A Distributed Generation Control Architecture for Islanded AC Microgrids

Stanton T. Cady, *Student Member, IEEE*, Alejandro D. Domínguez-García, *Member, IEEE*, and Christoforos N. Hadjicostis, *Senior Member, IEEE*

Abstract

In this paper, we propose a distributed architecture for generation control in islanded ac microgrids with both synchronous generators and inverter-interfaced power supplies. Although they are smaller and have lower ratings, the generation control objectives for an islanded microgrid are similar to those in large power systems, e.g., bulk power transmission networks; specifically, without violating limits on generator power output, frequency must be regulated and generation costs should be minimized. However, in large power systems, the implementation of the generation control functions is centralized, i.e., there is a computer that resides in a centralized location, e.g., a control center, with measurements and control signals telemetered between the generating units and the centrally-located computer. The architecture for generation control that we propose in this paper does not rely on such a centrally-located computer. Instead, the implementation of the control functions is distributed and relies on iterative algorithms that combine local measurements and certain information acquired from neighboring generating units with local, low-complexity computations. We provide analytical and experimental results that verify the effectiveness of the proposed architecture for generation control in islanded microgrids, and illustrate the performance of the aforementioned distributed algorithms under a variety of scenarios.

I. Introduction

In general, a group of interconnected loads and generators is considered to be a microgrid as long as, relative to larger systems such as bulk power transmission networks, they are physically smaller, have lower ratings and capacity, and are capable of islanded operation (see, e.g., [1]–[4] and the references therein). Although the archetypical example is a land-based collection of small generation units and loads which must be capable of operating in isolation from the utility grid, microgrids can also be found in applications such as more-electric aircraft (see, e.g., [5], [6] and the references therein), ships with electric propulsion (see, e.g., [7], [8]), and electrical installations for supporting telecommunication equipment (see, e.g., [9] and the references therein).

S. T. Cady and A. D. Domínguez-García are with the ECE Department at the University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA. E-mail: {scady2, aledan}@ILLINOIS.EDU.

C. N. Hadjicostis is with the ECE Department at the University of Cyprus, Nicosia, Cyprus, and also with the ECE Department at the University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA. E-mail: chadjic@UCY.AC.CY.

The work of S. T. Cady and A. D. Domínguez-García was supported in part by the National Science Foundation (NSF) under grant ECCS-CPS-1135598. The work of C. N. Hadjicostis was supported in part by the European Commission (EC) Seventh Framework Programme (FP7/2007-2013), under grant agreements INFSO-ICT-223844 and PIRG02-GA-2007-224877. Any opinions, findings, and conclusions or recommendations expressed here are those of the authors and do not necessarily reflect the views of NSF or EC.

Despite the aforementioned differences, the generation control objectives for islanded ac microgrids and large ac power systems are similar. More specifically, for any type of ac power system: (i) generation and demand must be constantly balanced, (ii) frequency must be regulated, and (iii) costs of generation must be minimized; importantly, all of these goals must be met without violating any power output limits of the generation units. In a large power system, each of these objectives is achieved independently using a three-layered generation control architecture. The bottom layer operates nearly instantaneously and is responsible for balancing generation and demand. The primary function of the middle layer is to regulate frequency, i.e., ensure the system operates as close as possible to the nominal frequency value, e.g., 60 Hz. In addition to regulating frequency, the middle layer is responsible for maintaining proper interchange power values between control areas; however, in the context of microgrids, there is no notion of control areas. The function of the top layer is to optimally dispatch the generating units, i.e., ensure that the units are generating power in such a way that total operation costs are minimized [10]. Although not the focus of this work, we note that, in general, the control objectives discussed above do not apply to dc power systems.

In a typical large power system, among the three previously mentioned control layers, only the bottom one operates using completely local information; the top two layers require a centralized decision maker to coordinate and control the generators. Moreover, this centralized generation control architecture is already in place, with no reasons compelling enough to motivate a system-wide change; in the nascent field of microgrids, however, there are no pre-established control architectures, allowing for new designs and alternatives. Additionally, the characteristics of microgrids make them amenable to a control architecture in which the decision-making is distributed among controllers located at the generating units. By combining local computations with information exchanged among neighboring controllers, distributed architectures can replicate the functionality of traditional generation control systems without requiring full knowledge of the number, type, or capabilities of generating units in the system, or a communication network connecting each generator to a central processor. Furthermore, compared to centralized ones, a distributed approach to generation control is more adaptable, allowing the system to operate regardless of additions or removals of generating units; together with the elimination of a centralized processor, this adaptability can result in higher system-level availability.

The main contribution of this work is the development and implementation of a distributed control architecture for islanded microgrids which replicates the functionality of the aforementioned top two control layers, i.e., frequency regulation and optimal dispatch. In our setup, we consider ac systems comprised of synchronous generators and inverter-interfaced power supplies, collectively referred to as distributed generation resources (DGRs), and assume that each DGR is equipped with a local controller and a transceiver through which information can be exchanged, possibly unidirectionally, with neighboring DGRs. By relying on this directed communication network and on simple computations executed by the local controllers, we provide two algorithms that allow the DGRs to determine their generation set-points in order to (i) regulate the system frequency and (ii) minimize total operational costs; like the control architectures from which ours is derived, both of these algorithms account for power output limits of the DGRs. Beyond introducing distributed alternatives to the top two generation control layers, a significant difference between our distributed control architecture and centralized counterparts used in bulk power systems is that ours is event triggered, i.e., generation set-points can be updated at non-fixed time instants, which eliminates unnecessary control efforts during periods in which the power demand remains relatively constant.

To demonstrate the effectiveness of our distributed generation control architecture, we tested it on a laboratory-grade microgrid. The electrical network of this microgrid is comprised of several small synchronous generators interconnected with resistive loads (inverter-interfaced power supplies are omitted due to lack of availability), whereas the cyber network for communication and control is comprised of Arduino microcontrollers outfitted with wireless transceivers. Using this microgrid, we experimentally verified the performance of the proposed control architecture under a variety of scenarios. Specifically, we provide results that illustrate the operation of the distributed frequency regulation and optimal dispatch functions following both an increase and a decrease in load. Additionally, we show how a DGR acting as spinning reserve can use the distributed frequency regulation function to independently determine if the collective power output limits of the online units is insufficient to balance the load, and act accordingly.

While the contributions outlined above describe the content discussed herein, they represent extensions of and improvements upon our preliminary work presented in [11], [12]. More specifically, we combine the frequency regulation and optimal dispatch algorithms proposed in the above papers into a complete distributed generation control architecture, outlining an event-triggered protocol which enables them to work together while accounting for microgrid-specific characteristics. Additionally, we provide a more thorough description of their operation and include some analysis on the frequency error dynamics of the closed-loop system. Finally, we build upon the experimental work presented in [11] to demonstrate both distributed generation control algorithms and the event-triggered protocol operating on a laboratory-grade microgrid.

Control of microgrids and distributed control of large power systems has received significant attention in the literature recently, but, to the authors' knowledge, no previous work matches the approach proposed herein. More specifically, [1]–[4], [13], [14] have only considered overall control strategies and centralized solutions to the problems we address in this paper. In [15], [16], the authors propose distributed methods for primary control; comparatively, our focus is on the top two control layers. Some work has been done to address the frequency regulation problem, with centralized approaches proposed in [2], [14], and, while [17]–[20] propose distributed solutions, all but [20] rely on undirected communication links. Although the approach in [20] also applies in directed communication networks, it is equivalent to a message passing protocol and requires an external signal to provide a control reference, which is not necessary in our work. While [21] proposes a distributed model predictive control approach for the middle generation control layer, the focus is on maintaining the proper power interchange between areas in bulk power transmission networks. Solving the optimal dispatch problem for microgrids has been addressed in a centralized manner in [4], with distributed solutions that rely on undirected information exchange proposed in [22]–[25]; among these, only [23] accounts for limits on generator output. Also, except for [25], no other previous work has proposed distributed solutions to both the frequency regulation and optimal dispatch problems, and compared to our approach, [25] does not account for generation limits. Finally, with the exceptions of [13], [19], much of the work proposed in the above-cited papers has been validated via simulations only. It should also

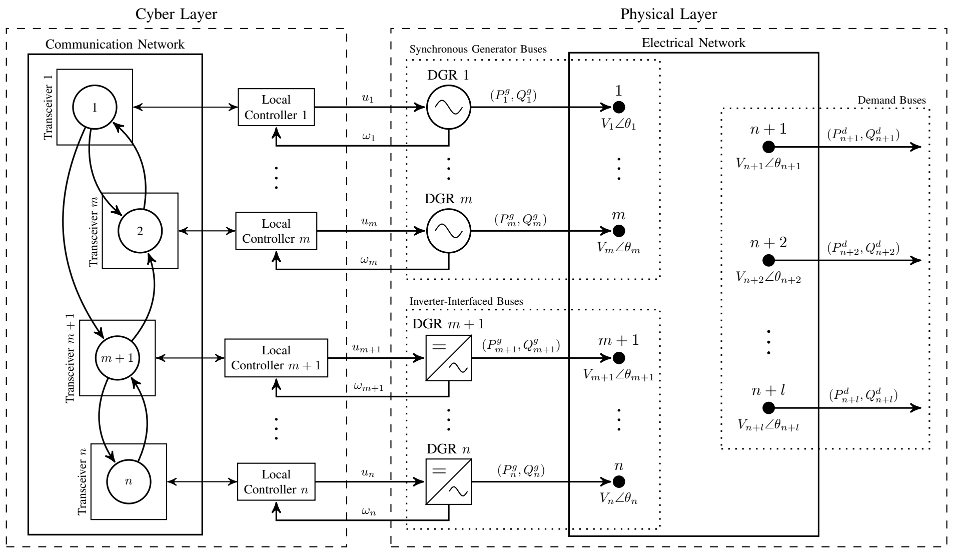


Fig. 1: Distributed generation control architecture block diagram: on the right, the connectivity between DGRs and loads is not explicitly shown; on the left, the communication topology linking the different transceivers is not explicitly shown.

be noted that, while we only consider active power control, there is a related body of literature which focuses on distributed control of reactive power for voltage regulation (see, e.g., [26], [27]); furthermore, a growing body of work has appeared recently which aims to solve similar problems to the ones we have addressed herein (see, e.g., [28], [29]).

The remainder of this paper is organized as follows. In Section II, we introduce models to describe (i) the physical layer of an islanded ac microgrid, including DGR dynamics, and (ii) the communication network describing the exchange of information among DGRs; additionally, we discuss a linear-iterative distributed algorithm that will be used as a primitive for implementing two algorithms on which the proposed control architecture relies. In Section III, we provide an overview of the desired control functions and outline specific control objectives. In Section IV, we describe the two algorithms that distributively implement the desired control functions. In Section V, we provide guidelines for choosing controller gains to ensure closed-loop stability. Section VI describes a laboratory-grade microgrid that we developed for testing the performance of the proposed architecture. Experimental results obtained using the aforementioned microgrid are presented in Section VII. Concluding remarks are presented in Section VIII.

II. Preliminaries

In this section, we introduce a nonlinear differential algebraic equation (DAE) model to describe the dynamics in the physical layer of an islanded ac microgrid, and characterize its synchronized solution. Next, we introduce a graph-theoretic model to describe a communication network which interconnects the local controllers of the power generators, henceforth referred to as distributed generation resources (DGRs), in the system. Figure 1 contains a block diagram that describes the interactions between these two networks; the components illustrated and the notation used in this diagram are introduced throughout this section and the next. Finally, we formulate a linear iterative algorithm that adheres to the aforementioned communication network which will serve as a primitive for distributively implementing the frequency regulation and optimal dispatch functions.

A. Physical Layer Model

Consider a system with $n + l$ buses. Assume that n of the buses are *DGR buses*, m of which only have synchronous generators connected to them, and $n - m$ of which only have inverter-interfaced power supplies connected to them; the remaining l buses are *demand buses*, i.e., they only have electric loads connected to them. Further, assume that the network interconnecting DGRs and demand buses is linear and that the admittances of the interconnections are represented by the matrix Y, where $G = \text{Re}\{Y\}$ and $B = \text{Im}\{Y\}$, $G, B \in \mathbb{R}^{(n+l)\times(n+l)}$, are the conductance and susceptance matrices, respectively. Without loss of generality, let $1, 2, \dots, m$ index the DGR buses with synchronous generators; let $m + 1, m + 2, \dots, n$ index the DGR buses with inverter-interfaced power supplies; let $n + 1, n + 2, \dots, n + l$ index the demand buses; and let V_i and θ_i denote the voltage magnitude and angle of bus $i = 1, 2, \dots, n + l$, respectively.

The dynamics of each synchronous generator is described by a structure-preserving model with constant complex voltage behind reactance (see, e.g., [30, Section 7.9.2]), augmented to include the governor dynamics. Thus, for a synchronous generator connected to DGR bus i, let $\delta_i(t)$ denote the angle of its voltage behind reactance (or "internal voltage") as measured with respect to a synchronous reference rotating at the nominal system electrical frequency, ω_0; $\omega_i(t)$ denote its rotor electrical angular speed; $P_i^m(t)$ denote its prime mover mechanical power; and $u_i(t)$ denote its generation set-point. Then, for all $i = 1, 2, \dots, m$, we have that

$$\frac{d\delta_i}{dt} = \omega_i - \omega_0, \tag{1}$$

$$M_i \frac{d\omega_i}{dt} = -D_i(\omega_i - \omega_0) + P_i^m - V_i \sum_{k=1}^{n+l} V_k \left[G_{ik}\cos(\theta_i - \theta_k) + B_{ik}\sin(\theta_i - \theta_k)\right], \tag{2}$$

$$\tau_i \frac{dP_i^m}{dt} = -P_i^m - \frac{1}{R_i\omega_0}(\omega_i - \omega_0) + u_i, \tag{3}$$

where D_i [s/rad] is the generator damping coefficient, M_i [s^2/rad] is a scaled inertia constant, τ_i [s] is the generator governor time constant, R_i [pu] is the speed-droop characteristic slope of the generator, and G_{ik} and B_{ik} are the (i, k) entries of the conductance and susceptance matrices, respectively.

