A Survey of Decentralized Online Learning

Xiuxian Li, Lihua Xie, and Na Li

Abstract—Decentralized online learning (DOL) has been increasingly researched in the last decade, mostly motivated by its wide applications in sensor networks, commercial buildings, robotics (e.g., decentralized target tracking and formation control), smart grids, deep learning, and so forth. In this problem, there are a network of agents who may be cooperative (i.e., decentralized online optimization) or noncooperative (i.e., online game) through local information exchanges, and the local cost function of each agent is often time-varying in dynamic and even adversarial environments. At each time, a decision must be made by each agent based on historical information at hand without knowing future information on cost functions. Although this problem has been extensively studied in the last decade, a comprehensive survey is lacking. Therefore, this paper provides a thorough overview of DOL from the perspective of problem settings, communication, computation, and performances. In addition, some potential future directions are also discussed in details.

Index Terms—Decentralized algorithms, autonomous agents, online optimization, online game, multi-agent networks, regret.

I. INTRODUCTION

Optimization and game have been extremely popular topics since the last century, which are ubiquitous across many realms, including computer science, systems and control, finance, biology, medical service, mathematics, machine learning, artificial intelligence, robotics, and so on [1]–[3]. Optimization and game have a similar objective, that is, seeking optimal decision vectors/variables, while the difference lies in that for a game, there are usually multiple agents (or players), each aims at computing its own best decision in a noncooperative fashion. It is worth noting that the environments in typical optimization and game are often stationary, i.e., the objective/cost/loss functions are time-invariant, which are of limited use in a multitude of practical applications in dynamic environments, such as the moving target (e.g., robot) tracking problem under either good or atrocious weather condition where the optimal variable can be viewed as the target's position at each time, which is time-varying as time evolves (note that the target is a part of the environment) [4].

In dynamic environments, online learning (OL) for optimization and game (or online optimization/game), as a notable tool for sequential decision making, has become an active research field in recent two decades, mostly because it can well handle a large number of realistic problems in portfolio management, auctions, transportation, smart grids, robotics, dictionary learning, online advertisement placement, online web ranking, neural networks, to just mention a few [5]–[7]. Online learning usually has three main features: 1) the decision maker does not have access to future information on cost functions in general; 2) the variation of cost functions is generally not imposed to obey any statistical distributions; 3) the environments may be even adversarial, i.e., intentionally preventing the decision maker from achieving the best decision (cf. surveys of centralized online learning [8]–[11]). With the above features, the decision maker has to choose a decision at each time instant only based on historical information at hand, and then the current objective information is revealed. In this setup, it is well known that no algorithms can be leveraged to exactly track the trajectory of best decisions or optimal variables, and usually two metrics, i.e., regret and competitive ratio, are introduced to measure the performance of proposed algorithms which basically drive the incurred total cost over a finite horizon to track the lowest cost achieved if knowing all the past and future cost functions in hindsight [5], [12], [13].

With the rapid development of science and technology, as well as the advent of large-scale network and big data in modern life, some limitations of the aforementioned so-called centralized online learning has emerged, for example, no one agent can possess all the information of an optimization/game problem at any time slot, which, however, is usually distributed over a group of agents who may be geographically dispersed [14]. In this case, decentralized online learning (DOL) has become a focal research topic in the last decade, where for optimization (i.e., decentralized online optimization (DOO)),

Fig. 1. Schematic illustration of a general framework for DOL, where $x_{i,t}$ is the decision vector/variable made by agent $i \in [N]$ at time step $t$, the environment is dynamic and even adversarial, $f_t$ is given in a generic form, which can be separable or nonseparable as studied in the literature, and all agents in the network can be cooperative or noncooperative.
the global cost function at each time step is inaccessible to any agent, while each of a collection of agents holds partial information on the global cost function and they cooperate to solve the global online optimization by information propagation to their local neighbors, and for game (i.e., online game), each agent is often unaware of the cost functions and strategies of other agents at each time. A schematic illustration of DOL’s framework is presented in Fig. 1. Compared with centralized online learning, DOL enjoys a plethora of prominent advantages, including privacy preserving, robustness to channel failures, resiliency to cyber-attacks, alleviation of computational burden to a centralized node, etc. [15].

Along this line, this paper aims to provides a comprehensive survey on DOL over multi-agent networks by reviewing over one hundred papers published in the last decade, encapsulated primarily from four perspectives: problem settings, communication issues, computation complexity, and performances, including full state information, communication delays, asynchronous algorithms, privacy-preserving, security-performance, including full state information, communication issues, computation complexity, and etc. To our best knowledge, this paper is the first to report an overview of DOL, which hopefully can motivate and facilitate further study in this field.

Notations. Let $[N] := \{1, 2, \ldots, N\}$ denote an integer set for an integer $N > 0$ and $\times_{i=1}^N X_i$ be the Cartesian product of set $X_i$’s. For simplicity, denote by $1$ and $0$ column vectors of all entries being $1$ and $0$, respectively, having compatible dimensions from the context, and $I_n$ the identity matrix of dimension $n \times n$. Let $\otimes$ be the Kronecker product. Denote by $\text{col}(x_1, \ldots, x_N)$ the column vector by piling up vectors $x_1, i \in [N]$ and $x \in \mathbb{R}^n$ the transpose of $x \in \mathbb{R}^n$. Denote by $P_X(\cdot)$ the projection operator onto a closed convex set $X \subseteq \mathbb{R}^n$ and $[\cdot]_+$ the projection operator onto the nonnegative orthant $\mathbb{R}^n_+$, i.e., $\mathbb{R}^n_+ := \{x = \text{col}(x_1, \ldots, x_n) \in \mathbb{R}^n | x_i \geq 0, \forall i \in [n]\}$. Let $\| \cdot \|$ denote the dual norm, i.e., $\| x \|_* := \sup_{\| y \| \leq 1} \langle x, y \rangle$.

In this setup, there are a group of $N$ agents, who hold their own private information and constitute a multi-agent network where each agent can communicate with its local neighbors via information exchange. At each time step $t \geq 1$, each agent $i \in [N]$ makes a decision $x_{i,t} \in X_i$ generally based on historical information at hand, i.e., partial or complete information on $f_i, X_i, l \leq t - 1$, and then the environment will reveal some partial or complete information on $f_i$ and $X_i$ to agent $i$ along with a suffered loss $f_i(x_{i,1}, \ldots, x_{N,t})$. After that, each agent interacts with its local neighbors by sharing its partial information, and then it continues the above similar process in next time instant until a pre-specified horizon $T > 0$ is reached.

Note that (1) is given in a general framework, which can be roughly classified into three main settings in the literature, i.e., consensus based DOL, multi-agent coordination based DOL, and online game, as introduced below.

1) Consensus based DOL. In this setting, the global cost function $f_i$ is in the form

$$f_i(x) = \sum_{i=1}^{N} f_{i,t}(x_i), \quad \text{s.t.} \quad x_i = x_j, \forall i,j \in [N], \quad (2)$$

where $x := \text{col}(x_1, \ldots, x_N) \in \mathbb{R}^{nN}$ and each $f_{i,t}$ is revealed only to agent $i$ at each time after making decision $x_{i,t}$. The goal is usually to minimize the cumulative cost over the total horizon as follows:

$$\min_{x_t \in \times_{i=1}^N X_i} \sum_{t=1}^{T} f_i(x_t), \quad (3)$$

2) Multi-Agent Coordination based DOL. In this scenario, $f_i$ is of the form

$$f_i(x_1, \ldots, x_N) := \sum_{i=1}^{N} f_{i,t}(x_1, \ldots, x_N), \quad (4)$$

where $f_{i,t}$ is the gradually revealed local cost function of agent $i$, possibly depending on all agents’ variables, but not necessary for all decision variables to be identical. And the case has the same goal as (3).

3) Online Game. All agents only care about its own time-varying interest (i.e., time-varying local cost functions) in this setup. To be specific, different from the cooperation among agents in (4), all agents are noncooperative with their gradually revealed local cost functions $f_{i,t}$’s, and the objective, in general, is for each agent $i$ to minimize its own cumulative cost as follows:

$$\min_{x_t \in \times_{i=1}^N X_i} \sum_{t=1}^{T} f_{i,t}(x_{i,t}, x_{-i,t}), \quad (5)$$

where $x_{-i,t}$ denotes the variables of all agents except $i$, i.e., $x_{-i,t} := \text{col}(x_{1,t}, \ldots, x_{i-1,t}, x_{i+1,t}, \ldots, x_{N,t})$.

