithm 1 by treating it as a randomized block-coordinate fixed-point iteration with delayed information, and using results in [22].

Denote $\varpi^i = col(x_i, \lambda_i, \{Z_l\}_{e_l \in \mathcal{E}_i^{out}})$ as the coordinates that should be updated by player i. Define a group of vectors $\Xi_1, \cdots, \Xi_N \in R^{n+Nm+Mm}$, such that for $j = 1, \cdots, n + Nm + Mm$, $[\Xi_i]_j = 1$ if the $j-$th coordinate of ϖ is also a coordinate of ϖ^i, and $[\Xi_i]_j = 0$, otherwise. Denote ξ as a random variable that takes values in $\{\Xi_1, \cdots, \Xi_N\}$, and $\mathbb{P}(\xi = \Xi_i) = \varsigma_i / \sum_{j=1}^N \varsigma_j$, where $\varsigma_i, i = 1, \cdots, N$ are the rates of the Poisson clock in Algorithm 1. Then let $\xi_k, k = 1, ...$ to be a sequence of i.i.d. random variables that are all equal to ξ. A randomized block-coordinate fixed-point iteration for (11), or equivalently for SYDNEY, is

$$\varpi_{k+1} = \varpi_k + \eta\xi_k \odot (T\varpi_k - \varpi_k). \tag{14}$$

In (14), only one agent i is activated at each iteration, with probability $\varsigma_i / \sum_{j=1}^N \varsigma_j$, which updates using SYDNEY. The convergence of (14) with nonexpansive T is fully investigated in [25]. However, (14) does not account for any delays. Rather it assumes that the operator T is applied to ϖ_k, which is the whole vector of all agents' iterates at the current time. This means that the communication and computation at each iteration should be performed "almost instantly", or that players should wait until the current iteration finishes before the next activation, such that each iteration is computed with the most recent information.

To remove this impractical assumption and to reduce the idle time of the players, we allow for delayed information when evaluating T in (14), which yields

$$\varpi_{k+1} = \varpi_k + \eta\xi_k \odot (T\hat{\varpi}_k - \varpi_k), \tag{15}$$

where $\hat{\varpi}_k$ is the delayed information at time k. The convergence of ADAGNES can be shown based on rewriting it as (15) for appropriately defined $\hat{\varpi}_k$, and resorting to the theoretical result in [22], on asynchronous coordinate updates of K-M iteration for nonexpansive operators.

Assumption 2: The maximal delay at each step is bounded by Ψ, i.e., $\sup_k \max_{i\in\mathcal{N}, l\in\mathcal{E}}\{\max\{\delta_i^k, \pi_l^k\}\} \leq \Psi$.

Theorem 2: Suppose Assumption 1 and 2 hold. Take $\theta > \frac{\chi^2}{2\upsilon}$, and take the step-sizes τ, γ, σ in Algorithm 1 such that $\Phi - \theta I$ is positive semi-definite. Denote $p_{min} = \min_{i\in\mathcal{N}}\{\frac{\varsigma_i}{\sum_{j=1}^N \varsigma_j}\}$, and choose $0 < \eta \leq \frac{cNp_{\min}}{2\Psi\sqrt{p_{\min}}+1} \frac{4\theta\upsilon-\chi^2}{2\theta\upsilon}$

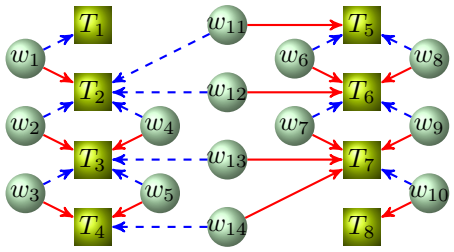

Fig. 1. Task allocation game: An edge from w_i to T_j on this graph implies that a part of worker w_i's output is allocated to task T_j.

for arbitrary $c \in (0, 1)$. Then with Algorithm 1, the players' decisions $\mathbf{x}_k$ converge to the variational GEN $\mathbf{x}^*$ of game in (1), and the players' local multipliers $\bar{\lambda}_k$ reach consensus and converge to the same λ^* with probability 1.