Similarly, the dynamics of each inverter-interfaced power supply are described by a constant voltage behind reactance model, augmented to include a frequency-droop controller (see, e.g., [17]). Thus, for an inverter-interfaced power supply connected to DGR bus i, let $\delta_i(t)$ denote the angle of its internal voltage as measured with respect to a synchronous reference rotating at ω_0, and let $u_i(t)$ denote its generation set-point. Then, for all $i = m + 1, m + 2, \dots, n$, we have that

$$\frac{d\delta_i}{dt} = \omega_i - \omega_0 = \frac{1}{H_i}\left[u_i - V_i \sum_{k=1}^{n+l} V_k \left[G_{ik}\cos(\theta_i - \theta_k) + B_{ik}\sin(\theta_i - \theta_k)\right]\right], \tag{4}$$

where H_i [s/rad] is the speed-droop characteristic slope of the inverter-interfaced power supply.

Let P_i^d denote the real power demand of a load connected to demand bus i; then, for all $i = n+1, n+2, \ldots, n+l$, we have that

$$0 = -P_i^d - V_i \sum_{k=1}^{n+l} V_k \left[G_{ik}\cos(\theta_i - \theta_k) + B_{ik}\sin(\theta_i - \theta_k)\right]. \tag{5}$$

It should be noted that we have omitted the reactive power models for synchronous generators, inverter-interfaced power supplies, and loads as they are unnecessary for the analysis presented in this paper. Also note that while not explicitly described, the model in (5) (and its analogous reactive power model) also includes constant-impedance loads as their effect can be captured by adding the appropriate terms to the diagonal entries of the network admittance matrix; similarly, constant-current loads can also be incorporated into the demand bus model (see, e.g., [30, Section 6.3] for the procedure).

1) DGR synchronization: Let $u_i^r \in \mathbb{R}$ denote a constant set-point applied to DGR $i,\ i = 1, 2, \ldots, n$, between time instants t_r and t_{r+1}, i.e., $u_i(t) = u_i^r$, $t_r \leq t < t_{r+1}$, and assume that the real power load at demand bus i, $P_i^d,\ i = n+1, n+2, \ldots, n+l$, is constant. Under synchronous operation, all DGRs will operate at a constant common electrical frequency, ω^r, i.e., $\omega_i = \omega^r,\ i = 1, 2, \ldots, n$, which can be obtained as follows. In (2) – (4), set the left-hand side to zero; then, together with (5), and the fact that $\omega_i = \omega^r,\ i = 1, 2, \ldots, n$ we have

$$0 = -D_i(\omega^r - \omega_0) + P_i^m - V_i \sum_{k=1}^{n+l} V_k \left[G_{ik}\cos(\theta_i - \theta_k) + B_{ik}\sin(\theta_i - \theta_k)\right], \quad i = 1, 2, \ldots, m, \tag{6}$$

$$0 = -P_i^m - \frac{1}{R_i\omega_0}(\omega^r - \omega_0) + u_i^r, \qquad i = 1, 2, \ldots, m, \tag{7}$$

$$0 = -H_i(\omega^r - \omega_0) + u_i^r - V_i \sum_{k=1}^{n+l} V_k \left[G_{ik}\cos(\theta_i - \theta_k) + B_{ik}\sin(\theta_i - \theta_k)\right], \quad i = m+1, m+2, \ldots, n, \tag{8}$$

$$0 = -P_i^d - V_i \sum_{k=1}^{n+l} V_k \left[G_{ik}\cos(\theta_i - \theta_k) + B_{ik}\sin(\theta_i - \theta_k)\right], \qquad i = n+1, n+2, \ldots, n+l. \tag{9}$$

Assume that between time instants t_r and t_{r+1} there exists $\theta_i,\ i = 1, 2, \ldots, n+l$, and $V_i,\ i = 1, 2, \ldots, n+l$, that satisfy (6) – (9) which we denote by θ_i^r, and V_i^r, respectively; this is essentially equivalent to the existence of a solution to the network power flow equations [30]. Then, by summing (6) – (9) over all i, it is easy to see that all the sine terms cancel out, which leads to

$$\omega_r = \omega_0 + \frac{\sum_{i=1}^{n} u_i^r + \sum_{i=n+1}^{n+l} P_i^d - P_l^r}{\sum_{i=1}^{m}(D_i + 1/R_i\omega_0) + \sum_{i=m+1}^{n} H_i}, \tag{10}$$

where $P_l^r = \sum_{i=1}^{n+l}\sum_{k=1}^{n+l} V_i^r V_k^r G_{ik}\cos(\theta_i^r - \theta_k^r)$. From (10) it follows that

$$\Delta\omega^r := \omega^r - \omega_0 = \frac{\sum_{i=1}^{n} u_i^r + \sum_{i=n+1}^{n+l} P_i^d - P_l^r}{\sum_{i=1}^{m}(D_i + 1/R_i\omega_0) + \sum_{i=m+1}^{n} H_i}, \tag{11}$$

which, in the remainder of this work, we will refer to as *frequency error*. As we will discuss later, one of the control objectives will be to adjust the u_i^r's in a distributed fashion so as to drive $\Delta\omega^r$ to zero.

Remark 1: It is important to note that we have not addressed questions of existence, uniqueness, and stability of synchronized solutions to (1) – (5) as these are topics that have been well studied in the literature (see, e.g., [31]–[33] and the references therein), and their rigorous analysis is outside the scope of this paper. However, the result in (11) is sufficient for the analysis in Section V. □

B. Cyber Layer Model

To realize a distributed implementation of the frequency regulation and optimal dispatch functions, we assume that each DGR is outfitted with a local controller that is capable of performing simple computations. Additionally, we assume that each local controller is equipped with a transceiver through which information can be exchanged (possibly unidirectionally) with the local controllers of other nearby DGRs.

In this work, we describe the communication network interconnecting the DGR local controllers by a directed graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} := \{1, 2, \ldots, n\}$ is the vertex set, with each vertex—or node—corresponding to a DGR; and where $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set, with the ordered pair $(i, j) \in \mathcal{E}$ if node i can receive information from node j; see the left-hand side of Fig. 1 for a graphical depiction of the cyber network model. By modeling the network using a directed graph, we allow for a very general communication modality with possibly asymmetric communication links; thus, we may have a flow of information from node j to node i and not the converse, i.e., $(i, j) \in \mathcal{E}$ while $(j, i) \notin \mathcal{E}$. For notational convenience, we require self-loops for all nodes, that is, $(i, i) \in \mathcal{E}$, $\forall i \in \mathcal{V}$.

We define the set of vertices from which the local controller of DGR i can receive information to be $\mathcal{N}_i^- := \{j \in \mathcal{V} : (i, j) \in \mathcal{E}\}$. Similarly, we define the set of vertices to which i can send information to be $\mathcal{N}_i^+ := \{j \in \mathcal{V} : (j, i) \in \mathcal{E}\}$, and we refer to $\mathcal{N}_i^-$ and $\mathcal{N}_i^+$ as the in- and out-neighborhood of node i, respectively. Furthermore, we denote the cardinality of the out-neighborhood, referred to as the out-degree, of node i as $\mathcal{D}_i^+ := |\mathcal{N}_i^+|$. Note that as a consequence of the self-loop requirement, each vertex is a member of its own in- and out-neighborhood, i.e., $i \in \mathcal{N}_i^-$ and $i \in \mathcal{N}_i^+$, $\forall i \in \mathcal{V}$. A directed path from node j to node i is a sequence of nodes $j \equiv l_0, l_1, \ldots, l_t \equiv i$ such that $(l_{\tau+1}, l_\tau) \in \mathcal{E}$ for $\tau = 0, 1, \ldots, t-1$. Throughout the remainder of this paper, we assume that the directed graph $\mathcal{G}$ is strongly connected, i.e., for each ordered pair of vertices $i, j \in \mathcal{V}$, there is a path from j to i [34].

C. The Ratio-Consensus Algorithm

The ratio-consensus algorithm (see [35]) relies on two linear iterations executed in parallel by the local controller of each DGR, appropriately initialized according to the requirements of the problem. This algorithm serves as a primitive for distributively implementing the frequency regulation and optimal dispatch functions.

Consider a microgrid with n DGRs and assume that the communication network describing the exchange of information between them can be described by the graph-theoretic model described above, i.e., a strongly connected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$. The local controller of each DGR i participating in the ratio-consensus algorithm maintains two values, y_i and z_i, referred to as internal states, which are (independently) updated at each iteration to be a linear combination of the previous internal states of all nodes in its in-neighborhood. Specifically, for each iteration

$k \geq 0$, node i updates its internal states as

$$y_i[k+1] = \sum_{j \in \mathcal{N}_i^-} \frac{1}{\mathcal{D}_j^+} y_j[k], \tag{12}$$

$$z_i[k+1] = \sum_{j \in \mathcal{N}_i^-} \frac{1}{\mathcal{D}_j^+} z_j[k], \tag{13}$$

where $\mathcal{D}_j^+$ is the out-degree of DGR $j \in \mathcal{N}_i^-$. Note that given the self-loop requirement, the update for each internal state includes a weighted value of the previous local internal state of each node. Assuming that $z_i[k] \neq 0,\ \forall k$, at each iteration, the local controller of generator i computes

$$\gamma_i[k] = \frac{y_i[k]}{z_i[k]}. \tag{14}$$

The following lemma establishes that (14) converges to a constant that is equal for every i (see [35] for a proof).

Lemma 1: Let $y_i[k],\ \forall i$, be the result of iteration (12) for some $y_i[0],\ \forall i$, and $z_i[k],\ \forall i$, be the result of iteration (13) for some $z_i[0]$, where $z_i[0] > 0,\ \forall i$; then, we have that $\lim_{k \to \infty} \gamma_i[k] = \frac{\sum_{j=1}^n y_j[0]}{\sum_{j=1}^n z_j[0]},\ \forall i$.

The result in Lemma 1 implies that, through the exchange of information over the directed graph $\mathcal{G}$, ratio consensus allows the DGR local controllers to compute $\sum_{j=1}^n y_j[0] / \sum_{j=1}^n z_j[0]$, thus enabling them to acquire knowledge that is not directly obtainable from their in-neighbors. It should be noted that the number of iterations required to acquire this information is highly dependent on the structure of the graph modeling the communication network. Specifically, the convergence rate of (12) and (13) will depend on the second-largest in magnitude eigenvalue of the weight matrix $P = [P_{ij}]$, with $[P_{ij}] = \frac{1}{\mathcal{D}_j^+}$ when $(i,j) \in \mathcal{E}$ and $[P_{ij}] = 0$ otherwise. In the following section, we will show how the ratio-consensus algorithm can be utilized to implement the frequency regulation and optimal dispatch functions in a distributed fashion.

III. Control Objectives and Protocol

First, we provide an overview of the control objectives to be met by our distributed architecture for generation control. These objectives are similar to those sought by centralized architectures for generation control used in bulk power systems, but they are achieved through a distributed computation over the communication network interconnecting the DGRs. This computation relies on two algorithms—their formal definition is deferred to Section IV—that enable the distributed implementation of the frequency regulation and optimal dispatch functions. Finally, we specify a protocol that allows the DGRs to determine when to utilize each of the two aforementioned algorithms; this protocol is based on purely local measurements and as such is completely distributed.

A. Overview of Control Functions and Objectives

Despite being smaller and having lower ratings, the generation control objectives for islanded ac microgrids are similar to those for large power systems. Therefore, the control functions of our distributed architecture are derived from the functionality provided by the typical three-layer generation control architecture used in bulk power transmission networks. In the discussion that follows, we briefly summarize the control functions of each of these three layers, highlighting the aspects that are relevant to the realization of our distributed architecture.

1) Droop Control: The purpose of this control function is to instantaneously balance generation with demand through local actions taken by the speed governors of each generation unit; in the synchronous generator model in (1) – (3), for example, the governor dynamic behavior is described by (3). Typically, the low-level control mechanism that provides this functionality, often referred to as *primary generation control* or *droop control*, is designed to allow instantaneous load changes to be divided proportionally with respect to generator ratings [36]. Regardless of power system size, droop control requires only local measurements and actuations and is inherently decentralized; thus, there is no need for a distributed alternative. As a result, in the remainder of this paper, we omit further discussion on droop control other than to facilitate discussion of other control functions.

2) Frequency Regulation: While primary generation control provides a simple and effective way to compensate for changes in load, it does so at the expense of frequency deviations. If not eliminated, these variations may result in equipment damage or otherwise undesirable operation, e.g., induction motors and their connected loads rotating at speeds for which they are not designed. In large power systems, there is a control function, commonly referred to as *secondary generation control* or *automatic generation control* (AGC), that is responsible for returning the frequency to its nominal value, e.g., 60 Hz, following transients induced by load changes. In addition to regulating frequency, secondary generation control is also responsible for maintaining proper interchange power values between control areas. However, in the context of microgrids, there is no notion of control areas; thus, in our distributed architecture, the only objective of the secondary control function is *frequency regulation*.

To compensate for frequency deviations arising from changes in load and the subsequent response of droop control, the secondary generation control in large-scale power systems is commonly implemented using a closed-loop controller that adjusts the set-point of each generator based upon the integral of the frequency error (see, e.g., [10], [36]). The implementation of AGC is centralized with measurements and control signals telemetered to and from the generating units and the control center where the computer implementing the controller resides.

In Section IV-A, we will see that, when executed over several rounds and properly initialized according to an estimate of the incremental demand for generation, the ratio-consensus algorithm can be used to replicate the functionality of the aforementioned centralized closed-loop controller for large power system AGC.