Along this line, recent works [16–18] have studied online game with/without constraints.
A. Further Discussion on Problem Settings

Although three main frameworks have been presented above for DOL, to summarize various studied problems in more details, we further classify these problems from other viewpoints, as discussed below.

Cost Functions. From this perspective, DOL problems can be classified as the conventional case, i.e., the local cost function of agent $i$ at time $t$ is viewed as a single one $f_{i,t}(x_i)$, which is the most widely investigated in the literature [19]–[22], and other cases, as follows.

- **DOL with Composite Functions.** In general, the cost function $f_i$ can be smooth or nonsmooth, and in order to exploit the fine structure of $f_i$ when it is nonsmooth, the cost function can be considered as a sum of two functions, one is smooth and the other is a nonsmooth regularizer, i.e., composite optimization. Such problems can be naturally found in realistic applications involving low-rank, sparsity, monotonicity, and so forth [23], [24].

- **DOL with Nonseparable Global Objectives.** Some practical applications may encounter the scenario where global $f_1(x_1, \ldots, x_N)$ is nonseparable, but each agent $i \in [N]$ only maintains a part (or a coordinate) $x_i$ of the whole variable $x = col(x_1, \ldots, x_N)$, such as in resource allocation [25], [26].

- **Decentralized Online Aggregative Learning.** In some practical applications, the cost function of each agent may depend on other agents’ variables besides its own variable [27]. Along this line, decentralized online aggregative learning has been recently proposed and studied [28], [29], where each local cost function relies not only on its own variable, but also on an aggregative term consisting of all agents’ variables over the network, i.e., the local cost function $f_{i,t}$ in (4) is of the form

$$f_{i,t}(x_i, \mu(x)) = \frac{1}{N} \sum_{j=1}^{N} f_j(x_i),$$

where $x = col(x_1, \ldots, x_N)$ with $x_i \in \mathcal{X}_i \subseteq \mathbb{R}^{n_i}$ being the decision vector of agent $i$, $f_j : \mathbb{R}^{n_j} \rightarrow \mathbb{R}$ is a local function of agent $i$, and $\mu(x)$ is an aggregative variable of all the agents.

Constraints. The constraint set $\mathcal{X}_{i,t}, i \in [N]$ is in a general form, meaning that $\mathcal{X}_{i,t}$ can be the entire Euclidean space, i.e., $\mathcal{X}_{i,t} = \mathbb{R}^{n_i}$, or can be of the following form:

$$\mathcal{X}_{i,t} = \{x_i \in \mathcal{X}_i | \text{s.t. (in)equality constraints}\},$$

where $\mathcal{X}_i \subseteq \mathbb{R}^{n_i}$ is a closed convex set, standing for the feasible set constraint for agent $i$, on which it is often relatively efficient to perform the projection operation. More details are introduced in the following.

- **DOL with Feasible Set Constraints.** In this case, the set $\mathcal{X}_{i,t}$ for each $i \in [N]$ is simply $\mathcal{X}_i$ (or an identical set $\mathcal{X}$), that is, $\mathcal{X}_{i,t} = \mathcal{X}_i$ (or $\mathcal{X}_{i,t} = \mathcal{X}$) for all $i \in [N], t \in [T]$, where $\mathcal{X}_i$ or $\mathcal{X}$ is usually assumed to be compact. This case is widely investigated in the literature [19]–[22], [30]. It is noteworthy that the compactness of $\mathcal{X}_i$ or $\mathcal{X}$ is generally imperative even in centralized online learning [5], since optimal variables can be arbitrary in the absence of this condition or other similar conditions (e.g., the boundedness of the constraint set’s diameter), probably incurring unbounded variations between consecutive cost functions. However, this condition can be eliminated in some cases, such as the unconstrained case with $f_i$ being strongly convex [31], [32].

- **DOL with Local Constraints.** Local constraints here subsume local equality and/or inequality constraints. This case has been specially studied, mostly due to the facts that constrained sets sometimes consist of equality and/or inequality constraints either on which it is computationally prohibitive to perform the projection operation or which have particular structures that can be elegantly exploited when developing algorithms (e.g., equality constraints of the form $Ax + By = c$ for variables $x \in \mathbb{R}^{n_1}, y \in \mathbb{R}^{n_2}$ with $A \in \mathbb{R}^{m \times n_1}, B \in \mathbb{R}^{m \times n_2}, c \in \mathbb{R}^m$, often handled by the alternating direction method of multipliers (ADMM) [33]–[37].

- **DOL with Coupled Constraints.** In this case, constraints are coupled (or separable) across all the agents, and each agent can only access partial information on constraint functions at each time $t$ [38]–[41], i.e.,

$$\mathcal{X}_i = \{x_i \in \mathcal{X}_i | \sum_{i=1}^{N} h_{i,t}(x_i) = 0_t\},$$

$$\mathcal{X}_i = \{x_i \in \mathcal{X}_i | \sum_{i=1}^{N} g_{i,t}(x_i) \leq 0_m\},$$

where $h_{i,t}$ and/or $g_{i,t}$ are gradually revealed only to agent $i$, and $\mathcal{X}_i$ simply denotes $\mathcal{X}_{i,t}$ for each $i \in [N]$, representing a global set constraint, which constitutes an identical constraint set imposed by all agents in this case. Moreover, it is worthwhile to note that all agents’ variables can be nonidentical in (8) and (9), i.e., no constraint $x_i = x_j$ for all $i \neq j$, and the separable constraints in (8) can simultaneously appear in a single problem.

- **DOL with Control Systems.** It is easy to observe that no system dynamics is considered in the aforesaid problems. In physical world, agents involved in real applications are often subject to some physical dynamics, such as bicycle dynamics for robots and Euler-Lagrange dynamics for manipulators, and thus, to better suit many applications in reality, control system dynamics has been integrated with the study of DOL recently [42], which can be viewed as sort of constraint for DOL.

Time-Varying Scenarios. According to distinct scenarios, the model of time-varying cost/constraint functions can be summarized in the following three categories.

- **Oblivious Model.** In this case, all cost/constraint functions are determined at the beginning by the environment or adversary, which cannot be changed in the process of learning.

- **Adaptive Online Model.** In this scenario, cost/constraint functions can be modified in the course of learning. However, the environment or adversary must change...
the functions before it is allowed to know the decision variables of all agents.

- **Adaptive Offline Model.** This is similar to adaptive online model, and the difference is that the environment or adversary can alter the functions after it knows all agents' decision variables.

- **Stochastic Model.** This model means the existence of some stochastic components for local cost functions, incurred by sampling data or other factors. Usually, the local cost function $f_{i,t}$ is dependent on an additional random variable $\xi_{i,t}$, which is subject to a generally unknown distribution $D_{i,t}$, i.e., $f_{i,t}(\cdot;\xi_{i,t})$ at time $t$.

To our best knowledge, all the existing works for DOL focus on the oblivious case among the first three models.

**Function Properties.**

- **Convex/Nonconvex DOL.** In existing literature, the convex case has been more frequently studied until now, where cost functions and constraint functions (if available) are all convex 19–21, 38. Meanwhile, the nonconvex case has been less addressed, where either cost or constraint functions are nonconvex, usually leading to more complex problems than the convex ones. To date, only a few papers considered the nonconvex case with only feasible set constraints 44, 45.

- **Strong Convexity and/or Smoothness.** To derive better performance, several elegant function properties can be particularly leveraged, which are also realistic in many applications, such as strong convexity, smoothness, and so forth. For example, strong convexity and smoothness have been harnessed to improve the performance for DOL in 31.

**B. Performance Metrics**

With the above problem settings in DOL, how to evaluate whether a developed algorithm is good or not is apparently an important task for effective algorithm proposals. For online learning, there are generally three classes of metrics which have been leveraged in the literature, as summarized below. For simplicity, the following metrics are introduced for the consensus based DOL 2 due to its most popularity, but similar metrics can be also defined for 4 and 5.