VI. Numerical studies

In this part, we consider a task allocation game with 8 tasks $\{T_1, \cdots, T_8\}$ and 14 processors (workers) $\{w_1, \cdots, w_{14}\}$. Each task T_j is quantified as a load of $C_j > 0$ that should be met by the workers. Each worker w_i decides its working output $x_i = col(x_i^1, x_i^2, x_i^3, x_i^4) \in \mathbf{R}^4$ within its capacity $\mathbf{0} \leq x_i \leq \Xi_i, \Xi_i \in \mathbf{R}_+^4$. If worker w_i allocates a part of its output to task T_j, there is an arrow $w_i \to T_j$ on Fig. 1. Specifically, if w_i allocates x_i^1, x_i^2 to T_j, there is a dashed blue arrow on Fig. 1, and if w_i allocates x_i^3, x_i^4 to T_j, there is a solid red arrow on Fig. 1. Define a matrix $A = [A_1, \cdots, A_{15}] \in \mathbf{R}^{8\times 56}$ with $A_i = [a_i^1, a_i^2, a_i^3, a_i^4] \in \mathbf{R}^{8\times 4}$ quantifying how the output of worker w_i is allocated to each task. Each column a_i^k has only one element being nonzero, and the jth element of a_i^1 or a_i^2 is nonzero if there is a dashed blue arrow $w_i \to T_j$ on Fig. 1, and the jth element of a_i^3 or a_1^4 is nonzero if there is a red solid arrow $w_i \to T_j$ on Fig. 1. The nonzero elements in A_i are randomly chosen from $[0.5, 1]$, which could be regarded as delivery loss factors. It is required that the tasks should be met by the working output of the players. Denote $C = col(C_1, \cdots, C_8)$, then the workers have an equality coupling constraint: $A\mathbf{x} = C$. The objective function of player (worker) w_i is $f_i(x_i, \mathbf{x}_{-i}) = c_i(x_i) - R^T(\mathbf{x})A_i x_i$. Here, $c_i(x_i)$ is a cost function, which is taken as $c_i(x_i) = \sum_{k=1}^4 q_i^k(x_i^k + 1)\log(x_i^k + 1) + (p_i^T x_i - d_i)^2 + x_i^T S_i x_i$. $R(\mathbf{x}) = col(R_1(\mathbf{x}), \cdots, R_8(\mathbf{x}))$ is a vector function that maps $\mathbf{x}$ to the award price of each task, and $R_j(\mathbf{x}) = \kappa_j - \chi_j[A\mathbf{x}]_j$.

The parameters of the problem are randomly drawn as follows: $C_j \in [1, 2]$, $\chi_j \in [0.5, 1]$, $\kappa_j \in [4, 9]$, $q_i^k \in [0.5, 1.5]$, $d_i \in [1, 2]$. $p_i \in \mathbf{R}^4$ is a randomly generated stochastic vector, $S_i \in \mathbf{R}^{4\times 4}$ is a randomly generated positive definite matrix, and each element of Ξ_i is drawn from $[1, 2]$. The parameters are numerically checked to ensure Assumption 1. The players communicates over a ring graph with alphabetic order, and edges are arbitrarily ordered. Here we assume that $\varsigma_1 =, \cdots, = \varsigma_N$, hence $\mathbb{P}(i_k = i) = \frac{1}{N}$. The maximal delay Ψ is taken to be 7 while the $\hat{\varpi}_k$ in (15) is constructed with each ϖ^i randomly chosen from $\{\varpi_{k-\Psi}^i, \cdots, \varpi_k^i\}$. Each

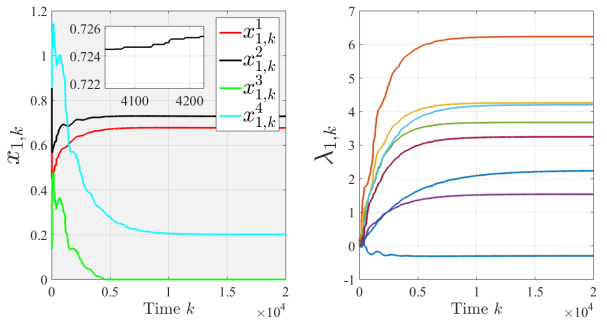

Fig. 2. The trajectories of player 1's $x_{1,k}$ and $\lambda_{1,k}$.

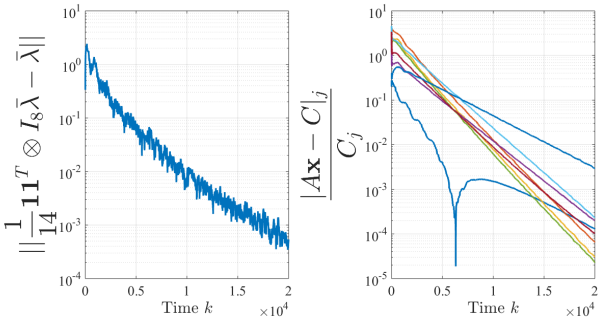

Fig. 3. The trajectories of the consensual error of all λ_i and the violation of equality constraint.

player has a local $C_i = \frac{1}{15}C$, and the step-sizes σ, γ, τ are randomly chosen from $[0.3, 0.5]$ such that Φ is positive definite, and $\eta = 0.4$. The initial $x_{i,0}$ is randomly chosen within $[\mathbf{0}, \Xi_i]$, and initial λ_i, z_l are chosen to be zero. We adopt the proposed asynchronous algorithm, ADAGNES. Fig 2 and 3 show trajectories generated by ADAGNES, demonstrating its convergence.