3) Optimal Dispatch: Although primary and secondary generation controls suffice to balance generation with demand and subsequently regulate frequency, they actuate without accounting for operational costs. Consequently, in a large power system, there is a third generation control function which seeks to determine the most economical way to allocate generation demand among all generators. That is, assuming the cost of each generator is a function of its power output, the objective of this tertiary control is to minimize the total generation cost subject to individual power output limits (see, e.g., [37]).

For large power systems, similar to secondary generation control, the optimal dispatch function relies on a decision-making algorithm executed on a computer that resides in a centralized location, e.g., a control center, with knowledge of the cost function and limits of each generator as well as the sum of the power output of all the generators. In Section IV-B, we will discuss an algorithm based on ratio consensus that is suitable for distributively implementing the optimal dispatch function.

B. Control Mode Protocol

Large power systems are vast, often containing thousands of loads that vary frequently; consequently, the actions of droop control result in near-constant frequency deviations. Furthermore, given that measurements and control signals must be telemetered to and from a central location, secondary and tertiary generation control in bulk power systems do not run continuously, relying instead on periodically triggered actuations performed approximately every two-to-five seconds and five minutes, respectively [10].

In the context of the class of microgrids considered in this work, and in order to motivate the control mode protocol proposed in this section, we make the following assumptions.

[A1.] There are fewer loads, each of which may be large relative to the total capacity of the DGRs.

[A2.] The collective inertia provided by certain types of DGRs (e.g., synchronous generators) may be small.

[A3.] Large frequency deviations may result following a load change.

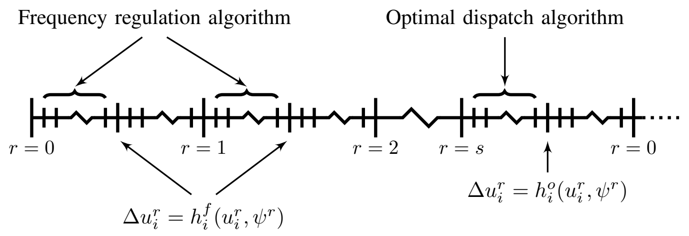
Given assumptions **[A1]** – **[A3]**, while minimizing cost is important, the main control objective in islanded microgrids is to regulate frequency [2]. As such, rather than operating on a fixed time interval, we adopt an event-triggered control architecture which only actuates when the frequency error exceeds a specified bound, and does not optimally dispatch the DGRs until the frequency is sufficiently close to the nominal value. In the discussion that follows, we specify the protocol utilized by the DGR local controllers to determine the appropriate control mode, and introduce notation that will be used in subsequent developments.

Remark 2: While assumptions **[A1]** – **[A3]** motivate our control mode protocol for microgrids, if they do not hold, scenarios may result in which the optimal dispatch algorithm is never executed. In particular, it is unlikely that these assumptions will be valid in, for example, bulk power transmission networks with a large number of frequently varying loads. Moreover, as noted in Section II-C, the distributed algorithm on which our control relies may not scale well for systems with a large number of generation units. □

To return the frequency to its nominal value following one or more load changes, and to subsequently dispatch the DGRs optimally, our architecture relies on the execution of two distributed algorithms (to be described in Section IV) over several rounds. Let r index the rounds over which the distributed algorithms are executed, and denote the generation set-point of DGR i at round r by u_i^r. Then, the relation between the generation set-point of DGR i between two consecutive rounds, r and $r+1$, is given by

$$u_i^{r+1} = u_i^r + \Delta u_i^r, \tag{15}$$

where Δu_i^r is the amount by which DGR i should adjust its output at round r. In subsequent developments, we assume that after the u_i^r's are set at each round r, sufficient time elapses such that a new synchronized operating point is reached wherein $\omega_1^r = \omega_2^r = \cdots = \omega_m^r =: \omega^r$ (see analysis in Section II-A1) before each DGR computes Δu_i^r and adjusts its set-point according to (15). Furthermore, if a feasible solution exists at each round, i.e.,



Frequency regulation algorithm
Optimal dispatch algorithm
$r = 0$
$r = 1$
$r = 2$
$r = s$
$r = 0$
$\Delta u_i^r = h_i^f(u_i^r, \psi^r)$
$\Delta u_i^r = h_i^o(u_i^r, \psi^r)$

Fig. 2: Timeline of distributed generation control algorithms.

$\sum_{i=1}^n \underline{P}_i \leq \sum_{i=1}^n u_i^r \leq \sum_{i=1}^n \overline{P}_i$, where $\underline{P}_i$ and $\overline{P}_i$ denote the lower and upper limits on the power output of DGR i, respectively, the algorithms developed in the next section will ensure that $\underline{P}_i \leq u_i^r \leq \overline{P}_i,\ i = 1, 2, \ldots, n$.

Let $\epsilon > 0$ denote the maximum allowable electrical frequency[1] error, let ξ_i^r be an indicator for the optimal dispatch function for DGR i, i.e.,

$$\xi_i^r = \begin{cases} 0, & \text{if DGR } i \text{ optimally dispatched since last freq. reg.,} \\ 1, & \text{otherwise,} \end{cases} \tag{16}$$

and define $u^r = [u_1^r, \ldots, u_n^r]^{\mathrm{T}}$. Then, each DGR i determines the value of Δu_i^r at round r as follows:

$$\begin{aligned} \Delta u_i^r &= h_i(u_i^r, \psi^r) \\ &:= \begin{cases} h_i^f(u_i^r, \psi^r), & \text{if } |\omega^r - \omega_0| > \epsilon, \\ h_i^o(u_i^r, \psi^r), & \text{if } |\omega^r - \omega_0| \leq \epsilon \text{ and } \xi_i^r = 1, \\ 0, & \text{otherwise,} \end{cases} \end{aligned} \tag{17}$$

for some function $h_i^f : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$ to be defined in Section IV-A, and some function $h_i^0 : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$ to be defined in Section IV-B; and where the input ψ^r is given by

$$\psi^r = \begin{cases} \phi^f(\omega^r, u^r), & \text{if } |\omega^r - \omega_0| > \epsilon, \\ \phi^o(u^r), & \text{if } |\omega^r - \omega_0| \leq \epsilon, \end{cases} \tag{18}$$

for some functions $\phi^f : \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}$ and $\phi^o : \mathbb{R}^n \mapsto \mathbb{R}$ to be defined in Sections IV-A and IV-B, respectively.

Given the protocol specified in (16) – (18), the timeline shown in Fig. 2 outlines the relative timescales over which the frequency regulation and optimal dispatch algorithms are executed. If we let s index the round during which the DGRs are optimally dispatched, i.e., $r = s$ is the round for which $|\omega_i^r - \omega_0| \leq \epsilon$ and $\xi_i^r = 1$, the figure illustrates that, for $r < s$, the magnitude of the frequency error exceeds the specified bound and the frequency regulation algorithm is executed. In particular, at rounds $r = 0, 1, \ldots, s-1$, the DGRs adjust their generation set-points using $h_i^f(\cdot)$, the collective effect of which replicates the functionality of a discrete-time integral controller. At $r = s$, the

[1]We assume that, aside from a proportionality constant, the electrical angular speed of DGR i, ω_i, is equivalent to the electrical frequency of the bus to which it is connected, and thus use the terms interchangeably.

magnitude of the frequency error has been reduced to within the specified bound and the DGR generation set-points are modified according to $h_i^o(\cdot)$, which yields the optimal generation allocation among the DGRs. Regardless of the algorithm used, after the set-points are adjusted, the system evolves according to (1) – (5).

From (17), we see that in order to determine the incremental set-point of each DGR, the *global* information captured in the value of ψ^r is needed; this value is determined by the output of the functions $\phi^f(\cdot)$ and $\phi^o(\cdot)$ in (18) for frequency regulation and optimal dispatch, respectively. Note also that, in order to evaluate these functions, it is necessary to have the set-points and electrical angular speeds of all the DGRs in the system. Additionally, as we will see in Section IV, $\phi^f(\cdot)$ and $\phi^o(\cdot)$ both depend on the minimum and maximum output power of all the DGRs in the system, while $\phi^o(\cdot)$ additionally requires the power output cost function parameters for each DGR. Although this implies that in order to evaluate (18), and in turn (17), it is necessary to pass all the above information to a centralized processor, in Section IV we will explain how $\phi_i^f(\cdot)$ and $\phi_i^o(\cdot)$ can be evaluated using a distributed computation which relies on the ratio-consensus algorithm discussed in Section II-C.

IV. Control Algorithms

In this section, we describe the two algorithms that serve to distributively compute functions $h_i^f(\cdot)$ and $h_i^o(\cdot)$ in (17), thus enabling the implementation of frequency regulation and optimal dispatch without the need for a centralized processor.

A. Distributed Frequency Regulation

Consider a microgrid consisting of n DGRs outfitted with local controllers, the interconnections of which can be represented by a strongly connected directed graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$. Assume that all the DGRs are synchronized to a common angular speed ω^r (not necessarily ω_0) determined by the values of the current DGR set-points, i.e., the u_i^r's (see analysis in Section II-A1). If we assume that each of the local controllers is capable of obtaining a measurement of the speed of the DGR to which it is connected,[2] an estimate for the amount by which DGR i should adjust its generation set-point to drive the electrical frequency to the nominal value at round r is

$$\Delta \hat{u}_i^r = \kappa_i(\omega^r - \omega_0) =: \kappa_i \Delta\omega^r, \tag{19}$$

where $\kappa_i < 0$ is some gain chosen by the local controller (in Section V, we give some guidelines on how each DGR can choose this gain).

Given the common electrical frequency at which the DGRs operate, and with no power output constraints, a simple solution to the frequency regulation problem is for each DGR to adjust its set-point at each round r as in (19) until the frequency error $\Delta\omega^r$ vanishes. However, if constraints due to DGR power ratings or operational limitations are considered, this solution may not be feasible. Furthermore, it is well known that, if the DGRs do not operate at a common electrical frequency, applying independent control to each DGR may result in an unstable

[2]While a speed measurement is required for primary generation control, we explicitly state this assumption to account for systems in which no droop controller is present, or the primary and secondary control functions are performed by separate controllers and sensors.

system (this is commonly referred to as isochronous governor control, see, e.g., [10, Section 9.5]). Thus, in order to address these problems, we use the ratio-consensus algorithm, with appropriately chosen initial conditions that account for DGR power output limits, to divide the estimated incremental generation demand among all DGRs.

Define $\Delta\underline{P}_i^r := \underline{P}_i - u_i^r$, and $\Delta\overline{P}_i^r := \overline{P}_i - u_i^r$ to be the incremental minimum and maximum power output of DGR i at round r, respectively. As mentioned previously, the characteristics of the loads and DGRs in islanded microgrids can lead to large frequency deviations; thus, we adopt an allocation scheme for the distributed frequency regulation algorithm whereby the incremental set-point of each DGR is adjusted proportionally to $\Delta\overline{P}_i^r - \Delta\underline{P}_i^r$ in order to eliminate the frequency error in as few rounds as possible. Furthermore, to ensure a feasible solution, we assume that, at each round r, the sum of the estimated incremental set-points lies within the collective incremental power output limits of the DGRs, i.e.,

$$\sum_{i=1}^{n} \Delta\underline{P}_i^r \leq \sum_{i=1}^{n} \Delta\hat{u}_i^r \leq \sum_{i=1}^{n} \Delta\overline{P}_i^r. \tag{20}$$

Then, the ratio consensus states in (12) and (13) are initialized to $y_i[0] = \Delta\hat{u}_i^r - \Delta\underline{P}_i^r$ and $z_i[0] = \Delta\overline{P}_i^r - \Delta\underline{P}_i^r$, $\forall i \in \mathcal{V}$, respectively. Given these initial conditions, it follows from Lemma 1 that DGR i can asymptotically obtain

$$\begin{aligned}\psi^r &= \lim_{k\to\infty} \gamma_i^r[k] = \lim_{k\to\infty} \frac{y_i^r[k]}{z_i^r[k]} = \frac{\sum_{j=1}^{n} \kappa_j(\omega^r - \omega_0) - \sum_{j=1}^{n} \Delta\underline{P}_j^r}{\sum_{j=1}^{n}\left(\Delta\overline{P}_j^r - \Delta\underline{P}_j^r\right)} = \frac{\sum_{j=1}^{n} \kappa_j(\omega^r - \omega_0) - \sum_{j=1}^{n}(\underline{P}_j - u_j^r)}{\sum_{j=1}^{n}\left(\overline{P}_j - \underline{P}_j\right)},\\ &=: \phi^f(\omega^r, u^r),\end{aligned} \tag{21}$$

which represents the magnitude of the sum of the estimated set-point adjustments as a percentage of the collective incremental power output limits of the DGRs.

After the ratio-consensus algorithm has converged and each DGR has obtained ψ^r, the incremental set-point of DGR i at round r is given as

$$\begin{aligned}\Delta u_i^r &= \Delta\underline{P}_i^r + \psi^r\left(\Delta\overline{P}_i^r - \Delta\underline{P}_i^r\right),\\ &= \underline{P}_i - u_i^r + \psi^r\left(\overline{P}_i - \underline{P}_i\right),\\ &=: h_i^f\left(u_i^r, \psi^r\right),\end{aligned} \tag{22}$$

and DGR i adjusts its reference command according to (15). Given this choice of initial conditions and the definition of $h_i^f(\cdot)$, we refer to the allocation scheme described in (19) – (22) as *fair splitting*.

In practice, the use of ratio consensus for obtaining ψ^r in (22) for frequency regulation will not extend over an infinite number of iterations but will only be executed for a finite number of iterations, k_f. A finite number of iterations implies a $\gamma_i[k_f]$ in (14) that is not necessarily the same for all nodes i, but can be bounded away from the true (limiting) value of ψ_r in (21) by a constant, ϵ_f. Moreover, the number of iterations, k_f, can be chosen, based upon the second-largest in magnitude eigenvalue of the weight matrix $P = [P_{ij}]$ (with $[P_{ij}] = \frac{1}{\mathcal{D}_j^+}$ when $(i, j) \in \mathcal{E}$ and $[P_{ij}] = 0$ otherwise) and the initial conditions of the ratio consensus algorithm, to make ϵ_f arbitrarily small.