1) **Dynamic Regret.** Equipped with the objective in (2), one direct metric is to compare the accumulated cost against a sequence of any comparators $\{q_t\}_{t=1}^T$, satisfying $q_t \in X_t := \cap_{i=1}^N X_{i,t}$. Then the dynamic regret is defined as [5]

$$D-\text{Reg}(\{q_t\}) := \sum_{t=1}^T \sum_{i=1}^N f_{i,t}(x_{i,t}) - \sum_{t=1}^T \sum_{i=1}^N f_i(q_t).$$

(10)

For (10), there are two paramount special cases: 1) the dynamic regret by setting $q_t = x_{i,t} := \arg\min_{x \in X} f_i(x)$; and 2) the case by setting $q_t = x^* := \arg\min_{x \in \cap_{i=1}^T X_i} \sum_{t=1}^T f_i(x)$, which is usually called static regret, denoted specially as $S-\text{Reg}$, i.e.,

$$S-\text{Reg} := \sum_{t=1}^T \sum_{i=1}^N f_{i,t}(x_{i,t}) - \sum_{t=1}^T \sum_{i=1}^N f_i(x^*).$$

(11)

where an underlying assumption is $\cap_{t=1}^T X_t \neq \emptyset$. It is easy to see that the static regret is to compare the total cost with the minimal cost with respect to a decision variable which is the same one (i.e., time-invariant) all over the time horizon. It should be noted that the dynamic regret is generally not sublinear with respect to $T$ in the worst case, hence requiring some regularities, such as path variation/length and gradient variation, on the comparator sequence or the cost function sequence 46, 47.

In DOL, another metric related to dynamic regret is so-called individual dynamic regret (or local/agent dynamic regret) for each $j \in [N]$ (e.g., 21, 48), defined as

$$D-\text{Reg}_j(\{q_t\}) := \sum_{t=1}^T f_j(x_{j,t}) - \sum_{t=1}^T f_j(q_t),$$

(12)

that is, the compared cumulative cost is with respect to the decision variables of agent $j$, instead of all agents’ variables, which can measure the special performance for individual agent in some sense. Similarly, we can define individual/local/agent static regret (or simply individual/local/agent regret) (e.g., 22, 33) as

$$S-\text{Reg}_j := \sum_{t=1}^T f_j(x_{j,t}) - \sum_{t=1}^T f_j(x^*), \ \forall j \in [N].$$

(13)

Note that associated with individual regrets in (12) and (13), the regrets in (10) and (11) can be called network regrets.

2) **Adaptive Regret.** The dynamic regret is defined over the entire horizon $t \in [T]$, i.e., from a global perspective. Another metric, called adaptive regret, is proposed from a local perspective, that is, it aims at comparing the cumulative cost over a subset of the entire horizon $[T]$, which includes two versions, i.e., strongly and weakly adaptive regrets. For strongly adaptive regret 49, the purpose is to minimize the maximal static regret over a fixed time interval, say $\tau > 0$, which is defined as

$$SA-\text{Reg}(\tau) := \max_{[r, r+\tau-1] \subset [T]} \sum_{t=r}^{r+\tau-1} \sum_{i=1}^N f_{i,t}(x_{i,t}) - \sum_{t=r}^{r+\tau-1} \min_{x \in \cap_{i=1}^T X_i} \sum_{t=r}^T f_i(x).$$

(14)

Furthermore, the weakly adaptive regret 50 aims to minimize the maximal static regret over any contiguous time intervals, specifically defined by

$$WA-\text{Reg} := \max_{[r, s] \subset [T]} \sum_{t=r}^s \sum_{i=1}^N f_{i,t}(x_{i,t}) - \min_{x \in \cap_{i=1}^T X_i} \sum_{t=r}^s f_i(x).$$

(15)

It is easy to observe that weakly adaptive regret is greater than strongly adaptive regret, which is dependent on the time interval $\tau$ and will incur different SA-Reg(\tau) values for distinct $\tau \in [1, T]$.\]
3) **Competitive Ratio.** Besides the above performance metrics, another natural metric is competitive ratio (CR for brevity) [51], defined as the division between the cumulative cost and the minimal overall cost over time horizon $T$, i.e.,

$$ CR := \frac{\sum_{t=1}^{T} \sum_{i=1}^{N} f_{i,t}(x_{i,t})}{\sum_{t=1}^{T} f_{i}(x_{i})} , $$

(16)

where $x_{i}^{*} = \arg \min_{x \in X_{i}} f_{i}(x)$.

Additionally, another type of metric related to competitive ratio is also employed in some literature, that is, an online algorithm is said to have a competitive ratio $c_{R} \geq 1$ [52], [53] if there exists a constant $c_{R} > 0$, which is independent of $T$, such that

$$ \sum_{t=1}^{T} \sum_{i=1}^{N} f_{i,t}(x_{i,t}) \leq c_{R} \sum_{t=1}^{T} f_{i}(x_{i}^{*}) + c_{R} , $$

(17)

which is only different from (16) by a constant $c_{R}$.

*“Good” Performance: With the above performance measures, a proposed algorithm is usually declared “good” if dynamic regret or adaptive regret is sublinear in $T$, i.e., $o(T)$, or the competitive ratio is a constant, i.e., $CR \leq c_{R}$ for some constant $c_{R} > 0$ or satisfying (17) with an additional constant $c_{R} > 0$.

It is worth noting that all the above metrics are introduced in the context of deterministic setting, which however can be easily accommodated to suit the stochastic scenario by replacing the cumulative cost with one in mathematical expectation with respect to any randomness in the context, e.g., stochastic cost functions, stochastic gradients, sample datasets, and stochastic algorithms, etc. Additionally, the aforesaid metrics can be also applied to the nonconvex setup in the sense of local optimum, and other special metrics for the nonconvex case were also proposed in the literature, such as the regret defined by first-order optimality condition [45].

It is also noteworthy that besides the aforementioned metrics, another metric is to directly track the time-varying optimal variables, which is often bounded by a constant that is determined by problem parameters (e.g., Lipschitz constant, gradient’s bound, etc.) and network factors (e.g., agents’ number, network connectivity, etc.), and this line of problems is called (centralized/decentralized) time-varying optimization [54]–[58]. Similarly, when the problem is to consider the fixed point computing for a series of operators in real Hilbert space, its goal is to track the sequence of time-varying fixed points, often called time-varying operator [59], [60]. This type of metric provides another perspective, but is different from the aforementioned metrics in DOL, for which a recent survey can be found in [61]. Also note that time-varying operator is now only addressed in the centralized manner. Nonetheless, the time-invariant scenario has been taken into account in the decentralized fashion in recent years [62]–[65], leaving the time-varying case as one of future directions.

Among the aforementioned performance metrics, the most widely exploited ones in online learning are static and dynamic regrets, followed by competitive ratio (more frequently leveraged in metrical task system (MTS) [66], etc.) and then adaptive regrets in the literature. Generally speaking, they provide different metrics for measuring designed online algorithms in online learning from diverse perspectives, which have vague relationships as discussed in the sequel. Up to date, quantitative relationship between dynamic regret and adaptive regret remains unclear and an attempt to simultaneously minimizing both dynamic and adaptive regrets was made in recent work [67]. Moreover, static regret is usually incompatible with competitive ratio, that is, they often cannot perform well simultaneously [68]. However, in recent work [69], the authors proposed a method for simultaneously ensuring a small static regret and a guaranteed constant competitive ratio for a special $N$-experts $D$-switching cost problem. Nonetheless, the problem of concurrently achieving the best static/dynamic regret and competitive ratio even in a general centralized convex setup remains open.

C. Applications

Numerous applications of DOL can be found in reality, including the binary classification problem from medical diagnosis [21], collaborative localization in sensor networks [70], heating, ventilation, and air-conditioning (HVAC) systems of commercial buildings [48], decentralized dynamic sparse recovery problem [23], decentralized estimation in sensor networks [19], decentralized target tracking in 2-D plane [71], target surrounding problem for robots in the plane [28], mobile edge computing [72], multi-class classification [73], regularized linear regression [24], robot formation control [25], support vector machines (SVM) for binary classification [22], decentralized energy resources for distribution grids [7], and so on. As examples, three applications in the literature are briefly introduced below.