VII. Conclusions

In this paper, we proposed an asynchronous distributed variational GNE computation algorithm for games with affine coupling constraints. With the help of edge variables and edge Laplacian matrix, eah agent can perform local iterations with local data and delayed neighbour information without any synchronization. A sufficient step-size choice is given for algorithm convergence. Our simulations verify only the convergence of the proposed algorithm. It would be interesting to perform the experiments on actual distributed systems and study the efficiency of the asynchronous algorithm compared with synchronous ones. Also interesting is to consider the case when players do not have access to all decisions that influence their local objective function, by using local consensus dynamics like [13]-[15].

References

[1] F. Facchinei, and J.S. Pang. "Nash equilibria: the variational approach". in *Convex optimization in signal processing and communications*, pp: 443-493, Cambridge University Press, 2010.

[2] G. Scutari, F. Facchinei, J.S. Pang, and D.P. Palomar. "Real and complex monotone communication games". *IEEE Transactions on Information Theory*, 60(7): 4197-4231, 2014.

[3] M. Ye, and G. Hu. "Game design and analysis for price-based demand response: An aggregate game approach". *IEEE transactions on cybernetics*, 47(3): 720-730, 2017.

[4] S. Grammatico. "Aggregative control of competitive agents with coupled quadratic costs and shared constraints". *55th IEEE Conference on Decision and Control (CDC)*, pp: 3597-3602, 2016.

[5] 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." In *55th IEEE Conference on Decision and Control (CDC)*, pp: 6123-6128, 2016.

[6] M. Zhu, and E. Frazzoli. "Distributed robust adaptive equilibrium computation for generalized convex games". *Automatica*, 63: 82-91, 2016.

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

[8] F. Parise, S. Grammatico, B. Gentile, and J. Lygeros. "Network aggregative games and distributed mean field control via consensus theory." *arXiv preprint*, arXiv:1506.07719, 2015.

[9] A. Fischer, M. Herrich, and L. Schonefeld. "Generalized Nash equilibrium problems-recent advances and challenges". *Pesquisa Operacional*, 34(3): 521-558, 2014.

[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*, 56(7): 1702-1708, 2011.

[11] S. Grammatico, F. Parise, M. Colombino, and J. Lygeros. "Decentralized convergence to Nash equilibria in constrained deterministic mean field control." *IEEE Transactions on Automatic Control*, 61(11): 3315-3329, 2016.

[12] 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*, 61(10): 2920-2935, 2016.

[13] F. Salehisadaghiani, and L. Pavel. "Distributed Nash equilibrium seeking: A gossip-based algorithm". *Automatica*, 72: 209-216, 2016.

[14] F. Salehisadaghiani, and L. Pavel. "Distributed Nash Equilibrium Seeking via the Alternating Direction Method of Multipliers". In *IFAC PapersOnLine 50-1*, pp: 6166-6171, 2017.

[15] D. Gadjov, and L. Pavel. Continuous-time distributed dynamics for Nash Equilibrium over networks via a passivity-based control approach. In *Proceedings of the IEEE Conference on Decision and Control (CDC)*, pp: 4600-4605, 2017.

[16] C.K. Yu, M. van der Schaar, and A.H. Sayed. "Distributed Learning for Stochastic Generalized Nash Equilibrium Problems". *IEEE Transactions on Signal Processing*, 65(15): 3893 - 3908, 2017.

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

[18] P. Yi, and L. Pavel. "A distributed primal-dual algorithm for computation of generalized Nash equilibria via operator splitting methods". In *Proceedings of the IEEE Conference on Decision and Control (CDC)*, pp: 3841-3846, 2017.

[19] P. Yi, and L. Pavel. "Distributed generalized Nash equilibria computation of monotone games via double-layer preconditioned proximal-point algorithms". *IEEE Transaction on Control of Network Systems*, accepted, 2017.

[20] T. Wu, K. Yuan, Q. Ling, W. Yin, and A. H. Sayed, "Decentralized Consensus Optimization with Asynchrony and Delay", *IEEE T. Signal and Information Processing over Networks*, DOI: 10.1109/TSIPN.2017.2695121, 2017.

[21] J. Lei, U. V. Shanbhag, J-S Pang, and S. Sen. "On Synchronous, Asynchronous, and Randomized Best-Response schemes for computing equilibria in Stochastic Nash games. *arXiv preprint*, arXiv:1704.04578, 2017.

[22] Z. Peng, Y. Xu, M. Yan, and W. Yin. "ARock: an algorithmic framework for asynchronous parallel coordinate updates." *SIAM Journal on Scientific Computing*, 38(5): A2851-A2879, 2016.

[23] P. Yi, and L. Pavel. "Asynchronous distributed algorithm for seeking generalized Nash equilibria." *arXiv preprint*, arXiv:1801.02967, 2018.

[24] H.H. Bauschke, and P. L. Combettes. *Convex analysis and monotone operator theory in Hilbert spaces*. Springer Science & Business Media, 2011.

[25] P. L. Combettes, and J. Pesquet. "Stochastic quasi-Fejér block-coordinate fixed point iterations with random sweeping." *SIAM Journal on Optimization*, 25(2): 1221-1248, 2015.