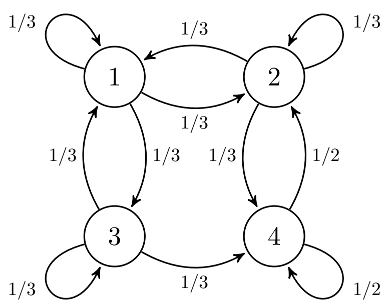


Fig. 3: Graph of 4-node network.

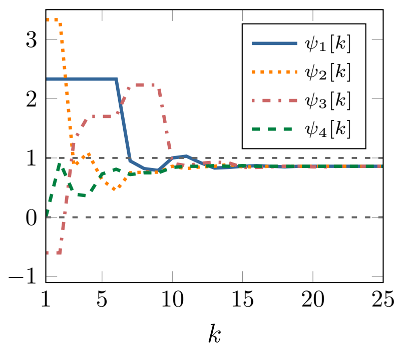


Fig. 4: Fair-splitting algorithm evolution with feasible solution.

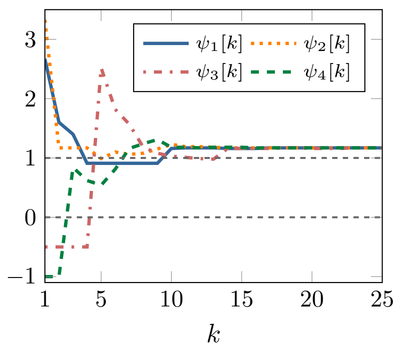


Fig. 5: Fair-splitting algorithm evolution with infeasible solution.

We omit a detailed discussion of this choice because it pertains to the convergence properties of the ratio consensus algorithm which is not the focus of this paper[3] and also because it was not an issue in our experiments. Next, we provide an example that illustrates the operation of the fair-splitting scheme and the choice of k_f.

Example 1 (Fair Splitting Feasible Solution): Consider the 4-node network in Fig. 3, where each node represents the local controller of a DGR participating in the fair-splitting algorithm. Omitting the index of the round, let $\Delta\hat{u} = \left[0.25, 0.4, -0.35, -0.2\right]^{\mathrm{T}}$, and let the incremental minimum and maximum power outputs of the DGRs be given, respectively, by $\Delta\underline{P} = \left[-0.1, -0.1, -0.2, -0.1\right]^{\mathrm{T}}$ and $\Delta\overline{P} = \left[0.05, 0.05, 0.05, 0.05\right]^{\mathrm{T}}$, such that $\sum_{i=1}^{4} \Delta\underline{P}_i \leq \sum_{i=1}^{4} \Delta\hat{u}_i \leq \sum_{i=1}^{4} \Delta\overline{P}_i$. Then, the initial values of y and z are set to $y[0] = \left[0.35, 0.5, -0.15, -0.1\right]^{\mathrm{T}}$, and $z[0] = \left[0.15, 0.15, 0.25, 0.15\right]^{\mathrm{T}}$.

Given the above initial conditions, the evolution of $\psi_i[k],\ i = 1, 2, 3, 4$, over 25 iterations is shown in Fig. 4. From the figure, we see that after approximately 15 iterations, all nodes converge to a solution in which $\psi = 0.86$. Thus, the nodes determine the solution is feasible (see the discussion below) and compute the incremental generation set-point according to (22), and we have that $\Delta u = \left[0.029, 0.029, 0.015, 0.029\right]^{\mathrm{T}}$. ■

If we assign an objective function that takes value zero when the feasible set defined by the inequality constraints in (20) is nonempty or ∞ when the feasible set is empty, the proposed distributed approach to frequency regulation is equivalent to solving a feasibility problem (see, e.g., [39]). In particular, the value of ψ^r found using the fair-splitting procedure can be utilized by each DGR to independently determine if the sum of the incremental set-points is within the collective power output limits of the available DGRs. Specifically, at round r, if $\psi^r < 0$ or $\psi^r > 1$, each DGR knows that the collective bounds on power output have been exceeded; we illustrate this idea next.

Example 2 (Fair Splitting Infeasible Solution): Consider the 4-node network in Fig. 3, where each node represents the local controller of a DGR participating in the fair-splitting algorithm. Omitting the index of the

[3]For example, our recent work in [38] goes a step further and discusses how the nodes can determine, in a distributed fashion, when to stop iterating given a desirable ϵ_f without prior knowledge of the network (i.e., without knowledge of the matrix P).

round, let $\Delta\hat{u} = \Big[0.3, 0.45, -0.2, -0.15,\Big]^{\mathrm{T}}$, and let the incremental minimum and maximum power outputs of the nodes be given respectively as $\Delta\underline{P} = \Big[-0.1, -0.05, -0.1, -0.05\Big]^{\mathrm{T}}$ and $\Delta\overline{P} = \Big[0.05, 0.1, 0.1, 0.05\Big]^{\mathrm{T}}$, such that $\sum_{i=1}^{4} \Delta\underline{P}_i = -0.3$ and $\sum_{i=1}^{4} \Delta\overline{P}_i = 0.3 < \sum_{i=1}^{4} \Delta\hat{u}_i = 0.4$. Then, we have that $y[0] = \Big[0.4, 0.5, -0.1, -0.1\Big]^{\mathrm{T}}$ and $z[0] = \Big[0.15, 0.15, 0.2, 0.1\Big]^{\mathrm{T}}$.

The evolution of $\psi_i[k],\ i = 1, 2, 3, 4$, over 25 iterations is shown in Fig. 5. From the figure, we see that after approximately 15 iterations, all nodes converge to a solution in which $\psi = 1.17$. Thus, the nodes determine the solution is infeasible (as they cannot adjust their output beyond their maximum power output). In Section VII, we show how the global information available through ψ can be used to, e.g., trigger reserve DGRs to come online. ■

Remark 3: If a DGR i does not participate in frequency regulation, it will set its u_i to some constant value u_i^0; however, this does not preclude DGR i from participating in the distributed algorithm for frequency regulation, i.e., the computation and communication aspects of the distributed protocol. To this end, the DGR only needs to set $\underline{P}_i = \overline{P}_i = u_i$; thus, it will initialize its z_i to $z_i[0] = \overline{P}_i - \underline{P}_i = 0$, and after the algorithm is executed, like all other DGRs, it will compute $\Delta u_i^r = \Delta\underline{P}_i^r + \psi^r(\Delta\overline{P}_i^r - \Delta\underline{P}_i^r) = \underline{P}_i - u_i^r + \psi^r(\overline{P}_i - \underline{P}_i) = 0$ such that $u_i^{r+1} = u_i^r + \Delta u_i^r = u_i^r = u_i^0$. Note that while Lemma 1 requires $z_i[0] > 0,\ \forall i$, in reality, only one $z_i[0]$ must be strictly positive while the others could be equal to zero. In practice, the nodes compute a close approximation to ψ^r after a sufficiently large number of iterations, k_f. Given that the graph is strongly connected, $z_i[k_f]$ is guaranteed to be bounded away from zero, which implies finite ψ^r. □

Remark 4: The fair-splitting allocation scheme is not the only viable choice for distributively regulating frequency using the ratio-consensus algorithm. Other options include allocating incremental set-points based upon: (i) the response time of the DGRs such that the estimated incremental demand is divided proportionally to, for example, the ramp rates of the DGRs or the time constant of the governors; or (ii) the proximity of each DGR to its stability limit. We leave investigation of such alternatives to future work. □

B. *Distributed Optimal Dispatch*

Assuming a feasible solution exists, the execution of several rounds of the frequency regulation scheme described previously will make the absolute value of the frequency error, $\Delta\omega^r$, smaller than some bound, ϵ. Let $r = s$ be the round at which this event occurs; then, from (11) it follows that the u_i^s's must be such that

$$\sum_{i=1}^{n} u_i^s = \sum_{i=n+1}^{n+l} P_i^d + \Delta\omega^s\big(\sum_{i=1}^{m}(D_i + 1/R_i\omega_0) + \sum_{i=m+1}^{n} H_i\big) := \chi.$$

As described previously, the fair-splitting procedure adjusts the incremental set-points of the DGRs proportionally with respect to incremental power output limits; thus, the u_i^s's might not necessarily be optimal, i.e., they might not minimize the overall cost of operation. Therefore, we would like to have a different generation allocation, $u_i^{s*},\ i = 1, 2, \dots, n$, among the DGRs while: (i) still satisfying their individual power output constraints, i.e., $\underline{P}_i \le u_i^{s*} \le \overline{P}_i,\ i = 1, 2, \dots, n$, and (ii) keeping their sum as before, i.e., $\sum_{i=1}^{n} u_i^{s*} = \sum_{i=1}^{n} u_i^s =: \chi$ (as we will show in Section V, this will ensure that $\Delta\omega^r$ remains within ϵ). That is, assuming that the cost associated with each DGR is quadratic, we would like to find the global solution, which we denote by u_i^{s*}, $i = 1, \dots, n$, of the following constrained optimization:

$$\begin{aligned} \underset{u_1, u_2, \dots, u_n}{\text{minimize}} \quad & \sum_{i=1}^{n} \frac{(u_i - \alpha_i)^2}{2\beta_i} = \sum_{i=1}^{n} c_i(u_i) \\ \text{subject to} \quad & \sum_{i=1}^{n} u_i = \chi \\ & 0 \leq \underline{P}_i \leq u_i \leq \overline{P}_i, \ i = 1, 2, \dots, n, \end{aligned} \tag{23}$$

where $\alpha_i \leq 0$, $\beta_i > 0$; this problem is commonly referred to as *optimal dispatch* [10]. The constants α_i and β_i are a characteristic of the primary source of energy for DGR i; in the case of a synchronous generator DGR, if, e.g., the prime mover is a gas turbine, these constants are determined by the heat-rate curve (see, e.g., [37] for a detailed discussion). For an inverter-interfaced DGR, if, e.g., the primary energy source is an array of photovoltaic panels, α_i represents the maximum power point (MPP). while β_i penalizes deviations from the MPP.

1) A Centralized Solution to the Optimal Dispatch Problem: The problem of optimally dispatching DGRs at round s given in (23) is convex and has a separable structure [40, pp. 502-506]; thus, its solution can be found by solving the Lagrange dual (see, e.g., [39], [40]), which is given by

$$\text{maximize} \quad \lambda\chi + \sum_{i=1}^{n} f_i(\lambda), \tag{24}$$

where $f_i(\lambda), \ \forall i$, are

$$f_i(\lambda) = \begin{cases} \frac{(\underline{P}_i - \alpha_i)^2}{2\beta_i} - \lambda \underline{P}_i, & \lambda < \underline{\lambda}_i, \\ -\lambda(\alpha_i + \lambda\frac{\beta_i}{2}), & \underline{\lambda}_i \leq \lambda \leq \overline{\lambda}_i, \\ \frac{(\overline{P}_i - \alpha_i)^2}{2\beta_i} - \lambda \overline{P}_i, & \overline{\lambda}_i < \lambda, \end{cases} \tag{25}$$

with $\underline{\lambda}_i = \frac{\underline{P}_i - \alpha_i}{\beta_i}$ and $\overline{\lambda}_i = \frac{\overline{P}_i - \alpha_i}{\beta_i}$.

The Lagrange dual problem defined by (24) is convex and, in this case, provides the optimal solution to the primal problem. Furthermore, the cost function in (25) is continuously differentiable, i.e., $f_i(\cdot) \in \mathcal{C}^1, \ \forall i$; thus, if the dual problem is feasible, the optimal solution λ^* must satisfy

$$\chi - \sum_{i=1}^{n} g_i(\lambda^*) = 0, \tag{26}$$

where $g_i(\lambda) = -\frac{df_i(\lambda)}{d\lambda}$, i.e.,

$$g_i(\lambda) = \begin{cases} \underline{P}_i, & \lambda < \underline{\lambda}_i, \\ \alpha_i + \lambda\beta_i, & \underline{\lambda}_i \leq \lambda \leq \overline{\lambda}_i, \\ \overline{P}_i, & \overline{\lambda}_i < \lambda. \end{cases} \tag{27}$$

Now, obtaining the value λ^* that satisfies (26) is equivalent to finding the intersection point of the functions $g(\lambda) := \sum_{i=1}^n g_i(\lambda)$ and $h(\lambda) := \chi$; this can be accomplished by evaluating the function $g(\cdot)$ for (at most) $2n$ values corresponding to the elements in $\mathcal{U} := \{\underline{\lambda}_1, \overline{\lambda}_2, \dots, \underline{\lambda}_n, \overline{\lambda}_n\}$. To this end, define $\mathcal{U}^+ := \{\lambda \in \mathcal{U} : g(\lambda) \geq \chi\}$, and $\mathcal{U}^- := \{\lambda \in \mathcal{U} : g(\lambda) \leq \chi\}$; then, as proposed in [41], the value of λ^* is given by

$$\begin{aligned} \lambda^* &= \lambda^+ - \big(g(\lambda^+) - \chi\big) \frac{\lambda^+ - \lambda^-}{g(\lambda^+) - g(\lambda^-)}, \\ &=: \phi^o(u^s), \end{aligned} \tag{28}$$

where

$$\lambda^+ = \underset{\lambda \in \mathcal{U}^+}{\operatorname{argmin}} |g(\lambda) - \chi|, \tag{29}$$

$$\lambda^- = \underset{\lambda \in \mathcal{U}^-}{\operatorname{argmin}} |g(\lambda) - \chi|, \tag{30}$$

and the solution to the primal problem given in (23) is

$$u_i^{s*} = g_i(\lambda^*), \ i = 1, 2, \dots, n. \tag{31}$$

From (28), it is easy to see that λ^* is obtained by a linear interpolation between the values of λ^- and λ^+; for this reason, we refer to the procedure outlined in (28) – (31) as the *interpolation method*.