**Example 1 (Decentralized Target Tracking [71]).** Consider a slowly moving (nearly constant velocity) target in a plane with independently evolving horizontal and vertical components of its position. The target’s decision vector (or state) is composed of four components at each time instant, i.e., horizontal position, vertical position, horizontal velocity, and vertical velocity. As a result, the optimal vector at each time $t$ is denoted by $x_{t}^{*} \in \mathbb{R}^{4}$, which obeys the physical motion dynamics as follows:

$$ x_{t+1}^{*} = Ax_{t}^{*} + \nu_{t} , $$

(18)

where $\nu_{t}$ is the system noise, and

$$ A = \begin{pmatrix} 1 & \delta \\ 0 & 1 \end{pmatrix} \otimes I_{2} $$

(19)

with $\delta > 0$ being the sampling interval. The objective is to track $x_{t}^{*}$ of the target by a sensor network of $N$ agents in a collaborative manner.

The agents are located on an $M \times M$ grid for some integer $M > 0$, aiming at tracking the moving target cooperatively. At each time step $t$, an observation $z_{i,t} \in \mathbb{R}$ on partial information $x_{t}^{*}$ can be performed by each agent $i$, satisfying

$$ z_{i,t} = c_{k_{i}}^{T} x_{t}^{*} + \omega_{i,t} , $$

(20)
where $c_k$ is the $k$th unit vector in $\mathbb{R}^L$, i.e., all entries are zero with the $k$th entry being 1, and $\omega_{i,t} \in \mathbb{R}$ is the observation noise. $k_i$ can be randomly chosen or pre-specified for each agent in order to ensure that every component of $x_i^*$ is observed by at least one sensor. In this scenario, each agent cannot observe the target on its own, but $x_i^*$ is globally identifiable from the viewpoint of the entire network. Then the local square cost function is of the form

$$f_i(t) = E[(z_{i,t} - e_k^T x)^2],$$

and the global cost function is

$$f(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x),$$

which boils down to the consensus based DOL [2].

**Example 2** (Decentralized Target Surrounding [28]). Consider a moving robot target in $\mathbb{R}^2$ space, denoted by $x_0(t)$, which tends to be attacked by a group of $M > 0$ intruders. Meanwhile, there are a family of $N$ robots, which aim at protecting the target by surrounding it, and each defender is only aware of its nearby intruders, that is, no single defender can grab the information on all intruders. However, by combining all defenders’ information, the attack from all intruders can be avoided in general. For simplicity, let us consider the case of $M = N$, and agent $i \in [N]$ is only aware of the intruder $i \in [N]$. In this scenario, at each time instant $t > 0$, each agent $i \in [N]$ makes a decision by deciding to move to a position $x_{i,t}$ (possibly subject to a position constraint $x_{i,t} \in X_{i,t}$), and then a loss will be revealed to agent $i$. Note that to better protect the target, it is preferable to drive all defenders to some positions such that the target is located at the center of all defenders. Therefore, the loss should be proportional to the distance from the average position of all defenders to the target. That is, the local loss function is of the form

$$f_i(x_{i,t}, \nu(x_t)), \quad \nu(x_t) := \frac{1}{N}\sum_{i=1}^{N} x_{i,t},$$

and the global loss function is

$$f(x_1, \ldots, x_N) = \sum_{i=1}^{N} f_i(x_{i,t}, \nu(x_t)),$$

which is an instance of multi-agent coordination based DOL [4].

**Example 3** (Online Nash-Cournot Game [16], [17]). The online Nash-Cournot game is denoted by $\Gamma([N], X, J_t)$, where $X = X_1 \times \cdots \times X_N$ denotes the action set of $N$ firms (or players), representing production constraints and market capacity constraints, and $J_t = (J_1(t), \ldots, J_N(t))$ is the cost function with $J_i(t)$ being the local cost function of firm $i$ at time slot $t$. Note that the same production is produced by all firms, which interact with each other via some underlying communication graph. Denote by $x_i \in \mathbb{R}$ the production quantity of firm $i$. In view of some instable factors such as the production cost, marginal costs and the demand price, the firm $i$’s production cost and demand price are given as $p_{i,t}(x_i) = \alpha_i(t)x_i$ and $d_{i,t}(x) = \beta_i(t) - \sum_{i=1}^{N} x_i$ for some $\alpha_i(t), \beta_i(t) > 0$, respectively. As a result, the local cost function of firm $i$ is of the form

$$J_{i,t}(x_i, x_{-i}) = p_{i,t}(x_i) - x_i d_{i,t}(x), \quad \forall t \in [0, T]$$

where $x_{-i} := col(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_N)$. Then this problem fits into online game as discussed in [5].

### III. Communication Perspective

This section is concerned with hot topics in DOL from the perspective of information exchanges among agents in the network, which is known to be pivotal to decentralized algorithms over multi-agent networks. Without agents’ interconnections, it is impossible to tackle the global online learning problem in the decentralized manner. In doing so, according to a variety of communication patterns, this section provides a comprehensive review on DOL, including stationary and/or time-varying communication graphs, full information communication, privacy-preserving communication, security, asynchronous algorithms, information quantization/compression, as illustrated in Fig. 2.

![Fig. 2. Schematic illustration of communication issues between any two agents $i \neq j \in [N]$ in DOL, where the dotted line connotes any possible underlying communication channel/link, which may be undirected or directed.](image)

1) **Communication Graphs**. As an important aspect in DOL, the interactions among all the agents in the network are quintessentially captured by graphs, which, according to different practical applications, can be mainly classified into four cases, i.e., undirected graphs [20], [22], [31], [33], [35], balanced/unbalanced fixed directed graphs [19], [30], [48], [76]–[78], balanced time-varying/switching directed graphs [21], [24], [28], [39], [40], [79]–[81], and unbalanced time-varying directed graphs [26], [38], [82]–[85], and random switching graphs [81], [86]. Generally speaking, undirected interaction graphs are relatively simple due to the elegant symmetry property of mixing matrix (or adjacency/Laplacian matrix), while time-varying directed graphs, especially unbalanced ones, are more complicated when studying DOL. For example, the authors in [31] investigated unconstrained DOL for strongly convex objectives, where an improved dynamic regret is established under stationary undirected interconnection graphs. However, it is nontrivial to extend the result in [31] to the case with unbalanced time-varying directed graphs, since some employed techniques are no longer available under unbalanced time-varying directed graphs.

2) **Full Information Communication**. This is the earliest and most frequently studied case in DOL, where the transmitted information from each agent to its out-neighbors...
is the agent’s full variables, i.e., exact information transmission. Relevant examples include [19–24], [31], [32], [34–40], [71], and so on, where full information on one or more variables is broadcast to neighboring agents. For example, an online distributed dual averaging algorithm was proposed using a sort of regularized projection in [19], where an exact variable information is propagated in each round. DOL with (strongly) convex functions was addressed in the presence of time-invariant and time-varying coupled inequality constraints in [38] and [39], respectively, where three full variables and one full variable are transmitted, respectively. In addition, a decentralized online mirror descent algorithm was developed for DOL with convex functions [71], where the optimal trajectory consisting of minimizers at all times is assumed to conform with a known linear dynamics corrupted by unknown and unstructured noises.

3) Privacy-preserving Communication. Usually, each agent broadcasts its accurate full information to its neighboring agents, which, however, is vulnerable to privacy disclosure. Provided that different mass data are distributed over a collection of agents, data privacy for each agent is paramount in a multitude of realistic applications, such as patients’ diagnosis information in hospital, personal daily data held on private mobile phone/computer, and users’ private information on Facebook, etc. Wherein, private data of each agent are undesirable to be disclosed to other agents. Along this line, an intrinsic privacy-preserving algorithm was proposed for DOL in [20], where sufficient and necessary conditions were established for privacy preservation, showing that other agents’ subgradients (and sensitive raw data) cannot be reconstructed by a malicious learner in networks with greater-than-one connectivity. Furthermore, differential privacy, firstly proposed in [87], has been taken into consideration in DOL [80], [85], [88], [89], where an independent and identically distributed (iid) noise drawn according to Laplace distribution is added to each agent’s variable before information transmission, thus guaranteeing that each agent’s exact information cannot be disclosed by other agents due to information masking by Laplace noises. Meanwhile, it is worth noting that except for injected noises, noisy information may be naturally propagated due to the noisy links/channels, or noisy measurements incurred by inexact sensing devices [90], [91].