2) A Distributed Implementation of the Interpolation Method: Although (31) provides the unique global solution to the optimal dispatch problem by solving its Lagrange dual, in order to determine $g(\lambda^+)$ and $g(\lambda^-)$, a centralized entity with knowledge of all the individual $g_i(\lambda)$'s and χ is required. If we rearrange (28), however, we will see that ratio consensus, together with a simple message-passing protocol, can be utilized to find λ^* without requiring a centralized decision maker or the need for each DGR to obtain all the individual functions $g_i(\cdot)$, the values they take for λ^+ or λ^-, or the value of χ.

First, note that (28) can be rewritten as

$$\lambda^* = \lambda^+ - \left(\frac{g(\lambda^+)}{\chi} - 1\right) \frac{\lambda^+ - \lambda^-}{\frac{g(\lambda^+)}{\chi} - \frac{g(\lambda^-)}{\chi}}, \tag{32}$$

where

$$\frac{g(\lambda^+)}{\chi} = \frac{\sum_{i=1}^n g_i(\lambda^+)}{\sum_{i=1}^n u_i^s}, \tag{33}$$

$$\frac{g(\lambda^-)}{\chi} = \frac{\sum_{i=1}^n g_i(\lambda^-)}{\sum_{i=1}^n u_i^s}, \tag{34}$$

which are ratios of sums of values that are known or can be computed by each local controller. As (33) and (34) imply, however, to find the optimal solution, each DGR must know λ^+ and λ^- *a priori*, that is, if each node knew which two $\lambda \in \mathcal{U}$ satisfied (29) and (30), ratio consensus could be used by the nodes to asymptotically obtain the ratios defined in (33) and (34), and consequently λ^*. From (32), we see that the criteria for determining λ^+ and λ^-

are equivalent to finding $\lambda \in \mathcal{U}$ such that $g(\lambda)/\chi$ is closest to 1 from above and below, respectively; thus, if each DGR obtains $g(\lambda)/\chi$, $\forall \lambda \in \mathcal{U}$, it can determine which ratios correspond to $g(\lambda^{+})/\chi$ and $g(\lambda^{-})/\chi$. Furthermore, if each DGR also knows which $\lambda \in \mathcal{U}$ correspond to λ^{+} and λ^{-}, λ^{*} can be computed and the optimal solution to the primal problem can be found using (31). Next, we provide a three-step procedure that enables the nodes to distributively perform the tasks above; the steps of this procedure are as follows.

[S1.] Without loss of generality, assume that initially, each node i only knows its respective $\underline{\lambda}_i$ and $\overline{\lambda}_i$. Then, by broadcasting these values, all nodes in the out-neighborhood of i can obtain them, and in turn, broadcast them to their out-neighbors. Proceeding in this fashion, after a finite number of steps, bounded by the diameter of the graph representing the communication network, each node i will obtain every $\lambda \in \mathcal{U}$.

[S2.] Once each node has acquired every $\lambda \in \mathcal{U}$, ratio consensus is used with $2n$ numerator states and a single denominator state. Let $\underline{y}_i^{(j)}$ and $\overline{y}_i^{(j)}$, for $j = 1, \dots, n$, and z_i, be the numerator and denominator states maintained by node i, respectively. Then, if the states are initialized such that $\underline{y}_i^{(j)}[0] = g_i(\underline{\lambda}_j)$, $\overline{y}_i^{(j)}[0] = g_i(\overline{\lambda}_j)$, for $j = 1, \dots, n$, and $z_i[0] = u_i^s$, it follows from Lemma 1 that each node i can asymptotically obtain

$$\underline{\gamma}_i^{(j)} = \lim_{k\to\infty} \frac{\underline{y}_i^{(j)}[k]}{z_i[k]} = \frac{\sum_{l=1}^{n} g_l(\underline{\lambda}_j)}{\sum_{l=1}^{n} u_l^s} = \frac{g(\underline{\lambda}_j)}{\chi}, \tag{35}$$

$$\overline{\gamma}_i^{(j)} = \lim_{k\to\infty} \frac{\overline{y}_i^{(j)}[k]}{z_i[k]} = \frac{\sum_{l=1}^{n} g_l(\overline{\lambda}_j)}{\sum_{l=1}^{n} u_l^s} = \frac{g(\overline{\lambda}_j)}{\chi}, \tag{36}$$

for $j = 1, \dots, n$, which correspond to $g(\lambda)/\chi$, $\forall \lambda \in \mathcal{U}$. Therefore, by determining which ratios are closest to 1 from above and below, and the corresponding λ^{+} and λ^{-}, respectively, each DGR can compute λ^{*} using (32).

[S3.] After the values $\underline{\gamma}_i^{(j)}$ and $\overline{\gamma}_i^{(j)}$, $i, j = 1, \dots, n$, have converged according to the ratio-consensus algorithm, and each node has determined λ^{*}, DGR i adjusts its reference command according to (15) where its incremental set-point at round $r = s$, with $\psi^s = \lambda^{*}$, is given by

$$\begin{aligned} \Delta u_i^s &= g_i(\psi^s) - u_i^s, \\ &= \begin{cases} \underline{P}_i - u_i^s, & 0 \le \psi^s < \underline{\lambda}_i, \\ \alpha_i + \psi^s \beta_i - u_i^s, & \underline{\lambda}_i \le \psi^s \le \overline{\lambda}_i, \\ \overline{P}_i - u_i^s, & \overline{\lambda}_i < \psi^s. \end{cases} \\ &=: h_i^o(u_i^s, \psi^s). \end{aligned} \tag{37}$$

Although the three-step procedure described above implies that each DGR must obtain every $\lambda \in \mathcal{U}$ (Step **S1**) before proceeding to run the ratio-consensus algorithm (Step **S2**), it is possible, with a slight modification, to execute the message-passing protocol and ratio consensus in parallel and still guarantee convergence of (35) and (36) (a formal proof can be found in [12]). In particular, each node i initializes z_i as before, but only node j initializes $\underline{y}_j^{(j)}[0] = g_j(\underline{\lambda}_j)$ and $\overline{y}_j^{(j)}[0] = g_j(\overline{\lambda}_j)$ since it is the only node that possesses $\underline{\lambda}_j$ and $\overline{\lambda}_j$ at $k = 0$. The remaining nodes initialize their numerator states corresponding to DGR j to zero, i.e., $\underline{y}_i^{(j)}[0] = 0$ and $\overline{y}_i^{(j)}[0] = 0$,

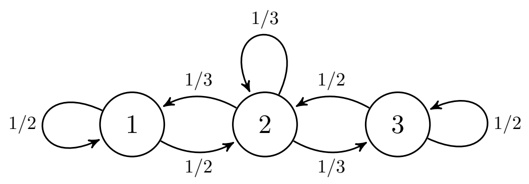


Fig. 6: Graph of linear 3-node network.

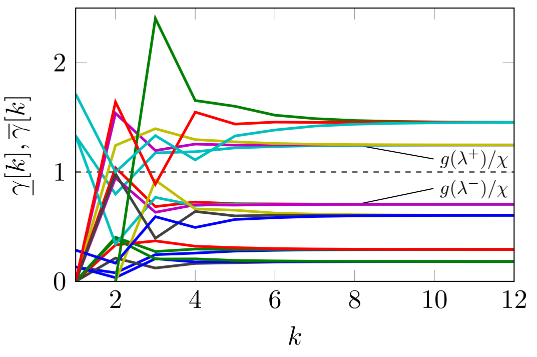


Fig. 7: Example evolution of the ratio-consensus algorithm.

for $i \neq j$, and, upon receiving $\underline{\lambda}_j$ and $\overline{\lambda}_j$, calculate $g_i(\underline{\lambda}_j)$ and $g_i(\overline{\lambda}_j)$, and add it to $\underline{y}_i^{(j)}$ and $\overline{y}_i^{(j)}$, respectively. The three-step procedure for distributively solving the optimal dispatch problem, including the modification to allow the message-passing protocol and ratio consensus to run in parallel, is described in Algorithm 1.

As was the case in the distributed algorithm for frequency regulation proposed in Section IV-A, the use of ratio consensus for obtaining $\underline{\gamma}_i^{(j)}$ and $\overline{\gamma}_i^{(j)}$ in (35) and (36) in practical optimal dispatch scenarios will only involve a finite number of iterations, k_o; this implies that the values $\underline{\gamma}_i^{(j)}$ and $\overline{\gamma}_i^{(j)}$ obtained at each node i will be bounded away from the true (limiting) values ($\frac{g(\underline{\lambda}_j)}{\chi}$ and $\frac{g(\overline{\lambda}_j)}{\chi}$, respectively) by a constant ϵ_o that can be made arbitrarily small by increasing k_o. The choice of ϵ_o affects the distance of the optimal solutions from the solution that will be chosen by the DGRs in a distributed fashion, but this distance can be made arbitrarily small by proper choice of k_o; we do not discuss the details of how to make this choice as this was not an issue in our experiments. The operation of Algorithm 1 is illustrated in the following example.

Example 3 (Optimal Dispatch): Consider the graph in Fig. 6 which represents the exchange of information between DGR local controllers in a 3-node network. We demonstrate the distributed optimal dispatch algorithm by showing the evolution of the $\underline{\gamma}$'s and $\overline{\gamma}$'s as computed by each of the local controllers. Let $\alpha = \left[-13/6, -3/2, -15/8\right]$, $\beta = \left[1/6, 1/2, 1/8\right]$, $\underline{P} = \left[10, 10, 20\right]$, and $\overline{P} = \left[100, 100, 120\right]$. Also, omitting the index of the round, let $u = \left[75, 75, 70\right]$ such that $\chi = 220$. Given the above values of α, β, $\underline{P}$, and $\overline{P}$, and the definitions given in Section IV-B, we have that $\underline{\lambda} = \left[73, 23, 175\right]$ and $\overline{\lambda} = \left[613, 203, 975\right]$.

The evolution of $\underline{\gamma}_i^{(j)}$ and $\overline{\gamma}_i^{(j)}$, for $i, j = 1, 2, 3$, is shown in Fig. 7. From the figure, we see that all of the values converge after around 9 iterations, allowing the nodes to determine which ratios are closest from below and above to 1, i.e., $g(\lambda^-)/\chi$ and $g(\lambda^-)/\chi$, the corresponding values of λ^- and λ^+, and the value of λ^* according to (32). After determining $\psi = \lambda^* = 425.29$, each node computes the amount by which it should adjust its generation set-point according to (37) and we have that $\Delta u = \left[-6.29, 25, -18.71\right]^{\mathrm{T}}$. ■

Algorithm 1: Distributed optimal dispatch

```
Each node $i,\ i = 1, \dots, n,$ separately does the following:
Input: $\alpha_i, \beta_i, \underline{P}_i, \overline{P}_i, u_i^s$
Output: $\Delta u_i^s$
begin
    initialize states:
        $\underline{\lambda}_i = \frac{\underline{P}_i - \alpha_i}{\beta_i}$        $\overline{\lambda}_i = \frac{\overline{P}_i - \alpha_i}{\beta_i}$
        $\underline{y}_i^{(i)}[0] = g_i(\underline{\lambda}_i)$        $\overline{y}_i^{(i)}[0] = g_i(\overline{\lambda}_i)$
        $\underline{y}_i^{(j)}[0] = 0$        $\overline{y}_i^{(j)}[0] = 0, \forall j \neq i$
        $z_i[0] = u_i^s$
    foreach iteration, $k = 0, 1, ..., k_o$ do
        if $\underline{\lambda}_i$, $\overline{\lambda}_i$ not broadcasted then
            broadcast $\underline{\lambda}_i$, $\overline{\lambda}_i$
        if received $\underline{\lambda}_j$, $\overline{\lambda}_j$ then
            $\underline{y}_i^{(j)}[k] = \underline{y}_i^{(j)}[k] + g_i(\underline{\lambda}_j)$
            $\overline{y}_i^{(j)}[k] = \overline{y}_i^{(j)}[k] + g_i(\overline{\lambda}_j)$
        update and broadcast states:
            $\underline{y}_i^{(j)}[k+1] = \sum_{l \in \mathcal{N}_i^-} \frac{1}{\mathcal{D}_l^+} \underline{y}_l^{(j)}[k]$
            $\overline{y}_i^{(j)}[k+1] = \sum_{l \in \mathcal{N}_i^-} \frac{1}{\mathcal{D}_l^+} \overline{y}_l^{(j)}[k]$
            $z_i[k+1] = \sum_{l \in \mathcal{N}_i^-} \frac{1}{\mathcal{D}_l^+} z_l[k]$
    compute:
        $\underline{\gamma}_i^{(j)} = \frac{\underline{y}_i^{(j)}[k_o]}{z_i[k_o]}$        $\overline{\gamma}_i^{(j)} = \frac{\overline{y}_i^{(j)}[k_o]}{z_i[k_o]}$
    determine $g(\lambda^+)/\chi$, $g(\lambda^-)/\chi$, $\lambda^+$, $\lambda^-$
    compute $\psi^s = \lambda^*$ according to (32)
    return $\Delta u_i^s = g_i(\psi^s) - u_i^s$
```

V. Choice of Controller Gains for Stabilizing Frequency

In this section, we will provide some guidelines for how to choose the κ_i's in (19) so that the distributed fair-splitting scheme for frequency regulation described in Section IV-A decreases the frequency error, $\Delta\omega^r = \omega^r - \omega_0$. Additionally, we will show that if $|\Delta\omega^r| \leq \epsilon$ for some $r = s$, such that the distributed algorithm for optimal dispatch described in Section IV-B is executed, then, $|\Delta\omega^{s+1}| \leq \epsilon$. To simplify the analysis in subsequent developments, we will make the following assumptions.

[A4.] The time constants associated with load changes are much slower than the time constants associated with the dynamics in (1) – (5), and the dynamics of the ratio-consensus algorithm.

[A5.] The error (with respect to the asymptotic solution) that results from executing ratio consensus a finite number of iterations is ignored. More specifically, the frequency regulation and optimal dispatch algorithms described in Section IV-A and Section IV-B are executed for a finite number of iterations, k_f and k_o, respectively, which causes inconsequential deviations from the asymptotic value defined in (21) and (37).

[A6.] Network losses are ignored, i.e., $P_l^r = 0$ in (11). While this simplifies the analysis, it does not preclude our distributed control approach from working on a lossy system. In fact, we show through experimental results in Section VII that our approach can work in the presence of lossy interconnections.

A. Error Dynamics During Frequency Regulation

Together with assumptions **[A4]** – **[A6]** and the analysis in Section II-A1, it can be seen that, at iteration r, and for constant $u_i^r,\ i = 1, 2, \ldots, n$, the common synchronous frequency at which the DGRs operate, ω^r, is given in (10), with the frequency error, $\Delta\omega^r$, defined in (11) (with $P_l^r = 0$). Therefore, since $u_i^{r+1} = u_i^r + \Delta u_i^r,\ i = 1, 2, \ldots, n$, it follows that

$$\begin{aligned}
\Delta\omega^{r+1} &= \frac{\sum_{i=1}^{n} u_i^{r+1} - \sum_{i=n+1}^{n+l} P_i^d}{\sum_{i=1}^{m}(D_i + 1/R_i\omega_0) + \sum_{i=m+1}^{n} H_i}, \\
&= \frac{\sum_{i=1}^{n} (u_i^{r} + \Delta u_i^r) - \sum_{i=n+1}^{n+l} P_i^d}{\sum_{i=1}^{m}(D_i + 1/R_i\omega_0) + \sum_{i=m+1}^{n} H_i}, \\
&= \frac{\sum_{i=1}^{n} u_i^{r} - \sum_{i=n+1}^{n+l} P_i^d}{\sum_{i=1}^{m}(D_i + 1/R_i\omega_0) + \sum_{i=m+1}^{n} H_i} + \frac{\sum_{i=1}^{n} \Delta u_i^r}{\sum_{i=1}^{m}(D_i + 1/R_i\omega_0) + \sum_{i=m+1}^{n} H_i}, \\
&= \Delta\omega^r + \frac{\sum_{i=1}^{n} \Delta u_i^r}{\sum_{i=1}^{m}(D_i + 1/R_i\omega_0) + \sum_{i=m+1}^{n} H_i}.
\end{aligned} \tag{38}$$

Now, it follows from (21) and (22) that $\sum_{i=1}^{n} \Delta u_i^r = \sum_{i=1}^{n} \kappa_i(\omega^r - \omega_0) = \sum_{i=1}^{n} \kappa_i \Delta\omega^r$, and by plugging this expression into (38) and rearranging terms, we obtain

$$\Delta\omega^{r+1} = \left(1 + \frac{\sum_{i=1}^{n} \kappa_i}{\sum_{i=1}^{m}(D_i + 1/R_i\omega_0) + \sum_{i=m+1}^{n} H_i}\right) \Delta\omega^r. \tag{39}$$

Thus, if $\left|1 + \frac{\sum_{i=1}^{n} \kappa_i}{\sum_{i=1}^{m}(D_i + 1/R_i\omega_0) + \sum_{i=m+1}^{n} H_i}\right| < 1$, then $\lim_{r\to\infty} \Delta\omega^r = 0$. Although there are many ways in which the κ_i's can be chosen such that $\Delta\omega^r$ goes to zero as r approaches infinity, a simple choice which requires only local information is to choose κ_i so as to satisfy

$$\begin{aligned}
-2(D_i + 1/R_i\omega_0) < \kappa_i < 0, &\qquad i = 1, 2, \ldots, m, \\
-2H_i < \kappa_i < 0, &\qquad i = m+1, m+2, \ldots, n.
\end{aligned}$$

B. Error Dynamics After Optimal Dispatch

As stated earlier, upon driving the frequency error to within the specified bound, i.e., $|\Delta\omega_r| < \epsilon$, the DGRs stop executing the distributed algorithm for frequency regulation and execute the distributed optimal dispatch algorithm described in Section IV-B. Let s denote the round at which the frequency error is driven below the error-tolerance, i.e., $|\Delta\omega^s| \leq \epsilon$. From the equality constraint in (23), it follows that the optimal dispatch solution $u_i^{s*},\ i = 1, 2, \ldots, n$, satisfies $\sum_{i=1}^n u_i^{s*} = \sum_{i=1}^n u_i^s$. But, from (15), (31), and (37), we also have that $u_i^{s+1} = u_i^s + \Delta u_i^s = u_i^{s*}$ such that $\sum_{i=1}^n \Delta u_i^s = 0$. Furthermore, given that

$$\Delta\omega^{s+1} = \Delta\omega^s + \frac{\sum_{i=1}^n \Delta u_i^s}{\sum_{i=1}^m (D_i + 1/R_i\omega_0) + \sum_{i=m+1}^n H_i}, \tag{40}$$

we have that $|\Delta\omega^{s+1}| = |\Delta\omega^s| \leq \epsilon$. Thus, after the frequency error is within the tolerance band, and the optimal dispatch algorithm is executed, the frequency error will remain within the tolerance band.

VI. Laboratory Testbed

We begin this section by describing the configuration and hardware specifications of the electrical network of a microgrid built in a laboratory to illustrate the effectiveness of the proposed distributed control architecture. Next, we provide the specifications for the hardware used to implement the microgrid's cyber network for communication, computation, and control. Then, we discuss the communication protocol created to exchange information for the distributed algorithms as well as a modification to ratio consensus which increases its robustness. Finally, we give an overview of a protocol used to synchronize the clocks of the nodes in the cyber network; this is important in order to ensure the correct execution of the control algorithms.

A. Physical Layer Hardware

To demonstrate the ability of the proposed distributed architecture to control a set of DGRs, we constructed the six-bus, 240 V, 3-phase power system shown in Fig. 8. The system is comprised of 3 Hampden Engineering synchronous machines, $\mathrm{G}_1, \mathrm{G}_2$, and G_3, (inverter-interfaced power supplies are omitted due to lack of availability) and 3 wye-connected resistive loads, labeled as $\mathrm{P}_1, \mathrm{P}_2$, and P_3. Each synchronous machine is connected at the shaft to a Kollmorgen Goldline brushless permanent magnet synchronous servomotor which serves as the prime mover.

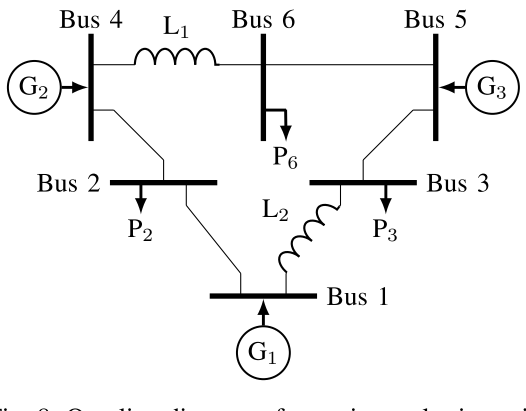


Fig. 8: One-line diagram of experimental microgrid.

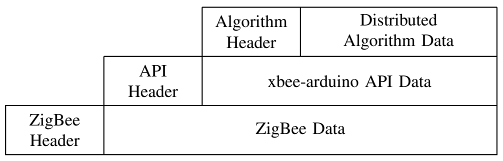


Fig. 9: Communication protocol stack.

The synchronous machines have 3 pole-pairs; thus, operation at a mechanical speed of 1200 rpm results in an electrical frequency of 60 Hz, i.e., $f_i = 3\omega_i$, where f_i is the electrical frequency at the terminals of DGR i and ω_i has units rpm. The per-phase resistance of each load is adjusted by adding 500 Ω resistors in parallel. Each load can have up to 10 resistors in parallel per phase, yielding resistances in the range $500, 250, \ldots, 50\,\Omega$. Extra impedance is added via series inductors between bus 4 and bus 6 as well as between bus 1 and bus 3, as shown in Fig. 8. The per-phase inductances, the synchronous generator, and prime mover data are given in Appendix A.

B. Cyber Layer Hardware

The hardware chosen to create the cyber network for communication and computation is based upon the open-source electronics prototyping platform Arduino. While there are other suitable hardware choices, we chose Arduino for its simplicity and for the extensive software libraries and extension circuit boards, called shields, that are available.

Each node in the cyber network is comprised of an Arduino Mega 2560 microcontroller board, connected to a MaxStream XB24-DMCIT-250 revB XBee module via a SparkFun Electronics XBee shield. The Arduino Mega 2560 is based on the Atmel AVR ATmega2560, an 8-bit microcontroller with 256 kB of flash memory, a clock speed of 16 MHz, and four universal asynchronous receiver/transmitter (UART) ports. The XBee shield serves as an interface between the Arduino board and the XBee module while providing the requisite 3.3 V power supply via a voltage regulator. The XBee is an embedded RF module with a built-in chip antenna operating at 2.4 GHz that allows the nodes in the testbed to exchange packetized data wirelessly. Although our testbed utilizes wireless communication and is based upon XBee transceivers, other wireless technologies such as WiFi or Bluetooth, or wired communication would also be suitable.

While the XBee shield allows for plug-and-play operation between the Arduino and the XBee, it is configured to utilize the same UART port as the onboard USB-to-serial converter, preventing the Arduino from simultaneously communicating with a computer via the USB port and other nodes in the testbed via XBee. Thus, to allow for concurrent connections, we use a modified shield which routes the XBee UART to an alternative port on the Arduino. While this modification is useful for logging the evolution of the distributed algorithms, it is also necessary for enabling the Arduinos to control the prime movers connected to the synchronous generators in the electrical network.

C. Cyber Layer Software

Next, we discuss the communication protocol developed to exchange information among nodes in the cyber network, as well as a modification to the ratio-consensus algorithm that enables robust operation in the presence of unreliable communication links. Additionally, we provide an overview of the protocol used to synchronize the clocks of the cyber network nodes.

1) Communication Protocol: The communication protocol created to exchange information among the cyber network nodes can be described by the three abstraction layers illustrated in Fig. 9. The lowest layer, commonly referred to as medium access control (MAC), is implemented on the XBee modules and is based upon the ZigBee (IEEE 802.15.4) standard. The middle layer is a version of the xbee-arduino API [42], modified to account for the aforementioned alteration to the XBee shield and to enable incoming and outgoing packets to be time-stamped in order to increase the accuracy of the synchronization mechanism discussed later in this section. Each packet generated by the xbee-arduino API contains information about the sender and the intended recipient, allowing the Arduinos to send unicast messages as well as determine the source of broadcasted packets. Note that to enable use of the xbee-arduino API, the XBee modules are placed in API mode (AP=2 with escapes). The header of the top layer contains information about which algorithm is being used, i.e., fair splitting or optimal dispatch, while the payload contains the actual information being exchanged.

In order to take advantage of the wireless medium used for communication among nodes, all of the packets used to exchange values for the distributed algorithms are broadcasted; that is, packets are not addressed to a particular node. Furthermore, to minimize network traffic, no acknowledgements are sent upon successful receipt of packets. To create a partially connected network despite the close proximity of the Arduinos during experimentation, each device is programmed to only accept packets received from the nodes in its in-neighborhood. In a more realistic setup, however, the testbed could be adapted to allow the availability of links between nodes to be determined dynamically based upon, for example, signal strength.

2) Ratio Consensus Implementation: Although the bottom layer of the protocol stack is designed to minimize packet collisions, it is still possible for packet loss to occur, particularly when attempting to reduce the time required to compute new generation commands using the distributed algorithms. To mitigate the effects of information lost due to temporarily unavailable communication links resulting from, e.g., neighboring nodes attempting to transmit simultaneously, we use a modified version of the ratio-consensus algorithm originally proposed in [43]. We refer to this variant of the ratio-consensus algorithm as *robust ratio consensus*; a brief overview of its operation as well as a short discussion on the implications with respect to its implementation in the laboratory testbed are provided in Appendix B. While the implementation details differ, it was shown in [43] that, under certain probabilistic assumptions on the link availability model, the asymptotic value of $\gamma_i[k]$ obtained with the robust ratio-consensus algorithm is identical to the original ratio-consensus algorithm described in Section II-C with probability one.

3) Clock Synchronization Protocol: Throughout the formulation of the ratio-consensus algorithm, we assumed that the local controller of all participating DGRs update the value of their state variables in unison; i.e., node i updates its state at iteration k at the same time node j updates its state, $\forall i, j, \in \mathcal{V}$. Without a common time reference and with no acknowledgements, however, it is possible for the local controllers to update their states at different times which may result in convergence to an inaccurate solution. While robust ratio consensus reduces the sensitivity of the system to timing errors, it is still possible for the nodes to converge to the wrong solution. Thus, before the distributed control algorithms are executed, all nodes are synchronized to a common time reference. The synchronization mechanism used in our hardware testbed is based on the hierarchy referencing time synchronization (HRTS) protocol proposed in [44]. This protocol has low overhead requirements and is capable of synchronizing the clocks of several nodes to a reference using only three packets. A detailed description of the synchronization process is provided in [45].

4) Initialization Protocol: Following the synchronization of the clocks, one of the nodes must initiate the distributed algorithms to ensure that all nodes in the network begin execution at roughly the same time. To facilitate this, the first node that detects that the frequency error has exceeded the specified bound or that the DGRs should be optimally dispatched will broadcast a scheduling packet containing information about the algorithm used, the start time, the number of iterations, and the period of each iteration. Upon receiving the scheduling packet, all other nodes will rebroadcast it to allow the information to propagate throughout the network, then wait until the scheduled start time. If multiple scheduling packets are received, the one with the earliest start time is chosen.