4) Security. Similar to privacy preservation during communications in multi-agent networks, another critical issue is malicious attacks or adversarial agents, including denial-of-service (DoS) attacks, replay attacks, false data injection attacks, and so on, which can severely vandalize the proposed online algorithms that perform well in benign environments, i.e., in the absence of any intentional attacks. It is well known that designing robust algorithms to adversarial attacks is nontrivial and challenging for multi-agent consensus, distributed learning and decentralized optimization, etc. (e.g., [92–94]). In this respect, adversarial agents were considered recently for DOL in [95], where Byzantine faulty agents can update its variable arbitrarily, which is then transmitted to its neighbors, with the purpose of preventing no-faulty agents from achieving the optimal solution, and individual static regret is ensured by establishing sufficient conditions on the graph topology, the number and location of the adversarial agents.

5) Asynchronous Algorithms. Most of existing works have focused on synchronous algorithms in DOL, that is, all the agents in the network can transmit their information, perform their calculations, and carry out their updating at the same time. As a matter of fact, synchronous algorithms depend on underlying assumptions that a global clock is accessible to all the agents and information transmissions along information channels do not undergo any delayed feedback. Nevertheless, this case may not be true in many practical applications, for example, wireless sensor networks [96], where individual agents often have different clocks and delayed feedback usually exists during information collection, computation and propagations, thereby incurring distinct time readings at the same global time and different information processing time for each agent, respectively. Along this line, asynchronous algorithms have been extensively investigated for DOL in recent years [74], [89], [97], [98], where delayed feedback of only local cost functions’ gradient is addressed in [97], delayed information transmission among agents is taken into account in [89], asymmetric gossiping communication is considered in [74], and a more general case is studied in [98], where inherent delays (the time needed to observe the effect of a decision), computation delays for processing an action at each agent (e.g., gradient computations), and communication delays for information transmission among agents are considered.

6) Information Quantization or Compression. One of particular important issues that appear in decentralized problems over multi-agent networks is the number of bits transmitted along information links/channels among all the agents, since the capacity of transmitted data cannot be arbitrarily large, but subject to physical limitations on transmission channels [99]. Because of this practical reason, information quantization (or compression), i.e., encoding information with a certain amount of bits, has been extensively studied in multi-agent control (e.g., [100], [101]) and centralized/decentralized optimization (e.g., [102–105]), etc. However, in contrast to the vast amount of literature in aforementioned fields, to our best knowledge, there are only few existing works focusing on DOL with information quantization, of which the proposed algorithms in [90] leverage only the signs of neighbors’ relative variables (a special one of quantization/compression strategies using only one bit), in order to alleviate the sensing and communication requirements for each agent.
TABLE I
State-of-the-art performance results with only feasible set constraint \(x_i \in \mathcal{X}, i \in [N]\) in DOL. The “plain case” means the direct availability of function gradients and projection operators in the algorithm design, “OB” stands for optimal bounds, and “N.F.” means that the case is not found in the literature.

| Metric         | Objective | Plain case | One-point bandit | Projection-free |
|----------------|-----------|------------|------------------|-----------------|
| Static regret  | Convex    | \(O(\sqrt{T})\) (OB) [19,22,30] | \(O(T^{\frac{1}{2}})\) [37] | \(O(T^{\frac{1}{2}})\) with \(O(\sqrt{T})\) communication complexity [105] |
|                | Strongly convex | \(O(\log T)\) (OB) [20,21,24] | \(T^{\frac{1}{2}} \log T\) [37] | \(O(T^{\frac{1}{2}})\) with \(O(T^{\frac{1}{4}})\) communication complexity (optimal up to logarithmic factors) [107] |
| Dynamic regret | Convex    | \(O(\sqrt{T}(1 + C_T))\) (near OB) [71] | N.F. | N.F. |
|                | Strongly convex | \(O(1 + P_T)\) (smooth, near OB) [108] | N.F. | N.F. |

\(a\) \(C_T := \sum_{t=1}^{T} \|x_{t+1}^{*} - Ax_t^{*}\|\) with \(x_t^{*} = \arg\min_{x \in \mathcal{X}} f_t(x)\), and \(A\) is a common knowledge on the deviation of the minimizer sequence, i.e., \(x_{t+1}^{*} = Ax_t^{*} + v_t\), where \(v_t\) is an unknown and unstructured noise.

\(b\) \(P_T := \sum_{t=2}^{T} \|x_t^{*} - x_{t-1}^{*}\|\) (called path variation/length) with \(x_t^{*} = \arg\min_{x \in \mathcal{X}} f_t(x)\).

IV. COMPUTATION PERSPECTIVE

In addition to communication aspects discussed in the last section, another important facet for DOL is the computation and memory/storage issues at each agent. As a consequence, this section aims at reviewing specific directions from the perspective of computation, including full gradients/subgradients, stochastic gradients/subgradients, gradient-free methods, projection-free methods, memory/storage requirement, as shown in Fig. 3.

1) Full Gradients/Subgradients. With regard to this case, full (sub)gradients of local cost functions need to be exactly computed for each agent, which is relatively easy in some special problems, such as linear cost functions, but usually computationally heavy for general convex and nonconvex functions. Nevertheless, analyzing full gradient case has elegant mathematical theories and can substantially shed light on more complicated scenarios, thus attracting numerous researchers (e.g., [19]–[22], [30], [31], [34], [35], [38], [39], to just mention a few), making it the most frequently studied case in the literature for DOL until now.

2) Stochastic Gradients/Subgradients. To meet more realistic applications frequently encountered in our daily life, where full (sub)gradients are usually computationally prohibitive for general cost and/or constraint functions, stochastic (sub)gradient methods, i.e., efficiently approximating exact gradients, have been increasingly addressed for DOL in recent years, for example, [28], [71], [83], [85], [89], where it is customarily assumed that stochastic (sub)gradients are unbiased, that is, the mathematical expectation of a stochastic gradient is equal to its true gradient, and have bounded variances.

3) Gradient-free Methods. Note that the above stochastic (sub)gradient literature often do not provide any concrete approaches on how to calculate stochastic (sub)gradients, instead postulating their unbiasedness and variance boundedness. As two specific instantiations, bandit feedback and sub-optimization solver have been viewed as important gradient-free approaches especially in large-scale learning. To be specific, bandit feedback means that only function values at a few specified points are returned by a calculation oracle, including one-point bandit feedback which returns function value at only one point [37], [40], [86], [90], [109], and multi-point bandit feedback which reveals function values at multiple points, where two-point bandit feedback is more widely exploited in the literature [24], [40], [41], [76], [78], [86], [97]. In addition, the sub-optimization solver method connotes that an optimal solution to a sub-optimization problem (often cheaper to resolve than the original problem) can be directly derived, which either has a closed-form solution or can be calculated usually by appealing to sophisticated optimizers, such as (stochastic) gradient descent and so forth, as done in [33], where two sub-optimization problems need to be efficiently solved for the proposed decentralized online ADMM algorithm.

4) Projection-free Methods. Another frequently used yet probably computationally heavy operation is to perform projections on feasible closed convex sets, which is only
computationally light for several special structured sets, e.g., hyper-box, sphere, and etc. As such, projection-free methods have been considered in DOL with the purpose of alleviating the computational burden for each agent usually by replacing projections with linear optimization over feasible sets \cite{73, 106}, where the Frank-Wolfe method (or the conditional gradient method) is borrowed to develop projection-free decentralized online algorithms.

5) Memory/Storage Requirement. As local computing is indispensable for each agent, for example, at each updating step, it is imperative for each agent to be capable of storing computing data with a certain amount of data memory. Generally speaking, one common and essential feature in DOL, as in conventional decentralized algorithms over multi-agent networks (e.g., decentralized optimization), is executing a weighted aggregation or mixing of the received neighboring information, including primal variables, dual variables, and other variables. In order to perform their weighted aggregation operations, one agent requires to store received neighboring information in its buffer, which however can be eliminated once completing this computation. In the meantime, another most common trait is calculating local (sub)gradients, needing to query a (sub)gradient oracle (e.g., full gradient oracle, one- and multi-point bandit oracles) for each agent in each updating round. It should be noted that in some scenarios with gradient-free methods, like the sub-optimization solver methods, gradients’ calculation is unnecessary, instead replaced by efficiently resolving sub-optimization problems, such as proximal point algorithm (PPA). In a nutshell, the above mentioned storage and memory are usually the minimal requirements in DOL (e.g., \cite{20, 23, 36, 71}). On the other hand, besides storing primal variables updated at last round, many works may further require to store other variables, including the widely employed dual variables \cite{22, 33}, especially for inequality and/or inequality constraints \cite{34, 35, 37, 39, 40} which often demand multiple gradient queries, one for local cost function and the others for local/global constrained functions. For example, an additional variable of problem-dimension different from dual variables was introduced and stored for each agent at each round in the literature, leveraged to either track the gradients of global cost functions \cite{23, 31}, or assist in mixing neighboring information in order to align them for all agents \cite{21}, or carry out the averaging of all its own historical decision variables for outputting a desirable decision \cite{19}. Moreover, when addressing unbalanced directed graphs, more scalar variable is generally necessary to be introduced, stored and transmitted for each agent with the aid of either the push-sum strategy \cite{82, 83, 85, 84} or the balancing weight approach \cite{85}. Meanwhile, another method for handling the network imbalance is to introduce a network-dimensional variable for each agent to estimate the left eigenvector of the unbalanced interaction matrix \cite{77}, which however is memory and computation prohibitive for large-scale networks since the new variable has the same dimension as the network.

V. STATE-OF-THE-ART PERFORMANCE

This section is devoted to summarizing various cutting-edge performance results on proposed decentralized online algorithms for DOL in the literature, where, to our best knowledge, almost all the existing works have adapted the static and/or dynamic regrets as the performance measure. In doing so, to make various best known results more clear, the overview of state-of-the-art regret bounds is divided into two parts, one is for the case with only feasible set constraints and the other is for the case with inequality constraints.

To this end, the best obtained regret performances for the feasible set constraint case are provided from three perspectives: the plain case, one-point bandit feedback, and projection-free methods, as shown in Table I and the best results for the case with inequality constraints are given in Table II including uncoupled and coupled inequality constraints, where the uncoupled constraints represent local (or global) inequality constraints gradually revealed to local agents (or all agents) in the network, and the employed notations are summarized as follows:

$$\text{CACV}_T := \sum_{t=1}^{T} \sum_{i=1}^{N} \sum_{l=1}^{m} [g_i^l(x_{i,t})]_+,$$  \hspace{1cm} (26)

$$\text{D-CCV}_T := \frac{1}{N} \sum_{t=1}^{T} \|g_t(x_{i,t})\|_+,$$  \hspace{1cm} (27)

$$\text{CV}_T^+ := \left\| \sum_{t=1}^{T} g_t(x_{i}) \right\|_+, \quad \text{and} \quad \text{V}_T^P := \left\| \sum_{t=2}^{T} g_t - g_{t-1} \right\|,$$  \hspace{1cm} (28)

where $g_i^l$ represents the $l$th component of time-invariant constraint functions $g$, i.e., $g = \text{col}(g^1, \ldots, g^m)$ with $g^j : \mathbb{R}^n \to \mathbb{R}$, {$g_t$}_{t=1}^T is a sequence of any comparators, as introduced when defining the dynamic regret in \cite{10}, and $V_T^P$ denotes the variation of the sequence of comparators, called path variation/length.

It should be noted that the case with two-point bandit feedback is not listed in Table I since results in this case are generally the same as the full information case. For example, the bounds on static regrets, $O(\sqrt{T})$ and $O(\log T)$, were established in \cite{78} with two-point bandit feedback for convex and strongly convex cost functions, respectively, which are exactly the same as the full information case and are even the optimal bounds as in centralized online learning. This fact can be also observed in Table II where a few two-point bandit feedback cases are given in \cite{119}.

To see if the up-to-date performances are good enough, it is a good avenue to compare them with the best known regret bounds in centralized online learning. To do so, we summarize the state-of-the-art performance results for online learning in the centralized setting in Table III where partial notations can
State-of-the-art performance results with inequality constraints in DOL. “IC” means inequality constraints, “A-OBF-C” represents that the result is also Applicable to the case with One-point Bandit Feedback for only Cost functions (but still using gradients of constraints functions), “A-TBF-C2” signifies that the result is also Applicable to the case with Two-point Bandit Feedback for both Cost and Constraint functions, “OBF-C” (resp. “OBF-C2”) means that the result is derived under One-point Bandit Feedback for only Cost functions (resp. both Cost and Constraint functions), and “SC” denotes the result derived under Slater’s condition.

| Metric         | Objective       | Uncoupled inequality constraints | Coupled inequality constraints |
|----------------|-----------------|---------------------------------|-------------------------------|
|                | Time-invariant  | Time-varying                    | Time-invariant               |
| Static regret  | Convex          | $O(T^{\frac{1}{2}})$           | $O(T^{\frac{1}{3}})$         |
|                | $CACV_T = O(T^{\frac{1}{2}})$ (A-OBF-C) [37] | $DCCV_T = O(T^{\frac{1}{3}})$ (A-TBF-C2) [110] | $CV_T^+ = O(T^{\frac{1}{3}})$ (SC) $\kappa \in (0, 1)$ |
|                | Strongly convex | $O(\log T)$                     | $O(T^\kappa)$                 |
|                | $CACV_T = O(\sqrt{T \log T})$ and $O(T^{\frac{1}{2}} \log T)$ | $DCCV_T = O(T^{1-\kappa})$ (A-TBF-C2) [110] | $CV_T^+ = O(T^{1-\kappa})$ or $CV_T^+ = O(T^{\max\{\kappa, 1-\kappa\}})$ (SC) $\kappa \in (0, 1)$ |
| Dynamic regret | Convex          | $O(T^{1-\kappa} + T^{\kappa} (1 + T^{\frac{1}{T}}))$ | $O(T^{1-\kappa} + T^{\kappa} (1 + T^{\frac{1}{T}}))$ |
|                | Strongly convex | N.F. [3]                        | N.F.                          |

1 Note that $CACV_T$ and $DCCV_T$ are strictly tighter metrics than $CV_T^+$ defined in [28] in general.
2 “N.B.” means no better results than the corresponding time-varying coupled inequality constraint case (note that time-invariant constraints are a special case of time-varying constraints).
3 “N.F.” means that the case is not found in the literature.

Note that all the above discussed works are in the setting of cooperative agents. For noncooperative agents in online game, there are only a few existing works in recent years, as summarized in Table IV, where individual regret is leveraged for convex and strongly convex cost functions, respectively, and the near-optimal bounds for dynamic regret with only feasible set constraints. However, most of results in DOL are still worse than the centralized case due to the complexity of local information communications.

Accelerated and Adaptive Methods. Even though several regret bounds are optimal in terms of $T$, as in centralized/decentralized optimization, there are also some methods to improve the regret performance or reduce the conditions while maintaining the same performance. For instance, the near-optimal bound $O(1 + P_T)$ on dynamic regret is usually obtained for strongly convex functions, which however was derived for convex functions in [28], but under a stronger assumption on the communication graph, i.e., every agent is connected to at least two other agents in the network. Moreover, some accelerated and improved methods, such as Nesterov accelerated gradient method and adaptive gradient methods, have been leveraged to further improve the performance. For example, the momentum acceleration technique was exploited to improve the performance under time-varying unbalanced communication graphs in [128], where an improved static regret bound $O(\sqrt{1 + \log T} + \sqrt{T})$ is established for convex cost functions. Also, adaptive gradient methods have been integrated into DOL in [129–131].