VII. Experimental Results

Using the laboratory testbed described in the previous section, we verify the effectiveness of our distributed generation control architecture under a variety of scenarios. We first present results demonstrating the distributed frequency regulation and optimal dispatch functions controlling the DGRs in the experimental microgrid following both an increase and a decrease in load. We then add a fourth DGR to the system which acts as a spinning reserve to illustrate how the local controller of each DGR can independently determine if the collective power output limits have been exceeded and act accordingly.

Throughout this section, all frequency is reported in hertz, and, to accommodate laboratory equipment features, the limits and set-points of the prime movers are given as torque rather than as power. To provide greater control while performing the experiments, instead of allowing a random DGR to initialize the algorithms as specified in Section VI-C4, in the results that follow, we select a single DGR, referred to as the leader, which is responsible for estimating the total torque that needs to be added or removed at each round, i.e., without loss of generality let "1" index the leader, then $\Delta\hat{u}_1^r = \kappa_1(f^r - f_0)$ while $\Delta\hat{u}_i^r = 0, \forall i \neq 1, \forall r$. Furthermore, to ensure that ψ^r can be reliably computed to sufficient accuracy at each round, we chose to have the nodes execute 50 iterations with a period of 50 ms for the frequency regulation algorithm, and 100 iterations with a 300 ms period for the optimal dispatch algorithm.[4] Note that the period and number of iterations to be executed were chosen conservatively to allow sufficient time for the algorithms to converge and to reduce the response time of the system in case of a malfunction during experimentation; both the number of iterations and their period could be reduced given more development time, and by making use of algorithms such as the one proposed in [38].

In each of the scenarios we tested experimentally, the coefficients defining the DGR cost functions are $\alpha = \left[-13/6, -3/2, -15/8\right]^{\mathrm{T}}$ N·m and $\beta = \left[1/6, 1/2, 1/8\right]^{\mathrm{T}}$ N·m. Additionally, the desired nominal frequency is 60 Hz, i.e., $f_0 = 60$ Hz, and the error bound used by the leader to determine when to trigger the optimal dispatch algorithm as defined in (17) is $\epsilon = 0.4$ Hz. Finally, in all following scenarios, the loads are three-phase balanced while the field excitation of each of the DGRs is left constant, i.e., voltage is unregulated.

[4]As implied by Lemma 1, convergence of the ratio-consensus algorithm is asymptotic. In practice, however, a finite number of iterations is sufficient to converge to a solution that is accurate to within the capabilities of the DGRs. For the experimental results presented in this section, the number of iterations necessary was determined *a priori* for each communication graph and was pre-programmed into the leader node.

A. Load Increase

We begin by illustrating the system response to a load increase. In this experiment, the exchange of information between local DGR controllers is represented by the graph shown in Fig. 6, where nodes $1, 2, 3$ correspond to generators G_1, G_2, G_3 in Fig. 8, respectively. The minimum output torque of all DGRs is 0 N·m while the maximum output torques for DGRs G_1, G_2, and G_3 are 4, 2.5, and 3.5 N·m, respectively. Additionally, the gain used by node 1 (the leader node) to compute $\Delta\hat{u}_1^r$ according to (19) is chosen to be $\kappa_1 = 85$ μN·m·s.

The plot in Fig. 10 shows the torque set-points for each of the three generators as well as the frequency of DGR G_1 as the system responds to two load increases at times $t = 100$ s and $t = 400$ s, and a subsequent generator re-dispatch at time $t = 775$ s which is triggered to minimize the overall generation cost. For the first and second load changes, the total power drawn is increased from 313 W to 396 W, and from 396 W to 465 W by adjusting the resistive load from 500 Ω/phase to 250 Ω/phase at buses 3 and 2, respectively.

From Fig. 10 we see that, following the first load increase, the frequency measured at DGR G_1 returns to 60 Hz after four rounds of the fair-splitting algorithm. If we let $r = 0$ be the index of the round before the first load increase and subsequent execution of the frequency regulation algorithm, then the values of ψ^r as defined in (21) for $r = 1, 2, 3, 4$ are 0.58, 0.59, 0.59, and 0.60, respectively. Similar to the first load change, three rounds of the fair-splitting algorithm, corresponding to $r = 5, 6, 7$, return the frequency to the nominal value following the second load change. The values of ψ^r for the last three rounds of the frequency regulation algorithm are 0.63, 0.65, and 0.65, respectively.

After the two load changes and the execution of 7 rounds of the fair-splitting algorithm, the frequency is within the specified bound, i.e., $|f_1 - f_0| \leq \epsilon$, and, as described by (17), the optimal dispatch function is used to determine the amount by which the DGRs should adjust their output in order to minimize the overall cost. If we let $s = 8$ be the index of the round that the DGRs are re-dispatched optimally, then the local controllers determine that $\psi^s = 276.36$ N·m, and, as Fig. 10 shows, the DGRs adjust their set-points according to (37) at time $t = 775$ s.

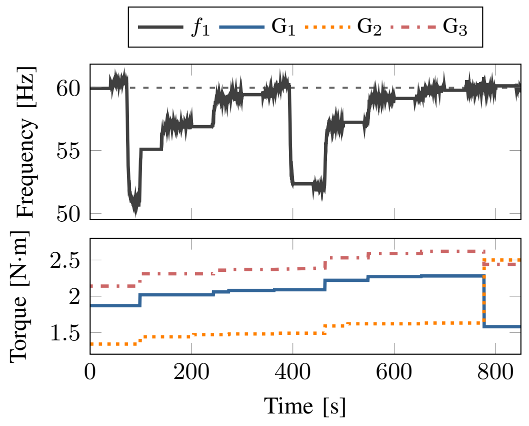


Fig. 10: System response to load increase.

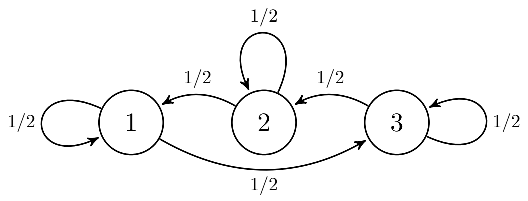

$$\begin{tikzcd}
1 \arrow[loop left, "1/2"] \arrow[rr, bend right, "1/2"'] & 2 \arrow[loop above, "1/2"] \arrow[l, bend right, "1/2"'] & 3 \arrow[loop right, "1/2"] \arrow[l, bend right, "1/2"']
\end{tikzcd}$$

Fig. 11: Graph of cyclic 3-node network.

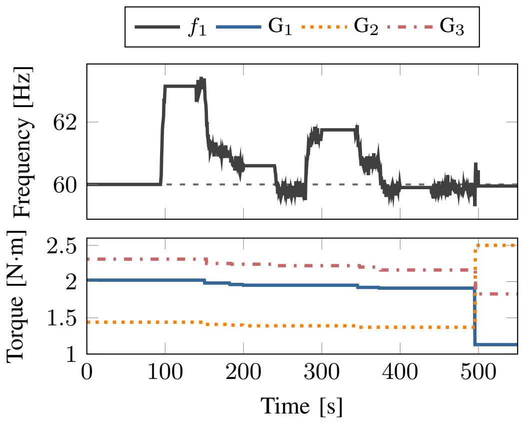


Fig. 12: System response to load decrease.

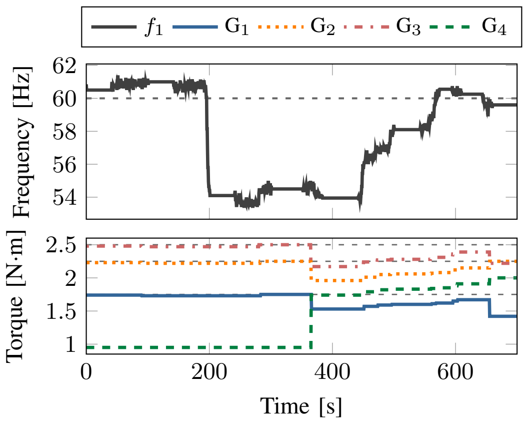


Fig. 13: System response before and after spinning reserve switch-in.

B. *Load Decrease*

We now illustrate the system response to a load decrease. For this experiment, the cyclic graph in Fig. 11 represents the exchange of information among the DGR local controllers. Similar to the previous experiment, DGR 1 is selected to be the leader with $\kappa_1 = 90$ μN·m·s, the minimum output torques of all DGRs are 0 N·m, and the maximum output torques for DGRs G$_1$, G$_2$, and G$_3$ are 4, 2.5, and 3.5 N·m, respectively.

The plot in Fig. 12 shows the torque set-points for each of the three generators as well as the frequency measured at DGR G$_1$ as the system responds to two load decreases at times $t = 100$ s and $t = 300$ s, and a subsequent generator re-dispatch at time $t = 500$ s, which is triggered to minimize the overall generation cost. For the first and second load changes, the total power drawn is decreased from 390 W to 365 W, and from 365 W to 352 W by adjusting the resistive load at bus 6 from 500 Ω/phase to 750 Ω/phase, and from 750 Ω/phase to 1000 Ω/phase, respectively.

Following the first load decrease, we see from Fig. 12 that the frequency of DGR G$_1$ returns to 60 Hz after three rounds of the fair-splitting algorithm. If we let $r = 0$ be the index of the round before the first load change and subsequent execution of the frequency regulation algorithm, then the values of ψ^r for $r = 1, 2, 3$ are 0.56, 0.56, and 0.56, respectively. After the second load change, two rounds of the fair-splitting algorithm, corresponding to $r = 4$ and 5, return the frequency to the nominal value. The values of ψ^r for the last two rounds of the frequency regulation algorithm are 0.55 and 0.54.

After the two load changes and the execution of 5 rounds of the fair-splitting algorithm, the frequency is within the specified bound and the generators are re-dispatched in order to minimize the overall cost. If we let $s = 6$ be the index of the round that the optimal dispatch function is executed, then the local controllers determine that $\psi^s = 243.57$ N·m and the DGRs adjust their set-points according to (37) at time $t = 500$ s as shown in Fig. 12.

C. Load Increase with Spinning Reserve

As illustrated in Example 2, the value of ψ^r obtained through the distributed computation of the frequency regulation function allows each DGR local controller to independently determine if the collective power output limits have been exceeded. Next, we demonstrate how this information can be used by a DGR acting as a spinning reserve to determine when to come online.

In order to demonstrate the capability of each generator to independently determine when the demand for generation exceeds the collective output limits of the online DGRs, we connected an additional DGR to bus 2, which we refer to as DGR G_4. Initially, the additional DGR limits its output to the minimum torque required to operate at nominal speed, but participates in the distributed algorithm with a maximum torque of 0.95 N·m. The actual maximum torque of the DGR G_4 is 2 N·m, while DGRs G_1, G_2, and G_3 can output up to 2.5, 2.25, and 1.75 N·m, respectively; the minimum torque output is 0.1 N·m for all DGRs. The exchange of information among the DGRs conforms to the graph in Fig. 3.

The plot in Fig. 13 shows the prime mover torque and the frequency of DGR G_1 as the electrical load is increased. At $t \approx 275$ s, the maximum output of DGRs G_1, G_2, and G_3 is reached, and the frequency error can no longer be driven to zero. At this point, by obtaining a value of ψ^r greater than one, DGR G_4 determines that the demand exceeds the collective limits and adjusts its maximum torque output to its actual value of 2 N·m. This occurs at $t \approx 360$ s, after which all DGRs are able to increase their output to drive the frequency to the nominal value of $f_0 = 60$ Hz. Once the reserve generator is switched in and the frequency is returned to the nominal value, the optimal dispatch function is executed and the DGRs adjust their set-points at time $t \approx 650$ s.

VIII. Concluding Remarks

In this paper, we proposed a distributed architecture for generation control in islanded ac microgrids. While microgrids are smaller and have lower ratings than, for example, bulk power transmission systems, the control objectives are similar; thus, the control functions of our architecture were derived from the three control functions provided by generation control architectures commonly adopted in large power systems. These functions are (i) droop control, (ii) frequency regulation, and (iii) optimal dispatch.

While droop control is completely decentralized, the implementation of the frequency regulation and optimal dispatch functions is typically centralized. However, by relying on local measurements, information obtained from neighboring generating units, and simple computations, we are able to achieve the same control objectives as those achieved by a centralized implementation. Compared to centralized ones, our distributed control approach can more easily adapt to changes, allowing the system to operate regardless of additions or removals of generating units.

A major component of this research was to experimentally verify the effectiveness of the proposed control architecture. To this end, we built an experimental microgrid comprised of several small synchronous generators and resistive loads all connected in a ring network. In this microgrid, the exchange of information among generating units was achieved via a wireless communications network which enabled the implementation of our distributed generation control algorithms for frequency regulation and optimal dispatch. We utilized this microgrid to verify the performance of these algorithms under numerous scenarios, including a case in which one of the generators, which was acting as spinning reserve, was able to come online after detecting that the collective power output limits were exceeded.

APPENDIX A
PHYSICAL LAYER HARDWARE DATA

TABLE I: Value of added inductances in power system

Parameter	Inductance [mH]
$L_{1,A}$	2.041
$L_{1,B}$	1.905
$L_{1,C}$	1.961
$L_{2,A}$	4.175
$L_{2,B}$	4.162
$L_{2,C}$	4.059

TABLE II: Hampden Engineering Corporation Synchronous Machine

Parameter	Value
Armature Voltage	133/230 RMS Volts
Armature Current	15.5/9 RMS Amps
Horsepower	2 Hp
Speed	1200 rpm
Frequency	60 Hz
Model	Syn-2

TABLE III: Kollmorgen Goldline Brushless Permanent Magnet Servomotor

Parameter	Value
Stall Current (Continuous)	10.3 RMS Amps
Stall Current (Peak)	33.0 RMS Amps
Torque (Continuous)	6.44 N·m
Torque (Peak)	19.5 N·m
Rated L/L Voltage	230 RMS Volts
Torque (Continuous)	6.44 N·m
Maximum Speed	4900 rpm
Frequency	164 Hz
Model	B-206-C-21

APPENDIX B
RATIO CONSENSUS IMPLEMENTATION DETAILS

Rather than broadcast the latest state values as in (12) – (13), nodes participating in the modified ratio-consensus algorithm broadcast the sum of the weighted states up to and including the current iteration k. For the case when all links are available, i.e., no packets are lost, the weighted states for iteration k can be inferred from the information a DGR receives from its in-neighbors. If a link is temporarily unavailable, however, the modification to the algorithm allows the receiving nodes to recover any lost information at the next successful iteration.