To compare Tables III, IV, some regret bounds in DOL are the same as the best/optimal ones in centralized online learning, such as $O(\sqrt{T})$ and $O(\log T)$ static regrets for convex and strongly convex cost functions, respectively, and the near-optimal bounds for dynamic regret with only feasible set constraints. However, most of results in DOL are still worse than the centralized case due to the complexity of local information communications.
TABLE III
State-of-the-art performance results in centralized online learning. To save the space, the following abbreviations are made: “IC” is the abbreviation of inequality constraints, “BCO” means bandit convex optimization, “SC” denotes the result derived under Slater’s condition, “OB” stands for optimal bounds, “EBC” means the result obtained under error bound condition (weaker than SC [111]), and “N.F.” means that the case is not found in the literature.

| Metric              | Objective       | Plain case          | One-point bandit       | Projection-free | Fixed IC                             | Time-varying IC       |
|---------------------|-----------------|---------------------|------------------------|-----------------|--------------------------------------|-----------------------|
| Static regret       | Convex          | $O(\sqrt{T})$ (OB) [112] | $O(\min\{\sqrt{nT}, T^{3\epsilon}\})$ (pseudo-1d BCO) (OB, up to logarithmic factors) [113] | $O(T^{\frac{3}{4}})$ (nonsmooth) [113] | $O(\sqrt{T})$ (smooth) [115] | $O(\sqrt{T})$ (SC) [119] |
| Strongly convex     | (OB) [120]      | $O(\log T)$ (OB) [107, 121] | $O(T^{\frac{3}{4}})$ | $CCV_T = O(\sqrt{T})$ [13] | $O(\log T)$ (OB) [13] | N.F. |
| Dynamic regret      | Convex          | $O(P_T)$ (smooth, $x_t^* \in X$) [47] | $O(\sqrt{T} + \frac{1}{2} P_T)$ (OB) [123] | N.F. | $O(\sqrt{T}(1 + P_T^2))$ | $O(\max\{\sqrt{T}, \sqrt{T} P_T\})$ (SC) or $CV^T = O(\sqrt{T})$ [124] |
| Strongly convex     | (OB) [125]      | $O(\min\{P_T, P_{T},P_{T}^2\})$ (smooth, OB) [126] | N.F. | $CCV_T = O(\sqrt{T})$ [13] | $O(\max\{\sqrt{T}, \sqrt{T} P_T\})$ (SC) or $CV^T = O(\sqrt{T})$ [124] |

* This result improves a lower bound of $O(n\sqrt{T})$ even for strongly convex and smooth cost functions in [127].

b $L_f$ is the Lipschitz constant for $\nabla f_i$.

TABLE IV
State-of-the-art performance results in online game.

| Metric              | Objective       | Time-invariant (Full gradient) | Coupled inequality constraints | Time-varying |
|---------------------|-----------------|-------------------------------|-------------------------------|--------------|
| Individual          | Strong mononicity | $O(\sqrt{T}(1 + P_T))$ | $O(T^{\frac{3}{4}} + \frac{1}{2} P_T)$ | $O(T^{\frac{3}{4}} + \frac{1}{4} \sqrt{T})$ |
| dynamic regret      | (Full gradient)  | $CV^T = O(\sqrt{T}(1 + P_T))$ | $CV^T = O(T^{\frac{3}{4}} + \frac{1}{2} \sqrt{T})$ | $E(CV^T) = O(T^{\frac{3}{4}} + \frac{1}{4} \sqrt{T})$ |

* $P_T$ is defined in the footnote of Table III. However, $x_t^*, t \in [T]$ used in $P_T$ is slightly different here, denoting the Nash equilibrium (instead of a minimizer) at time instant $t$.

It is worth noticing that the benchmark used in [37] and [38] is Nash equilibria. Other alternatives can be defined as

$$D-R^g_i := \sum_{t=1}^{T} f_{i,t}(x_{i,t}, x_{-i,t}) - \min_{y_{i,t} \in X_{i,t}} \sum_{t=1}^{T} f_{i,t}(y_{i,t}, x_{-i,t}), \quad (37)$$

where $x_t^* = col(x_t^*, x_{-i,t}^*)$ is a (generalized) Nash equilibrium of online game at round $t$. Similarly, individual/local static regret for online game can be defined as

$$S-R^g_i := \sum_{t=1}^{T} f_{i,t}(x_{i,t}, x_{-i}) - \min_{y_{i,t} \in X_{i,t}} \sum_{t=1}^{T} f_{i,t}(y_{i,t}, x_{-i}), \quad (38)$$

where $x^* = col(x_t^*, x_{-i}^*)$ is a (generalized) Nash equilibrium of an offline game with $\sum_{t=1}^{T} f_{i,t}(x_{i,t}, x_{-i})$ being the cost function of agent $i$ for all $i \in [N]$.

VI. FUTURE DIRECTIONS
With the above discussions on DOL, it can be found that the research on DOL is still in its infancy, and to facilitate further
studies in this domain for both beginners and specialists, this section aims at pointing out possible future research directions in DOL.

1) **Better Performance.** By comparing existing performance results on DOL in Tables I and II with the centralized best-known or optimal results in Table III, a multitude of gaps still need to be bridged for DOL. For example, the static regret bound in the decentralized case with one-point bandit feedback is far beyond the optimal bound as obtained in the centralized setup, and in the case with uncoupled/coupled inequality constraints, lots of regret bounds are also inferior to the best known results in the centralized setting. Moreover, there are a number of open issues in DOL to be addressed. As an example, the dynamic regret bound for DOL with one-point bandit feedback is still vacant.

2) **Network Effect.** One of important issues in decentralized algorithms is to consider the effect of network, such as the agent number $N$ and other network parameters, on the algorithms’ performances. In this respect, a dynamic regret bound for convex cost functions is established in [71] as

$$O\left( \sqrt{\frac{NT(1 + C_T)}{1 - \lambda_2(W)}} \right),$$

(39)

where $\lambda_2(W)$ denotes the second largest eigenvalue of the network matrix (e.g., adjacency or Laplacian matrices) $W = (w_{ij})$ in magnitude, $C_T := \sum_{t=1}^{T} \| x_{t+1}^* - Ax_t^* \|$. $A$ is a common knowledge on the deviation of the minimizer sequence, i.e., $x_{t+1}^* = Ax_t^* + \eta_t$, where $\eta_t$ is an unknown and unstructured noise. It is easy to see that the regret bound is proportional to $\sqrt{N}$ and inversely proportional to $\sqrt{1 - \lambda_2(W)}$.

In addition, a dynamic regret bound, i.e.,

$$GP_T + O\left( \frac{NG^2}{\gamma(1 - \gamma^2/\xi)} + \frac{N^2G^2}{1 - w_{\max}} \right),$$

(40)

is derived in [48] for convex functions under a stronger assumption on the communication graph, that is, every agent is connected to at least two other agents in the network [48], where $\gamma \in [0, 1), \zeta \geq 1$ are graph-related parameters, $w_{\max} := \max_{i,j \in N} w_{ij}$, and $G$ is a constant such that $\|f_i(\cdot)\|_2 \leq G$. One can observe that the bound is proportional to $N^2$ and inversely proportional to some graph parameters. The above bounds are neat results in the literature, and however, it is still unclear whether the dependence on those graph parameters are sharp or not.

3) **Information Quantization/Compression.** It is already known that the transmission capacity along information channels among neighboring agents is paramount for decentralized algorithms over multi-agent networks. Therefore, information quantization/compression for both transmitted message and local computations (e.g., gradient calculation) is imperative in future research directions, given that the current relevant research is still lacking in DOL (one exception is [30]), although it has been extensively considered in multi-agent control (e.g., [100], [101]) and centralized/decentralized optimization (e.g., [102], [105]), etc.

4) **DOL with Control Systems.** Most of existing works focus on decentralized online learning in the absence of system dynamics, that is, without considering physical control systems, which can be viewed as information-layer problems. Nevertheless, an agent often has its physical operating dynamics, such as bicycle dynamics for robots and Euler-Lagrange dynamics for manipulators, which should be appropriately considered and controlled, thought of as physical layer problems. Although recent research [42] has integrated the control system dynamics into DOL, the related research is yet to be fully explored in order to smoothly apply decentralized online algorithms to real-world problems.