As before, each node maintains two states, $y_i[k]$ and $z_i[k]$, that are updated at each iteration. Using the modified algorithm, however, each node maintains two additional states, $\mu_i[k]$ and $\sigma_i[k]$, which are the values broadcasted to the out-neighbors of DGR i at iteration k. The values of $\mu_i[k]$ and $\sigma_i[k]$ are the sum of $y_i[k]/\mathcal{D}_i^+$ and $z_i[k]/\mathcal{D}_i^+$

since the iterative process began, and thus, they are updated as follows:

$$\mu_i[k+1] = \mu_i[k] + \frac{1}{\mathcal{D}_i^+} y_i[k] = \sum_{t=0}^{k} \frac{1}{\mathcal{D}_i^+} y_i[t], \tag{41}$$

$$\sigma_i[k+1] = \sigma_i[k] + \frac{1}{\mathcal{D}_i^+} z_i[k] = \sum_{t=0}^{k} \frac{1}{\mathcal{D}_i^+} z_i[t], \tag{42}$$

with $\mu_i[0] = 0$ and $\sigma_i[0] = 0$. To account for the fact that the values received from in-neighbors are accumulated sums, each DGR i updates its states as

$$\begin{aligned} y_i[k+1] &= \frac{1}{\mathcal{D}_i^+} y_i[k] + \sum_{\substack{j \in \mathcal{N}_i^- \\ i \neq j}} (\nu_{ij}[k+1] - \nu_{ij}[k]), \\ z_i[k+1] &= \frac{1}{\mathcal{D}_i^+} z_i[k] + \sum_{\substack{j \in \mathcal{N}_i^- \\ i \neq j}} (\tau_{ij}[k+1] - \tau_{ij}[k]), \end{aligned} \tag{43}$$

where the values of $\nu_{ij}[k+1]$ and $\tau_{ij}[k+1]$ depend on the successful receipt of a packet from DGR j during iteration k, and are given by[5]

$$\begin{aligned} \nu_{ij}[k+1] &= \begin{cases} \mu_j[k+1], & \text{if } (i,j) \in \mathcal{E}[k], \\ \nu_{ij}[k], & \text{if } (i,j) \notin \mathcal{E}[k], \end{cases} \\ \tau_{ij}[k+1] &= \begin{cases} \sigma_j[k+1], & \text{if } (i,j) \in \mathcal{E}[k], \\ \tau_{ij}[k], & \text{if } (i,j) \notin \mathcal{E}[k]. \end{cases} \end{aligned} \tag{44}$$

Upon examining (44), we see that in order to implement the robust ratio-consensus algorithm, each node must store the most recent successfully received values, necessitating a unique identifier for each node in the in-neighborhood as well as a list of the identifiers of all possible in-neighbors given all available communication links. To identify the senders in our communication and computation testbed, we utilize the 64-bit hardware address associated with the XBee modules which is included in the xbee-arduino API headers. Furthermore, in our setup, the list of possible in-neighbors is programmed on the devices rather than generated at runtime.

Another important implementation detail that is evident from (44) is the fact that the previous successfully received values from each in-neighbor must be saved. In particular, each local controller i must store $\nu_{ij}[k]$ and $\eta_{ij}[k]$ for all $j \in \mathcal{N}_i^-$. While these values can be overwritten upon successfully receiving a packet from an in-neighbor, we see that an additional variable must be utilized for each numerator and denominator of ratio consensus, which, in the case of the distributed optimal dispatch algorithm, amounts to $2n+1$ extra variables.

[5]In order to take into account for the possibility that communication links may not be available at every iteration, it is necessary to slightly modify the graph-theoretic model describing the exchange of information among DGRs introduced earlier. To this end, we denote the graph representing the network interconnecting the DGRs as $\mathcal{G}[k] = \{\mathcal{V}, \mathcal{E}[k]\}$, where $\mathcal{V}$ is independent of k as defined before, and $\mathcal{E}[k]$ is the set of edges where $(i,j) \in \mathcal{E}[k]$ if DGR i can receive information from DGR j at iteration k. We assume that $\mathcal{E}[k] \subseteq \mathcal{E}$, $\forall k \geq 0$, where $\mathcal{E}$ is as defined in Section II-B, and describes the scenario in which all communication links are available.

References

[1] R. Lasseter *et al.*, "Integration of distributed energy resources: The CERTS microgrid concept," Lawrence Berkeley National Laboratory, Tech. Rep. LBNL-50829, Apr. 2002.

[2] J. Peças Lopes, C. Moreira, and A. Madureira, "Defining control strategies for microgrids islanded operation," *IEEE Transactions on Power Systems*, vol. 21, no. 2, pp. 916–924, May 2006.

[3] F. Katiraei and M. Iravani, "Power management strategies for a microgrid with multiple distributed generation units," *IEEE Transactions on Power Systems*, vol. 21, no. 4, pp. 1821–1831, Nov. 2006.

[4] A. Tsikalakis and N. Hatziargyriou, "Centralized control for optimizing microgrids operation," *IEEE Transactions on Energy Conversion*, vol. 23, no. 1, pp. 241–248, Mar. 2008.

[5] A. Emadi and M. Ehsani, "Aircraft power systems: technology, state of the art, and future trends," *IEEE Aerospace and Electronic Systems Magazine*, vol. 15, no. 1, pp. 28–32, Jan. 2000.

[6] J. Rosero, J. Ortega, E. Aldabas, and L. Romeral, "Moving towards a more electric aircraft," *IEEE Aerospace and Electronic Systems Magazine*, vol. 22, no. 3, pp. 3–9, Mar. 2007.

[7] K. Butler, N. Sarma, and V. Ragendra Prasad, "Network reconfiguration for service restoration in shipboard power distribution systems," *IEEE Transactions on Power Systems*, vol. 16, no. 4, pp. 653–661, Nov. 2001.

[8] A. Monti, D. Boroyevich, D. Cartes, R. Dougal, H. Ginn, G. Monnat, S. Pekarek, F. Ponci, E. Santi, S. Sudhoff, N. Schulz, W. Shutt, and F. Wang, "Ship power system control: a technology assessment," in *Proc. of IEEE Electric Ship Technologies Symposium*, 2005, pp. 292–297.

[9] T. Gruzs and J. Hall, "Ac, dc or hybrid power solutions for today's telecommunications facilities," in *Proc. of International Telecommunications Energy Conference*, 2000, pp. 361–368.

[10] A. Wood and B. Wollenberg, *Power Generation, Operation, and Control*. New York, NY: Wiley, 1996.

[11] S. T. Cady and A. D. Dominguez-Garcia, "Distributed generation control of small-footprint power systems," in *Proc. of North American Power Symposium*, 2012, pp. 1–6.

[12] A. D. Domínguez-García, S. T. Cady, and C. N. Hadjicostis, "Decentralized optimal dispatch of distributed energy resources," in *Proc. of IEEE Conference on Decision and Control*, 2012, pp. 3688–3693.

[13] N. Hatziargyriou, H. Asano, R. Iravani, and C. Marnay, "Microgrids," *IEEE Power and Energy Magazine*, vol. 5, no. 4, pp. 78–94, July 2007.

[14] A. Micallef, M. Apap, C. Spiteri-Staines, and J. Guerrero, "Secondary control for reactive power sharing in droop-controlled islanded microgrids," in *Proc. of IEEE International Symposium on Industrial Electronics*, 2012, pp. 1627–1633.

[15] M. Marwali, J.-W. Jung, and A. Keyhani, "Control of distributed generation systems - part II: Load sharing control," *IEEE Transactions on Power Electronics*, vol. 19, no. 6, pp. 1551–1561, Nov. 2004.

[16] G. Larsen, N. van Foreest, and J. Scherpen, "Distributed control of the power supply-demand balance," *IEEE Transactions on Smart Grid*, vol. 4, no. 2, pp. 828–836, June 2013.

[17] J. W. Simpson-Porco, F. Dörfler, and F. Bullo, "Synchronization and power sharing for droop-controlled inverters in islanded microgrids," *Automatica*, vol. 49, no. 9, pp. 2603–2611, 2013.

[18] M. Andreasson, H. Sandberg, D. Dimarogonas, and K. Johansson, "Distributed integral action: Stability analysis and frequency control of power systems," in *Proc. of IEEE Conference on Decision and Control*, 2012, pp. 2077–2083.

[19] Q. Shafiee, J. Guerrero, and J. Vasquez, "Distributed secondary control for islanded microgrids – a novel approach," *IEEE Transactions on Power Electronics*, vol. 29, no. 2, pp. 1018–1031, Feb. 2014.

[20] A. Bidram, A. Davoudi, F. Lewis, and Z. Qu, "Secondary control of microgrids based on distributed cooperative control of multi-agent systems," *IET Generation, Transmission Distribution*, vol. 7, no. 8, pp. 822–831, Aug. 2013.

[21] A. Venkat, I. Hiskens, J. Rawlings, and S. Wright, "Distributed MPC strategies with application to power system automatic generation control," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 6, pp. 1192–1206, 2008.

[22] M. Arnold, R. R. Negenborn, G. Andersson, and B. D. Schutter, "Multi-area predictive control for combined electricity and natural gas systems," in *Proc. of European Control Conference*, 2009.

[23] A. Maknouninejad, W. Lin, H. G. Harno, Z. Qu, and M. A. Simaan, "Cooperative control for self-organizing microgrids and game strategies for optimal dispatch of distributed renewable generations," *Energy Systems*, pp. 23–60, March 2012.

[24] Y. Zhang, N. Gatsis, and G. Giannakis, "Robust distributed energy management for microgrids with renewables," in *Proc. of IEEE Conference on Smart Grid Communications*, 2012, pp. 510–515.

[25] M. Andreasson, D. Dimarogonas, K. Johansson, and H. Sandberg, "Distributed vs. centralized power systems frequency control," in *Proc. of European Control Conference*, 2013, pp. 3524–3529.

[26] S. Bolognani and S. Zampieri, "Distributed control for optimal reactive power compensation in smart microgrids," in *Proc. of IEEE Conference on Decision and Control and European Control Conference*, 2011, pp. 6630–6635.

[27] E. Dall'Anese, H. Zhu, and G. Giannakis, "Distributed optimal power flow for smart microgrids," *IEEE Transactions on Smart Grid*, vol. 4, no. 3, pp. 1464–1475, Sept. 2013.

[28] F. Dörfler, J. W. Simpson-Porco, and F. Bullo, "Breaking the Hierarchy: Distributed Control & Economic Optimality in Microgrids," *IEEE Transactions on Control of Network Systems*, January 2014, submitted. Available at http://arxiv.org/pdf/1401.1767v1.pdf.

[29] N. Li, L. Chen, C. Zhao, and S. Low, "Connecting automatic generation control and economic dispatch from an optimization view," in *Proc. of American Control Conference*, 2014, pp. 735–740.

[30] P. Sauer and A. Pai, *Power System Dynamics and Stability*. Upper Saddle River, NJ: Prentice Hall, 1998.

[31] P. Varaiya, F. Wu, and R.-L. Chen, "Direct methods for transient stability analysis of power systems: Recent results," *Proc. of the IEEE*, vol. 73, no. 12, pp. 1703–1715, Dec. 1985.

[32] H.-D. Chiang, *Direct Methods for Stability Analysis of Electric Power Systems: Theoretical Foundation, BCU Methodologies, and Applications*. New York, NY: Wiley, 2011.

[33] F. Dorfler, M. Chertkov, and F. Bullo, "Synchronization in complex oscillator networks and smart grids," *Proc. of the National Academy of Sciences*, vol. 110, no. 6, pp. 2005–2010, February 2013.

[34] D. B. West, *Introduction to Graph Theory*, 2nd ed. Prentice Hall, 2001.

[35] A. D. Domínguez-García and C. N. Hadjicostis, "Distributed algorithms for control of demand response and distributed energy resources," in *Proc. of IEEE Conference on Decision and Control*, 2011, pp. 27–32.

[36] A. S. Debs, *Modern Power Systems Control and Operation*. Boston, MA: Kluwer Academic Publishers, 1988.

[37] A. Bergen and V. Vittal, *Power System Analysis*. Upper Saddle River, NJ: Prentice Hall, 2000.

[38] N. Manitara and C. N. Hadjicostis, "Distributed stopping strategies for average consensus in digraphs," in *Proc. of IEEE International Symposium on Communications, Control, and Signal Processing*, May 2014.

[39] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY: Cambridge University Press, 2004.

[40] D. Bertsekas, *Nonlinear Programming*. Belmont, MA: Athena Scientific, 2004.

[41] M. Madrigal and V. Quintana, "An analytical solution to the economic dispatch problem," *IEEE Power Engineering Review*, vol. 20, no. 9, pp. 52–55, Sep. 2000.

[42] A. Rapp. xbee-arduino. [Online]. Available: http://code.google.com/p/xbee-arduino/

[43] A. D. Domínguez-García, C. N. Hadjicostis, and N. Vaidya, "Resilient networked control of distributed energy resources," *IEEE Journal on Selected Areas in Comm.*, vol. 30, no. 6, pp. 1137–1148, Jul. 2012.

[44] H. Dai and R. Han, "TSync: a lightweight bidirectional time synchronization service for wireless sensor networks," *ACM SIGMOBILE Mobile Computing Communications Review*, vol. 8, pp. 125–139, Jan. 2004.

[45] S. T. Cady, A. D. Domínguez-García, and C. N. Hadjicostis, "Robust implementation of distributed algorithms for control of distributed energy resources," in *Proc. of North American Power Symposium*, 2011, pp. 1–5.