5) **Continuous-time Algorithms.** Most of proposed algorithms in DOL are discrete-time iterated mainly due to the discrete-time computation fact of realistic implementations, for example, by computers. Even so, many physical systems or phenomena in practice are in continuous-time domain, such as the continuous-time operating dynamics of electric current flow and so forth. On the other hand, continuous-time algorithms can also be a powerful tool for providing insight into discrete-time algorithms (e.g., [134], [135]). Along this line, the continuous-time setup has been addressed with local inequality constraints in [136]. However, this type of problems are still less addressed in the literature, and thereby one of future directions is to pay more attention to this scenario.

6) **Nonconvex DOL.** Nonconvex cost/constraint functions can be frequently encountered in realistic applications, although the current research mostly focuses on the convex case for DOL. In this respect, a few works have studied the nonconvex case for centralized online learning, for example, locally Lipschitz and nonconvex cost functions in [137] with $O(1 + P_T)$ dynamic regret by proposing online Newton’s method and nonconvex cost functions but satisfying local proximal-PL inequality (a generalization of Polyak-Łjasišewicz (PL) condition for unconstrained optimization) in [138] with $O(1 + P_T)$ dynamic regret by developing online projected gradient descent with desirable initialization. However, only a few works have thus far investigated the nonconvex case in the decentralized setup [44], [45]. Consequently, it is nonnegligible to further take into account the nonconvex DOL in future research directions.

7) **Adaptive Gradient Methods.** It is well known that adaptive methods are important in centralized optimization and deep learning due to its easy implementation and superior performance in practical applications. The most popular adaptive gradient method is Adam by estimating first- and second-order moments of gradients [139]. As a result, one natural idea is to apply adaptive methods to DOL for improving the performance, which is exactly done in [129], [131]. However, the research along this line is not far fully explored, leaving an enormous...
 possibility for future directions.
8) **Second-order Methods.** It is easy to observe that all
developed algorithms for DOL are at most first-order
methods, that is, depending on first-order gradients or
zeroth-order function values. Nonetheless, it is well
known that second-order methods, such as the Newton
method, can generally improve the performance, usually
outperforming first-order methods. As such, second-
order online methods have been investigated in central-
ized online learning, e.g., [137], [140]–[143]. In contrast,
the decentralized case is yet to be explored.
9) **DOL on Remannian Manifolds.** Aside from the Eu-
clidean space studied for DOL in the literature, Remann-
ian manifolds, as a generalization of Euclidean spaces,
have long been an intriguing topic in deep learning
and centralized/decentralized optimization, possessing a
large number of applications such as in principal com-
ponent analysis (PCA), independent component analysis
(ICA), radar signal processing, dictionary learning, and
mixture modeling [144]. However, the study of DOL
on Remannian manifolds is still missing, only having a
few works on the centralized setup [145], [146], thereby
posing the necessity of addressing this case in future.
10) **The Case with Switching Cost.** It is worth noting that it
may incur a strategy changing cost when altering one’s
decision or action in practice, for example, the moving
cost from the current position to the next selected
position when the decision vector is the position of a
robot. In this scenario, the local cost function of each
agent $i \in [N]$ at time step $t$ is of the form
$$ f_{i,t}(x_{i,t}, x_{i,t-1}) + d(x_{i,t}, x_{i,t-1}), \quad (41) $$
where $x_{i,t}$ is still the decision vector of agent $i$ at time
t, and $x_{i,t-1}$ denotes a set of some other decision vectors
except $x_{i,t}$, which may be empty, or neighboring agents’
decision vectors, or all other agents’ decision vectors,
such as the case of decentralized online aggregative optimization [23], [29]. In (41), $d(x_{i,t}, x_{i,t-1})$ denotes a
generic distance from $x_{i,t}$ to $x_{i,t-1}$, e.g., $\|x_{i,t} - x_{i,t-1}\|_1, \|x_{i,t} - x_{i,t-1}\|_2$, and Bregman divergence $D_\psi(x_{i,t}|x_{i,t-1})$ for a strictly/strongly convex function $\psi$. Generally speaking, $f_{i,t}(x_{i,t}, x_{i,t-1})$ is called
hitting cost, operating cost, or stage cost in the literature,
and $d(x_{i,t}, x_{i,t-1})$ is called switching cost, smoothing
cost, or movement cost in the literature, which has been
extensively studied in centralized online learning (e.g., [147]–[151], to just name a few). However, it has thus
far not been considered in DOL, hence being regarded
as one of interesting future research directions.
11) **The Case with Predictions.** In online learning, future
information on cost functions is usually unaccessible
and even adversarial. One natural question is if the
performace can be improved when a few future infor-
mation is available or can be predicted in some sense,
including the gradient of next round at the current
time and a lookahead window of future cost functions,
etc. The answer is intuitively positive, usually leading
to a lower constant improvement in big $O$ term, as
confirmed in centralized online learning (e.g., [152]–
[154]). In comparison, there are still no works on this
study for DOL until now, thereby motivating its possible
investigation in future.
12) **Competitive Ratio.** It can be found that all related works
in DOL adapt the static and/or dynamic regrets as the
performance metric. However, as presented in Section
II-B, competitive ratio and adaptive regrets are also
considered as performance measures in the centralized
online learning, although the relationships of all the
metrics are yet to be fully understood, as discussed in
the last paragraph of Section II-B including a few works
focusing on the pertinent study [67]–[69]. In this connec-
tion, it is interesting yet challenging to further address
all the metrics (especially competitive ratio and adaptive
regret) and their relationships in both centralized and
decentralized online learning.
13) **Online Game.** In contrast with decentralized online
optimization, online game in dynamic environments
has so far been less explored in the literature, where
local or private cost functions are time-varying, even
adversarial and gradually revealed to each agent without
accessible to future information. Currently, only a few
works have researched online game with time-invariant
and time-varying coupled inequality constraints in [16]
and [17], respectively, where both works have adapted
individual dynamic regret as the performance regret and
sublinear regret and constraint violation are established,
as shown in Table [LV] Along this line, it is imperative to
make more efforts to investigate online game in future,
including improving regret bounds, studying individual
static regret $S$-$Reg^{\psi}_{i}$ and $D$-$R_{i}^{\psi}$, $S$-$R_{i}^{\psi}$ as well as their
relationships, and so forth.

VII. CONCLUSION

This paper has presented a comprehensive survey of DOL,
including decentralized online optimization and online game,
for which the overview has been performed from four view-
points, that is, problem settings, communication issues, com-
putation complexity, and up-to-date performances, includ-
ing full state information, communication delays, asynchrono-
sus algorithms, privacy-preserving, security-guaranteeing,
information quantization/compression, full gradient calcula-
tion, bandit feedback, projection-free algorithms, etc. With regard
to these aspects, state-of-the-art results have been summarized
and reported in this paper. Finally, possible future research
directions have also been elaborated, which, hopefully, is
conducive to further investigations on DOL in future.

REFERENCES

[1] M. J. Osborne and A. Rubinstein, *A Course in Game Theory*. MIT
press, 1994.
[2] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1999.
[3] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge U.K.:
Cambridge University Press, 2004.
[4] S. Shahrampour and A. Jadbabaie, “An online optimization approach
for multi-agent tracking of dynamic parameters in the presence of
adversarial noise,” in *Proceedings of American Control Conference*,
Seattle, USA, 2017, pp. 3306–3311.
[151] P. Zhao, Y.-X. Wang, and Z.-H. Zhou, “Non-stationary online learning with memory and non-stochastic control,” arXiv preprint arXiv:2102.03758, 2021.

[152] A. Lesage-Landry, I. Shames, and J. A. Taylor, “Predictive online convex optimization,” Automatica, vol. 113, p. 108771, 2020.

[153] Y. Li, X. Chen, and N. Li, “Online optimal control with linear dynamics and predictions: Algorithms and regret analysis,” in Advances in Neural Information Processing Systems, Vancouver, Canada, 2019, pp. 1–13.

[154] Y. Li, G. Qu, and N. Li, “Online optimization with predictions and switching costs: Fast algorithms and the fundamental limit,” IEEE Transactions on Automatic Control, vol. 66, no. 10, pp. 4761–4768, 2021.